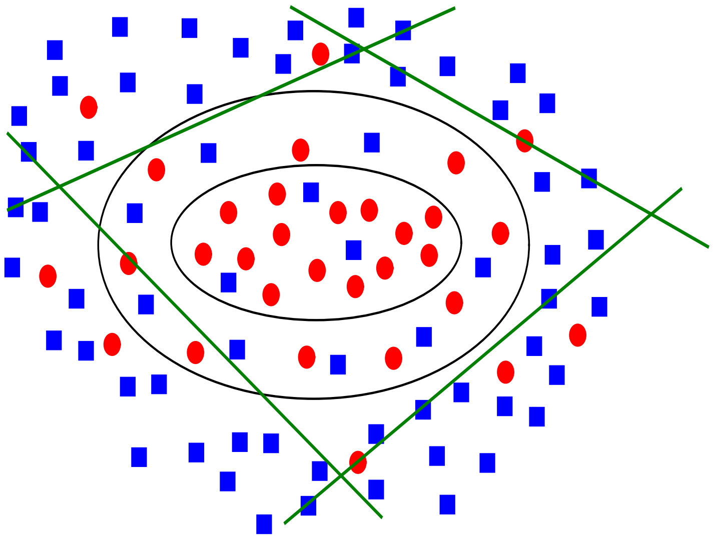
Boosting Tutorial

Ron Meir

Department of Electrical Engineering
Technion, Israel



Course Outline

Sources of Information:

Internet www.boosting.org

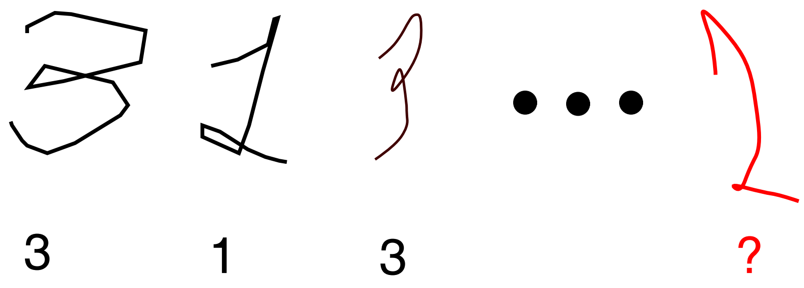
Journals and papers Machine Learning, Journal of Machine Learning Research, Neural Computation, Annals of Statistics, ...
Many papers available from www.boosting.org

People List available at www.boosting.org

Software

- Matlab http://mlg.anu.edu.au/~raetsch/Software.html
- Matlab http://tiger.technion.ac.il/~eladyt/Classification_toolbox.html (general purpose Matlab toolbox - only basic AdaBoost supported)
- Splus http://www-stat.stanford.edu/~jhf/MART.html

Learning - Problem Formulation I



The 'World':

Data: $\{(x_i, y_i)\}_{i=1}^{m}$, $x_i \in \mathbb{R}^d$, $y_i \in \{\pm 1\}$

Unknown target function: $y = f(x)$ (or $y \sim P(y|x)$)

Unknown distribution: $x \sim p(x)$

Objective: Given **new** x, predict y

Problem: $P(x, y)$ is unknown!

Learning - Problem Formulation II

The 'Model'

Hypothesis class: $\mathcal{H} : \mathbb{R}^d \mapsto \{\pm 1\}$

Loss: $\ell(y, h(x)) \quad$ (e.g. $I[y \neq h(x)]$)

Objective: Minimize the true (expected loss) - generalization

$$\min_{h \in \mathcal{H}} \{\mathbf{E}\ell(Y, h(X))\}$$

Caveat: Only have **data** at our disposal

'Solution': Form empirical estimator which 'generalizes well'

Question: How can we efficiently construct complex hypotheses with good generalization?

Notation

Data:	$D_m = \{(x_1, y_1), \ldots, (x_m, y_m)\}$
Source:	$P(X, Y)$
Hypothesis space:	$\mathcal{H}$
Loss:	$\ell(y, h(x))$
Empirical loss:	$\sum_{i=1}^{m} \ell(y_i, h(x_i))$
True loss:	$L(h) = \mathbf{E}\ell(Y, h(X))$
Bayes loss:	$L^* = \text{argmin}_h L(h)$ (h unrestricted)
Optimum in class:	$L^*_{\mathcal{H}} = \text{argmin}_{h \in \mathcal{H}} L(h)$
Empirical estimator:	$\hat{h}_m \in \mathcal{H}$ - based on D_m
	A random variable

PAC LEARNING

Input: Sample $D_m = (x_1, y_1), \ldots, (x_m, y_m) \sim P(X, Y)$

Accuracy parameter ϵ

Confidence parameter δ

A hypothesis class $\mathcal{H}$

Algorithm: A mapping from D_m to $\mathcal{H}$

Output: A hypothesis $\hat{h}_m \in \mathcal{H}$

Requirements:

- Show that $\hat{h}_m$ obeys

$$\mathbf{Pr}\left\{D_m : \; L(\hat{h}_m) - L^*_{\mathcal{H}} > \epsilon\right\} < \delta$$

- Require that algorithm run in time polynomial in $m, 1/\epsilon, 1/\delta$.

Dependence on the Distribution

Distribution free Require that PAC property hold for every distribution

Distribution dependent Demand that PAC hold only for given, fixed distribution

Intermediate case Require that property holds for a large class of distributions

Levels of Generality

Recall

$$h_{\mathrm{B}} = \mathrm{argmin}_h \, L(h) \qquad \text{(}h \text{ unrestricted)}$$
$$h^* = \mathrm{argmin}_{h \in \mathcal{H}} \, L(h)$$

The restricted setting Assume that

$$h_{\mathrm{B}} \in \mathcal{H}$$

The agnostic setting Assume nothing about h_{B}, require

$$L(\hat{h}_m) \xrightarrow{P} L(h^*)$$

Universal setting Require that

$$L(\hat{h}_m) \xrightarrow{P} L(h_{\mathrm{B}})$$

Weak and Strong Learning

Assumption: Restricted model, $h_{\mathrm{B}} \in \mathcal{H}$

Strong PAC **Learning** - Demand small error with high probability

$$\mathbf{Pr}\left\{L(\hat{h}_m) - L^* > \epsilon\right\} < \delta \qquad (*)$$

for small ϵ and δ.

Weak PAC **Learning** - Demand that $(*)$ holds for 'large' (but not trivial) ϵ

- Example: Binary classification, require that

$$\epsilon \leq \frac{1}{2} - \gamma \qquad (\gamma > 0)$$

Are Weak and Strong Learning Related?

Question: Can a weak learning algorithm be transformed into a strong learning algorithm?

Answer: Yes - in a distribution free setting (Schapire, 90)

No - in a distribution dependent setting (Kearns and Valiant 94)

Early Algorithms:

Boosting by Filtering (Schapire 1990)

Boosting by Majority (Freund 1995)

Basic Issues in Boosting

Main sources:

- Breiman 1996
- Freund 1995, Freund and Schapire 1996
- Schapire, Freund, Bartlett and Lee 1998

Early Algorithms

Boosting by Filtering (Schapire 1990)

- ⋆ Run weak learners on different distributions of the examples
- ⋆ Combine the weak learners
- ⋆ Complex; Requires prior knowledge about performance of weak learners

Boosting by Majority (Freund 1995)

- ⋆ Run weak learners on different distributions of the examples
  - – Different learners excel on different subsets
- ⋆ Combine the weak learners using Majority
- ⋆ Requires prior knowledge about performance of weak learners

Combining Classifiers

Input: A pool of binary classifiers $h_1, h_2, \ldots, h_k$

Objective: A composite classifier

$$f(x) = \operatorname{sgn}\left(\sum_{i=1}^{k} \alpha_i h_i(x)\right)$$

Question: When can this procedure succeed?

Require diversity of the classifiers

Diversity: Use different subsets of the data for each h_i

Use different features

Decorrelate classifiers during training

BAGGING I

Idea: Generate diversity in pool by training on different subsets

Bootstrap sample: Given $S = (x_1, y_1), \ldots, (x_m, y_m)$ generate S' by choosing i.i.d. pairs with replacement

Bagging (Breiman 1996)

Input: Training set S, Integer T

- For $t = 1, \ldots, T$
  - S_t = bootstrap sample from S
  - Construct classifier h_t based on S_t
- End For
- Output classifier: Majority vote of $\{h_1, h_2, \ldots, h_T\}$

Bias/variance - Regression I

Regression Setting: Easier to understand than classification

Data: $D = \{(x_i, y_i)\}_{i=1}^m x_i \in \mathbb{R}^d, \; y_i \in \mathbb{R}$

Source: (x_i, y_i) i.i.d. $P(X, Y)$

Objective: Find $f \in \mathcal{F}$ such that

$\mathbf{E}(Y - f(X))^2$ is minimal

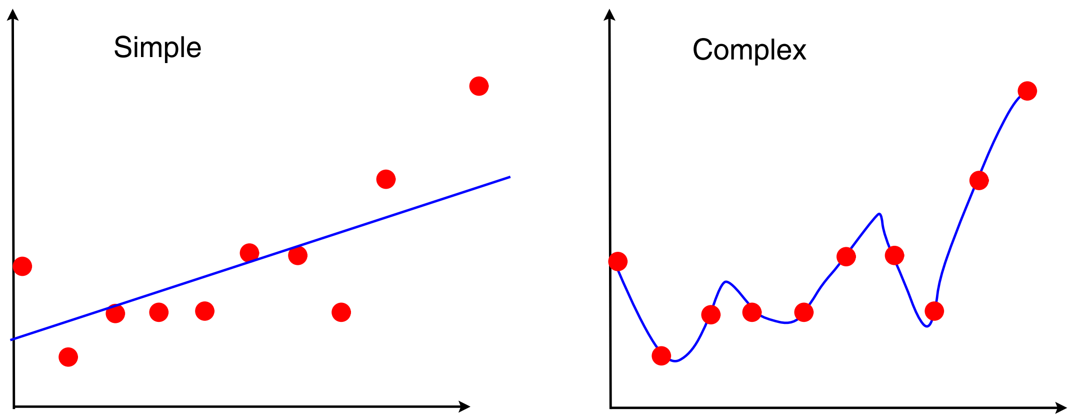
Optimum: $f^*(x) = \mathbf{E}(Y|x)$

Estimator: $\hat{f}_m$

Two problems: Only have a finite training set

Complexity of $\mathcal{F}$ is unknown

Bias/variance - Regression II



Trade-off:

Complex $\mathcal{F}$	overfitting
Simple $\mathcal{F}$	underfitting
Objective:	Find ‘best balance’ between the two
	Split error into Bias + Variance

Bias/variance - Regression II

Identify error components:

$$\mathbf{E}_D \mathbf{E}_{X,Y} \left(\hat{f}(X) - Y\right)^2$$

$$= \mathbf{E}_D \mathbf{E}_X \left(\hat{f}(X) - \mathbf{E}(Y|X)\right)^2 + \mathbf{E}_{X,Y} \left(Y - \mathbf{E}(Y|X)\right)^2$$

$$= \mathbf{E}_X \mathbf{E}_D \left(\hat{f}(X) - \mathbf{E}_D \hat{f}(X)\right)^2 \qquad \text{(variance)}$$

$$+ \mathbf{E}_X \left(\mathbf{E}_D \hat{f}(X) - \mathbf{E}(Y|X)\right)^2 \qquad (\text{bias})^2$$

$$+ \mathbf{E}_X \left(Y - \mathbf{E}(Y|X)\right)^2 \qquad \text{(noise)}$$

Unbiased estimator: bias $= 0$.

Bias/variance Tradeoff

Objective: Minimize bias and variance simultaneously - usually impossible

Tradeoff: Small data sets and large $\mathcal{F}$

variance large, bias small

Large data sets and small $\mathcal{F}$

variance small, bias large

Combining Regressors - Bias

Set of estimators: $\hat{f}_1(x), \hat{f}_2(x), \ldots, \hat{f}_k(x)$

Simple average: $\hat{f}(x) = \frac{1}{k}\sum_{i=1}^{k} \hat{f}_i(x)$

Bias:

$$
\begin{aligned}
B_f(x) &= \mathbf{E}_D \hat{f}(x) - \mathbf{E}(Y|x) \\
&= \frac{1}{k}\sum_{i=1}^{k} \mathbf{E}_D \hat{f}_i(x) - \mathbf{E}(Y|x)
\end{aligned}
$$

Unbiased estimators remain unbiased - more likely for complex estimators

Combining Regressors - Variance I

$$
\begin{aligned}
V_f(x) &= \mathbf{E}_D \left(\hat{f}(X) - \mathbf{E}_D \hat{f}(X) \right)^2 \\
&= \mathbf{E}_D \left(\frac{1}{k} \sum_{i=1}^{k} \hat{f}_i(x) - \frac{1}{k} \sum_{i=1}^{k} \mathbf{E}_D \hat{f}_i(x) \right)^2 \\
&= \mathbf{E}_D \left(\frac{1}{k} \sum_{i=1}^{k} [\hat{f}_i(x) - \mathbf{E}_D \hat{f}_i(x)] \right)^2 \\
&= \frac{1}{k^2} \sum_{i=1}^{k} \mathrm{Var}\left\{ \hat{f}_i(x) \right\} + \frac{1}{k^2} \sum_{i \neq j} \sum \mathrm{Cov}\left\{ \hat{f}_i(x), \hat{f}_j(x) \right\}
\end{aligned}
$$

Combining Regressors - Variance II

Recall

$$V_f(x) = \frac{1}{k^2}\sum_{i=1}^{k} \mathrm{Var}\left\{\hat{f}_i(x)\right\} + \frac{1}{k^2}\sum_{i\neq j}\sum \mathrm{Cov}\left\{\hat{f}_i(x), \hat{f}_j(x)\right\}$$

Assume:

Covariances small: $\mathrm{Cov}\left\{\hat{f}_i(x), \hat{f}_j(x)\right\} \approx 0$

Variances similar: $\mathrm{Var}\left\{\hat{f}_i(x)\right\} \approx v$

Then

$$\boxed{V_f(x) \approx \frac{v}{k}}$$

Reduction: by a factor of $1/k$ - main effect if bias unchanged

Bias/Variance - Classification

Note: No generally agreed upon split of classification error into bias variance

Alternatives: Will not discuss

Intuition:

Bias: Measures the average correctness of the classifier across many data sets

Variance: Measures the fluctuations in the classifier's performance

Bagging II

Why does Bagging work? (A simple explanation)

Covariances small:	Due to using different subsets for training
Variances similar:	Estimator from each sub-sample behaves similarly (on average)
Biases:	Weakly affected

More elaborate explanation - Bühlman and Yu 1999

Takeaway Messages:

- ⋆ It is advantageous to reduce dependence between component estimators
- ⋆ Can we reduce bias and variance simultaneously?

The Idea of Boosting

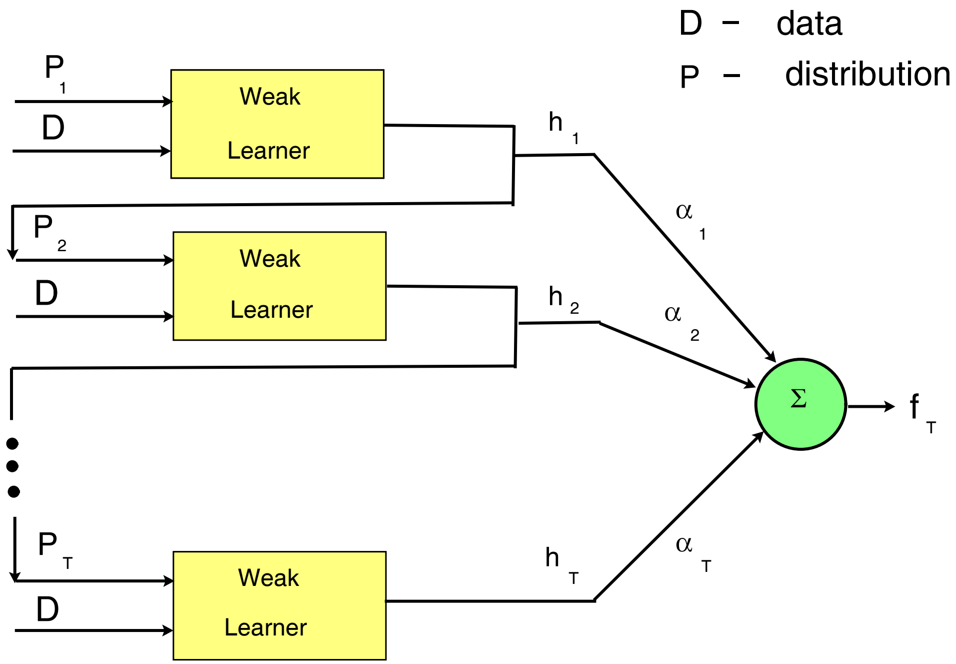
Basic idea: An adaptive combination of poor learners, forced to differ from each other, leads to an excellent (complex) classifier!

Base class: $\mathcal{H}$ - base class of simple classifiers (e.g., linear)

Output Classifier: $f_T(x) = \mathrm{sgn}\left(\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x})\right), \quad (h_t \in \mathcal{H})$

Idea outline: Train a sequence of simple classifiers on modified data distributions, and form a weighted average

AdaBoost



Weak Learner: (h_t binary)

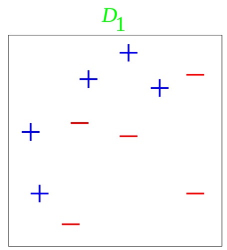
$$\epsilon_t \triangleq \sum_{i=1}^{m} P_t(i) I[h_t(x_i) \neq y_i]$$

AdaBoost

1. **Initialize:** $P_1(i) = 1/n$, $t = 1$
2. **While** $t \leq T$ **&** $\epsilon_t < 1/2$
   - **Construct binary weak classifier:**
   $$h_t = \operatorname*{argmin}_{h \in \mathcal{H}} \left\{ \sum_{i=1}^{m} P_t(i) I[y_i \neq h(x_i)] \right\}$$
   - **Update distribution:**
   $$P_{t+1}(i) = \frac{P_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$
   - **Compute weights:** $\alpha_t = \frac{1}{2} \ln \left(\frac{1-\epsilon_t}{\epsilon_t} \right)$
   - Set $t \leftarrow t + 1$
3. **Final hypothesis:** $f(x) \triangleq \left(\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x}) \right)$

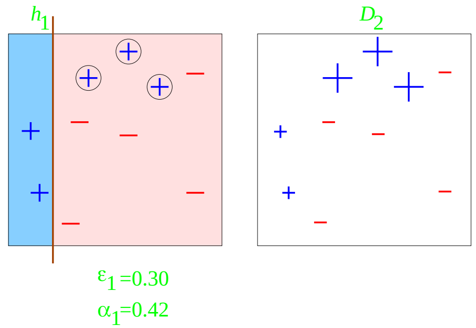
AdaBoost in Action I

Toy Example



Weak hypotheses == vertical or horizontal half-planes

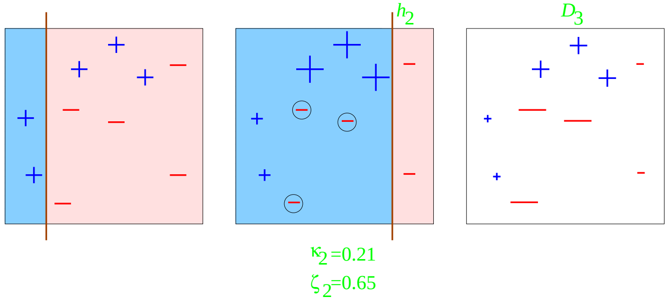
Round 1



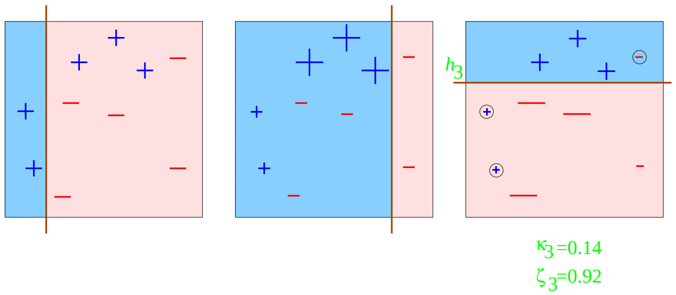
Source: Singer and Lewis Tutorial

AdaBoost in Action II

Round 2



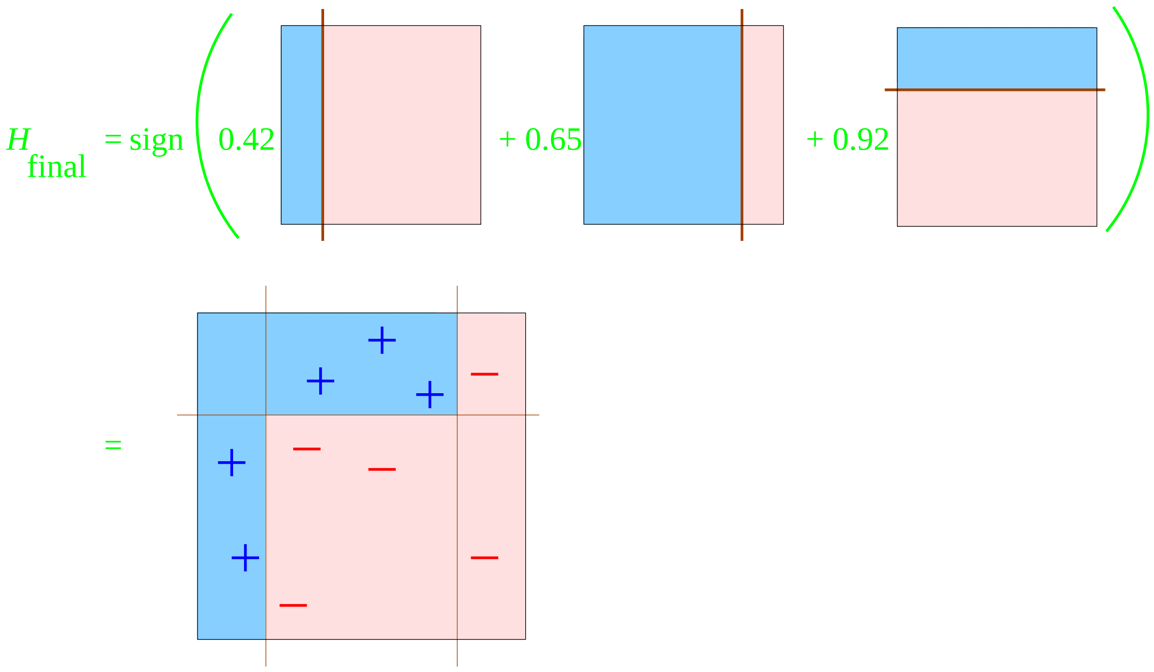
Round 3



Source: Singer and Lewis Tutorial

AdaBoost in Action III

Final Hypothesis



Source: Singer and Lewis Tutorial

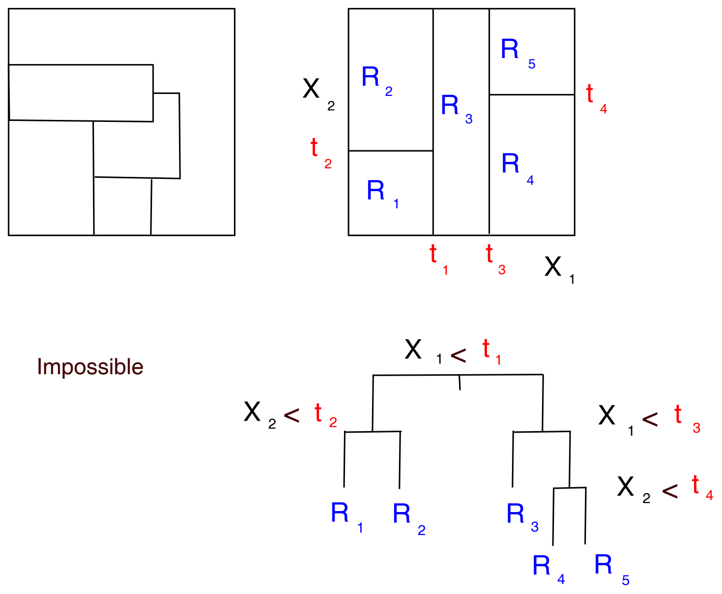
Weak Learners Used for Boosting

Stumps:	Single axis parallel partition of space
Decision trees:	Hierarchical partition of space
Multi-layer perceptrons:	General non-linear function approximators
Radial basis functions:	Non-linear expansions based on kernels

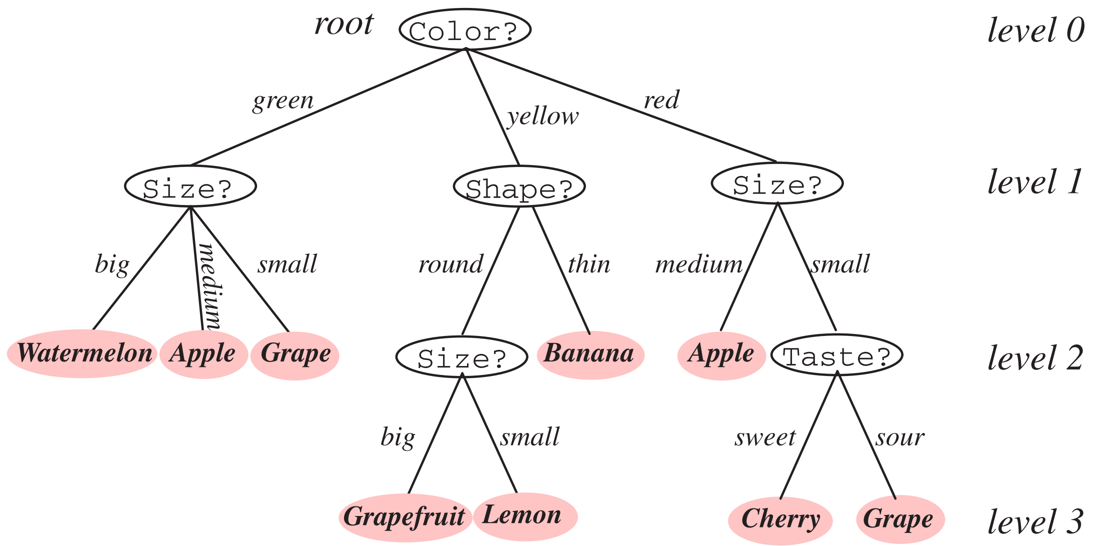
Decision Trees

Basic idea:

- ⋆ Hierarchical and recursive partitioning of the feature space
- ⋆ A simple model (e.g., constant) is fit in each region
- ⋆ In many approaches, split is axis-parallel

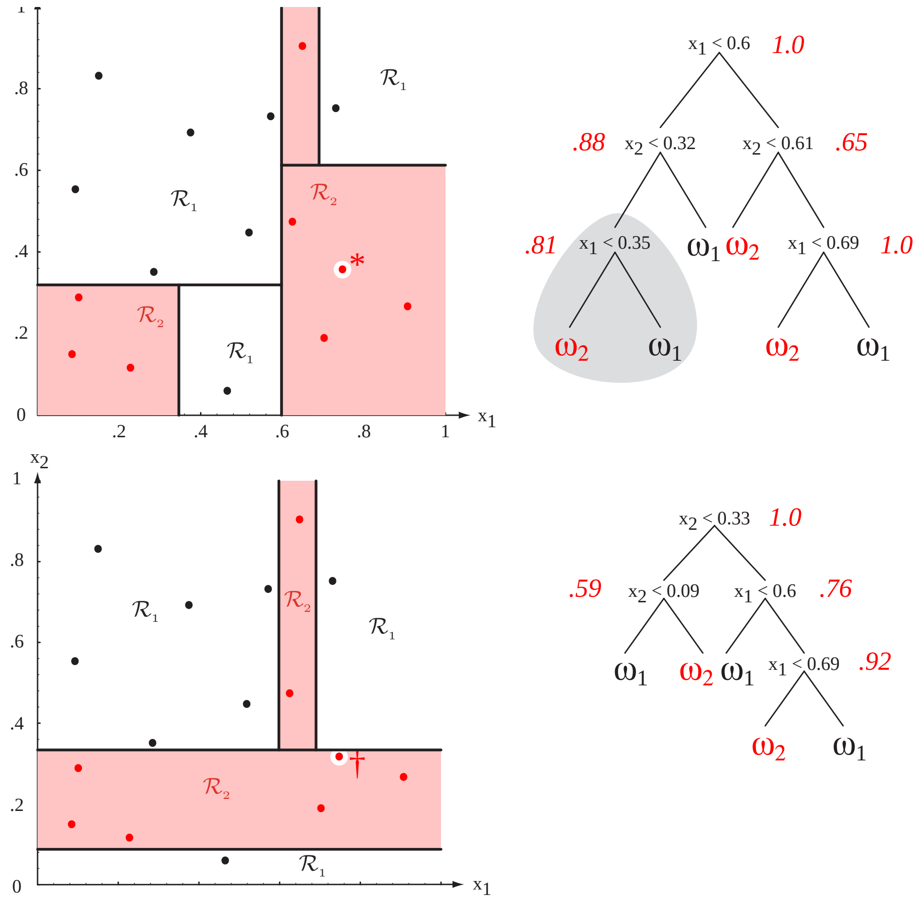


DECISION TREES - NOMINAL FEATURES



Source: Duda, Hart and Stork textbook

Decision Trees - Instability

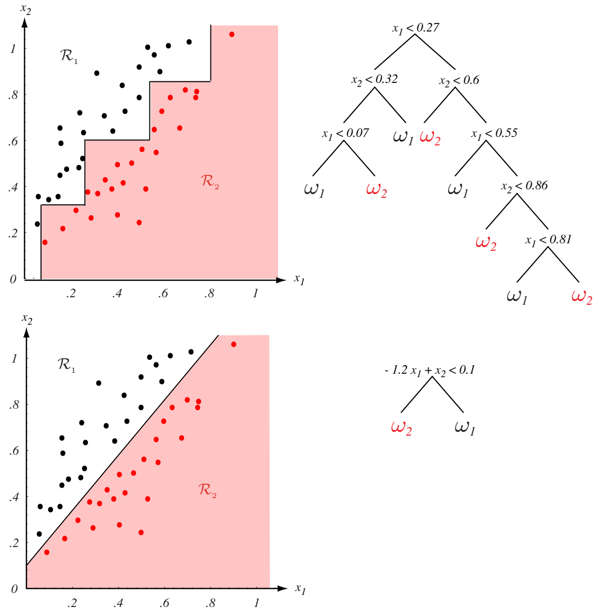


Source: Duda, Hart and Stork textbook

Decision Trees - Oblique Splits

Parallel splits: Representationally Restrictive, but easy to interpret and construct

Oblique splits: Richer; hard to interpret and construct



Source: Duda, Hart and Stork textbook

Variations on AdaBoost

Will be discussed in Part 6

The Behavior on the Sample

Main sources:

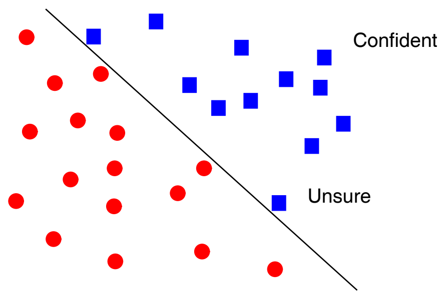
- Schapire and Singer 1999
- Schapire, Freund, Bartlett and Lee 1998

MARGINS AND CLASSIFICATION

Motivation: We lose information in classifying a point as ± 1

Observation: We have higher confidence in correctly classified points, which are far from the decision boundary

Suggestion: Use real number to indicate confidence



Margin: $\text{margin}_f(x, y) = yf(x)$

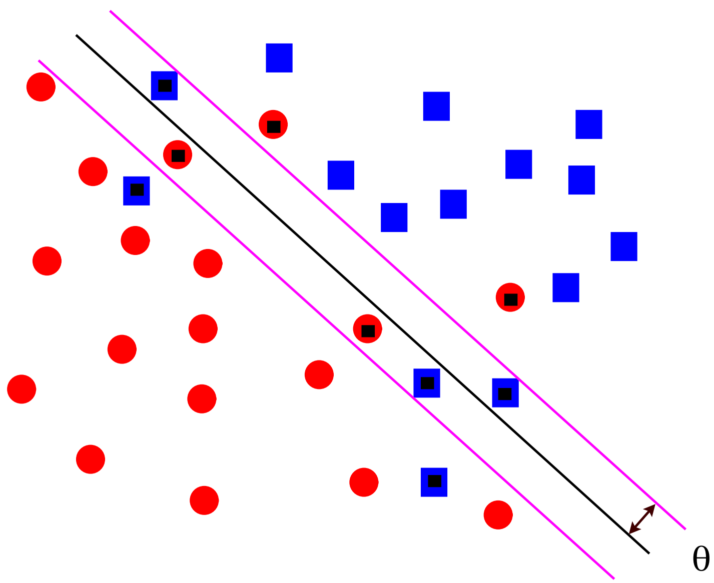
Hyper-plane: $\text{margin}_{w,w_0}(x, y) = y(w^T x + w_0)$

THE EMPIRICAL MARGIN ERROR

Empirical margin error:

$$\hat{L}^{\theta}{}_m(f) = \frac{1}{m}\sum_{i=1}^{m} I[y_i f(x_i) \leq \theta]$$

(Scale of θ and f must match!)



The Empirical Margin Error

Lemma: Assume $h_t(x) \in \{-1, +1\}$, set

$$f_T(x) = \frac{\sum_{t=1}^T \alpha_t h_t(x)}{\sum_{t=1}^T \alpha_t} \qquad (f \in [-1, +1]),$$

where $\{\alpha_t\}$ obtained from AdaBoost. Then

$$\begin{aligned} \hat{L}_m^\theta(f_T) &= \frac{1}{m} \sum_{i=1}^m I[y_i f_T(x_i) \leq \theta] \\ &\leq e^{\theta \sum_{t=1}^T \alpha_t} \left(\prod_{t=1}^T Z_t \right) \end{aligned}$$

Proof

$$
\begin{aligned}
Z_t &= \sum_{i=1}^{n} P_t(i) e^{-y_i \alpha_t h_t(x_i)} \\
&= \sum_{i: y_i = h_t(x_i)} P_t(i) e^{-\alpha_t} + \sum_{i: y_i \neq h_t(x_i)} P_t(i) e^{\alpha_t} \\
&= (1 - \epsilon_t) e^{-\alpha_t} + \epsilon_t e^{\alpha_t}
\end{aligned}
$$

$$
y f_T(x) \leq \theta \quad \Rightarrow \quad y \sum_{t=1}^{T} \alpha_t h_t(x) \leq \theta \sum_{t=1}^{T} \alpha_t
$$

$$
\exp\left(-y \sum_{t=1}^{T} \alpha_t h_t(x) + \theta \sum_{t=1}^{T} \alpha_t \right) \geq 1 \quad (y f(x) \leq \theta)
$$

$$
I[y f(x) \leq \theta] \leq \exp\left(-y \sum_{t=1}^{T} \alpha_t h_t(x) + \theta \sum_{t=1}^{T} \alpha_t \right)
$$

Proof, Cont'd

$$P_{t+1}(i) = P_t(i)\frac{\exp\left(-\alpha_t y_i h_t(x_i)\right)}{Z_t}$$

$$= \frac{\exp\left(-\sum_{\tau=1}^{t} \alpha_\tau y_i h_\tau(x_i)\right)}{m\prod_\tau Z_\tau} \qquad \text{(Induction)}$$

$$\hat{\mathbf{E}}\{I[yf(x) \le \theta]\} \le \hat{\mathbf{E}}\left\{e^{-y\sum_{t=1}^{T}\alpha_t h_t(x) + \theta\sum_{t=1}^{T}\alpha_t}\right\}$$

$$= \frac{1}{m} e^{\theta\sum_{t=1}^{T}\alpha_t} \sum_{i=1}^{n} e^{-y_i\sum_{t=1}^{T}\alpha_t h_t(x_i)}$$

$$= e^{\theta\sum_{t=1}^{T}\alpha_t}\left(\prod_{t=1}^{T} Z_t\right)\underbrace{\sum_{i=1}^{n} P_{T+1}(i)}_{=1}$$

Empirical Margin Error Bound

Claim: Selecting $\alpha_t = (1/2)\ln((1-\epsilon_t)/\epsilon_t)$ leads to the bound

$$Z_t = 2\sqrt{\epsilon_t(1-\epsilon_t)}.$$

Proof Straightforward by substitution

Conclusion: Recall $\hat{L}_m^\theta(f) = \frac{1}{m}\sum_{i=1}^m I[y_i f(x_i) \le \theta]$

$$\boxed{\hat{L}_m^\theta(f_T) \le \prod_{t=1}^{T} \sqrt{4\epsilon_t^{1-\theta}(1-\epsilon_t)^{1+\theta}}}$$

If	$\epsilon_t = 1/2 - \gamma_t, \quad \gamma_t \ge \theta \ \forall t$
Then	$\hat{L}_m^\theta(f_T) \to 0$ exponentially fast!

Training Error

Consider $\theta = 0$,

$$\hat{L}_m(f) = \frac{1}{m}\sum_{i=1}^{m} I[y_i f(x_i) \leq 0]$$

Using $\ln x \leq x - 1$

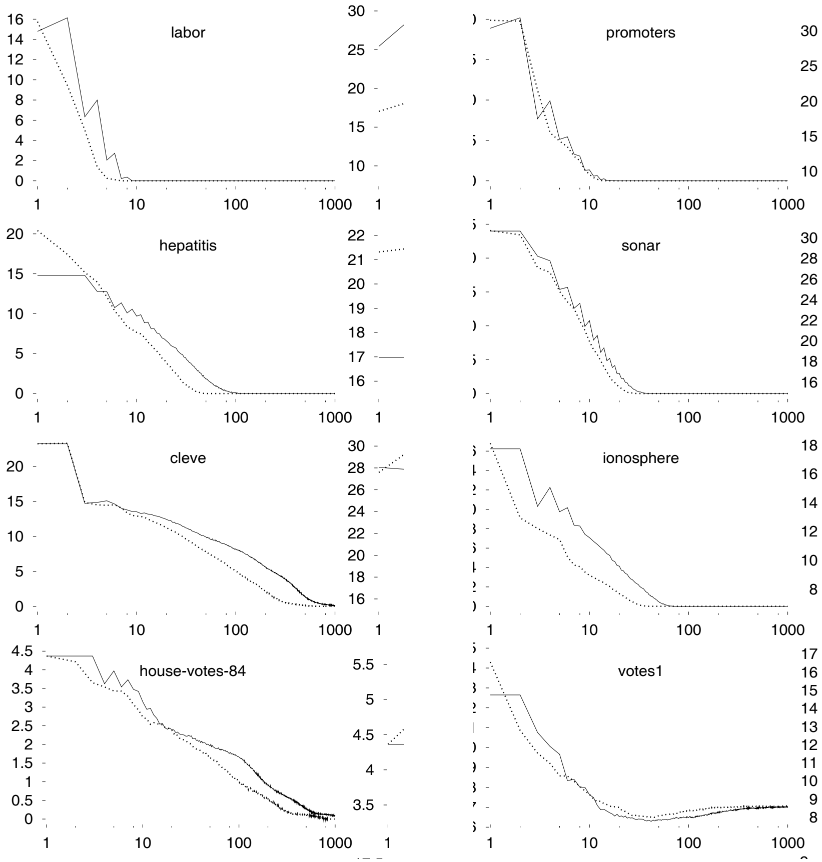
$$\hat{L}_m(f_T) \leq \prod_{t=1}^{T} \sqrt{(1-4\gamma_t^2)}$$

$$\leq e^{-2\sum_{t=1}^{T}\gamma_t^2}$$

$$\to 0 \quad \text{if} \sum_{t=1}^{T}\gamma_t^2 \to \infty$$

Training Error can be driven to zero, if weak learners are sufficiently strong

Training Error for Real Data



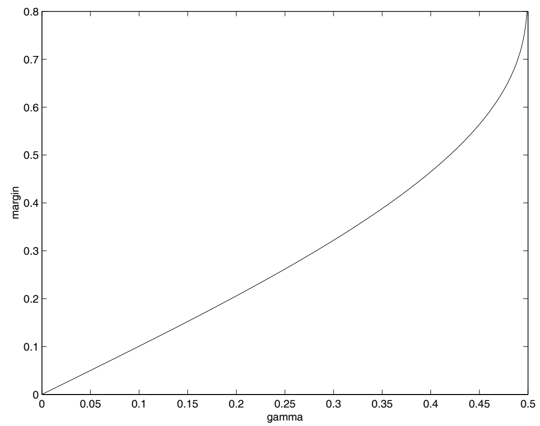
Source: Schapire and Singer 1999

The Minimal Margin

Assume $\epsilon_t \leq 1/2 - \gamma$

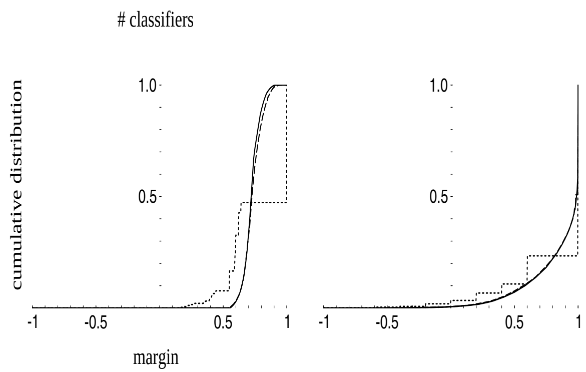
Then

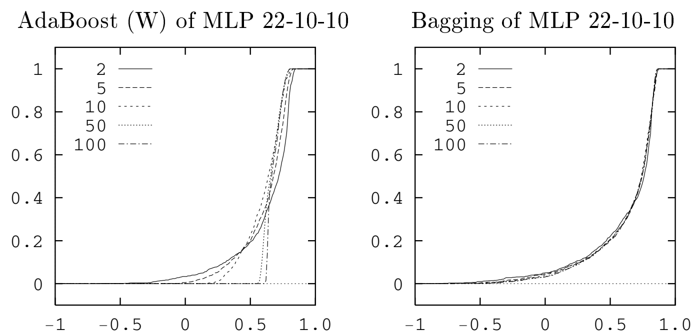
$$\text{Minimal margin} \geq \frac{\log \frac{1/4}{(1/2-\gamma)(1/2+\gamma)}}{\log \frac{1/2+\gamma}{1/2-\gamma}}$$



Margin Plots I

Margin plots: Histogram of $y_i f_T(x_i)$, $i = 1, 2, \ldots, m$





Decison Trees

Schapire *et al.* 1998

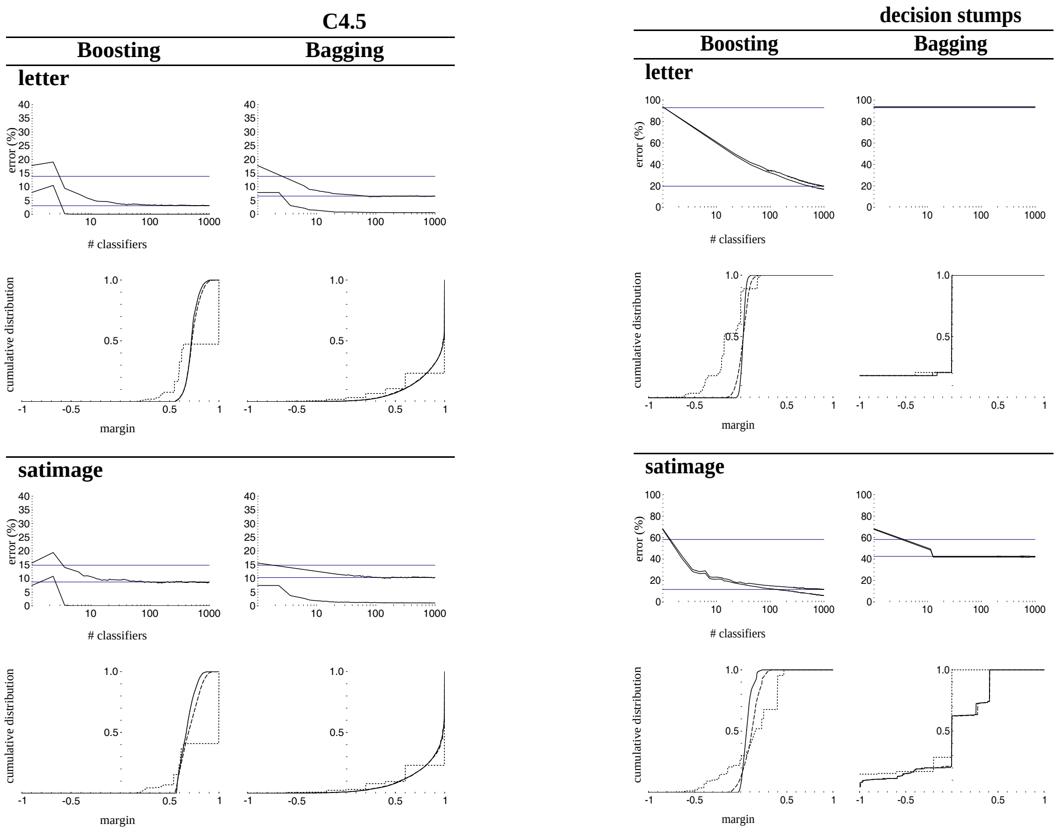
Neural Networks

Schwenk and Bengio 2000

Observation: AdaBoost increases the margins of most points

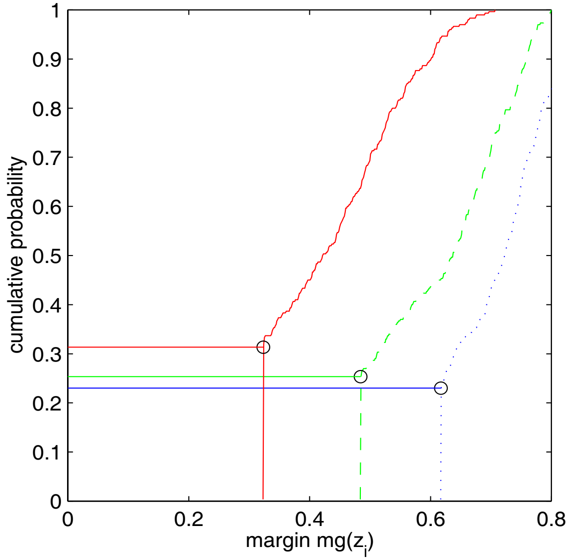
Bagging has a broader spectrum of distributions

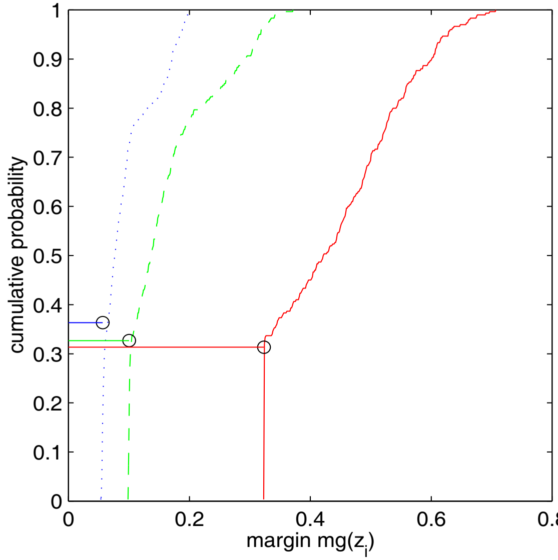
Margin Plots I



Source: Schapire, Freund, Bartlett and Lee 1998

The effect of Noise and Complexity





Effect of Noise on margin dist.

Noise ⇒ smaller margins

Effect of complexity on margin dist.

Complexity ⇒ larger margins

Source: Rätsch *et al.*, 2000

Interpreting the Boosting Weights

Recall that

$$\hat{L}(f_T) \leq \prod_{t=1}^{T} Z_t(\alpha) \quad ; \quad Z_t(\alpha) = \sum_{i=1}^{m} P_t(i) e^{-\alpha y_i h_t(x_i)}$$

Minimizing $Z_t(\alpha)$,

$$\frac{dZ_t(\alpha)}{d\alpha} = -\sum_{i=1}^{m} P_t(i) y_i h_t(x_i) e^{-\alpha y_i h_t(x_i)} = -Z_t(\alpha) \sum_{i=1}^{m} P_{t+1}(i) y_i h_t(x_i) = 0$$

Conclude

$$\sum_{i=1}^{m} P_{t+1}(i) y_i h_t(x_i) = 0$$

Interpretation: The new distribution is uncorrelated with the previous hypothesis h_t, so that the new hypothesis, based on P_{t+1} will also be uncorrelated with $h_t(x)$

Generalization Error

Main sources:

- Schapire, Freund, Bartlett and Lee 1998
- Schapire and Singer 1999
- Kégl, Linder and Lugosi 2001

Setup

Expected error: $L(f) = \mathbf{E}\{I[yf(x) \leq 0]\}$

Data: $(x_1, y_1), \ldots, (x_m, y_m)$

Empirical error: $\hat{L}_m(f) = \frac{1}{m}\sum_{i=1}^{m} I[y_i f(x_i) \leq 0]$

Empirical margin error: $\hat{L}_m^{\theta}(f) = \frac{1}{m}\sum_{i=1}^{m} I[y_i f(x_i) \leq \theta]$

Empirical classifier: $\hat{f}_m$

Main issue: Bound $L(\hat{f}_m)$ in terms of $\hat{L}_m^{\theta}(\hat{f}_m)$

Standard VC bounds: Use $\hat{L}_m(\hat{f}_m)$

Margin bounds: Use $\hat{L}_m^{\theta}(\hat{f}_m)$

VC DIMENSION

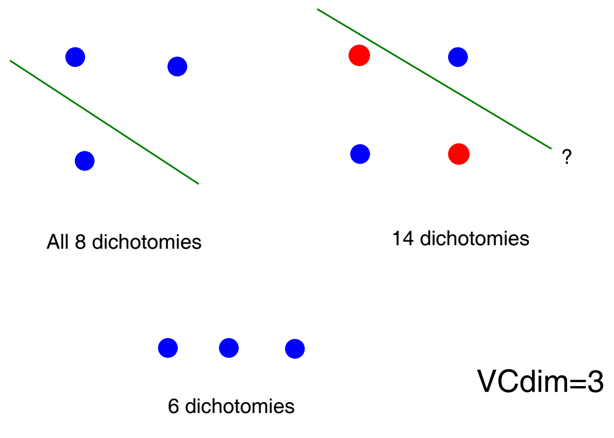
Given: $\mathcal{F} = \{f : \mathbb{R}^d \mapsto \{-1, +1\}\}$

Question: How complex is the class?

§

Shattering: $\mathcal{F}$ shatters a set X if $\mathcal{F}$ achieves all dichotomies on X

VC-dimension The size of the largest shattered subset of X



VC Bounds

For any $f \in \mathcal{F}$, with probability larger than $1 - \delta$

$$|L(f) - \hat{L}_m(f)| \leq c_1 \sqrt{\frac{\mathrm{VCdim}(\mathcal{H})}{m}} + c_2 \sqrt{\frac{\log \frac{1}{\delta}}{m}}$$

Confidence interval: Standard statistical interpretation

Let

$$\hat{f}_m = \operatorname*{argmin}_{f \in \mathcal{F}} \hat{L}_m(f)$$

Then

$$\mathbf{E} L(\hat{f}_m) \leq \inf_{f \in \mathcal{F}} L(f) + c \sqrt{\frac{\mathrm{VCdim}(\mathcal{H})}{m}}$$

(Original proof: Vapnik and Chervonenkis 1971)

Proof for Finite Hypothesis Class

Estimate: $\mathbf{Pr}\left\{\left|L(\hat{f}) - \hat{L}_m(\hat{f})\right| > \epsilon\right\}$

Assume: $\mathcal{F} = \left\{f^{(1)}, \cdots, f^{(N)}\right\}$, $N < \infty$

$$
\begin{aligned}
\mathbf{Pr}\left\{\left|L(\hat{f}_m) - \hat{L}_m(\hat{f}_m)\right| > \epsilon\right\} &\leq \mathbf{Pr}\left\{\max_{1\leq i\leq N}\left|L(f^{(i)}) - \hat{L}_m(f^{(i)})\right| > \epsilon\right\} \\
&\leq \sum_{i=1}^{N} \mathbf{Pr}\left\{\left|L(f^{(i)}) - \hat{L}_m(f^{(i)})\right| > \epsilon\right\} \\
&\leq N \max_{1\leq i\leq N} \mathbf{Pr}\left\{\left|L(f^{(i)}) - \hat{L}_m(f^{(i)})\right| > \epsilon\right\} \\
&\leq 2Ne^{-2m\epsilon^2} \qquad \text{(Hoeffding's inequality)}
\end{aligned}
$$

Proof (Cont'd)

Hoeffding inequality: Let $\{x_i\}_{i=1}^n$ be independent with $|x_i| \le B$ for all i. Then

$$\mathbf{Pr}\left\{\left|\frac{1}{m}\sum_{i=1}^m x_i - \mathbf{E}\left\{\frac{1}{m}\sum_{i=1}^m x_i\right\}\right| > \epsilon\right\} \le 2e^{-2m\epsilon^2/B^2}$$

Using previous slide

$$\mathbf{Pr}\left\{\left|L(f^{(i)}) - \hat{L}_m(f^{(i)})\right| > \epsilon\right\} \le 2Ne^{-2m\epsilon^2}$$

Set r.h.s. to δ, obtaining for all $f \in \mathcal{F}$

$$\left|L(f) - \hat{L}_m(f)\right| \le \sqrt{\frac{\log(2N/\delta)}{2m}}$$

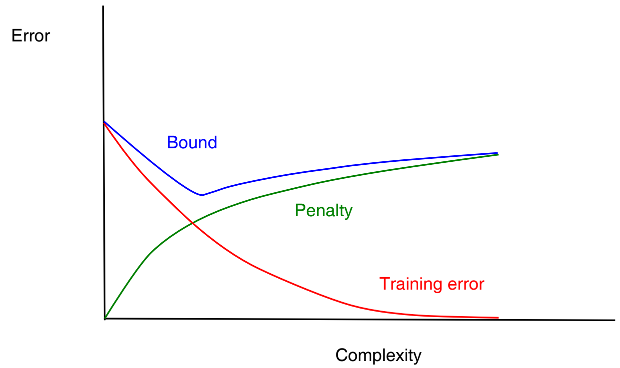
with probability larger than $1-\delta$.

Interpretation of VC Bounds

Recall

$$L(f) \leq \hat{L}_m(f) + c_1 \sqrt{\frac{\mathrm{VCdim}(\mathcal{H})}{m}} + c_2 \sqrt{\frac{\log \frac{1}{\delta}}{m}}$$

Tightness:	Bound becomes tight for increasing sample size
Empirical Error:	Decreases with complexity of $\mathcal{H}$
Confidence term:	Increases with complexity classes $\mathcal{H}$
Overfitting:	Occurs when $\mathrm{VCdim}(\mathcal{H})$ becomes very large



VC Bounds for Convex Combination

Recall

$$f_T(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$$

Let

$$\mathrm{co}_T(\mathcal{H}) = \left\{ f : \; f(x) = \sum_{t=1}^{T} \alpha_t h_t(x), \;\; \alpha_i \geq 0, \;\; \sum_{t=1}^{T} \alpha_t = 1 \right\}$$

For any $f \in \mathrm{co}_T(\mathcal{H})$, with probability $1 - \delta$

$$L(f) \leq \hat{L}_m(f) + c_1 \sqrt{\frac{\mathrm{VCdim}(\mathrm{co}_T(\mathcal{H}))}{m}} + c_2 \sqrt{\frac{\log \frac{1}{\delta}}{m}}$$

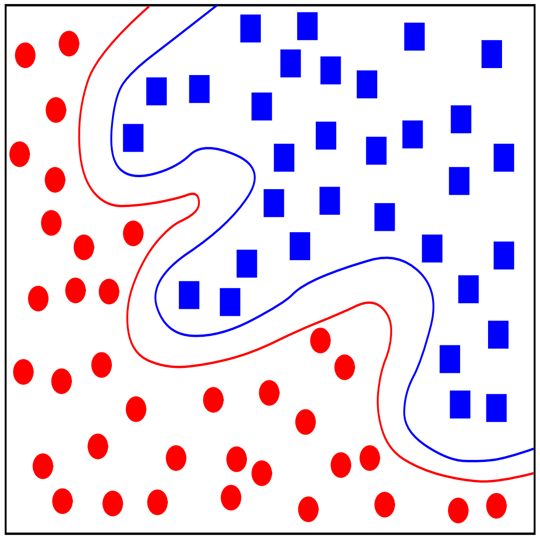
VC Bounds

Problem with VC Bound: $\mathrm{VCdim}(\mathrm{co}_T(\mathcal{H}))$ may be large, often

$$\mathrm{VCdim}(\mathrm{co}_T(\mathcal{H}) \approx T\ \mathrm{VCdim}(\mathcal{H})$$

Luckiness: Does not take into account possible luckiness

$\hat{L}_m^\theta(f)$ may be small



Margin Based Bounds

For any $f \in \mathrm{co}_T(\mathcal{H})$, with probability $1-\delta$

$$L(f) \leq \hat{L}_m^{\theta}(f) + \frac{c_1}{\theta}\sqrt{\frac{\mathrm{VCdim}(\mathcal{H})}{m}} + c_2\sqrt{\frac{\log\frac{1}{\delta}}{m}}$$

Observe:

- ⋆ No dependence on T - is overfitting absent?
- ⋆ Luckiness incorporated through $\hat{L}_m^{\theta}(f)$
- ⋆ Limit $\theta \to 0$ diverges
- ⋆ Much better that VC bounds (if lucky)
- ⋆ Overfitting absent (if lucky) - no T-dependence

Luckiness Tradeoff

$$L(f) \leq \hat{L}_m^{\theta}(f) + \frac{c_1}{\theta}\sqrt{\frac{\mathrm{VCdim}(\mathcal{H})}{m}} + c_2\sqrt{\frac{\log\frac{1}{\delta}}{m}}$$

Large θ: $\hat{L}_m^{\theta}(f)$ small (if lucky)

$\frac{1}{\theta}\sqrt{\frac{\mathrm{VCdim}(\mathcal{H})}{m}}$ small

Small θ: $\hat{L}_m^{\theta}(f) \approx \hat{L}_m(f)$

$\frac{1}{\theta}\sqrt{\frac{\mathrm{VCdim}(\mathcal{H})}{m}}$ large

Optimum: If large θ can be achieved with small error

Basic Idea of the Proof

Source: Original proof Schapire *et al.* (98), outline from Kégl *et al.* (01)

Double sample: Generate an independent fictitious sample $\{x'_1, \ldots, x'_m\}$, set $L'(f) = (1/m)\sum_{i=1}^{m} I[y'_i f(x'_i) \le \theta]$

Transform problem:

$$\mathbf{E} \max_{f \in \mathcal{F}} \left(L(f) - \hat{L}_m^{\theta}(f) \right) \le \mathbf{E} \max_{f \in \mathcal{F}} \left(L'_m(f) - \hat{L}_m^{\theta}(f) \right)$$

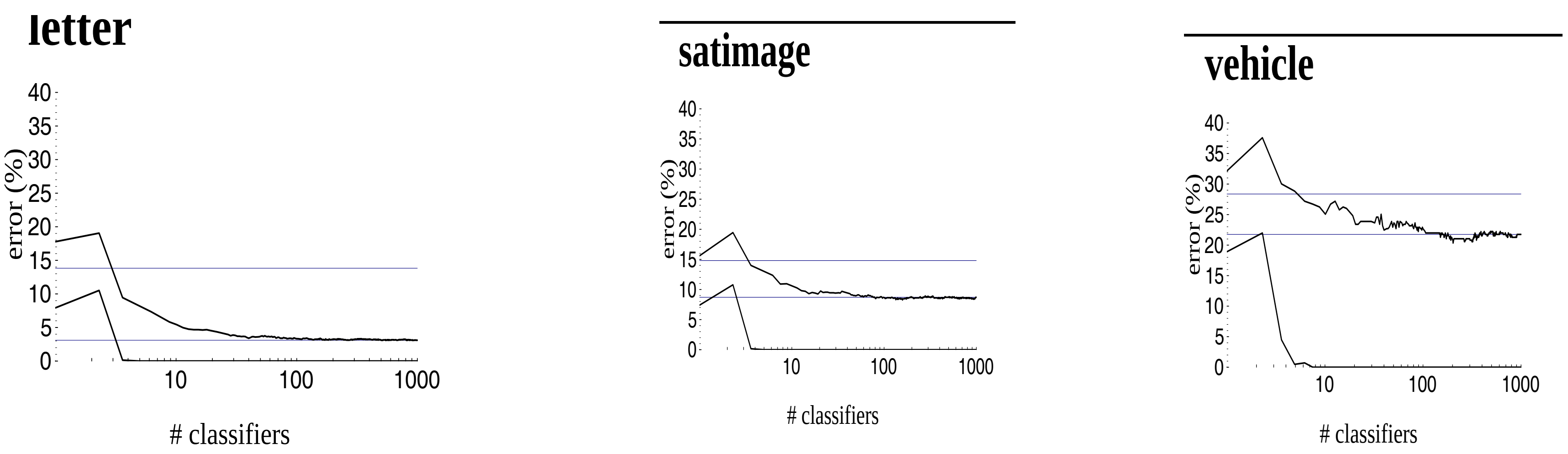
Symmetrize problem: Use freedom in θ

$$\mathbf{E} \max_{f \in \mathcal{F}} \left(L'_m(f) - \hat{L}_m^{\theta}(f) \right) \approx \mathbf{E} \max_{f \in \mathcal{F}} \left(L_m'^{\theta/2}(f) - \hat{L}_m^{\theta/2}(f) \right)$$

Maxima of linear programs: Occur at an extreme point

Experiments - Is the theory Corroborated?

Weak learner: C4.5 decision tree



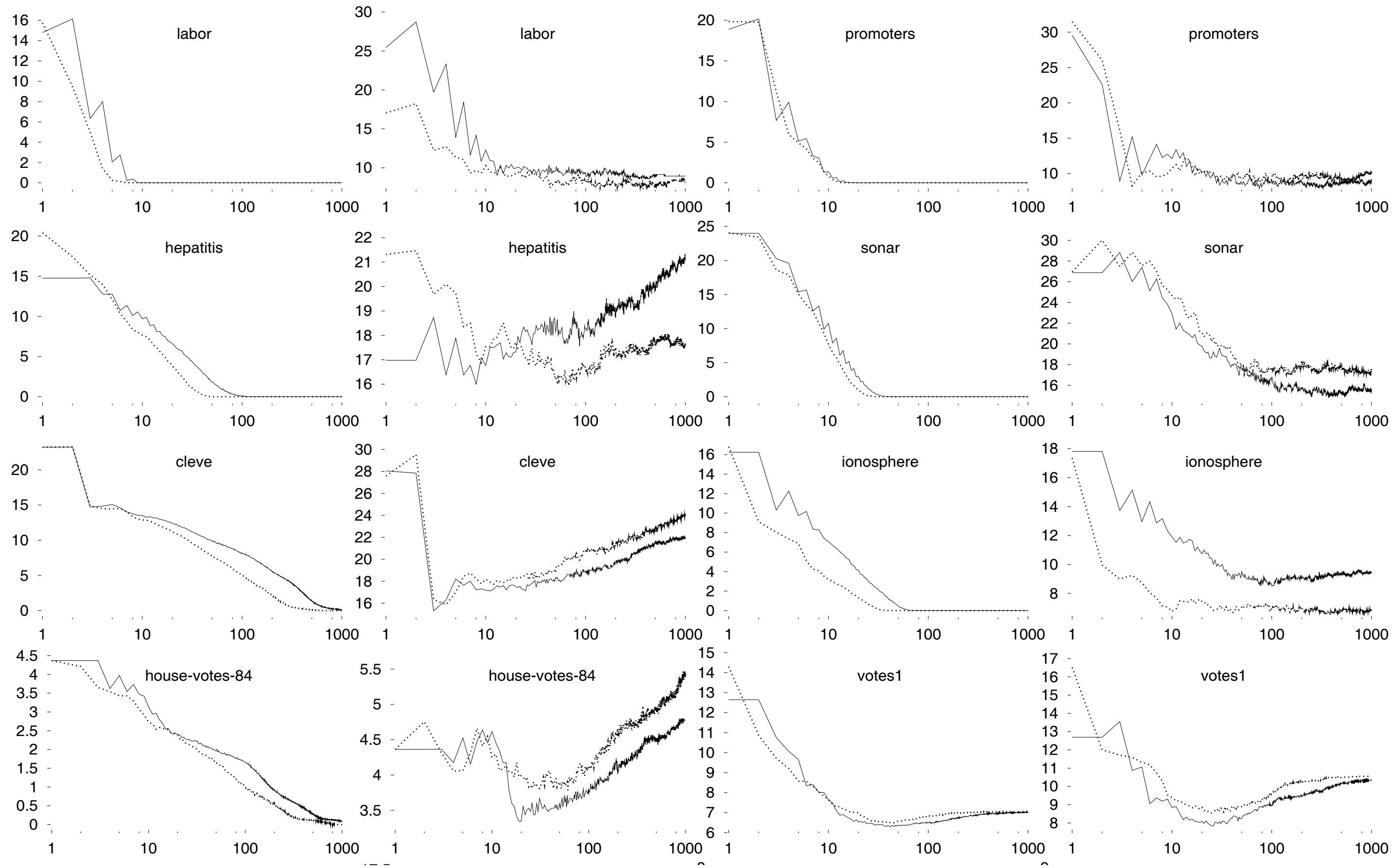
Training Error: Converges to zero

Test Error: Asymptotes - no overfitting observed

Continues to decrease after training error vanishes

Source: Schapire, Freund, Bartlett and Lee 1998

Overfitting Observed



Source: Schapire and Singer 1999

Regularizing Boosting

Observe: If weak learner attains large advantage, can 'boost forever'

Question: What happens in noisy situations?

Regularization: Need to control complexity

Do not insist on driving training error to zero

Early stopping

Do not allow weights on examples to become very different

Restrict the optimization process

Soft Margins

Basic idea: (Rätsch *et al.* 2000)

Introduce slack variables as in SVM,

$$y_i \sum_{t=1} \alpha_t h_t(x_i) \geq \rho \qquad \Longrightarrow \qquad y_i \sum_{t=1} \alpha_t h_t(x_i) \geq \rho - C\zeta_i$$

Soft margin:

$$\tilde{\rho}_i = y_i \sum_{t=1} \alpha_t h_t(x) + C\zeta_i$$

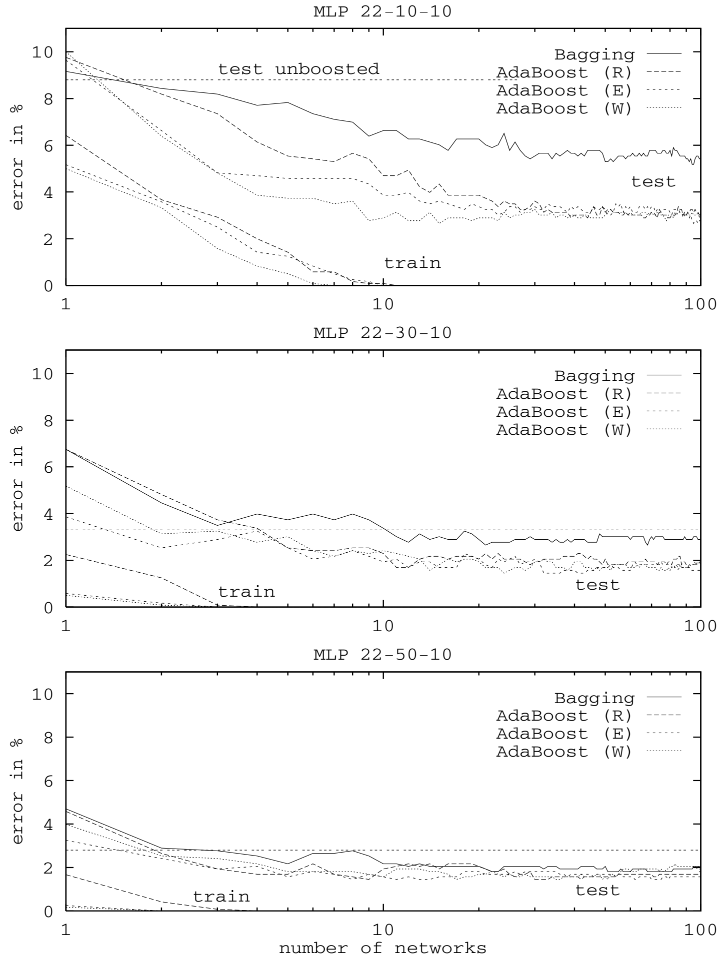
New algorithm: Maximize the margin subject to constraint on the number of errors

Trade-off determined by value of C

Early Stopping

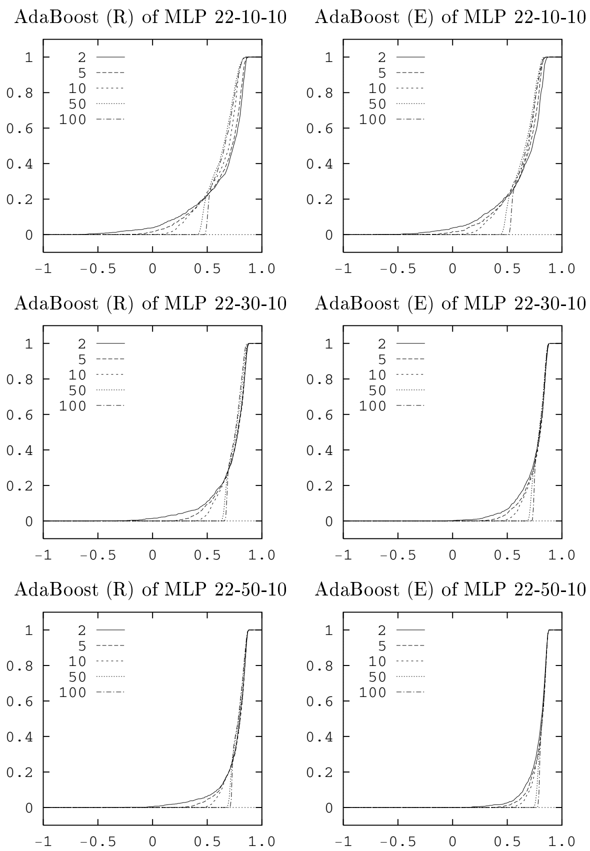
Idea:	Run the Boosting algorithm for T steps
Stopping time:	Upper bound on the true error
	Cross-validation
Problem with bound:	Bound discussed independent of T!
	Need more refined bounds
Another idea:	Constrained optimization (discuss in Section 8)

Experiments - Boosting Neural Networks



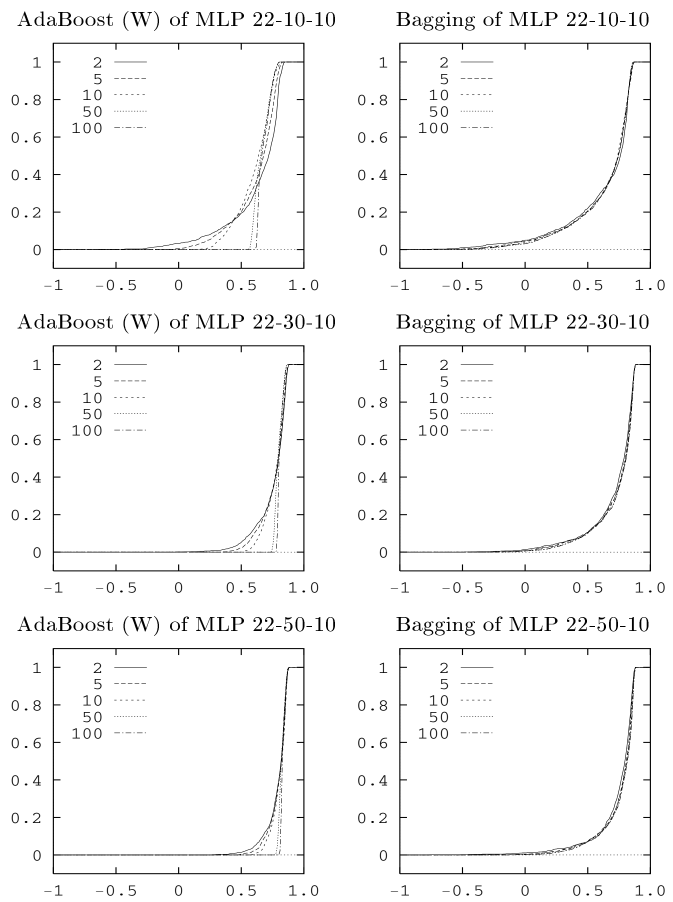
Source: Schwenk and Bengio

Margins - Neural Networks



Source: Schwenk and Bengio

Margins - Boosting vs. Bagging



Source: Schwenk and Bengio 2000

On the Existence of Weak Learners

- Main source: Mannor and Meir (2001, 2002)

Weak Learner

Weighted Error: Let $h : \mathcal{X} \mapsto \{-1, +1\}$,

$$\epsilon(P, h) = \sum_{i=1}^{m} P(i) I[h(x_i) \neq y_i]$$

Weak Learner: Boosting will 'work' if for any P,

$$\epsilon(P, h) \leq \frac{1}{2} - \gamma, \qquad \gamma \text{ sufficiently large advantage}$$

Trivial solution: $\epsilon(P_t, h_t) \leq 1/2$ is easily achievable

Weak Learners and Boolean Functions

f	Arbitrary Boolean function, $f : \{-1,+1\}^d \mapsto \{-1,+1\}$
H	Set of Boolean functions
Distribution:	$\mathcal{D}$ - distribution over $\{-1,+1\}^d$
Correlation:	$\mathrm{Corr}_{\mathcal{D}}(f,H) = \max_{h \in H} \mathbf{E}_{\mathcal{D}}[f(x)h(x)]$
	$\mathrm{Corr}(f,H) = \min_{\mathcal{D}} \mathrm{Corr}_{\mathcal{D}}(f,H)$

Claim: (Freund 1995)

$$k > (2\log 2) d \mathrm{Corr}(f,H)^{-2} \qquad \Longrightarrow \qquad f(x) = \mathrm{sgn}\left(\sum_{i=1}^{k} h_i(x)\right)$$

Conclude: Combination of weak learners can represent any f if it is correlated with H

Problem: f unknown; Proof relies heavily on Boolean nature

Basic Issues

Observe: A weak error of 1/2 is useless for learning!

Effective weak learner: A weak learner leading to 'good' generalization

Main Questions:

I. How large does γ_t need to be?

II. When does an effective weak learner exist?

Observe: It suffices to consider the case where $P^+ = P^-$, namely

$$\sum_{x_i \in X^+} P(i) = \sum_{x_i \in X^-} P(i)$$

On the Scale of the Advantage

Recall

$$L(f) \leq \hat{L}_m^{\theta}(f) + \frac{c_1}{\theta}\sqrt{\frac{\mathrm{VCdim}(\mathcal{H})}{m}} + c_2\sqrt{\frac{\log\frac{1}{\delta}}{m}}$$

and

$$\hat{L}_m^{\theta}(f_T) \leq \prod_{t=1}^{T}\sqrt{4\epsilon_t^{1-\theta}(1-\epsilon_t)^{1+\theta}} \qquad \left(\epsilon_t = \frac{1}{2} - \gamma_t\right)$$

Sufficient condition for vanishing bound:

Margin error: $L_m^{\theta}(f_T) \to 0$ if $\gamma_t \geq \theta$

Complexity: $\theta \gg 1/\sqrt{m}$

Effective weak learner: Demand that $\gamma_t \geq 1/\sqrt{m}$

$\Omega(1/m)$ Linear Weak Learner Always Exists

Assumption: Work with linear classifier (learner)

$$\boxed{h(x) = \operatorname{sgn}\left(w^\top x + w_0\right)}$$

Claim: For any set of distinct points $(x_1, y_1), \ldots, (x_m, y_m)$ and distribution P, there exists a linear classifier h such that

$$\epsilon(P, h) \leq \frac{1}{2} - \frac{1}{4m - 2}$$

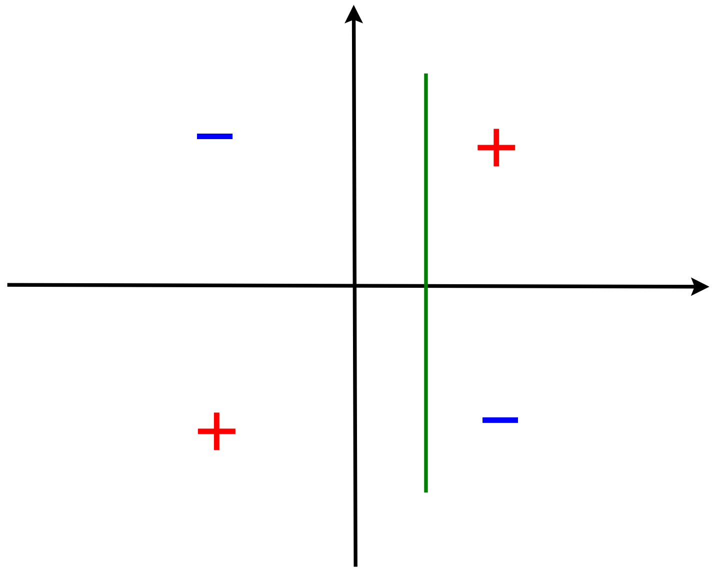
Proof: Project onto 1D and use **pigeonhole principle**

This is not good enough: Applies to arbitrarily labeled points

Stumps are not Weak Learners

Stumps: Hyper-planes parallel to axes

Fail on simple XOR configuration

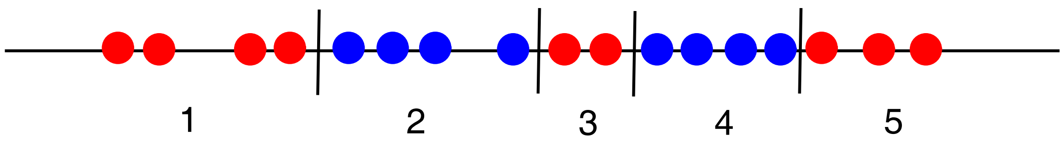


Bounds in terms of the number of regions

Motivation: Error reduction is hard if $\pm$ points are highly intermixed

Single dimension: From previous result, if there are K uniform regions, obtain error

$\epsilon \leq \frac{1}{2} - \frac{c}{K}$



Multiple dimensions: Problem becomes very hard!

New tools are required

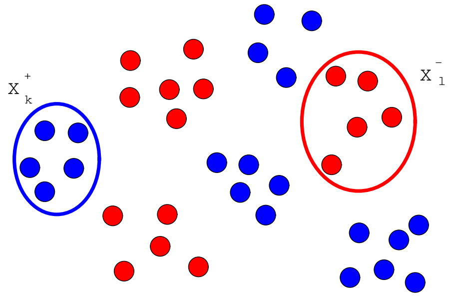
Condition for Effective Linear Learner

Let

$$\inf_h \epsilon(P, h) = \frac{1}{2} - \gamma^*$$

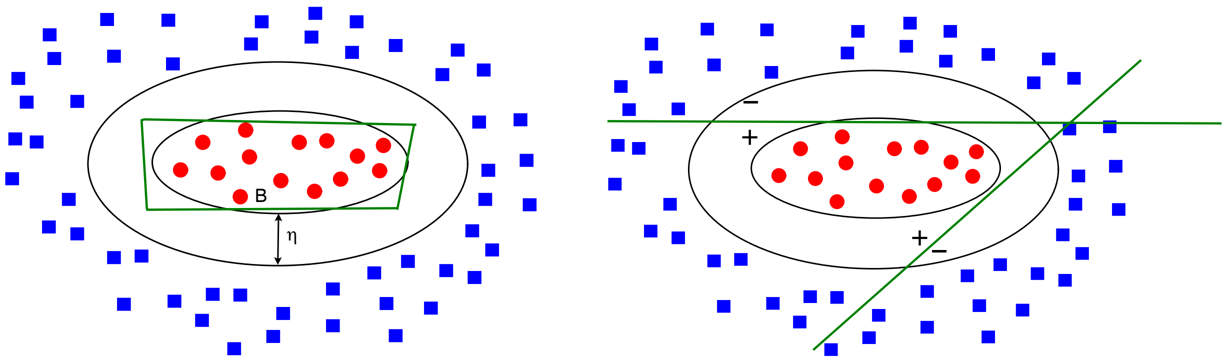
Question: When is $\gamma^* \gg \frac{1}{m}$? (Ideally, $\gamma^* > c \geq 0$)

Will show: Occurs when data 'approximately clusters'



Generalization: Under such conditions boosting generalizes well

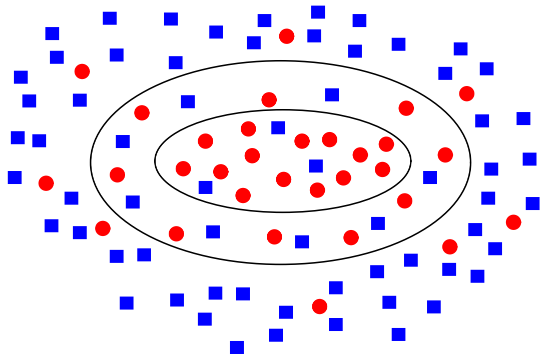
Simple Example



Claim: $\inf_h \epsilon(P,h) \leq \frac{1}{2} - \frac{c}{K}$ (K faces)

- ⋆ Define K classifiers $h_1, \ldots, h_K$ as in the right figure
- ⋆ Construct non-linear classifier $h = \min(h_1, \ldots, h_K)$
- ⋆ h yields zero error
- ⋆ There must exist a classifier h_i with error smaller than $1/2 - 1/2K$

A More Complex Example



Will need more advanced tools

Geometric Discrepancy - hyperplanes

$$S = (x_1, y_1), \ldots, (x_m, y_m) \qquad x_i \in \mathbb{R}^d, \; y_i \in \{\pm 1\}$$

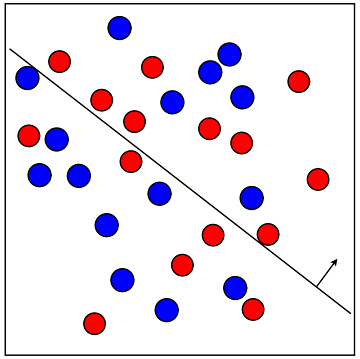
$X^{\pm}$ = subset of points for which $y = \pm 1$

$$\mathrm{disc}(S, H) \triangleq \frac{1}{m} \left| |X^+ \cap H| - |X^- \cap H| \right|$$

Objective: Find halfspace H such that

$\mathrm{disc}(S, H)$ is maximal

$\epsilon(U, h) = \frac{1}{2} - \mathrm{disc}(S, H)$



Geometric Discrepancy - Lower Bounds

Positive results:

Alexander (90,91): For any set of m points there exists a halfspace H for which

$$\boxed{\mathrm{disc}(S,H) \geq \frac{c(d)(\delta/L)^{1/2}}{\sqrt{m}}}$$

δ - minimal distance between points

L - maximal distance between points

Well-separated points:

$$(L/\delta) \leq cm^{1/d} \quad \Rightarrow \quad \mathrm{disc}(S,H) \geq \frac{c'(d)}{m^{1/2+1/2d}}$$

Geometric Discrepancy - Upper Bounds

Negative results:

Objective: Find the most difficult set and associated coloring

Matoušek (95): There **exists** a set of colored points such that for any halfspace H

$$\boxed{\mathrm{disc}(S, H) \leq \frac{c'(d)}{m^{1/2+1/2d}}}$$

Conclusion: Precludes general-purpose effective weak learners

Regularity: Some structural assumptions are essential to achieve effective weak learning

Geometric Characterization

Recall

$$\epsilon(P, h) = \sum_{i=1}^{m} P(i) I[h(x_i) \neq y_i]$$
$$= \frac{1}{2} - \gamma(P, h)$$

Using Discrepancy Theory

$$\boxed{\sup_h \{\gamma(P, h)\} \geq \sqrt{-(C_d/L)I(P)}}$$

where

$$I(P) = \sum_{i \neq j} \sum \|x_i - x_j\| y_i y_j P_i P_j$$

is a purely geometric quantity

Interpretation

$$I(P) = \sum\sum_{i \neq j} \|x_i - x_j\| y_i y_j P_i P_j$$

Split sum into equally labeled and oppositely labeled pairs

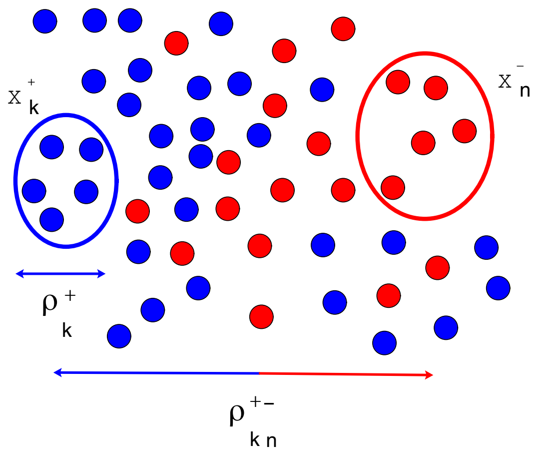
$$-I(P) = \sum\sum_{\{y_i \neq y_j\}} \|x_i - x_j\| P_i P_j - \sum\sum_{\{y_i = y_j\}} \|x_i - x_j\| P_i P_j$$

This is large if:

oppositely labeled points	far apart
equally labeled points	closely clustered

INTUITION

Objective: Obtain an error bound with a large advantage



Intuition: Bulk of blue points shifted from bulk of red points

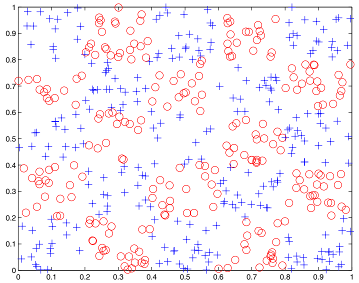
Proof: Quantify notion of 'shifted'

Advantage Bound I

- Split points into $K^{\pm}$ homogeneous regions
- Define appropriate 'size measure' for each region

  ρ^{++} average separation between 'positive' clusters

  ρ^{+-} average separation between 'positive-negative' clusters
- **Clustering measure:** $\Delta^{\pm}$
- **Characterization:** If $\rho^{+-} \geq \rho^{++} + \rho^{--}$ then γ is large

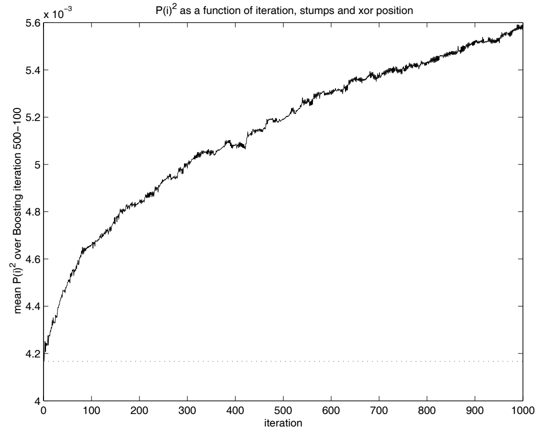


Advantage Bound II

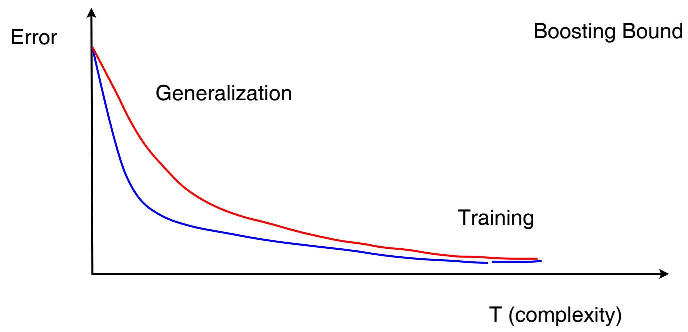
$$\sup_{h}\{\gamma(P,h)\} \geq \frac{\sqrt{C_d/4L}}{\sqrt{K^+/\Delta^+ + K^-/\Delta^-}} + \left(\frac{C_d}{4L}\sum_{i=1}^{m} P_i^2\right)^{\frac{1}{2}}$$

First term: Large (ind. of m) under favorable conditions

Second term: Large if distribution is skewed: $\sum_{i=1}^{m} P_i^2$ large



Stopping Criteria for Boosting I



Question: How does the number of boosting iterations depend on the problem?

Relevant parameters: Sample size

Geometry and coloring of points

Skewness of boosted distribution (path-dependent)

Stopping Criterion for Boosting II

Question: How many boosting iterations are required to reduce generalization error to ϵ?

Answer: Require that

$$\hat{L}_m^\theta(f_T) + \frac{1}{\theta}\sqrt{\frac{\mathrm{VCdim}(\mathcal{H})}{m}} \leq \epsilon$$

If

$$m \geq \frac{4\mathrm{VCdim}(\mathcal{H})}{\epsilon^2\theta^2},$$

it suffices that

$$\boxed{T \geq \left(\frac{K^+}{\Delta^+} + \frac{K^-}{\Delta^-}\right)\log\frac{2}{\epsilon}}$$

Summary

- Linear weak learners are can drive training error to zero if points are 'well-separated'
- Can show, that if a gap exists between positive/negative points, generalization error converges to zero
- Stopping criterion under favorable conditions
- Results apply to any weak learner based on linear classifiers (neural networks, decision trees with oblique splits)
- **Main drawback:** does not allow overlapping distributions

Applications of Boosting

Main sources:

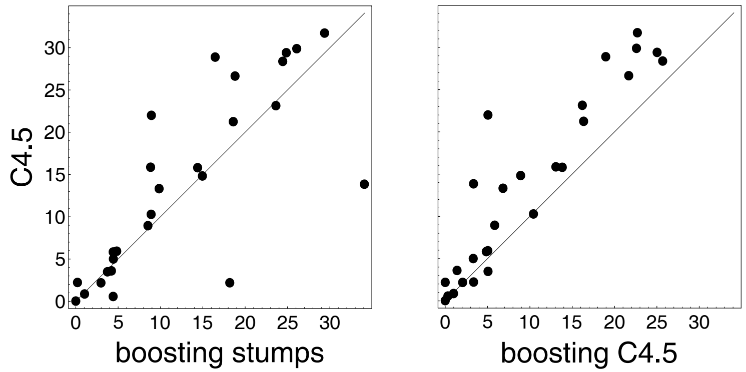
- Dietterich 2000
- Rätsch *et al.* 2001
- Schapire 2002 and references therein

Practical Issues

Advantages:	Difficulties:
A general meta-algorithm - use any 'reasonable' weak learner	Not clear how to incorporate prior knowledge effectively
Single parameter to be tuned (# iterations) - in principle	Regularization often essential - best strategy unclear
Fast and easy to program	The best choice of weak learner is not obvious
Theoretical performance guarantees	Decision boundaries generated using parallel-split based methods often very rugged

Performance on UCI Benchmarks

Weak Learners: Stumps and C4.5 decision trees



Source: Freund and Schapire 1996

Performance on Benchmarks

Weak Learners: Radial basis functions, regularization used

	RBF	AB	AB$_R$	LP$_R$-AB	QP$_R$-AB	SVM
Banana	*10.8±0.6*	12.3±0.7	10.9±0.4	**10.7±0.4**	10.9±0.5	11.5±0.6
B.Cancer	27.6±4.7	30.4±4.7	26.5±5.5	26.8±6.1	**25.9±4.6**	*26.0±4.7*
Diabetes	24.1±1.9	26.5±2.3	*23.9±1.6*	24.1±1.9	25.4±2.2	**23.5±1.7**
German	24.7±2.4	27.5±2.5	*24.3±2.1*	24.8±2.2	25.2±2.1	**23.6±2.1**
Heart	17.1±3.3	20.3±3.4	16.6±3.7	**14.5±3.5**	17.2±3.4	*16.0±3.3*
Image	3.3±0.6	**2.7±0.7**	**2.7±0.6**	*2.8±0.6*	**2.7±0.6**	3.0±0.6
Ringnorm	*1.7±0.2*	1.9±0.3	**1.6±0.1**	2.2±0.5	1.9±0.2	*1.7±0.1*
F.Sonar	34.4±2.0	35.7±1.8	*34.2±2.2*	34.8±2.1	36.2±1.8	**32.4±1.8**
Splice	*9.9±1.0*	10.3±0.6	**9.5±0.7**	*9.9±1.4*	10.3±0.6	10.8±0.6
Thyroid	*4.5±2.1*	**4.4±2.2**	**4.4±2.1**	4.6±2.2	**4.4±2.2**	4.8±2.2
Titanic	23.3±1.3	*22.6±1.2*	*22.6±1.2*	24.0±4.4	22.7±1.1	**22.4±1.0**
Twonorm	*2.9±0.3*	3.0±0.3	**2.7±0.2**	3.2±0.4	3.0±0.3	3.0±0.2
Waveform	10.6±1.0	10.8±0.6	**9.8±0.8**	10.5±1.0	10.1±0.5	*9.9±0.4*
Mean %	6.6±5.8	11.9±7.9	1.7±1.9	8.9±10.8	5.8±5.5	4.6±5.4
Winner %	14.8±8.5	7.2±7.8	26.0±12.4	14.4±8.6	13.2±7.6	23.5±18.0

⋆ Results highly competitive with state-of-the-art SVM classifier

Source: Rätsch *et al.* 2000

Effect of Noise

Noise = 0%	C4.5	Adaboost C4.5	Bagged C4.5
Random C4.5	5 – 0 – 4	1 – 6 – 2	3 – 3 – 3
Bagged C4.5	4 – 0 – 5	0 – 5 – 4	
Adaboost C4.5	6 – 0 – 3		

Noise = 5%	C4.5	Adaboost C4.5	Bagged C4.5
Random C4.5	5 – 2 – 2	3 – 2 – 4	1 – 5 – 3
Bagged C4.5	6 – 0 – 3	5 – 1 – 3	
Adaboost C4.5	3 – 3 – 3		

Noise = 10%	C4.5	Adaboost C4.5	Bagged C4.5
Random C4.5	4 – 1 – 4	5 – 1 – 3	1 – 6 – 2
Bagged C4.5	5 – 0 – 4	6 – 1 – 2	
Adaboost C4.5	2 – 3 – 4		

Noise = 20%	C4.5	Adaboost C4.5	Bagged C4.5
Random C4.5	5 – 2 – 2	5 – 0 – 4	0 – 2 – 7
Bagged C4.5	7 – 0 – 2	6 – 0 – 3	
Adaboost C4.5	3 – 6 – 0		

Boosting (without regularization) inferior to Bagging for high noise

Source: Dietterich 2000

Other Applications

Some examples:

Text classification	Schapire and Singer - Used stumps with normalized term frequency and multi-class encoding
OCR	Scwenk and Bengio - used neural networks
Natural language Processing	Collins; Haruno, Shirai and Ooyama
Image retrieval	Thieu and Viola
Medical diagnosis	Merle *et al.*

Fuller list: Schapire's 2002 review

Greedy Algorithms

Main sources:

- Friedman 2001, Friedman, Hastie and Tibshirani 2000
- Mason, Bartlett, Baxter and Frean 2000
- Schapire and Singer 1999
- Tong 2002

Greedy Algorithms - Background

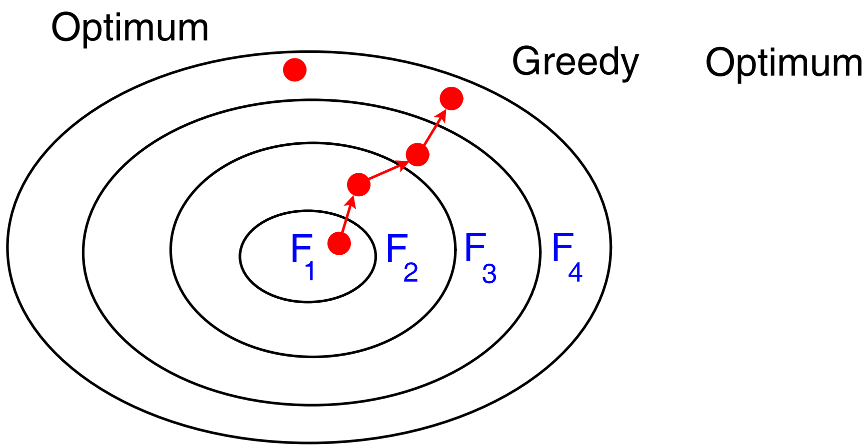
Objective:

$$\min_{f \in \mathcal{F}} f(\mathbf{x}) \qquad \mathcal{F} \text{ `very large'}$$

Solution: Split into sequence of 'easy' sub-problems

Solve easy sub-problem

Use solution as starting point for more complex problem

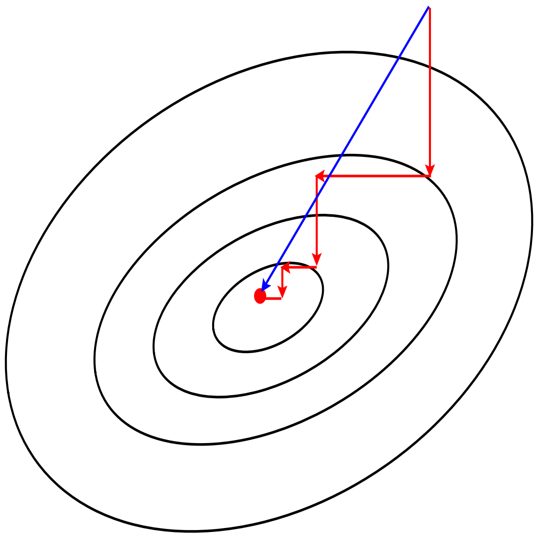


Greedy Coordinate Descent I

Objective: Minimize $f(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^d$

Problem: Derivatives hard to compute

Solution: Greedy coordinate descent - iteratively choose direction of maximal decrease



Greedy Coordinate Descent II

- Select $\mathbf{x}_0$; Set $t = 0$
- While Stopping Condition not obeyed
  - Compute best axis direction

$$f_i^* = \min_{\alpha} f(\mathbf{x}_t + \alpha \mathbf{e}_i) \qquad ; \qquad i^* = \operatorname*{argmin}_{1 \leq i \leq d} f_i^*$$

  - Set $\mathbf{x}_{t+1} = \mathbf{x}_t + \alpha^* \mathbf{e}_{i^*} \quad ; \quad t \leftarrow t + 1$
- Stopping condition: $t > T$ or Error tolerance achieved

Greedy Algorithms for Classification

Objective: Greedily construct a complex classifier

$$f_T(x) = \sum_{t=1}^{T} \alpha_t h_t(x) \qquad h_i \in \mathcal{H} \qquad ; \qquad \hat{L}_m(f) = \frac{1}{m}\sum_{i=1}^{m} I[y_i f(x_i) \leq \theta]$$

Based on 'base' classifiers $h \in \mathcal{H}$

1. Choose $f_0 = \mathrm{argmin}_{h \in \mathcal{H}} \hat{L}_m(h)$
2. For $t = 1, 2, \ldots, T$

$$h_t = \underset{h \in \mathcal{H}}{\mathrm{argmin}}\, \hat{L}_m(f_{t-1} + h)$$

$$\alpha_t = \underset{\alpha}{\mathrm{argmin}}\, \hat{L}_m(f_{t-1} + \alpha h_t)$$

$$f_t = f_{t-1} + \alpha_t h_t$$

Problems with Greedy Classification

Computation: Minimization ay be intractable even for simple base learners

NP-hard even for linear classifier

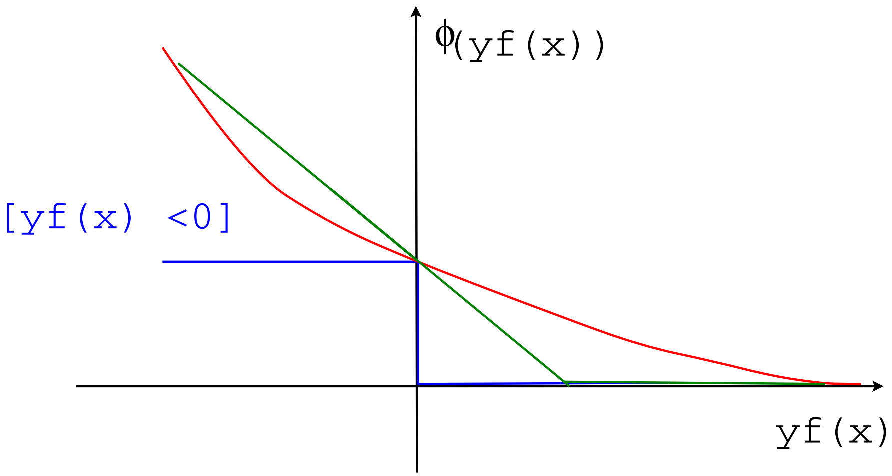
Overfitting: Minimizing $0-1$ loss may lead to overfitting

Remedy: Construct a convex function which bounds the $0-1$ loss, and minimize it greedily

Loss Functions for Boosting

Require

$$\phi(yf(x)) \geq I[yf(x) \leq 0]$$



Let

$$\hat{A}(f) = \frac{1}{m}\sum_{i=1}^{m}\phi(y_i f(x_i))$$

Repeat greedy procedure with convex loss

Greedy Classification Based on Convex Loss

$$\hat{A}(f) = \frac{1}{m}\sum_{i=1}^{m} \phi(y_i f(x_i))$$

1. Choose $f_0 = \mathrm{argmin}_{h \in \mathcal{H}} \hat{A}(h)$
2. For $t = 1, 2, \ldots, T$

$$h_t = \operatorname*{argmin}_{h \in \mathcal{H}} \hat{A}(f_{t-1} + h)$$

$$\alpha_t = \operatorname*{argmin}_{\alpha} \hat{A}(f_{t-1} + \alpha h_t)$$

$$f_t = f_{t-1} + \alpha_t h_t$$

Approximately Going Down the Gradient

Question: How do we minimize $\hat{A}(f + h)$?

Functional gradient: Set $\mathbf{g} = (g(x_1), \ldots, g(x_m))$,

$$\nabla \hat{A}(f) = \left(\frac{\partial \hat{A}(f(x_1) + g(x_1))}{\partial g(x_1)}, \ldots, \frac{\partial \hat{A}(f(x_1) + g(x_m))}{\partial g(x_m)} \right)_{\mathbf{g}=\mathbf{0}}$$

$$= \frac{1}{m} \left(\phi'(y_1 f(x_1)) y_1, \ldots, \phi'(f(x_m)) y_m \right)$$

Optimality of gradient: Negative gradient is optimal direction for small step sizes

$$\hat{A}(f + \epsilon h(\approx \hat{A}(f) + \epsilon \langle \nabla \hat{A}(f), g \rangle$$

Restriction: Must choose g from $\mathcal{H}$

Approximately Going Down the Gradient

Compromise: Choose $h \in \mathcal{H}$ which maximizes

$$\langle -\nabla \hat{A}(f), h \rangle = \frac{1}{m^2} \sum_{i=1}^{m} y_i h(x_i) \phi'(y_i f(x_i))$$

Set

$$P(i) = \frac{\phi'(y_i f(x_i))}{\sum_{j=1}^{m} \phi'(y_j f(x_j))}$$

Assume: $\phi'(yf(x))$ is positive

Objective is

$$\max_{h \in \mathcal{H}} \left\{ \sum_{i=1}^{m} P(i) y_i h(x_i) \right\}$$

Approximately Going Down the Gradient

Assume: h binary

$$
\begin{aligned}
h &= \operatorname*{argmax}_{h\in\mathcal{H}} \left\{ \sum_{i=1}^{m} P(i) y_i h(x_i) \right\} \\
&= \operatorname*{argmax}_{h\in\mathcal{H}} \left\{ \sum_{i:y_i = h(x_i)} P(i) - \sum_{i:y_i \neq h(x_i)} P(i) \right\} \\
&= \operatorname*{argmin}_{h\in\mathcal{H}} \left\{ \sum_{i:y_i \neq h(x_i)} P(i) \right\} \\
&= \operatorname*{argmin}_{h\in\mathcal{H}} \left\{ \sum_{i=1}^{m} P(i) I[y_i \neq h(x_i)] \right\} \\
&= \operatorname*{argmin}_{h\in\mathcal{H}} \{\text{Weighted error}\} \qquad (\text{as in AdaBoost})
\end{aligned}
$$

Selecting the Step Size

Fix h and

$$\min_{\alpha} \left\{ \sum_{i=1}^{m} \phi(y_i f(x_i) + \alpha h(x_i)) \right\}$$

Assume

- Exponential loss $\phi(yf(x)) = e^{-yf(x)}$
- Binary hypotheses

Obtain

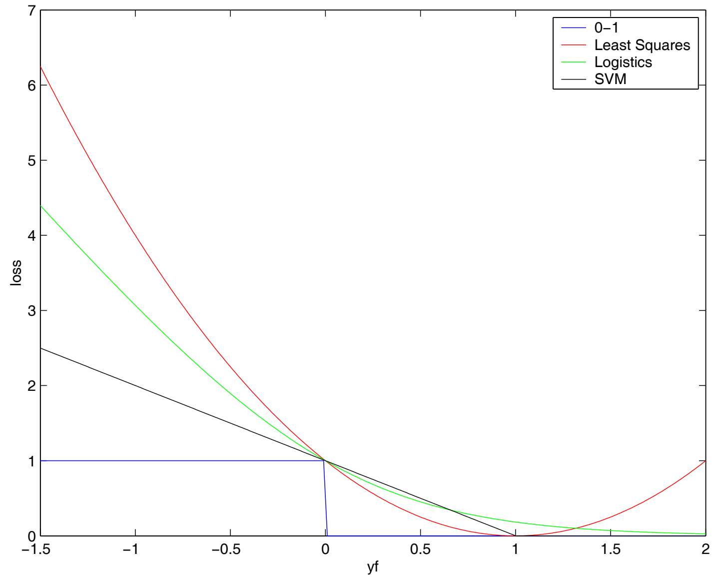
$$\alpha = \frac{1}{2} \log\left(\frac{1-\epsilon}{\epsilon}\right) \qquad \left(\epsilon = \sum_{i=1}^{m} P(i) I[y_i \neq h(x_i)]\right)$$

Conclude: Greedy based optimization with binary hypotheses and exponential loss reproduces AdaBoost

Other Cost Functions

Possibilities for $\phi(u)$:

Least Squares	$(1-u)^2$
SVM	$\max(1-u, 0)$
Logistic	$\log_2(1+\exp(-u))$



Greedy Algorithms for L_2 Approximation

Objective: Greedily locate function in $\mathrm{co}(\mathcal{H})$ which minimizes $\|f-g\|_2$ for any $g \in \mathrm{co}(\mathcal{H})$. Require bound on

$$\|f-g\|_2 \qquad f \in \mathrm{co}_t(\mathcal{H}) \text{ obtained greedily}$$

Greedy procedure: Loop over $\tau = 1, 2, \ldots$

$$h_\tau = \operatorname*{argmin}_{h \in \mathcal{H}} \|(1-\alpha) f_{\tau-1} + \alpha h\|_2$$

$$\alpha_\tau = \operatorname*{argmin}_{0 \le \alpha \le 1} \|(1-\alpha) f_{\tau-1} + \alpha h_\tau\|_2$$

$$f_\tau = (1-\alpha) f_{\tau-1} + \alpha h_\tau$$

Convergence rate: Barron (1993)

$$\boxed{\|f_t - g\|_2^2 \le \frac{c}{t}}$$

Greedy Algorithms for Convex Loss Functions

Objective: Greedily minimize $\hat{A}(f)$, $f \in \mathrm{co}(\mathcal{H})$

Input: A sample D_m; a stopping time t; a constant β

Algorithm:

1. Set $\hat{f}^1_{\beta,m} = \mathrm{argmin}_{h \in \mathcal{H}} \hat{A}(h)$

2. For $t = 2, 3, \ldots, T$

$$\hat{h}_t, \hat{\alpha}_t = \underset{h \in \mathcal{H}, 0 \le \alpha \le \beta}{\mathrm{argmin}} \hat{A}((\beta - \alpha)\hat{f}^{t-1}_{\beta,m} + \alpha h)$$

$$\hat{f}^t_{\beta,m} = (\beta - \hat{\alpha}_t)\hat{f}^{t-1}_{\beta,m} + \alpha_t \hat{h}_t$$

Output: Classifier $\hat{f}^T_{\beta,m}$

Greedy Algorithms for Convex Loss Functions

Observe:

$$\hat{f}^T_{\beta,m} \in \beta\mathrm{co}_T(\mathcal{H})$$

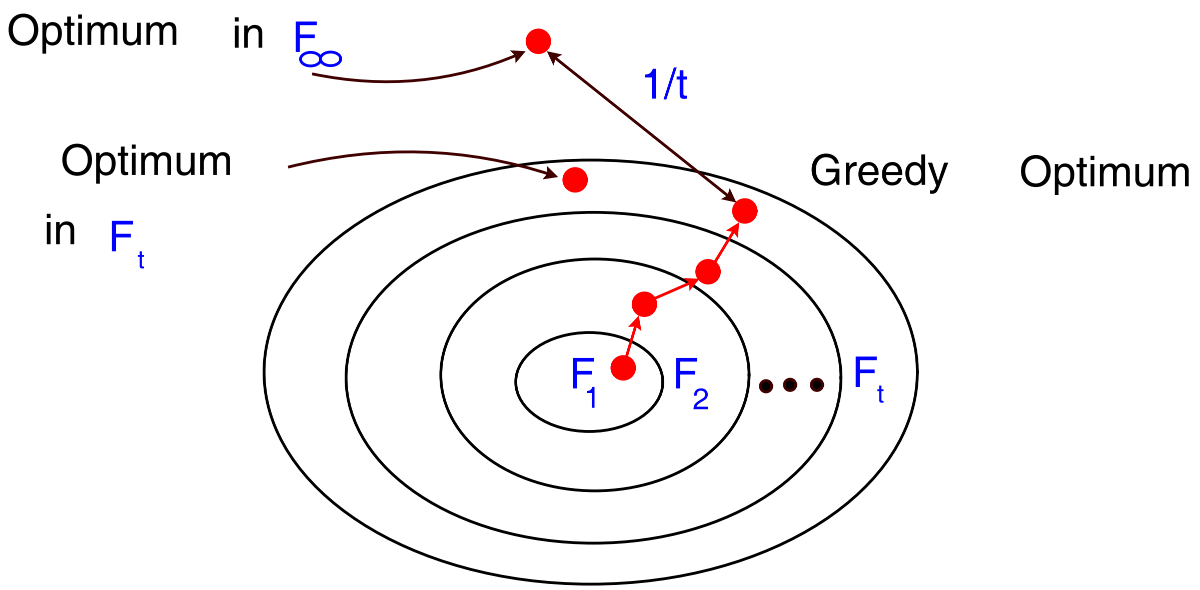
$$\beta\mathrm{co}_T(\mathcal{H}) = \left\{ f: \ f(x) = \sum_{t=1}^{T} \alpha_t h_t(x), \ \alpha_t \geq 0, \ \sum_{t=1}^{T} \alpha_t = 1, \ h_t \in \mathcal{H} \right\}$$

Convergence: (Tong 2002) Assume that ϕ is strictly convex and $\phi'' \leq M$, then

$$\boxed{\hat{A}(f^t_{\beta,m}) - \inf_{f \in \beta\mathrm{co}(\mathcal{H})} \hat{A}(f) \leq \frac{2\beta^2 M}{t}}$$

Implication: Greedy algorithm will converge to the unique global minimum of $\hat{A}(f)$ over $f \in \mathrm{co}(\mathcal{H})$

Greedy Algorithms Convergence



Greedy Minimization of the Log-Likelihood

Bernoulli Model:

$$P(y=1|x) = p(x) \triangleq \frac{1}{1+e^{-2f(x)}}$$

For $y \in \{-1, +1\}$

$$P(y|x) = p(x)^{(1+y)/2}(1-p(x))^{(1-y)/2}$$

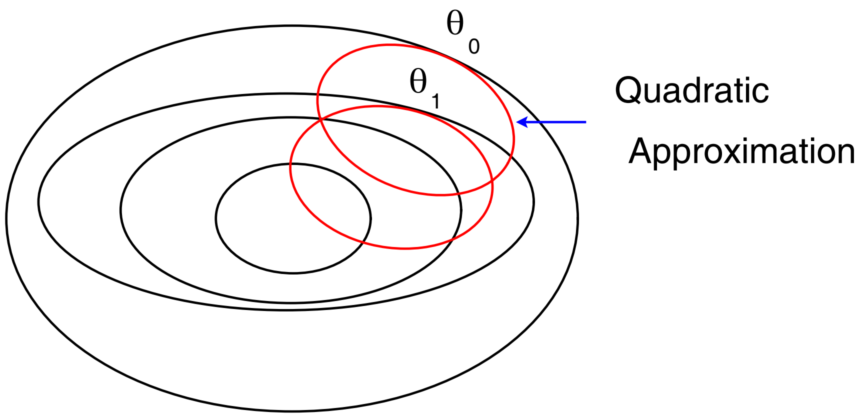
Log-Likelihood:

$$\begin{aligned}
\ell(f) &= \log \prod_{i=1}^{m} P(y_i|x_i) \\
&= \sum_{i=1}^{m} \left\{ \frac{1+y_i}{2} \log p(x_i) + \frac{1-y_i}{2} \log(1-p(x_i)) \right\} \\
&= \sum_{i=1}^{m} \left\{ (1+y_i) f(x_i) - \log\left(1+e^{2f(x_i)}\right) \right\}
\end{aligned}$$

REMINDER - NEWTON'S ALGORITHM I

Objective: Minimize a multi-variate function $f(\theta)$

Basic idea At each step minimize a quadratic approximation of f around current point and iterate



Hessian:

$$\left(\nabla^2 f(\theta)\right)_{ij} = \frac{\partial f(\theta)}{\partial\theta_i \partial\theta_j}$$

Reminder - Newton's Algorithm II

Exact solution for quadratic approximation:

$$\min_{\theta} \left\{ f(\theta_t) + \nabla f(\theta_t)^T(\theta - \theta_t) + \frac{1}{2}(\theta - \theta_t)^T \nabla^2 f(\theta_t)(\theta - \theta_t) \right\}$$

$$\theta_{t+1} = \theta_t - \left(\nabla^2 f(\theta_t)\right)^{-1} \nabla f(\theta_t)$$

Convex functions: $\nabla^2 f(\theta)$ positive-definite for any θ

Quadratic function: Converges in a single step

General functions: Very sensitive to initial conditions. Hessian may not be positive-definite.

Greedily Maximizing the Log-Likelihood

$$\ell(f+h) = \sum_{i=1}^{m} \left\{ (1+y_i)(f(x_i)+h(x_i)) - \log\left(1+e^{2(f(x_i)+h(x_i))}\right) \right\}$$

First and second order derivatives

$$s(x_i) = \left.\frac{\partial \ell(f(x_i)+h(x_i))}{\partial h(x_i)}\right|_{h(x_i)=0} = 2\left\{(1+y_i)/2 - p(x_i)\right\}$$

$$H(x_i) = \left.\frac{\partial^2 \ell(f(x_i)+h(x_i))}{\partial h(x)^2}\right|_{h(x_i)=0} = -4p(x_i)(1-p(x_i))$$

Newton algorithm: (note that $\ell(f)$ is convex in f)

$$f(x) \leftarrow f(x) - H(x)^{-1} s(x)$$

Gain: Replace line search by exact calculation

Greedily Maximizing the Log-Likelihood

LogitBoost (Hastie, Friedman and Tibshirani2000)

1. Set $w(i) = 1/m,\ i = 1, 2, \ldots, m,\ f_0(x) = 0$ and $p(x_i) = 1/2$
2. For $t = 1, 2, \ldots, T$
   - ⋆ Compute
     $$z_i = 2[(1 + y_i)/2 - p(x_i)]$$
   - ⋆ Estimate h_t using a Newton step based on f_t
   - ⋆ Update
     $$f_{t+1}(x) = f_t(x) + h_t(x) \qquad ; \qquad p_{t+1}(x) = 1/\left(1 + e^{-2f_{t+1}(x)}\right)$$
3. Output the classifier $\sum_{t=1}^{T} f_t(x)$

On the Choice of Loss Function

Observation: AdaBoost may lead to significant overfitting in noisy situations

Possible explanation: Exponentially high cost paid for misclassification

Suggested remedy: Reduce cost of misclassified examples

Least Squares: $(y - f(x))^2$

Logistic: $\log(1 + \exp(-yf(x)))$

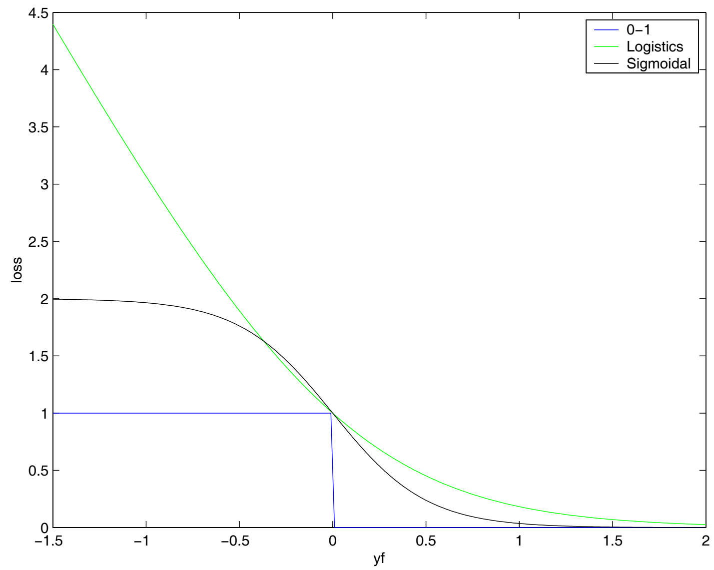
Huber:

$$\ell(y, f(x)) = \begin{cases} |y - f(x)|^2 & \text{if } |y - f(x)| \leq \delta \\ 2\delta(|y - f(x)| - \delta/2) & \text{otherwise} \end{cases}$$

Aggressively Suppressing Noise

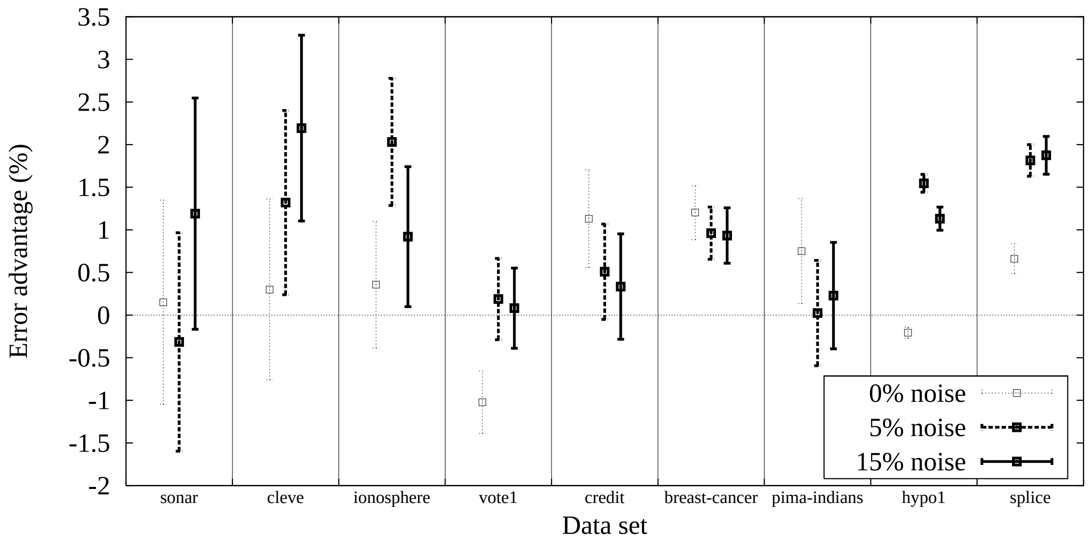
Mason *et al.* (2000) suggested

$$\ell(y, f(x)) = [1 - \tanh(\lambda y f(x))] \qquad \text{DOOM II}$$



Caveat: Non-convex optimization problem

Aggressively Suppressing Noise - Results



Advantage of DOOM II over AdaBoost in noisy situations

Source: Mason *et al.* 2000

Boosting for Regression

Setup: $D_m = (x_1, y_1), \ldots, (x_m, y_m)$, $y_i \in \mathbb{R}$

Objective: Model $P(y|x)$ or $\mathbf{E}(Y|x)$

Scheme: Greedily construct additive model

Basic issues similar to optimizing convex upper bound for classification

Cost function:

$$\ell(y, f(x)) \qquad \text{e.g. } (y - f(x))^2, \quad |y - f(x)|$$

$$\hat{\Lambda}(f) = \frac{1}{m} \sum_{i=1}^{m} \ell(y_i, f(x_i))$$

Greedy Regression

General greedy algorithm

1. Choose $f_0 = \text{argmin}_{h \in \mathcal{H}}$
2. For $t = 1, 2, \ldots, T$

$$h_t = \underset{h \in \mathcal{H}}{\text{argmin}}\, \hat{\Lambda}(f_{t-1} + h)$$

$$\alpha_t = \underset{\alpha}{\text{argmin}}\, \hat{\Lambda}(f_{t-1} + \alpha h_t)$$

$$f_t(x) = f_{t-1}(x) + \alpha_t h_t(x)$$

Computation: Minimizing $\hat{\Lambda}(f + \alpha h)$ over $\mathcal{H}$ may be hard

Suggestion: Minimize least squares distance from negative gradient

Parametric form: $h(x) = h(x, \theta)$

Gradient Boosting

Gradient Boost (Friedman 1999)

1. Choose $f_0 = \text{argmin}_{h \in \mathcal{H}} \hat{\Lambda}(h)$

2. For $t = 1, 2, \ldots, T$

$$\tilde{y}_i = -\left[\frac{\partial \ell(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x)=f_{t-1}(x)}, \qquad i = 1, 2, \ldots, m$$

$$\theta_t, \beta_t = \underset{\theta,\beta}{\text{argmin}} \sum_{i=1}^{m} [\tilde{y}_i - \beta h(x_i, \theta)]^2$$

$$\alpha_t = \underset{\alpha}{\text{argmin}}\, \hat{\Lambda}(f_{t-1}(x) + \alpha h(x, \theta_t))$$

$$f_t(x) = f_{t-1}(x) + \alpha_t h(x, \theta_t)$$

Advantage: Many loss functions ℓ accommodated using least-squares optimization

Consistency of Boosting Algorithms

Main sources:

- Tong 2001, 2002
- Mannor, Meir and Tong 2002

Related work:

- Jiang 2001
- Lugosi and Vayatis 2002

Consistency Defined

Recall setup:

Examples:	$D_m = (x_1, y_1), \ldots, (x_m, y_m)$
Source distribution:	Each pair drawn independently from $P(X, Y)$
Hypothesis class:	$\mathcal{F} : \mathcal{X} \mapsto \{-1, +1\}$ or $\mathbb{R}$
Algorithm:	Based on D_m, select $\hat{f}_m$
Consistency:	Show that $L(\hat{f}_m) \xrightarrow{P} L^*$
	$L^* = \min_f L(f)$
Universal consistency:	$P(X, Y)$ unrestricted
Should we care?	While asymptotic, consistency is the only guarantee that ultimately we perform well

Convergence Rates

$P(X, Y)$ unrestricted: Universality possible; no rates available

Interesting restrictions: $\eta(x) = P(Y = 1|x)$ is 'smooth'

$\eta(x)$ is Lipschitz

$\eta(x)$ is r-times differentiable

$\eta(x)$ possesses a bounded variation

Minimax rates: Provide yardstick to quality

$$\inf_{\hat{f}_m} \sup_{P \in \mathcal{P}} \left(L(\hat{f}_m) - L^* \right) \geq \frac{c}{m^a}$$

Typical behaviors:

Highly smooth P: $a = 1/2$, or other constant

r times differentiable $a = r/d$ (Curse of dimensionality)

Examples of Universally Consistent Classifiers

k Nearest Neighbor Classifier:	Choose $k \to \infty, \; k/n \to 0$
Adaptive Nearest Neighbor Classifier:	Select k based on the data
Kernel approaches:	Similar results using width
Neural Networks:	Increase size of hidden layer at appropriate rate
Support Vector Machine:	Recently shown for certain kernels

Basic issue: Are greedy algorithms based on convex losses consistent?

Problem: Why should minimizing an upper bound work?

Minimizing an Upper Bound

Recall, we greedily minimize

$$\hat{A}(f) = \frac{1}{m}\sum_{i=1}^{m} \phi(y_i f(x_i))$$

If distribution were known, minimize

$$\begin{aligned} A(f) &= \mathbf{E}_{X,Y}\phi(Yf(X)) \ , \\ &= \mathbf{E}_X \left\{ \eta(X)\phi(f(X)) + (1-\eta(X))\phi(-f(X)) \right\} \end{aligned}$$

Consider

$$G(\eta, f) = \eta\phi(f) + (1-\eta)\phi(-f)$$

For all cost functions discussed $\eta > 1/2 \Longrightarrow f > 0$

Conclude: The sign of f gives correct classification

Minimizing an Upper Bound

Claim: (Tong 2002) Let $f_{\text{opt}} = \text{argmin}_f A(f)$, then

$$\boxed{L(f) - L^* \leq c\left(A(f) - A(f_{\text{opt}})\right)^{1/2}}$$

Assume: $\hat{f}^t_{\beta,m}$ obtained from a greedy algorithm, based on minimizing $\hat{A}(f)$ over $\beta\text{co}(\mathcal{H})$

Show: If $A(f) - A(f_{\text{opt}}) \to 0$ then $L(f) - L^* \to 0$

Obtain rates

Regularization: β serves as a regularization parameter

Large β- Good approximation, poor estimation

Small β- Poor approximation, good estimation

Universal Consistency and Convergence Rates

Claim: (Mannor, Meir and Tong 2002) **(i)** The AdaBoost, Least-Squares and Logistic loss functions lead to universal consistency; **(ii)** Obtain best rates of convergence for Logistic loss.

Proof idea:

- Using Tong's results, work with A instead of L
- Select a 'large' base class $\mathcal{H}$ (dense in class of continuous functions)
- Let the parameter β increase with sample size
- Control the estimation and approximation errors through β
- **Conclude:** establish (i) universal consistency and (ii) rates of convergence for 'smooth' decision boundaries

Multi-Category Classification

Main sources:

- Allwein, Schapire and Singer 2000
- Guruswami and Sahai 1999

Boosting for Multi-Category Problems

Multi-Class: Each input x can be classified into one of k classes, $k > 2$.

$\mathcal{Y} = \{1, 2, \ldots, k\}$

Multi-Label: A single input may be classified into several categories

Data: $(x_1, Y_1), \ldots, (x_m, Y_m)$, $Y_i \subseteq \mathcal{Y}$

Generalized Hypothesis: Each hypothesis predicts several labels

$H : \mathcal{X} \mapsto 2^{\mathcal{Y}}$

Example: $H(x) = \{2, 3, 5\}$

Boosting for Multi-Category Problems

Hamming Loss: Fraction of labels which differ

$$\Lambda(h(x), Y) = \frac{1}{k}|h(x)\Delta Y| \qquad \text{(symmetric difference)}$$

Translating to binary: For $Y \subseteq \mathcal{Y}$

$$Y\{\ell\} = \begin{cases} +1 & \text{if } \ell \in Y, \\ -1 & \text{if } \ell \notin Y. \end{cases}$$

Example: For $Y = \{2, 3, 5\}$, $Y\{\ell\} = +1$ if $\ell = 2, 3, 5$, -1 otherwise

Boosting idea: Replace each example (x_i, Y_i) by k examples

$$(x_i, Y_i) \mapsto ((x_i, \ell), Y_i\{\ell\})$$

Boosting for Multi-Category Problems

AdaBoost.MH (Schapire and Singer 1999)

1. Input: $(x_1, Y_1), \ldots, (x_m, Y_m)$, $x_i \in \mathbb{R}$, $Y_i \subseteq \mathcal{Y}$
2. **Initialize:** $P_1(i, \ell) = 1/mk$, $t = 1$
3. **For** $t = 1, 2, \ldots, T$
   - Construct *binary* weak classifiers $h_t(x_i, \ell)$ based on $\mathcal{D}_t$
   - Update distribution:

$$\mathcal{D}_{t+1}(i, \ell) = \frac{\mathcal{D}_t(i, \ell) \exp(-\alpha_t Y_i\{\ell\} h_t(x_i, \ell))}{Z_t}$$

   - Compute weights α_t
   - Set $t \leftarrow t + 1$
4. **Final hypothesis:** $H(x, \ell) = \mathrm{sgn}\left(\sum_{t=1}^{T} \alpha_t h_t(x, \ell)\right)$

Output Coding

Problem: Most standard classifiers operate naturally on binary problems

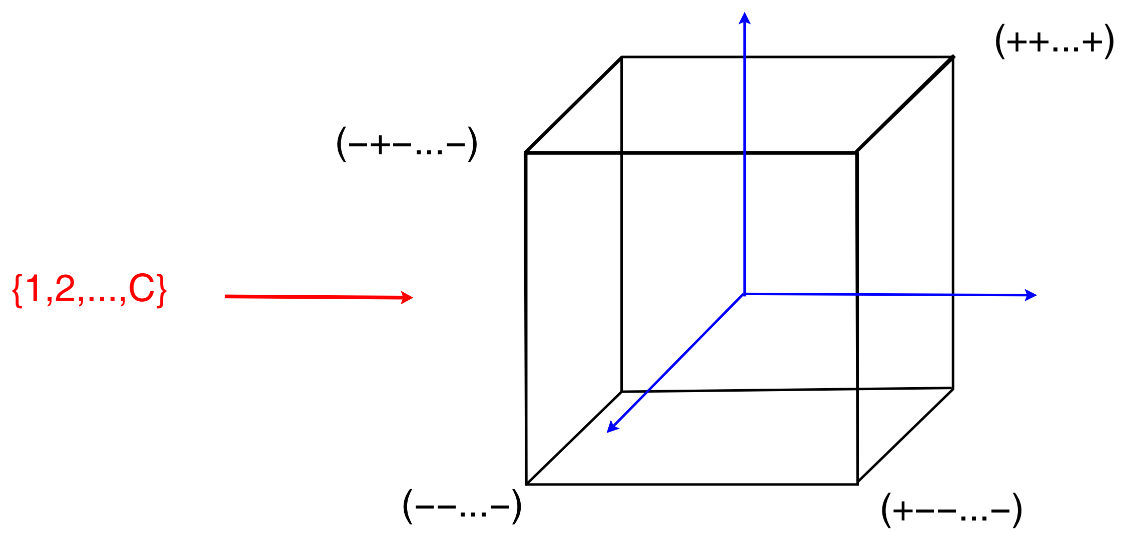
Coding: How do we naturally encode multi-category problems?

- ⋆ **Unary:** $1 \mapsto \{1, 0, \ldots, 0\}$, $2 \mapsto \{0, 1, \ldots, 0\}$ (k bits)
- ⋆ **Binary:** Map each class to its binary representation ($\log k$ bits)

Output Coding: Basic idea -

- ⋆ **Coding:** Map each class into a binary code word of size ℓ
- ⋆ **Learning:** Learn a set of ℓ binary classifiers
- ⋆ **Decoding:** Given an input $\mathbf{x}$ find 'closest' row read out class label

Solution by Output Coding II



Objective: Attempt to generate high error-correcting ability

Solution by Output Coding III

Coding:

$$\begin{pmatrix} 1 \\ 2 \\ \vdots \\ k \end{pmatrix} \mapsto \begin{pmatrix} + & + & - & + & + \\ - & - & + & + & - \\ & & \cdots & & \\ + & - & + & - & + \end{pmatrix}$$

Binary Learning: Construct ℓ binary classifiers

$$\text{Data} \quad \mapsto \quad \hat{f}_1(\mathbf{x}), \hat{f}_2(\mathbf{x}), \cdots, \hat{f}_\ell(\mathbf{x})$$

Hamming Decoding:

$$\hat{F}(\mathbf{x}) = \operatorname*{argmin}_{1 \le i \le c} \left\{ \sum_{j=1}^{\ell} |M(i,j) - \hat{f}_j(\mathbf{x})|/2 \right\}$$

What guarantees a good code?

Focusing on training-error demand:

Large inter-row distance	Error correcting property
Small binary classifier error	Depends on data, classifiers and matrix properties

Examples: (3-class problem) ρ- minimal row distance

One-against-all

$$\begin{pmatrix} + & - & - \\ - & + & - \\ - & - & + \end{pmatrix}$$

$\rho = 2$

Random

$$\begin{pmatrix} + & - & + & - \\ - & + & - & + \\ + & - & - & + \end{pmatrix}$$

$\rho = 2$

Solution by Output Coding VII

Some observations:

- One-against-all often leads to poor results
- Performance depends on several factors:
  - Matrix properties: inter-row and inter-column distances and correlations
  - Empirical classifier performance
- Many open questions - optimal choice far from clear

AdaBoost with ECC

Problem with AdaBoost.MH: Input domain augmented to $\mathcal{X} \times \mathcal{Y}$

- ⋆ Computational time increased
- ⋆ Unclear how to effectively make use of extra input

AdaBoost.ECC: Construct coding matrix sequentially and Boost

Advantage:

- ⋆ Only standard binary classifiers used (original input)

AdaBoost.ECC (Guruswami and Sahami 1999)

1. Input: $(x_1, Y_1), \ldots, (x_m, Y_m)$, $x_i \in \mathcal{X}$, $Y_i \in \mathcal{Y}$, $|\mathcal{Y}| = k$

2. **Initialize:** $\tilde{\mathcal{D}}_1(i, \ell) = 1/m(k-1)$ if $\ell = Y_i$, 0 otherwise

3. **For** $t = 1, 2, \ldots, T$

   - Form t'th binary problem, $\mu_t : Y \mapsto \{-1, +1\}$
   - Let $\mathcal{D}_t(i) = \sum_{\ell \in Y} \tilde{\mathcal{D}}_t(i, \ell) I[\mu_t(y_i) \neq \mu_t(\ell)]$
   - Compute binary weak learner based on $\mathcal{D}_t(i)$
   - Compute $\alpha_t(x)$
   - Update distribution:

$$\tilde{\mathcal{D}}_{t+1}(i, \ell) = \mathcal{D}_t(i, \ell) \exp[-\alpha_t(x_i) h_t(x_i)(\mu_t(\ell) - \mu_t(y_i))/2] \Big/ Z_t$$

4. **Final hypothesis:** $H(x) = \text{argmax}_{\ell \in Y} \left\{ \sum_{t=1}^{T} \alpha_t \mu_t(\ell) \right\}$

AdaBoost with ECC

$$\alpha_t = \frac{1}{2} \log \left(\frac{\sum_{i:h_t(x_i)=\mu_t(y_i)} \mathcal{D}_t(i)}{\sum_{i:h_t(x_i)\neq\mu_t(y_i)} \mathcal{D}_t(i)} \right)$$

Convergence of the training error: For both AdaBoost.MH and AdaBoost.ECC, if

$$\epsilon_t = \frac{1}{2} - \gamma_t \qquad \text{where } \gamma_t \geq \gamma > 0$$

then the training error decreases exponentially to zero

Main References Used

Comments:

- ⋆ The list is woefully incomplete. A more extensive list can be found in Schapire's 2002 review.
- ⋆ These and many other references available at www.boosting.org

References:

- ⋆ E.L. Allwein and R.E. Schapire and Y. Singer,“Reducing multiclass to binary: a unifying approach for margin classifiers”,J. Machine Learning Research, Vol. 1:113-141, 2000.
- ⋆ L. Breiman, “Bagging predictors”, Machine Learning, Vol. 24:123-140, 1996.
- ⋆ T. Dietterich, “An experimental comparison of three methods for

constructing ensembles of decision trees: Bagging, boosting and randomization", Machine Learning, Vol. 40(2):139-158, 2000.

⋆ J. Friedman, Greedy function approximation: a gradient boosting machine, The Annals of Statistics, 38(2):337-374, 2001.

⋆ J. Friedman, T. Hastie and R. Tibshirani, "Additive logistic regression: a statistical view of boosting", The Annals of Statistics, Vol. 38(2): 337-374, 2000.

⋆ Y. Freund, "Boosting a weak learning algorithm by majority", Information and Computation, Vol. 121:256-285, 1995.

⋆ Y. Freund and R.E. Schapire, "Experiments with a new boosting algorithm", Proceeding of the Thirteenth International Conference on Machine Learning, pp. 148-156, 1996.

⋆ V. Guruswami and A. Sahai, "Multiclass Learning, Boosting, and error-correcting codes", Proceedings of the Twelfth Annual Conference on Computational Learning Theory, 1999

- ⋆ W. Jiang, “Some theoretical aspects of boosting in the presence of noisy data”, Proceedings of the Eighteenth International Conference on Machine Learning, 2001
- ⋆ B. Kégl, T. Linder and G. Lugosi, “Data-dependent margin-based generalization bounds for classification”, Proceedings of the Fourteenth Annual Conference on Computational Learning Theory, 2001
- ⋆ G. Lugosi and N. Vayatis, “On the bayes-risk consistency of boosting methods”, Technical Report, Pompeu Fabra University, 2001
- ⋆ S. Mannor and R. Meir, “On the existence of weak learners and applications to boosting”, Machine Learning, 2002 (In press)
- ⋆ S. Mannor and R. Meir,“Geometric bounds for generalization in Boosting”, Proceedings of the Fourteenth Annual Conference on Computational Learning Theory, pp. 461-472, 2001
- ⋆ S. Mannor, R. Meir and T. Zhang,“The consistency of greedy algorithms for classification”, Submitted (2002)

- ⋆ L. Mason, P. Bartlett, J. Baxter and M. Frean,"Functional Gradient Techniques for Combining Hypotheses", in Advances in Large Margin Classifiers, Eds. A. Smola, P. Bartlett, B. Schölkopf and D. Schuurmans, MIT Press 2000
- ⋆ G. Rätsch, T. Onoda and K.R. Müller, "Soft margins for AdaBoost", Machine Learning, Vol. 42(3):287-320, 2001
- ⋆ R.E. Schapire, "The Boosting approach to machine learning: an overview", MSRI Workshop on Nonlinear Estimation and Classification, 2002.
- ⋆ R.E. Schapire, Y. Freund, P. Bartlett and W.S. Lee, "Boosting the margin: a new explanation for the effectiveness of voting methods", The Annals of Statistics, Vol. 26(5):1651-1686, 1998.
- ⋆ R.E. Schapire and Y. Singer,"Improved boosting algorithms using confidence-rated predictions", Machine Learning, 37(3): 297-336, 1999.
- ⋆ T. Zhang,"Statistical behavior and consistency of classification methods based on convex risk minimization, Technical Report, IBM T.J. Watson, 2001.

⋆ T. Zhang, "Sequential greedy approximation for certain convex optimization problems", Technical Report, IBM T.J. Watson, 2002.