An Introduction to Boosting
Part I

Gunnar Rätsch

Friedrich Miescher Laboratory
of the Max Planck Society

http://www.fml.tuebingen.mpg.de/~raetsch

Help with earlier presentations: Ron Meir, Klaus-R. Müller

Contents of this Course

- Part I: Introduction to Boosting
  1. Basic Issues, PAC Learning
  2. The Idea of Boosting
  3. AdaBoost's Behavior on the Sample
- Part II: Some Theory
  4. Generalization Error Bounds
  5. Boosting and Large Margins
  6. Leveraging as Greedy Optimization
- Part III: Extensions and Applications
  7. Boosting vs SVMs
  8. Improving Robustness & Extensions
  9. Applications

Slides available from the wiki and from
http://www.fml.mpg.de/~raetsch/boostintro

Sources of Information

Internet http://www.boosting.org*
http://www.cs.princeton.edu/~schapire/boost.html

Conferences Computational Learning Theory (COLT), Neural Information Processing Systems (NIPS), Int. Conference on Machine Learning (ICML), . . .

Journals Machine Learning, Journal of Machine Learning Research, Information and Computation, Annals of Statistics

People List available at http://www.boosting.org

Software Only few implementations: algorithms are "too simple" (cf. http://www.boosting.org)

Review Papers Schapire, 1999, and Meir & Rätsch, 2002.

* New version: http://www.fml.mpg.de/boosting

Learning – Problem Formulation I



The 'World':

Data:	$\{(\mathbf{x}_n, y_n)\}_{n=1}^N, \mathbf{x}_n \in \mathbf{R}^d, y_n \in \{\pm 1\}$
Unknown target function:	$y = f(\mathbf{x})$ (or $y \sim P(y\|\mathbf{x})$)
Unknown distribution:	$\mathbf{x} \sim p(\mathbf{x})$
Objective:	Given new $\mathbf{x}$, predict y

Problem: $P(\mathbf{x}, y)$ is unknown!

Learning – Problem Formulation II



The 'Model'

Hypothesis class: $\mathcal{H} = \{h \mid h : \mathbf{R}^d \to \{\pm 1\}\}$

Loss: $l(y, h(\mathbf{x}))$ (e.g. $\mathbf{I}[y \neq h(\mathbf{x})]$)

Objective: Minimize the true (expected) loss – "generalization error"

$$h^* = \operatorname*{argmin}_{h \in \mathcal{H}} L(h) \text{ with } \boxed{L(h) := \mathbf{E}_{\mathbb{X} \times \mathbb{Y}}\, l(\mathbb{Y}, h(\mathbb{X}))}$$

Problem: Only have a data sample available, $P(\mathbf{x}, y)$ is unknown!

Solution: Find empirical minimizer

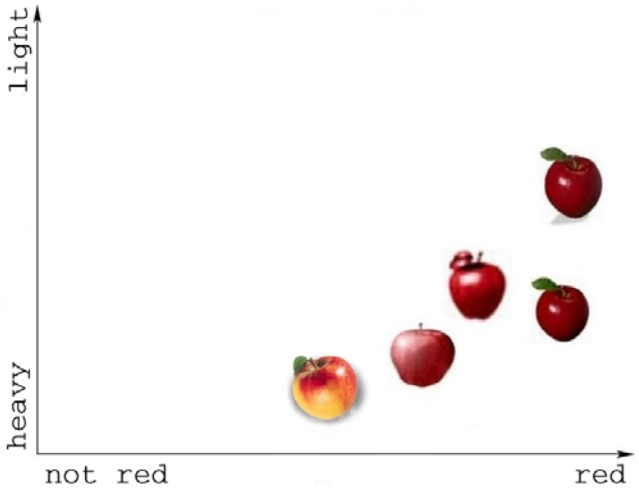
$$\hat{h}_N = \operatorname*{argmin}_{h \in \mathcal{H}} \frac{1}{N} \sum_{n=1}^{N} l(y_n, h(\mathbf{x}_n))$$

How can we efficiently construct complex hypotheses with small generalization errors?

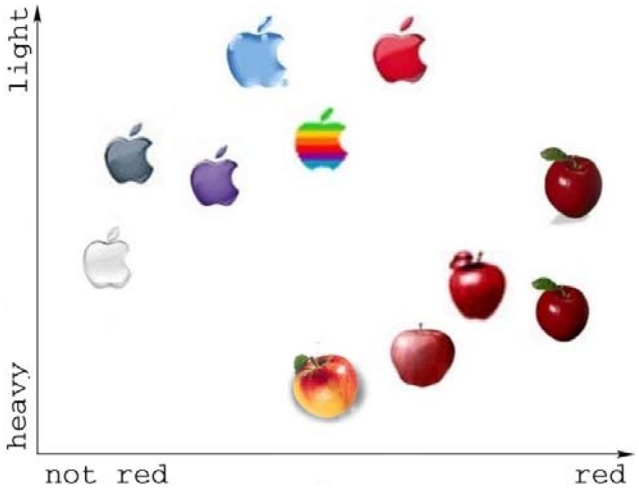
Example: natural vs. plastic Apples



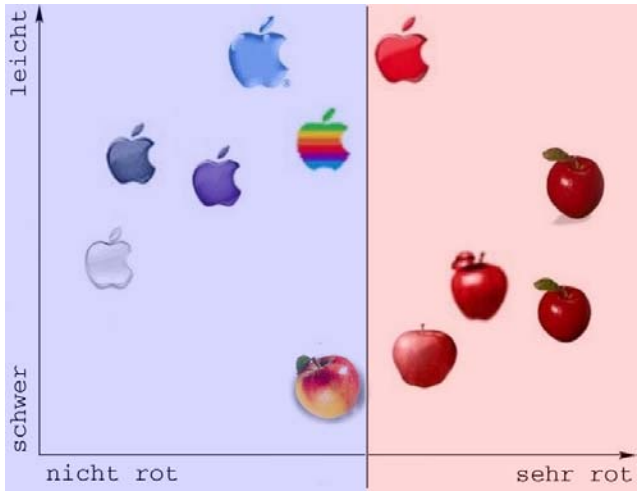
Example: natural vs. plastic Apples



Example: natural vs. plastic Apples



Simple Hypotheses: Cuts on Coordinate Axes



PAC Learning

Input: Sample $S_N = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_n) \sim P(\mathbf{x}, y)$
Hypothesis class $\mathcal{H} \ni h : \mathbf{x} \mapsto y$
Accuracy parameter ϵ
Confidence parameter δ

Algorithm a mapping from S_N to $\mathcal{H}$, $\mathcal{L} : S_N \mapsto \hat{h}_N$

Output function $\hat{h}_N$ from $\mathcal{H}$

Requirements 1. **P**robably **A**pproximately **C**orrect:

$$\mathbf{Pr}\left(S_N : L(\hat{h}_N) - L(h^*) \geq \epsilon\right) \leq \delta$$

where $h^* = \operatorname{argmin}_{h \in \mathcal{H}} \mathbf{E}\, l(\mathbb{Y}, h(\mathbb{X}))$

2. "efficient":
$\Rightarrow$ polynomial runtime in $N, 1/\epsilon, 1/\delta$

Levels of Generality

Recall

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

Restricted Setting: Assume $h^B = h^* \in \mathcal{H}$, require

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

Agnostic Setting: Assume nothing about h^B, require

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

Universal Setting: Assume nothing about h^B, require

$$L(\hat{h}_N) \xrightarrow{P} L(h^B)$$

Weak and Strong Learning

Assumption: Restricted setting, i.e. $h^B \in \mathcal{H}$

Strong PAC Learning: Demand small error with high probability:

$$\mathbf{Pr}\left(S_N : L(\hat{h}_N) - L(h^*) \geq \epsilon\right) \leq \delta \qquad (1)$$

for arbitrary (small) ϵ and δ

Weak PAC Learning: Demand that (1) holds for ϵ and δ

- Example: Binary classification, require that error ϵ is smaller than $1/2$:

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

With Boosting one can obtain a strong learner from a weak learner!

2. The Idea of Boosting

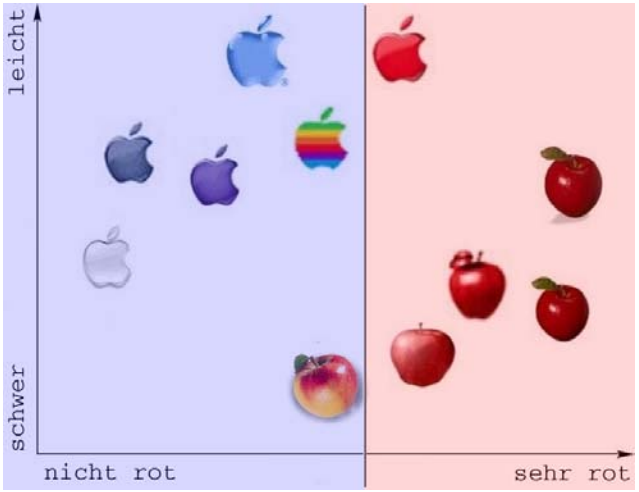


Hill Inlet, Whitsunday Island

Main Sources:

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

Simple Hypotheses: Cuts on Coordinate Axes

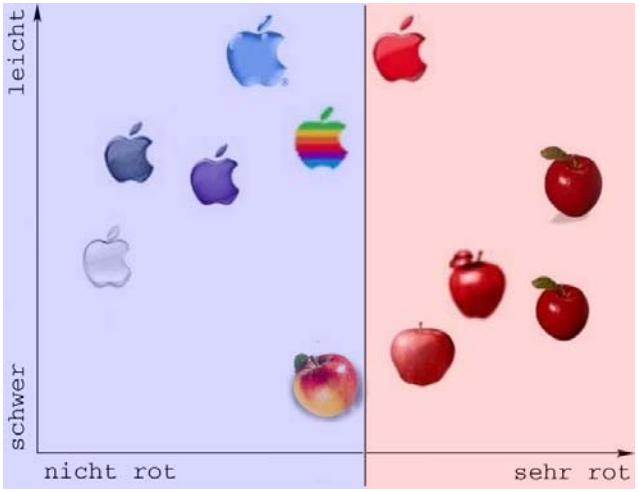


AdaBoost (Freund & Schapire 1996)

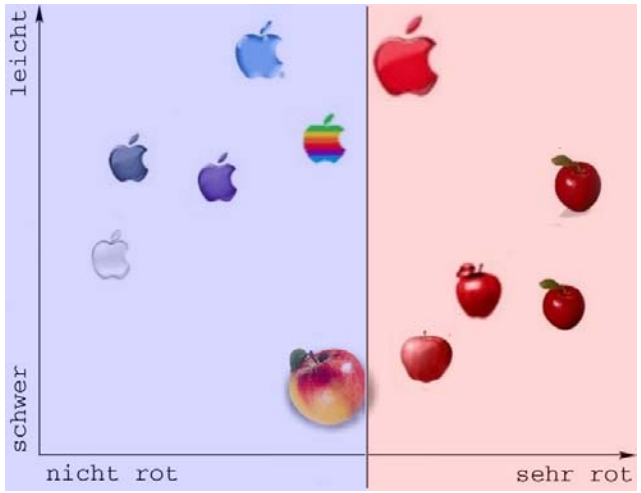
- **Idea:**
  - Simple hypotheses are not perfect!
  - Hypotheses combination $\Rightarrow$ increased accuracy
- **Problems:**
  - How to generate different hypotheses?
  - How to combine them?
- **Method:**
  - Compute distribution $d_1, \ldots, d_N$ on examples
  - Find hypothesis on the weighted training sample $(\mathbf{x}_1, y_1, d_1), \ldots, (\mathbf{x}_N, y_N, d_N)$
  - Combine hypotheses $h_1, h_2, \ldots$ linearly:

$$f = \sum_{t=1}^{T} \alpha_t h_t$$

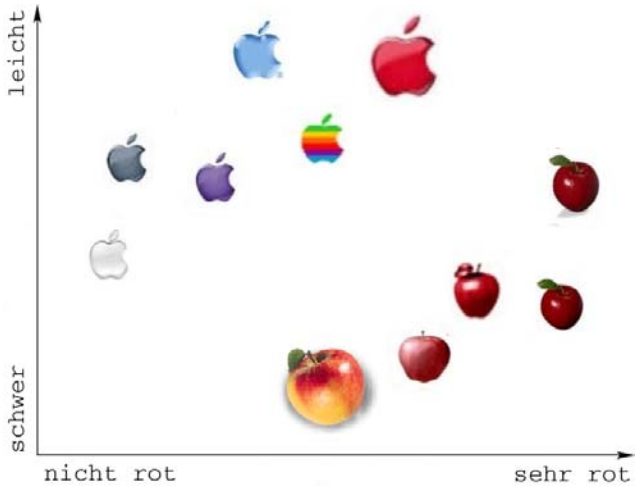
Boosting: 1st Iteration



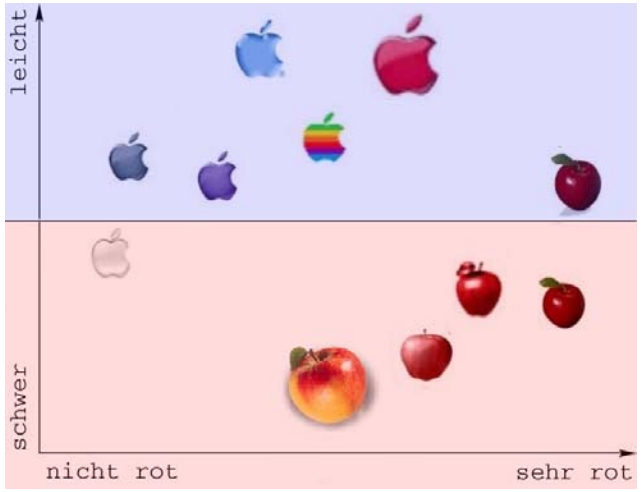
Boosting: Recompute weighting



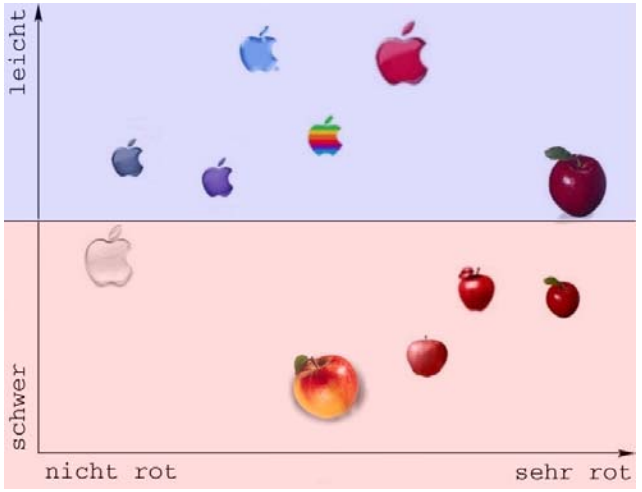
Boosting: 2nd Iteration



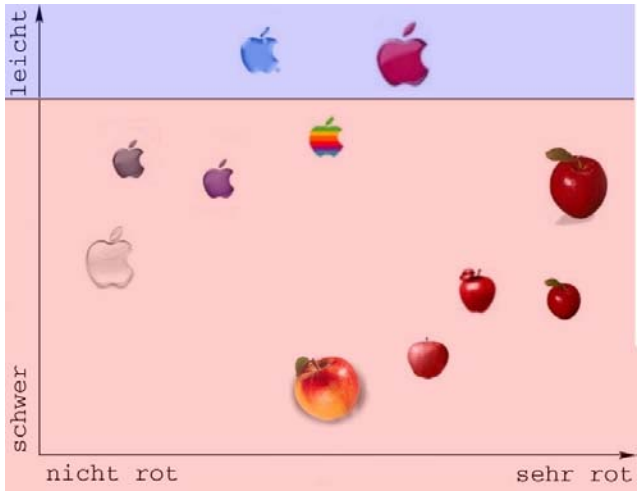
Boosting: 2nd Hypothesis



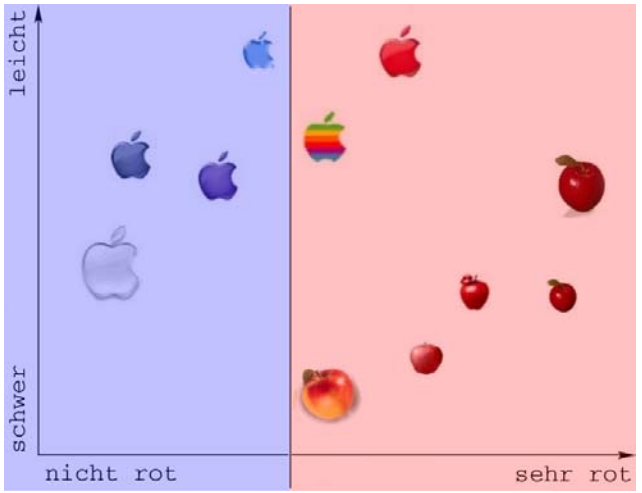
Boosting: Recompute weighting



Boosting: 3nd Hypothesis



Boosting: 4th Hypothesis



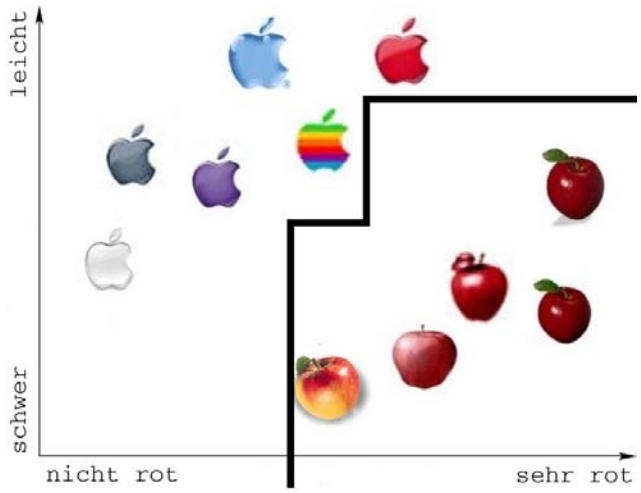
Combination of Hypotheses



Decision



Decision



Ensemble learning for classification

- Ensemble for binary classification consists of
  - Hypotheses (basis functions) $\{h_t(\mathbf{x}) : t = 1, \ldots, T\}$
    - of some hypothesis class $\mathcal{H} = \{h \mid h : \mathbf{x} \mapsto \pm 1\}$
  - Weights $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_T]$
    - satisfying $\alpha_t \geq 0$
- Classification Output: weighted majority of the votes
  - $f_{\text{Ens}}(\mathbf{x}) = \sum_{t=1}^{T} \alpha_t h_t(\mathbf{x})$
- How to find the hypotheses and their weights?
  - $+$ Boosting by Filtering (Schapire 1990)
  - $+$ Boosting by Majority (Freund 1995)
  - $-$ Bagging (Breiman 1996): $\alpha_t = 1/T$
  - $+$ AdaBoost (Freund & Schapire 1996)

Early Algorithms

Boosting by Filtering (Schapire 1990)

- Run weak learner on differently filtered example sets
- Combine the weak hypotheses; **Complex**
- Requires knowledge on the performance of weak learner

Boosting by Majority (Freund 1995)

- Run weak learner on same, but differently weighted set
- Combine the weak hypotheses linearly – Majority vote
- Requires knowledge on the performance of weak learner

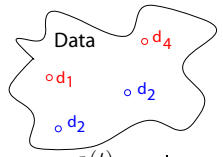
Bagging (Breiman 1996)

- Run learner on bootstrap replicates of the training set
- Average weak hypotheses – Majority vote
- Shown to reduce the variance (in regression)
- **not emphasizing "difficult" examples**

AdaBoost algorithm

Input: N examples $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$

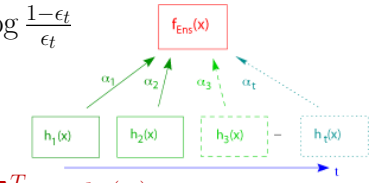
Initialize: $d_n^{(1)} = 1/N$ for all $n = 1 \ldots N$



Do for $t = 1, \ldots, T$,

1. Train base learner according to example distribution $\mathbf{d}^{(t)}$ and obtain hypothesis $h_t : \mathbf{x} \mapsto \{\pm 1\}$.
2. compute weighted error $\epsilon_t = \sum_{n=1}^{N} d_n^{(t)} \mathbf{I}(y_n \neq h_t(\mathbf{x}_n))$
3. compute hypothesis weight $\alpha_t = \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$
4. update example distribution

$$d_n^{(t+1)} = d_n^{(t)} \exp\left(-\alpha_t y_n h_t(\mathbf{x}_n)\right) / Z_t$$



Output: final hypothesis $f_{\text{Ens}}(\mathbf{x}) = \sum_{t=1}^{T} \alpha_t h_t(\mathbf{x})$

Desirable Properties of the Learner



- generates “simple” hypotheses

  $\Rightarrow$ Uses small base hypothesis set $\mathcal{H}$
- Should (approximately) minimize weighted error

$$h_t = \operatorname*{argmin}_{h \in \mathcal{H}} \epsilon_t(h)$$

where

$$\epsilon_t(h) = \sum_{n=1}^{N} d_n^{(t)} \mathbf{I}(y_n \neq h(\mathbf{x}_n))$$

- Often resampling techniques are used
  - Randomly draw subsets according to distribution $\mathbf{d}^{(t)}$
  - Run weak learner on the subset

$\Rightarrow$ Allows usage of arbitrary weak learner

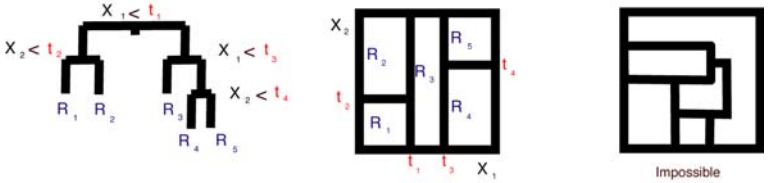
Weak Learners for Boosting



- Decision stumps (axis parallel splits)
- Decision trees (e.g. C4.5 by Quinlan 1996)
- Multi-layer Neural networks (e.g. for OCR)
- Radial basis function networks (e.g. UCI benchmarks, etc)

Decision trees:

- Hierarchical and recursive partitioning on the input space
- Many approaches, usually axis parallel splits



Hypothesis weight $\alpha_t = \frac{1}{2}\log\frac{1-\epsilon_t}{\epsilon_t}$



. . . is equivalent to minimizing the following loss function in each iteration:

$$E(\alpha) = \sum_{n=1}^{N} d_n^{(t)} \exp(-\alpha y_n h_t(\mathbf{x}_n))$$

Excercise!
Use that $h_t(\mathbf{x}_n) \in \{-1, +1\}$ and $\epsilon_t = \sum_{n=1}^{N} d_n^{(t)} \mathbf{I}(y_n \neq h(\mathbf{x}_n))$

3. AdaBoost’s Behavior on the Sample





Hill Inlet, Whitsunday Island

Main Sources:

- Freund & Schapire 1996
- Schapire, Freund, Bartlett & Lee 1998

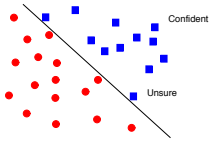
Margins and Classification



Motivation: Lose information in classifying a point as ± 1

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

Suggestion: Use real number to indicate confidence



Margin: $\rho_f(\mathbf{x}, y) = yf(\mathbf{x})$

Hyperplane: $\rho_{\mathbf{w},b}(\mathbf{x}, y) = y(\mathbf{w}^\top \mathbf{x} + b)$

Which Hyperplane?



- Function set used in boosting: Convex Hull of $\mathcal{H}$

$$\mathcal{F} := \left\{ f : \mathbf{x} \mapsto \sum_{h \in \mathcal{H}} \alpha_h h(\mathbf{x}) \;\middle|\; \alpha_h \geq 0, \sum_{h \in \mathcal{H}} \alpha_h = 1 \right\}$$

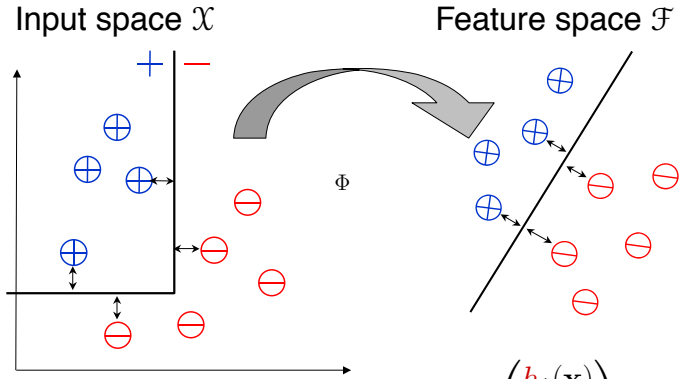
- the α's are the parameters
- Find a hyperplane in the Feature Space spanned by the hypotheses set $\mathcal{H} = \{h_1, h_2, \ldots\}$
- Margin ρ for an example $(\mathbf{x}_i, y_n)$ and $f \in \mathcal{F}$:

$$\rho_n(\boldsymbol{\alpha}) := y_n f(\mathbf{x}_n) = y_i \sum_{t=1}^{T} \frac{\alpha_t}{\sum_t \alpha_t} h_t(\mathbf{x}_i)$$

- Margin ρ for a function $f \in \mathcal{F}$ by $\rho(\boldsymbol{\alpha}) := \min_{n=1,\ldots,N} \rho_n(\boldsymbol{\alpha})$

Large Margin and Linear Separation





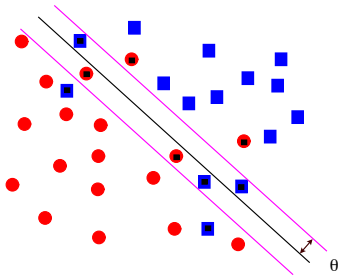
Linear separation in $\mathcal{F}$ is nonlinear separation in input space

$$\Phi(\mathbf{x}) = \begin{pmatrix} h_1(\mathbf{x}) \\ h_2(\mathbf{x}) \\ \vdots \end{pmatrix},$$

$$\mathcal{H} = \{h_1, h_2, \ldots\}$$

The Empirical Margin Error





Empirical margin error:

$$\hat{L}_N^\theta(f) = \frac{1}{N} \sum_{n=1}^{N} I[y_n f(x_n) \leq \theta]$$

(Scale of θ and f must match!)

The Empirical Margin Error



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

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

Then

$$\begin{aligned}\hat{L}_N^\theta(f_T) &= \frac{1}{N}\sum_{n=1}^N I[y_n f_T(x_n) \le \theta] \\ &\le e^{\theta\sum_{t=1}^T \alpha_t}\left(\prod_{t=1}^T Z_t\right),\end{aligned}$$

where Z_t is computed in AdaBoost:

$$Z_t = \sum_{n=1}^N d_n^t e^{-y_n \alpha_t h_t(\mathbf{x}_n)}$$

Proof



$$\begin{aligned}Z_t &= \sum_{n=1}^N d_n^t e^{-y_n \alpha_t h_t(x_n)} \\ &= \sum_{i:y_n = h_t(x_n)} d_n^t e^{-\alpha_t} + \sum_{i:y_n \neq h_t(x_n)} d_n^t e^{\alpha_t} \\ &= (1-\epsilon_t)e^{-\alpha_t} + \epsilon_t e^{\alpha_t}\end{aligned}$$

$$\begin{aligned} y f_T(x) \le \theta \quad &\Rightarrow \quad y\sum_{t=1}^T \alpha_t h_t(x) \le \theta \sum_{t=1}^T \alpha_t \\ &\Rightarrow \quad \exp\left(-y\sum_{t=1}^T \alpha_t h_t(x) + \theta\sum_{t=1}^T \alpha_t\right) \ge 1\end{aligned}$$

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

Proof, Cont'd



$$\begin{aligned} d_n^{t+1} &= d_n^t \frac{\exp\left(-\alpha_t y_n h_t(x_n)\right)}{Z_t} \\ &= \frac{\exp\left(-\sum_{\tau=1}^t \alpha_\tau y_n h_\tau(x_n)\right)}{m\prod_\tau Z_\tau} \qquad \text{(Induction)}\end{aligned}$$

$$\begin{aligned}\hat{\mathbf{E}}\{I[yf(x) \le \theta]\} &\le \hat{\mathbf{E}}\left\{e^{-y\sum_{t=1}^T \alpha_t h_t(x) + \theta\sum_{t=1}^T \alpha_t}\right\} \\ &= \frac{1}{N} e^{\theta\sum_{t=1}^T \alpha_t}\sum_{n=1}^N e^{-y_n\sum_{t=1}^T \alpha_t h_t(x_n)} \\ &= e^{\theta\sum_{t=1}^T \alpha_t}\left(\prod_{t=1}^T Z_t\right)\underbrace{\sum_{n=1}^N d_n^{T+1}}_{=1}\end{aligned}$$

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 (α_t is minimizer of $Z_t(\alpha_t)$)

Conclusion: Recall $\hat{L}_N^\theta(f) = \frac{1}{N}\sum_{n=1}^N I[y_n f(x_n) \le \theta]$

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

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

Training Error



Consider $\theta = 0$: $\hat{L}_N(f) = \frac{1}{N}\sum_{n=1}^{N} I[y_n f(x_n) \leq 0]$

Using $\ln x \leq x - 1$

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

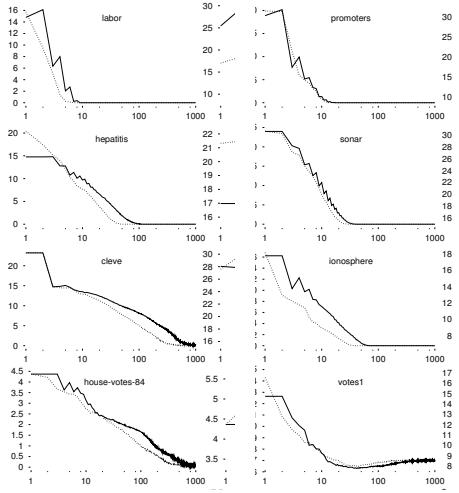
$$\leq e^{-2\sum_{t=1}^{T}\gamma_t^2} \to 0 \quad \text{if} \sum_{t=1}^{T} \gamma_t^2 \to \infty$$

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

If $\gamma_t \geq \gamma$ $(\forall t)$, then training error is zero after

$$T \leq \left\lceil \frac{2 \log N}{\gamma^2} \right\rceil$$

Training Error for Real Data

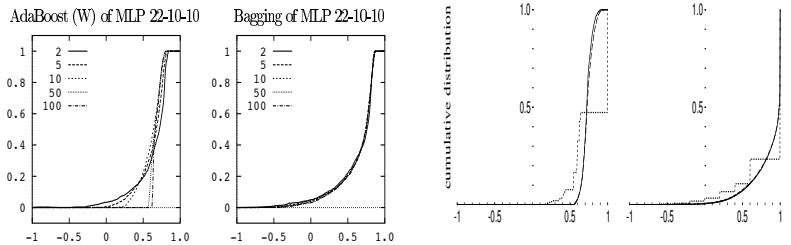


Source: Schapire and Singer 1999

Margin Plots I



Margin plots: Histogram of $y_n f_T(x_n)$, $n = 1, 2, \ldots, N$



Neural Networks

Schwenk and Bengio 2000

Decison Trees

Schapire *et al.* 1998

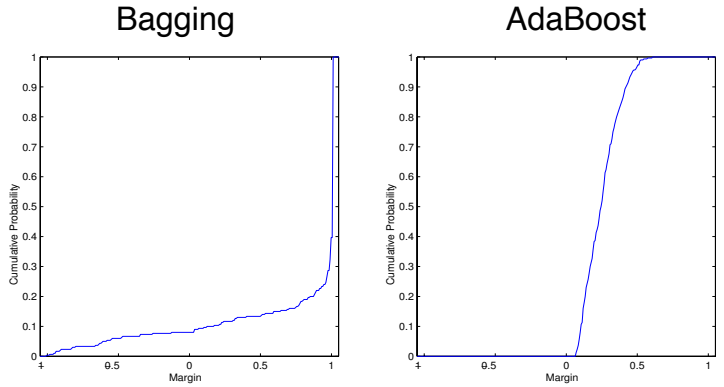
Observation:

- AdaBoost increases the margins of most points
- Bagging has a broader spectrum of distributions

Margin Plots II



AdaBoost tends to increase small margins, while decreasing large margins



Source: Schapire et al. 1998, Rätsch et al., 1998



An Introduction to Boosting
Part II

Gunnar Rätsch

Friedrich Miescher Laboratory
of the Max Planck Society

http://www.fml.tuebingen.mpg.de/~raetsch

Help with earlier presentations: Ron Meir, Klaus-R. Müller



Contents of this Course

- Part I: Introduction to Boosting
  1. Basic Issues, PAC Learning
  2. The Idea of Boosting
  3. AdaBoost's Behavior on the Sample
- Part II: Some Theory
  4. Generalization Performance
  5. Boosting and Large Margins
  6. Leveraging as Greedy Optimization
- Part III: Extensions and Applications
  7. Boosting vs SVMs
  8. Improving Robustness & Extensions
  9. Applications

Slides available from the wiki and from
http://www.fml.mpg.de/~raetsch/boostintro



4. Generalization Performance



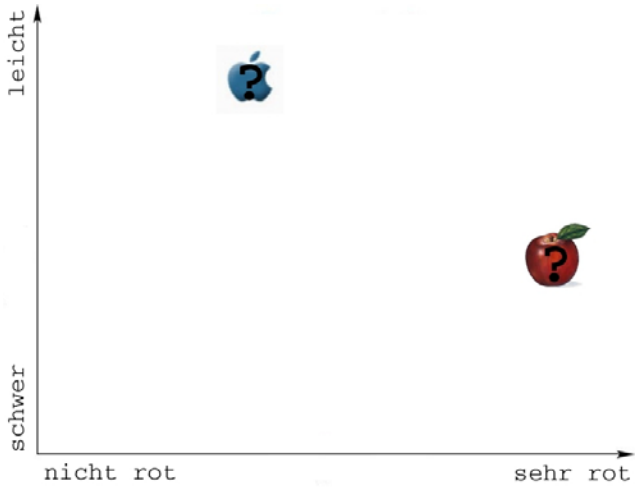
Hill Inlet, Whitsunday Island

Main Sources:

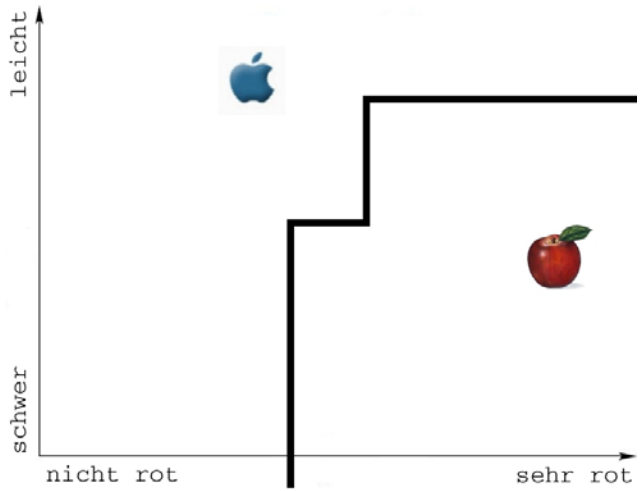
- Freund & Schapire 1996
- Schapire, Freund, Bartlett & Lee 1998



Unseen Examples



Unseen Examples

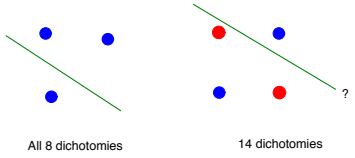


Setup

Expected error:	$L(f) = \mathbf{E}\{I[yf(x) \leq 0]\}$
Data:	$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$
Empirical error:	$\hat{L}_N(f) = \frac{1}{N}\sum_{n=1}^{N} I[y_n f(x_n) \leq 0]$
Empirical margin error:	$\hat{L}_N^{\theta}(f) = \frac{1}{N}\sum_{i=1}^{N} I[y_n f(x_n) \leq \theta]$
Empirical classifier:	$\hat{f}_N$
Main issue:	Bound $L(\hat{f}_N)$ in terms of $\hat{L}_N(\hat{f}_N)$
Standard VC bounds:	Use $\hat{L}_N(\hat{f}_N)$
Margin bounds:	Use $\hat{L}_N^{\theta}(\hat{f}_N)$

VC Dimension

Given:	$\mathcal{F} = \{f : \mathbf{R}^d \mapsto \{-1, +1\}\}$
Question:	How complex is the class?
Shattering:	$\mathcal{F}$ shatters a set $X \subset \mathbf{R}^d$, if $\mathcal{F}$ achieves all dichotomies on X
VC-dimension:	The size of the largest shattered subset of X





$d_{\mathcal{F}} = 3$

VC Bounds

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

$$|L(f) - \hat{L}_N(f)| \leq c_1\sqrt{\frac{d_{\mathcal{F}}}{N}} + c_2\sqrt{\frac{\log\frac{1}{\delta}}{N}}$$

Confidence interval: Standard statistical interpretation

Let

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

Then

$$\mathbf{E}L(\hat{f}_N) \leq \inf_{f \in \mathcal{F}} L(f) + c\sqrt{\frac{d_{\mathcal{F}}}{N}}$$

(Original proof: Vapnik and Chervonenkis 1971)

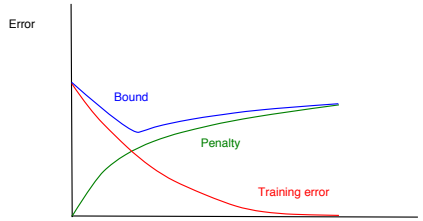
Interpretation of VC Bounds



Recall

$$L(f) \leq \hat{L}_N(f) + c_1\sqrt{\frac{d_{\mathcal{F}}}{N}} + c_2\sqrt{\frac{\log\frac{1}{\delta}}{N}}$$

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



VC Bounds for Convex Combination



Recall

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

Let

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

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

$$L(f) \leq \hat{L}_N(f) + c_1\sqrt{\frac{d(\mathrm{co}_T(\mathcal{H}))}{N}} + c_2\sqrt{\frac{\log\frac{1}{\delta}}{N}}$$

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

$$d(\mathrm{co}_T(\mathcal{H})) \approx T\log(T)\; d(\mathcal{H})$$

But sufficient to get original Boosting results . . .

PAC Boosting – VC dim. of comb. hypothesis

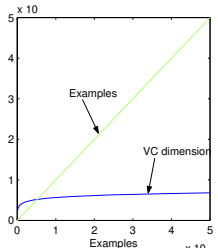


- AdaBoost needs at most $\lceil \frac{2\log N}{\gamma^2} \rceil$ iterations to obtain consistent hypothesis
- Let d be the VC dimension of the hypothesis class $\mathcal{H}$.
- Then the VC dimension of the class of AdaBoost's combined functions is bounded by

$$d_{Ens}(N,\gamma) = \mathcal{O}\Big(\underbrace{d\frac{\log(N)}{\gamma^2}}_{\sim T}\log\left(\frac{\log(N)}{\gamma^2}\right)\Big) = \mathcal{O}\left(d\log(N)\log^2(N)\right).$$

An Example

- VC dimension $d = 2$ (e.g. decision stumps)
- $\epsilon_t \leq 0.4 = \frac{1}{2} - \frac{1}{2}\gamma$ $\Rightarrow \gamma \geq 0.2$



PAC Boosting – putting things together



Assume weak learner is strong enough (weak PAC learner):

- AdaBoost generates consistent hypothesis: $\hat{L}_N(f) = 0$
- VC dimension of a consistent combined hypotheses class

$$d_N \leq \mathcal{O}(d\log(N)\log^2(N))$$

- With probability $1-\delta$ holds

$$L(f) \leq 0 + c_1\sqrt{\frac{d_N}{N}} + c_2\sqrt{\frac{\log\frac{1}{\delta}}{N}}$$

- The **generalization** error "$0 + \varepsilon(N)$" can be made arbitrarily small by increasing the sample size N.

PAC Booosting – Digestion

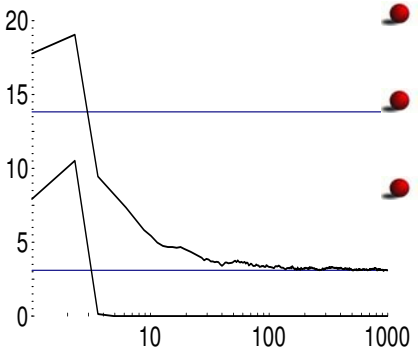


- properties of weak learner imply exponential convergence to a consistent hypothesis
- Fast convergence ensures small VC dimension of the combined hypothesis
- small VC implies small deviation from the empirical risk
- for any $\varepsilon > 0$ and $\delta > 0$ exists a sample size N, such that with probability $1 - \delta$ the expected risk is smaller than ε
- Any weak learner can be boosted to achieve an arbitrary high accuracy! ($\rightsquigarrow$ strong learner)

A "strange" Phenomenon



boosting C4.5 on "letter" data



- test error does not increase
  $\rightsquigarrow$ even after 1000 iterations!
- it continues to drop
  $\rightsquigarrow$ even after training error is 0!
- Occam's razor predicts simpler rule is better
  $\rightsquigarrow$ wrong in this case!?

Needs a better explanation!

Margin Bounds (Schapire, Freund, Bartlett & Lee 1998)



For any $f \in \mathrm{co}_T(\mathcal{H})$ and $\theta \in (0, 1]$, with probability $1 - \delta$

$$L(f) \leq \hat{L}_N^\theta(f) + \frac{c_1}{\theta}\sqrt{\frac{d(\mathcal{H})}{N}} + c_2\sqrt{\frac{\log\frac{1}{\delta}}{N}}$$

Observe:

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

Luckiness Tradeoff



$$L(f) \leq \hat{L}_N^\theta(f) + \frac{c_1}{\theta}\sqrt{\frac{d(\mathcal{H})}{N}} + c_2\sqrt{\frac{\log\frac{1}{\delta}}{N}}$$

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

$\frac{1}{\theta}\sqrt{\frac{d(\mathcal{H})}{N}}$ small

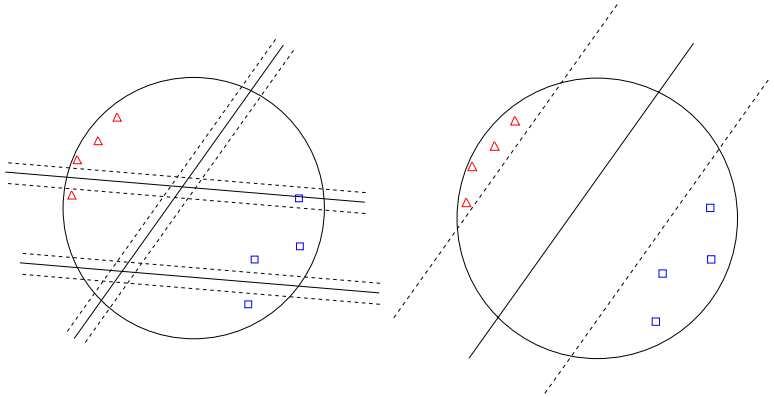
Small θ: $\hat{L}_N^\theta(f) \approx \hat{L}_N(f)$

$\frac{1}{\theta}\sqrt{\frac{d(\mathcal{H})}{N}}$ large

Optimum: If large θ can be achieved with small error
Extreme: AdaBoost – maximize θ while $\hat{L}_N^\theta(f) = 0$

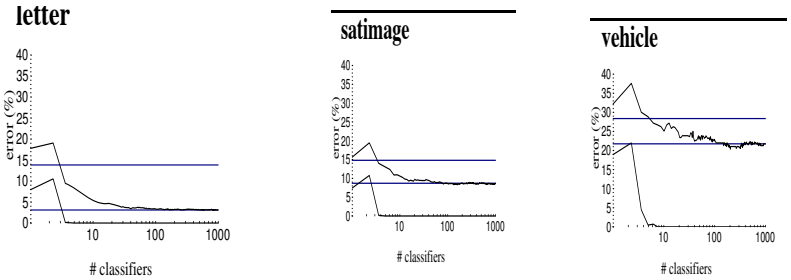
(cf. next section)

Large Margin ⇒ low Complexity? – Why?



Experiments - Is the Theory Corroborated?

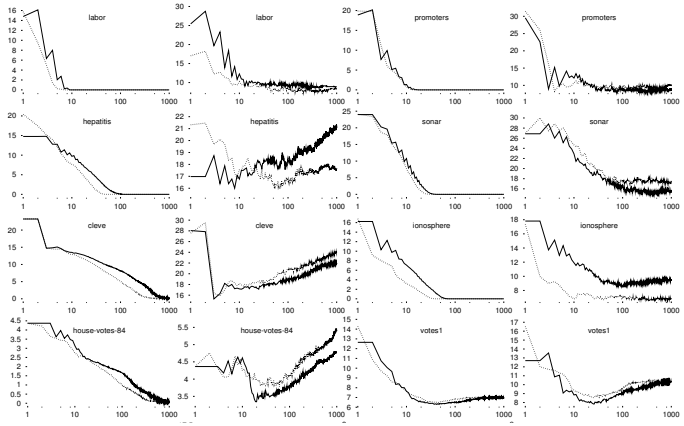
Weak learner: C4.5 decision tree



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
"Don't insist on driving training error to 0"
"Early stopping"
"Do not allow weights on examples to become very different"
"Restrict the optimization process"

New algorithms:

- Maximize the margin subject to constraint on the number of errors
- Trade-off determined by a regularization parameter

5. Boosting and Large Margins

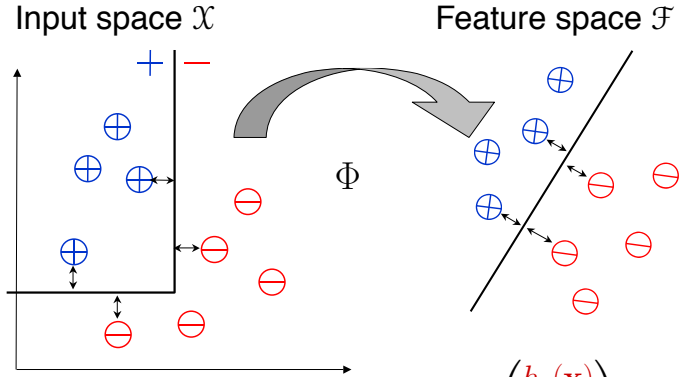


Hill Inlet, Whitsunday Island

Main Sources:

- Schapire, Freund, Bartlett & Lee 1998
- Breiman 1999
- Bennett, Demiriz & Shawe Taylor 2000
- Rätsch 2001, Rätsch & Warmuth 2002

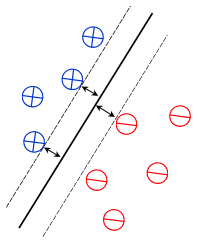
Large Margin and Linear Separation



Linear separation in $\mathcal{F}$ is nonlinear separation in input space

$$\Phi(\mathbf{x}) = \begin{pmatrix} h_1(\mathbf{x}) \\ h_2(\mathbf{x}) \\ \vdots \end{pmatrix}, \quad \mathcal{H} = \{h_1, h_2, \ldots\}$$

Boosting and Large Margins



- Hyperplane = combined hypothesis

$$f_{\boldsymbol{\alpha}}(\mathbf{x}) = (\Phi(\mathbf{x}) \cdot \boldsymbol{\alpha}) = \sum_{j=1}^{J} \alpha_j h_j(\mathbf{x})$$

- margin $\rho(\boldsymbol{\alpha}) = \min_n \underbrace{\frac{y_n(\Phi(\mathbf{x}_n) \cdot \boldsymbol{\alpha})}{\|\boldsymbol{\alpha}\|_1}}_{\text{distance}}$
- distance = ℓ_∞-distance (Mangasarian 1999)

Weak Learner $\mathcal{L}$

- One uses an example weighting $d_1, \ldots, d_N$
- Weak learner $\mathcal{L}$ returns hypotheses $\mathcal{H} \ni h : \mathbf{x} \mapsto \{\pm 1\}$
  ... which are consistently *better than random guessing*

$$\underbrace{\sum_{n=1}^{N} d_n \ \mathrm{I}(h(\mathbf{x}_n) \neq y_n)}_{\text{weighted tr. error of } h} \leq \frac{1}{2} - \frac{1}{2}\gamma$$

  for any weighting $\mathbf{d}$ and some fixed $\gamma > 0$
- For real valued hypotheses $h : \mathbf{x} \mapsto [-1, 1]$

$$\underbrace{\sum_{n=1}^{N} d_n \ y_n h(\mathbf{x}_n)}_{\text{Edge of } h} \geq \gamma$$

Edge and Margin



- Edge of a weak learner $\mathcal{L}$ for a probability vector $\mathbf{d}$ on the examples

$$\gamma(\mathbf{d}) := \max_{h \in \mathcal{H}} \sum_{n=1}^{N} d_n \, y_n h(x_n)$$

- Margin of the combined hypothesis $f_{\boldsymbol{\alpha}}$

$$\rho(\boldsymbol{\alpha}) := \min_{n=1\ldots N} y_n \frac{f_{\boldsymbol{\alpha}}(\mathbf{x}_n)}{\sum_t \alpha_t} = \min_{n=1\ldots N} y_n \sum_{t=1}^{T} \frac{\alpha_t}{\sum_r \alpha_r} h_t(\mathbf{x}_n)$$

Min-Max-Theorem:

$$\min_{\mathbf{d}} \gamma(\mathbf{d}) = \max_{\boldsymbol{\alpha}} \rho(\boldsymbol{\alpha})$$

If $\gamma(\mathbf{d}) \geq \gamma$, the margin is at least γ!

Discussion



What is the margin achieved by AdaBoost?

$\Rightarrow$ finds a solution with large, but not maximal margin

$\Rightarrow$ only about a half of the maximal margin

Modify AdaBoost for maximizing margin

- Arc-GV (Breiman 1999) asymptotically maximizes the margin
  - no converge rates
- AdaBoost* and TotalBoost requires $\frac{2\log(N)}{\epsilon^2}$ iterations to get margin in $[\rho^* - \epsilon, \rho^*]$
- Column Generation uses a Linear Programming solver
  - usually very fast
  - no converge rates

AdaBoost$_\varrho$ (Breiman 1997)



1. **Input:** $S = \langle (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N) \rangle$, T, margin target ϱ
2. **Initialize:** $d_n^1 = 1/N$ for all $n = 1 \ldots N$
3. **Do for** $t = 1, \ldots, T$,
   (a) Train classifier on $\{S, \mathbf{d}^t\}$ and obtain $h_t : \mathbf{x} \mapsto [-1, 1]$
   (b) Calculate the edge γ_t of h_t: $\gamma_t = \sum_{n=1}^{N} d_n^t y_n h_t(\mathbf{x}_n)$.
   (c) **if** $\gamma_t \leq \varrho$ or $\gamma_t = 1$, **then** break
   (d) Set $\alpha_t = \frac{1}{2} \log \frac{1+\gamma_t}{1-\gamma_t} \boxed{- \frac{1}{2} \log \frac{1+\varrho}{1-\varrho}}$.
   (e) Update $d_n^{t+1} = d_n^t \exp\left\{-\alpha_t y_n h_t(\mathbf{x}_n)\right\} / Z_t$, s.t. $\sum_{n=1}^{N} d_n^{t+1} = 1$
4. **Output:** $f(\mathbf{x}) = \sum_{t=1}^{T} \frac{\alpha_t}{\sum_r \alpha_r} h_t(\mathbf{x})$

Asymptotic Margin for AdaBoost$_\varrho$



Corollary 1 *Assume AdaBoost$_\varrho$ generates hypotheses $h_1, h_2, \ldots$ with edges $\gamma_1, \gamma_2, \ldots$ and coefficients $\alpha_1, \alpha_2, \ldots$. Let $\gamma = \inf_{t=1,2,\ldots} \gamma_t$ and assume $\gamma > \varrho$. Then, asymptotically ($t \to \infty$), the margin ρ of the combined hypothesis is bounded from below by*

$$\rho \geq \frac{\log(1-\varrho^2) - \log(1-\gamma^2)}{\log\left(\frac{1+\gamma}{1-\gamma}\right) - \log\left(\frac{1+\varrho}{1-\varrho}\right)} \geq \frac{\varrho + \gamma}{2}.$$

Arc-GV: Set $\varrho := \max_{r=1,\ldots,t-1} \rho_t$

$\Rightarrow$ easy convergence proof

$\Rightarrow$ no convergence rate!

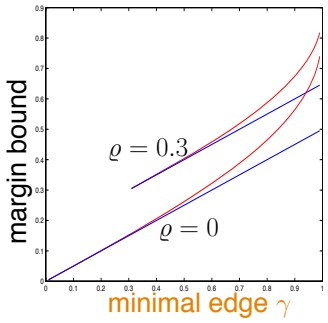
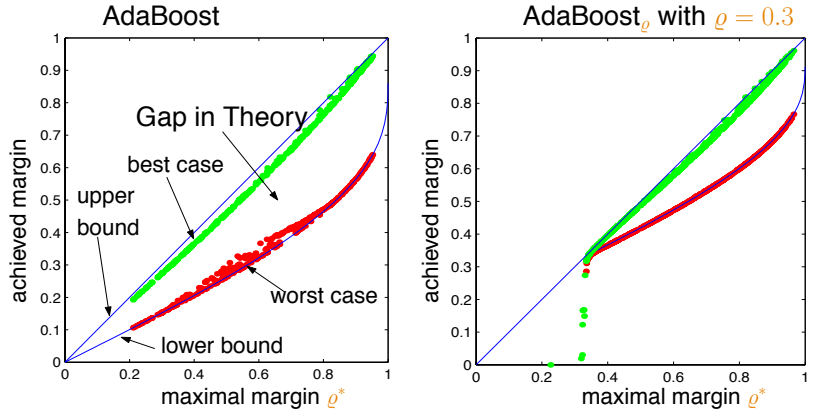


Illustration on Random Problems



$\Rightarrow$ There is a big GAP!

$\Rightarrow$ AdaBoost might not converge

Some Results to get to AdaBoost*

Lemma Assume that in a run of AdaBoost$_\varrho$ holds $\alpha_t \geq 0$ for all t. Then we have

$$\frac{1}{N}\sum_{n=1}^{N} \mathrm{I}\left(\mathbf{y}_n f_{\boldsymbol{\alpha}}(\mathbf{x}_n) \leq \theta\right) \leq \prod_{t=1}^{T} \exp\left\{\theta\alpha_t + \log Z_t\right\}$$

where Z_t is as in AdaBoost$_\varrho$.

Lemma Let γ_t be the edge of h_t in the t-th step of AdaBoost$_\varrho$. Assume $-1 \leq \varrho_t \leq \gamma_t$. Then holds for all $\theta \in [-1, 1]$

$$\exp\left\{\theta\alpha_t + \log Z_t\right\} \leq \exp\left(-\frac{1+\theta}{2}\log\left(\frac{1+\varrho_t}{1+\gamma_t}\right) - \frac{1-\theta}{2}\log\left(\frac{1-\varrho_t}{1-\gamma_t}\right)\right)$$

A Margin-Maximizing Algorithm: AdaBoost*

Lemma If $\theta \leq \varrho_t \leq \gamma_t$:

$$\exp\left(-\frac{1+\theta}{2}\log\left(\frac{1+\varrho_t}{1+\gamma_t}\right) - \frac{1-\theta}{2}\log\left(\frac{1-\varrho_t}{1-\gamma_t}\right)\right)$$

$$\leq \exp\left(-\frac{1+\varrho_t}{2}\log\left(\frac{1+\varrho_t}{1+\gamma_t}\right) - \frac{1-\varrho_t}{2}\log\left(\frac{1-\varrho_t}{1-\gamma_t}\right)\right) \leq 1 - \frac{1}{2}\frac{(\varrho_t-\gamma_t)^2}{1-{\varrho_t}^2}$$

- Assume: $\gamma_t \geq \rho^*$ for all $t = 1, \ldots, T$
- Would like to achieve $\theta = \varrho^* - \epsilon \quad \Rightarrow \quad \theta \leq \min_t \gamma_t - \epsilon$
- Set $\varrho_t = \min_t \gamma_t - \epsilon \quad \Rightarrow \quad \theta \leq \varrho_t \leq \gamma_t \quad \Rightarrow$ apply Lemma

$$\frac{1}{N}\sum_{n=1}^{N} \mathrm{I}\left(\mathbf{y}_n f_{\boldsymbol{\alpha}}(\mathbf{x}_n) \leq \theta\right) \leq \prod_{t=1}^{T}\left(1 - \frac{1}{2}\frac{(\varrho_t-\gamma_t)^2}{1-{\varrho_t}^2}\right) \leq \left(1 - \frac{\epsilon^2}{2}\right)^T$$

$\Rightarrow$ needs $\dfrac{2\log(N)}{\epsilon^2}$ iterations to achieve margin $\rho^* - \epsilon$

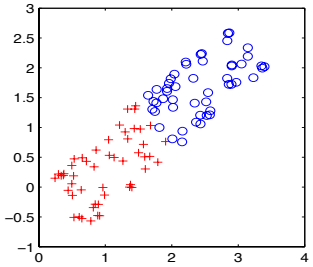
AdaBoost*

1. **Input:** $S = \langle(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\rangle$, desired accuracy ν
2. **Initialize:** $d_n^1 = 1/N$ for all $n = 1 \ldots N$
3. **Do for** $t = 1, \ldots$
   (a) Train classifier on $\{S, \mathbf{d}^t\}$ and obtain $h_t : \mathbf{x} \mapsto [-1, 1]$
   (b) Calculate the edge γ_t of h_t: $\gamma_t = \sum_i d_i^t y_i h_t(\mathbf{x}_i)$
   (c) Set $\varrho_t = \left(\min_{q=1,\ldots,t} \gamma_q\right) - \nu$
   (d) Set $\alpha_t = \frac{1}{2}\log\frac{1+\gamma_t}{1-\gamma_t}\ \boxed{-\frac{1}{2}\log\frac{1+\varrho_t}{1-\varrho_t}}$.
   (e) Update $d_n^{t+1} = d_n^t \exp\left\{-\alpha_t y_n h_t(\mathbf{x}_n)\right\}/Z_t$, s.t. $\sum_{n=1}^{N} d_n^{t+1} = 1$
4. **Output:** $f_{\boldsymbol{\alpha}}(\mathbf{x}) = \sum_{t=1}^{T} \alpha_t h_t(\mathbf{x})$, where the coefficients α_t maximize margin over hypotheses set $\{h_1, \ldots, h_T\}$.

Does a Larger Margin Help?



- 100 dimensional data with two discriminative directions (other noise)
- C4.5, AdaBoost and Marginal AdaBoost on 200 realizations of data



	C4.5	AB	Marginal AB
E_{gen}	$7.4 \pm 0.11\%$	$4.0 \pm 0.11\%$	$\mathbf{3.6 \pm 0.10\%}$
ϱ	–	0.31 ± 0.01	$\mathbf{0.58 \pm 0.01}$
wins	$1/200$	$59/200$	$\mathbf{140/200}$

- generalization error improves with larger margins
- this is only good for separable data $\Rightarrow$ **Soft-Margins**

(cf. Section 8)

Efficient Solution of Semi-infinite Problems



- Marginal AdaBoost and AdaBoost* can solve:

$$\max_{\boldsymbol{\alpha} \in \mathbf{R}^J} \rho \quad \text{with} \quad y_n \sum_{j=1}^{J} \alpha_j h_j(\mathbf{x}_n) \geq \rho \quad \text{for all} \quad n = 1, \ldots, N$$

with $\|\boldsymbol{\alpha}\|_1 = 1$

- Dual optimization Problem:

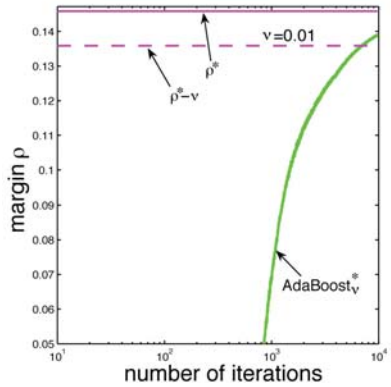
$$\min_{\varepsilon, \mathbf{d} \in \mathbf{R}^N} \varepsilon \quad \text{with} \quad \sum_{n=1}^{N} d_n \, y_n h_j(\mathbf{x}_n) \leq \varepsilon \quad \text{for all} \quad h_j \in \mathcal{H}$$

and $\mathbf{d} \geq \mathbf{0}$, $\sum_{n=1}^{N} d_n = 1$

- if $\mathcal{H}$ has infinite number of elements, there are infinitely many variables (primal) and constraints (dual)
- AdaBoost* works anyway and can solve the problem quite efficiently!

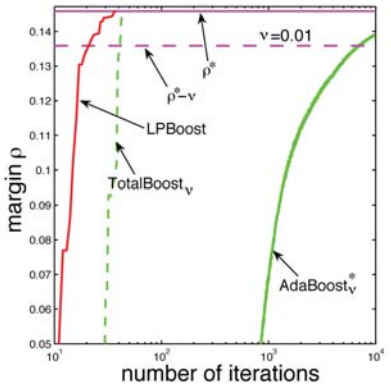
($\mathcal{H}$ needs to be compact)

Discussion: An Example



- AdaBoost* is as fast as expected . . .

Discussion: An Example



- AdaBoost* is as fast as expected . . .
  . . . but much slower than other techniques
  . . . Many calls to the base learning algorithm
- Difference: only optimizes one weight

AdaBoost* as Entropy Projection

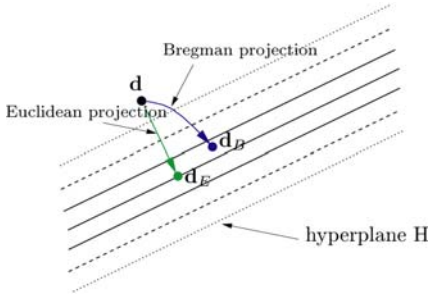
In each iteration of AdaBoost:

$$d_n^{t+1} = d_n^t \exp(-\alpha_t y_n h_t(\mathbf{x}_n))$$

It turns out that is the solution to

$$\min_{\mathbf{d}\in\mathcal{P}_+^N} \underbrace{\sum_{n=1}^N d_n \log(d_n^t) - d_n + d_n^t}_{\text{KL divergence}}$$

with $\underbrace{\sum_{n=1}^N d_n y_n h_t(\mathbf{x}_n) \le \varrho_t}_{\text{edge of } h_t}$



$\mathbf{d}^{t+1}$ is generalized projection of $\mathbf{d}^t$ onto hyperplane given by h_t

TotalBoost$_\nu$

1. **Input:** $S = \langle(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\rangle$, desired accuracy ν
2. **Initialize:** $d_n^1 = 1/N$ for all $n = 1 \ldots N$
3. **Do for** $t = 1, \ldots$
   (a) Train classifier on $\{S, \mathbf{d}^t\}$ and obtain $h_t : \mathbf{x} \mapsto [-1, 1]$
   (b) Calculate edge γ_t of h_t: $\gamma_t = \sum_i d_i^t y_i h_t(\mathbf{x}_i)$
   (c) Set $\varrho = (\min_{q=1,\ldots,t} \gamma_q) - \nu$
   (d) Update $\mathbf{d}^{t+1} = \operatorname{argmin}_{\{\mathbf{d}\in\mathcal{P}^N \mid \sum_i d_i y_i h_q(\mathbf{x}_i) \le \varrho, \text{ for } 1\le q\le t\} = \mathcal{C}_t} \Delta(\mathbf{d}, \mathbf{d}^1)$
   (e) **If** above infeasible or at boundary, then break
4. **Output:** $f_{\boldsymbol{\alpha}}(\mathbf{x}) = \sum_{t=1}^T \alpha_t h_t(\mathbf{x})$, where the coefficients α_t maximize margin over hypotheses set $\{h_1, \ldots, h_T\}$.

Iteration Bound $\lceil \frac{2\ln N}{\nu^2} \rceil$

Theorem 2 *For $\mathbf{d}^t, \mathbf{d}^{t+1} \in \hat{P}^N$ and $\mathbf{u} \in [-1, 1]^N$, if $\Delta(\mathbf{d}^{t+1}, \mathbf{d}^t)$ finite and $\mathbf{d}^{t+1} \cdot \mathbf{u} \neq \mathbf{d}^t \cdot \mathbf{u}$ then*

$$\Delta(\mathbf{d}^{t+1}, \mathbf{d}^t) > \frac{(\mathbf{d}^{t+1} \cdot \mathbf{u} - \mathbf{d}^t \cdot \mathbf{u})^2}{2}$$

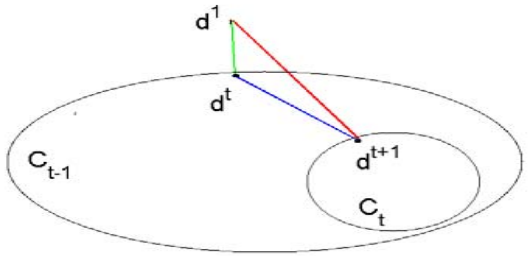
- $\mathbf{d}^t \cdot \mathbf{u}^t = \gamma_t$
- $\mathbf{d}^{t+1} \cdot \mathbf{u}^t \le \varrho \le \gamma_t - \nu$, where $u_i^t := y_i h_t(\mathbf{x}_i)$
- Thus $\mathbf{d}^t \cdot \mathbf{u}^t - \mathbf{d}^{t+1} \cdot \mathbf{u}^t \ge \nu$ and $\Delta(\mathbf{d}^{t+1}, \mathbf{d}^t) > \frac{\nu^2}{2}$

Generalized Pythagorean Theorem

$\mathcal{C}_t = \{\mathbf{d} \in \hat{P}^N | \mathbf{d} \cdot \mathbf{u}^q \le \hat{\gamma}_t, 1 \le q \le t\}, \mathcal{C}_0 = \hat{P}^N, \mathcal{C}_t \subseteq \mathcal{C}_{t-1}$
$\mathbf{d}^t$ is projection of $\mathbf{d}^1$ onto $\mathcal{C}_{t-1}$ at iteration $t-1$

$$\mathbf{d}^t = \operatorname*{argmin}_{\mathbf{d}\in\mathcal{C}_{t-1}} \Delta(\mathbf{d}, \mathbf{d}^1)$$

$$\Delta(\mathbf{d}^{t+1}, \mathbf{d}^1) \ge \Delta(\mathbf{d}^t, \mathbf{d}^1) + \Delta(\mathbf{d}^{t+1}, \mathbf{d}^t)$$



Sketch of Proof

$$
\begin{array}{rccccc}
1: & \Delta(\mathbf{d}^2,\mathbf{d}^1) - \Delta(\mathbf{d}^1,\mathbf{d}^1) & \geq & \Delta(\mathbf{d}^2,\mathbf{d}^1) & > & \frac{\nu^2}{2} \\
2: & \Delta(\mathbf{d}^3,\mathbf{d}^2) - \Delta(\mathbf{d}^2,\mathbf{d}^1) & \geq & \Delta(\mathbf{d}^3,\mathbf{d}^2) & > & \frac{\nu^2}{2} \\
 & \dots & & \dots & & \\
t: & \Delta(\mathbf{d}^{t+1},\mathbf{d}^1) - \Delta(\mathbf{d}^t,\mathbf{d}^1) & \geq & \Delta(\mathbf{d}^{t+1},\mathbf{d}^t) & > & \frac{\nu^2}{2} \\
 & \dots & & \dots & & \\
T-1: & \Delta(\mathbf{d}^T,\mathbf{d}^1) - \Delta(\mathbf{d}^{T-1},\mathbf{d}^1) & \geq & \Delta(\mathbf{d}^T,\mathbf{d}^{T-1}) & > & \frac{\nu^2}{2}
\end{array}
$$

$$
+: \quad \underbrace{\Delta(\mathbf{d}^T,\mathbf{d}^1)}_{\leq \ln N} - \underbrace{\Delta(\mathbf{d}^1,\mathbf{d}^1)}_{=0} \quad > \quad (T-1)\frac{\nu^2}{2}
$$

Therefore, $T \leq \lceil \frac{2 \ln N}{\nu^2} \rceil$

LPBoost: Column Generation

Recall: Dual optimization Problem:

$$
D(\mathcal{H}): \quad \min_{\varepsilon, \mathbf{d} \in \mathcal{P}^N} \varepsilon \quad \text{with} \quad \sum_{n=1}^{N} d_n \, y_n h_j(\mathbf{x}_n) \leq \varepsilon \quad \text{for all} \quad h_j \in \mathcal{H}
$$

Add one hypothesis per iteration and resolve the restricted master problem:

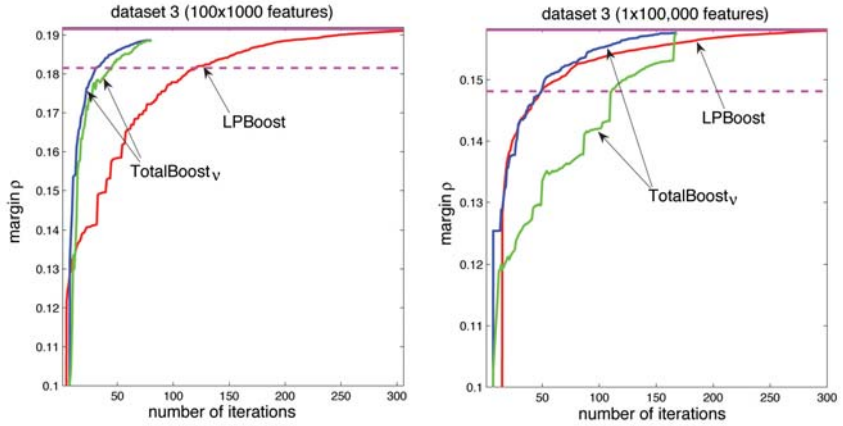
- Solve $D(\mathcal{H}_t)$ starting with $\mathcal{H}_1 = \emptyset$
- Find $h \in \mathcal{H}$ violating constraint in $D(\mathcal{H})$ at solution $D(\mathcal{H}_t)$
  - that is one with large enough edge
- Add hypothesis $\mathcal{H}_{t+1} = \mathcal{H}_t \cup \{h_t\}$

$\Rightarrow$ Converges in finite steps, when $\mathcal{H}$ finite
$\Rightarrow$ Converges asymptotically, when $\mathcal{H}$ infinite (how fast?)

We have also solved the primal problem if $\mathcal{H}$ is compact!

LPBoost can perform much worse

. . . when using many redundant features:



LPBoost vs. TotalBoost$_\nu$ on two 100,000 dimensional datasets: [*left*] many redundant features ($N_1 = 1,000, N_2 = 100$) and [*right*] independent features ($N_1 = 100,000, N_2 = 1$).

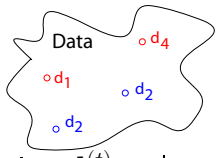
6. Boosting as Greedy Optimization



Main Sources:

- Kivinen & Warmuth 1999, Lafferty 1999
- Collins, Schapire & Singer 2002
- Rätsch, Mika & Warmuth 2001, Rätsch 2001

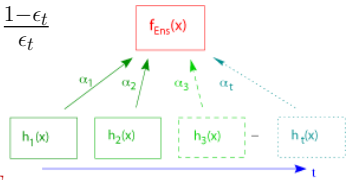
AdaBoost algorithm



Input: N examples $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$
Initialize: $d_n^{(1)} = 1/N$ for all $n = 1 \ldots N$
Do for $t = 1, \ldots, T$,

1. Train base learner according to example distribution $\mathbf{d}^{(t)}$ and obtain hypothesis $h_t : \mathbf{x} \mapsto \{\pm 1\}$.
2. compute weighted error $\epsilon_t = \sum_{n=1}^N d_n^{(t)} \mathbf{I}(y_n \neq h_t(\mathbf{x}_n))$
3. compute hypothesis weight $\alpha_t = \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$
4. update example distribution

$$d_n^{(t+1)} = d_n^{(t)} \exp\left(-\alpha_t y_n h_t(\mathbf{x}_n)\right) / Z_t$$



Output: final hypothesis $f_{\text{Ens}}(\mathbf{x}) = \sum_{t=1}^T \alpha_t h_t(\mathbf{x})$

Loss Function for AdaBoost

- AdaBoost stepwise minimizes a function

$$\mathrm{G}[f_{\boldsymbol{\alpha}}] = \sum_{n=1}^N \exp\left\{-y_n f_{\boldsymbol{\alpha}}(\mathbf{x}_n)\right\}$$

- The gradient of $\mathrm{G}[f_{\boldsymbol{\alpha}^{(t)}}]$ gives the example weights used for AdaBoost:

$$\frac{\partial\, \mathrm{G}[f_{\boldsymbol{\alpha}^{(t)}}]}{\partial f(\mathbf{x}_n)} = \exp\left\{-y_n f_{\boldsymbol{\alpha}^{(t)}}(\mathbf{x}_n)\right\} =: d_n^{(t+1)}$$

- The hypothesis coefficient α_t is chosen, such that $\mathrm{G}[f_{\boldsymbol{\alpha}^{(t)}}] = \mathrm{G}[f_{\boldsymbol{\alpha}^{(t-1)}} + \alpha_t h_t]$ is minimized:

$$\alpha_t = \operatorname*{argmin}_{\alpha_t \in \mathbf{R}} \mathrm{G}[f_{\boldsymbol{\alpha}^{(t)}}] = \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$$

The Generic Leveraging Algorithm

Input: loss $\mathrm{G} : \mathbf{R}^N \to \mathbf{R}$, N examples $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$
Initialize: $f_0 \equiv 0$, $\mathbf{d}^{(1)} = \nabla \mathrm{G}[f_0]$ (often $\sim 1/N$)
Do for $t = 1, \ldots, T$,

1. Run weak learner according to distribution $\mathbf{d}^{(t)}$ and obtain hypothesis h_t.
2. Compute hypothesis weight α_t such that $\mathrm{G}[f_{t-1} + \alpha_t h_t]$ is minimized
3. Update the combined hypothesis: $f_t = f_{t-1} + \alpha_t h_t$
4. Update example distribution: $\mathbf{d}^{t+1} = \nabla \mathrm{G}[f_t]$

Output: final hypothesis $f_T(\mathbf{x}) = \sum_{t=1}^T \alpha_t h_t(\mathbf{x})$

Leveraging = Boosting without PAC Boosting property

Examples

- AdaBoost (Freund & Schapire, 94) & Confidence Rated Boosting

$$\mathrm{G}[f_{\boldsymbol{\alpha}}] = \sum_{n=1}^N \exp\left\{-y_n f_{\boldsymbol{\alpha}}(\mathbf{x}_n)\right\} \qquad d_n = \exp(-y_n f_{\boldsymbol{\alpha}}(\mathbf{x}_n))$$

- Logistic Regression Boosting (Friedman et al., Collins et al., 2000)

$$\mathrm{G}[f_{\boldsymbol{\alpha}}] = \sum_{n=1}^N \log\left(1 + \exp\left\{-y_n f_{\boldsymbol{\alpha}}(\mathbf{x}_n)\right\}\right) \qquad d_n = \frac{\exp(-y_n f_{\boldsymbol{\alpha}}(\mathbf{x}_n))}{1 + \exp(-y_n f_{\boldsymbol{\alpha}}(\mathbf{x}_n))}$$

- Least Square Boosting (Friedman, 1999)

$$\mathrm{G}[f_{\boldsymbol{\alpha}}] = \sum_{n=1}^N (y_n - f_{\boldsymbol{\alpha}}(\mathbf{x}_n))^2 \qquad d_n = y_n - f_{\boldsymbol{\alpha}}(\mathbf{x}_n)$$

- Boosting with Soft Margins (Rätsch et al., 2001a, 2001b)

$$\mathrm{G}[f_{\boldsymbol{\alpha}}] = \sum_{n=1}^N \log\left(1 + \exp\left\{1 - y_n f_{\boldsymbol{\alpha}}(\mathbf{x}_n)\right\}\right) \qquad d_n = \frac{\exp(1 - y_n f_{\boldsymbol{\alpha}}(\mathbf{x}_n))}{1 + \exp(1 - y_n f_{\boldsymbol{\alpha}}(\mathbf{x}_n))}$$

Coordinate Descent



- Task: minimize a loss function of the form:

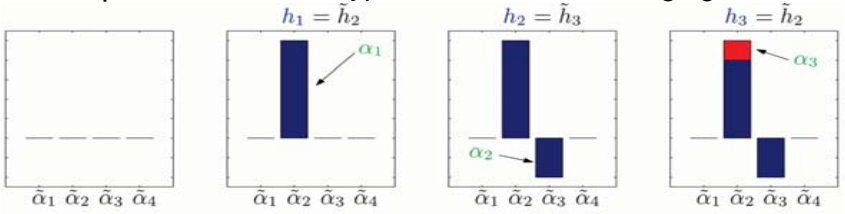
$$\mathrm{F}(\tilde{\boldsymbol{\alpha}}) = \mathrm{F}(\tilde{\alpha}_1, \ldots, \tilde{\alpha}_J)$$

- Particularly simple strategy: Iteratively select a coordinate, say the j-th, and find $\tilde{\alpha}_j$, such that $\mathrm{F}(\tilde{\alpha}_1, \ldots, \tilde{\alpha}_j, \ldots, \tilde{\alpha}_J)$ is minimized.
- How to choose j? Different strategies:
  - At random.
  - Round robin fashion.
  - Gauss-Southwell Scheme (GS): select component with largest, absolute gradient value.
  - Maximum-Improvement Scheme (MI): Choose the coordinate such, that the objective decreases maximally.
- Converges under certain conditions

Leveraging as Coordinate Descent



- Leveraging implements a coordinate gradient descent method in $\tilde{\boldsymbol{\alpha}}$-coefficient space $\mathbf{R}^J$ to minimize $\mathrm{G}[f_{\tilde{\boldsymbol{\alpha}}}]$
- Example: four different hypotheses, three Leveraging iterations



$\tilde{h}_2$ is selected twice and its weight is corrected in the 3rd iteration.

- If the learner selects a hypothesis more than once, we merge the weights by:

$$\tilde{\alpha}_j^{(t)} = \sum_{r=1}^{t} \alpha_r \mathrm{I}(h_r = \tilde{h}_j), \quad j = 1, \ldots, J, \qquad (1)$$

Base Learner and Selection Schemes



- If the base learner minimizes the $\mathbf{d}^{(t)}$-weighted training error ϵ_t, then it
  - maximizes the edge γ_t (assume $\tilde{h}_j(\mathbf{x}) \in \{-1, +1\}$) and
  - finds the hypothesis $h \in \mathcal{H}$ that has the largest gradient component (Gauss-Southwell Scheme), since

$$\frac{\partial \mathrm{G}[f_{\tilde{\boldsymbol{\alpha}}^{(t-1)}}]}{\partial \tilde{\alpha}_j} = \sum_{n=1}^{N} d_n^{(t)} \tilde{h}_j(\mathbf{x}_n) = \gamma_t,$$

- The base learner can be used as an "Oracle" to select new hypotheses/directions for coordinate descent.
- In confidence rated boosting (Schapire & Singer, 98) one uses the base learner to find the hypothesis that minimizes G (MI Scheme).

Preliminaries & Conditions



- The general form we consider is:

$$\min_{\tilde{\boldsymbol{\alpha}} \in \mathcal{S}} \mathrm{F}(\tilde{\boldsymbol{\alpha}}), \quad \text{where} \quad \mathrm{F}(\tilde{\boldsymbol{\alpha}}) = \mathrm{G}(H\tilde{\boldsymbol{\alpha}}) + \boldsymbol{\kappa}^\top \tilde{\boldsymbol{\alpha}}, \qquad (2)$$

  where G twice continously differenctiable and strictly convex on $\mathrm{dom}\ \mathrm{G}$, H has no zero columns, and $\mathcal{S}$ is a (not necessarily bounded) box constrained set.
- When/how will coordinate descent for such an loss function (think: AdaBoost) converge? **Only depends on base learner!**
- AdaBoost, Logistic Regression, LS-Boost all have such loss functions where $\boldsymbol{\kappa}$ is zero (hence: forget $\boldsymbol{\kappa}$ for now).
- Here we will only consider finite hypothesis sets (closed under complementation).

Condition: δ-Optimality



Let $\mathcal{H}$ be a finite, complementation closed hypothesis set and G be a functional as e.g. in AdaBoost.

A weak learner is called δ-optimal, if it always returns a hypothesis $\tilde{h}_j \in \mathcal{H}$ that satisfies

1. $\gamma_j \geq \delta \max_{j'=1,\ldots,J} \gamma_{j'}$, where $\gamma_j = \sum_{n=1}^{N} d_n \tilde{h}_j(\mathbf{x}_n)$,

or

2. $\mathrm{G}(\tilde{\boldsymbol{\alpha}}^t) - \mathrm{G}(\tilde{\boldsymbol{\alpha}}^{t+1,j}) \geq \delta \max_{j'=1,\ldots,J} \left(\mathrm{G}(\tilde{\boldsymbol{\alpha}}^t) - \mathrm{G}(\tilde{\boldsymbol{\alpha}}^{t+1,j'}) \right)$

for some fixed $\delta \in (0, 1]$ and all $\mathbf{d} = \nabla \mathrm{G}(\boldsymbol{\alpha})$.

$\Rightarrow$ Approximative Gauss-Southwell vs. Maximum Improvement Scheme

Convergence Theorem for Leveraging



Theorem: Given a training sample S, a finite, complementation closed hypothesis set $\mathcal{H}$. Let $\mathrm{G} : \mathbf{R}^N \to \mathbf{R}$ be twice differentiable, strictly convex and its Hessian be uniformly upper bounded. Suppose Leveraging generates a sequence of combined hypotheses $f_{\tilde{\boldsymbol{\alpha}}^0}, f_{\tilde{\boldsymbol{\alpha}}^1}, \ldots$ using a δ-optimal learning algorithm $\mathcal{L}$ for $\mathcal{H}$.
Then any limit point of $\{\tilde{\boldsymbol{\alpha}}^t\}$ is a solution and the sequence converges *linearly*.

Note: Linear convergence means, that the objective function decreases exponentially fast towards its minimum:

$$\mathrm{G}[f_{\tilde{\boldsymbol{\alpha}}^{(t)}}] - \mathrm{G}[f^*] \leq c^{-t} \quad \text{for some } c < 1$$

And what about AdaBoost?



- **Corollary** Suppose AdaBoost, Logistic Regression or LS-Boost generates a sequence of combined hypotheses $f_{\tilde{\boldsymbol{\alpha}}^{(1)}}, f_{\tilde{\boldsymbol{\alpha}}^{(2)}}, \ldots$ using a δ-optimal base learner.
  Assume solutions of the optimization problem are finite.
  Then $\{f_{\tilde{\boldsymbol{\alpha}}^{(t)}}\}$ converges linearly to a solution!
- Already known for AdaBoost and Logistic Regression if the data is separable (PAC boosting property).
  But here it also holds for nonseparable data and a large class of loss functions!

Dual Optimization – AdaBoost



In each iteration of AdaBoost:

$$d_n^{t+1} = d_n^t \exp(-\alpha_t y_n h_t(\mathbf{x}_n))$$

It turns out that is the solution to

$$\min_{\mathbf{d} \in \mathbf{R}_+^N} \underbrace{\sum_{n=1}^{N} d_n \log(d_n^t) - d_n + d_n^t}_{\text{KL divergence}}$$

$$\text{with} \quad \underbrace{\sum_{n=1}^{N} d_n y_n h_t(\mathbf{x}_n) = 0}_{\text{edge of } h_t}$$

$\mathbf{d}^{t+1}$ is generalized projection of $\mathbf{d}^t$ onto hyperplane given by h_t

Dual Optimization – Leveraging



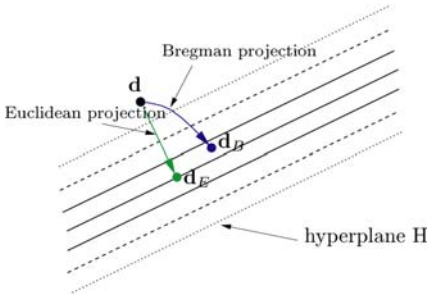
Define generalized "Distance" (Bregman Divergence):

$$\Delta_{\mathcal{G}}(\mathbf{d}, \mathbf{d}^t) := \mathcal{G}(\mathbf{d}) - \mathcal{G}(\mathbf{d}^t) - \langle \nabla \mathcal{G}(\mathbf{d}^t), \mathbf{d} - \mathbf{d}^t \rangle$$

Find the point "closest" to $\mathbf{d}^t$ which is on the hyperplane, i.e. solve

$$\min_{\mathbf{d} \in \mathbf{R}^N} \quad \Delta_{\mathcal{G}}(\mathbf{d}, \mathbf{d}^t)$$

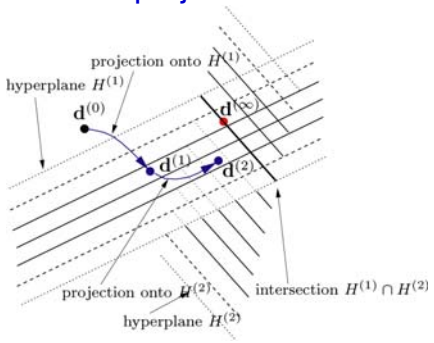
$$\text{with} \quad \sum_{n=1}^{N} d_n y_n h_t(\mathbf{x}_n) = 0$$



Bregman Algorithm and Leveraging



Bregman Algorithm:
– Iterate projections



Leads to Dual solution of:

$$\min_{\mathbf{d} \in \mathbf{R}^N} \quad \Delta_{\mathcal{G}}(\mathbf{d}, \mathbf{d}^0)$$

$$\text{with} \quad \sum_{n=1}^{N} d_n y_n h_j(\mathbf{x}_n) = 0 \qquad \forall h_j \in \mathcal{H}$$

Primal Problem:

$$\min_{\boldsymbol{\alpha}} \quad \mathrm{G}[f_{\boldsymbol{\alpha}}] + \text{const.}$$

G is convex conjugate function to $\mathcal{G}$

Digestion I



Convergence of Leveraging:

- Primal Domain (Coordinate Descent)
- Dual Domain (Generalized Projections)

$\Rightarrow$ obtain similar results as previously for AdaBoost

Many more convergence results!

Most general results:

- Rätsch et al. 2001 (global minimum; finite $\mathcal{H}$; linear convergence)
- Collins et al. 2001 (global minimum; finite $\mathcal{H}$; no rates)
- Zhang 2001 (global minimum; modified alg.; order one convergence)
- Mason et al. 1998 (local convergence; "AnyBoost")

Digestion II



Observe:

- AdaBoost$^{(*)}$ solves Linear Program (Section 5), but ...
- ... also minimizes exponential function (Section 6)!

Many solutions for exponential problem!

Questions:

- **Why is AdaBoost solving the LP?**
- **What is the connection between the exponential approach and the LP?**

$\Rightarrow$ **Barrier Optimization!**

How to solve a Linear Program?



$$\begin{aligned} \max_{\boldsymbol{\alpha},\rho} \quad & \rho \\ \text{subject to} \quad & y_n \sum_{j=1}^{J} \alpha_j h_j(\mathbf{x}_n) \geq \rho \quad n = 1,\ldots,N \\ & \alpha_j \geq 0 \qquad\qquad\qquad j = 1,\ldots,J \\ & \textstyle\sum_j \alpha_j = 1 \end{aligned}$$

- Simplex methods
- Barrier/Penalty methods
- Interior point methods
- ...

The number of hypotheses can be very large or infinite!
How is boosting solving this task?

Barrier Optimization



Constraint Optimization:

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & g(\boldsymbol{\theta}) \\ \text{s.t.} \quad & c_n(\boldsymbol{\theta}) \geq 0 \\ & n = 1,\ldots,N \end{aligned}$$

$$E_\beta(\boldsymbol{\theta}) = g(\boldsymbol{\theta}) + \sum_{n=1}^{N} \kappa_\beta(c_n(\boldsymbol{\theta}))$$

Convergence: $\lim_{\beta\to 0} \operatorname{argmin}_{\boldsymbol{\theta}} E_\beta(\boldsymbol{\theta}) = \boldsymbol{\theta}^*$ $\rightsquigarrow$ solution to problem

Barrierfunctions: $\kappa_\beta(t) = -\beta\log(t)$ $\rightsquigarrow$ Log-Barrier $\rightsquigarrow$ Interior Point
$\kappa_\beta(t) = \beta\exp(-t/\beta)$ $\rightsquigarrow$ Exp-Barrier $\rightsquigarrow$ Boosting

Barrier Optimization



Constraint Optimization:

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & g(\boldsymbol{\theta}) \\ \text{s.t.} \quad & c_n(\boldsymbol{\theta}) \geq 0 \\ & n = 1,\ldots,N \end{aligned} \qquad \begin{aligned} \min_{\boldsymbol{\alpha},\rho} \quad & -\rho \\ \text{s.t.} \quad & \rho_n(\boldsymbol{\alpha}) - \rho \geq 0 \\ & n = 1,\ldots,N \end{aligned}$$

$$E_\beta(\boldsymbol{\theta}) = g(\boldsymbol{\theta}) + \sum_{n=1}^{N} \kappa_\beta(c_n(\boldsymbol{\theta})) \qquad E_\beta(\boldsymbol{\alpha},\rho) = -\rho + \sum_{n=1}^{N} \kappa_\beta(\rho_N(\boldsymbol{\alpha}) - \rho)$$

Convergence: $\lim_{\beta\to 0} \operatorname{argmin}_{\boldsymbol{\theta}} E_\beta(\boldsymbol{\theta}) = \boldsymbol{\theta}^*$ $\rightsquigarrow$ solution to problem

Barrierfunctions: $\kappa_\beta(t) = -\beta\log(t)$ $\rightsquigarrow$ Log-Barrier $\rightsquigarrow$ Interior Point
$\kappa_\beta(t) = \beta\exp(-t/\beta)$ $\rightsquigarrow$ Exp-Barrier $\rightsquigarrow$ Boosting

AdaBoost$_\varrho$ as Barrier Algorithm



- The exp-barrier objective for the large margin LP:

$$E_\beta(\boldsymbol{\alpha}) = -\rho + \beta \sum_{n=1}^{N} \exp\left(-\frac{\rho_n(\boldsymbol{\alpha}) - \rho}{\beta}\right)$$

- We need $\beta \to 0$, and have $\|\boldsymbol{\alpha}\| \to \infty$: set $\beta := 1/\|\boldsymbol{\alpha}\|_1$:

$$E_\beta(\boldsymbol{\alpha}) = -\rho + \frac{1}{\|\boldsymbol{\alpha}\|_1} \underbrace{\sum_{n=1}^{N} \exp\left(-\rho_n(\boldsymbol{\alpha})\|\boldsymbol{\alpha}\|_1 + \rho\|\boldsymbol{\alpha}\|_1\right)}_{\text{AdaBoost}_\varrho\text{'s Objective}}$$

$\Rightarrow$ **AdaBoost$_\varrho$ minics a Barrier technique for solving a LP!**

An Introduction to Boosting
Part III

Gunnar Rätsch

Friedrich Miescher Laboratory
of the Max Planck Society

http://www.fml.tuebingen.mpg.de/~raetsch

Help with earlier presentations: Ron Meir, Klaus-R. Müller

Contents of this Course

- Part I: Introduction to Boosting
  1. Basic Issues, PAC Learning
  2. The Idea of Boosting
  3. AdaBoost's Behavior on the Sample
- Part II: Some Theory
  4. Generalization Error Bounds
  5. Boosting and Large Margins
  6. Leveraging as Greedy Optimization
- Part III: Extensions and Applications
  7. Boosting vs. SVMs
  8. Improving Robustness & Extensions
  9. Applications

Slides available from the wiki and from
http://www.fml.mpg.de/~raetsch/boostintro

7. Boosting vs. SVMs



Hill Inlet, Whitsunday Island

Main Sources:

- Schapire, Freund, Bartlett & Lee 1998
- Rätsch et al. 2000, 2001

SVM's Mathematical Program

The SVM minimization of

$$\begin{aligned} &\min_{\mathbf{w}\in\mathcal{F}_\Phi} && \tfrac{1}{2}\|\mathbf{w}\|^2 \\ &\text{subject to} && y_n\langle \mathbf{w}, \Phi(\mathbf{x}_n)\rangle \geq 1, \quad n = 1,\ldots,N. \end{aligned} \tag{1}$$

reformulate as maximization of the margin ρ

$$\begin{aligned} &\max_{\mathbf{w}\in\mathcal{F}_\Phi,\rho\in\mathbf{R}_+} && \rho \\ &\text{subject to} && y_n \sum_{j=1}^{D} w_j \Phi_j(\mathbf{x}_n) \geq \rho \quad \text{for} \quad n = 1,\ldots,N \\ & && \|\mathbf{w}\|_2 = 1, \end{aligned} \tag{2}$$

where $D = \dim(\mathcal{F})$ and Φ_j is the j-th component of Φ in feature space:

$$\Phi_j = P_j[\Phi]$$

Boosting's Mathematical Program



- master hypothesis

$$f(\mathbf{x}) = \sum_{t=1}^{T} \frac{\alpha_t}{\|\boldsymbol{\alpha}\|_1} h_t(\mathbf{x})$$

- base hypotheses h_t produced by the base algorithm.

AdaBoost* & Arc-GV solutions are (asymptotically) the same as linear program solution, maximizing the smallest margin ρ:

$$\max_{\mathbf{w}\in\mathbf{R}^J, \rho\in\mathbf{R}_+} \quad \rho$$

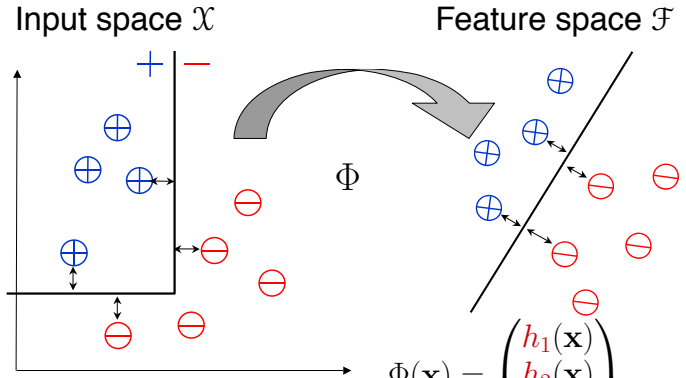
$$\text{subject to} \quad y_n \sum_{j=1}^{J} w_j h_j(\mathbf{x}_n) \geq \rho \quad \text{for} \quad n = 1, \ldots, N \qquad (3)$$

$$\|\mathbf{w}\|_1 = 1,$$

where J is the number of hypotheses in $\mathcal{H}$.

Large Margin and Linear Separation





Linear separation in $\mathcal{F}$ is nonlinear separation in input space

$$\Phi(\mathbf{x}) = \begin{pmatrix} h_1(\mathbf{x}) \\ h_2(\mathbf{x}) \\ \vdots \end{pmatrix},$$

$$\mathcal{H} = \{h_1, h_2, \ldots\}$$

$$k(\mathbf{x}, \mathbf{y}) = \Phi(\mathbf{x}) \cdot \Phi(\mathbf{y})$$

Even more explicit



Observe that any hypothesis set $\mathcal{H}$ implies a mapping Φ by

$$\Phi : \mathbf{x} \mapsto [h_1(\mathbf{x}), \ldots, h_J(\mathbf{x})]^\top,$$

and a kernel

$$k(\mathbf{x}, \mathbf{y}) = (\Phi(y) \cdot \Phi(\mathbf{y})) = \sum_{j=1}^{J} h_j(\mathbf{x}) h_j(\mathbf{y})$$

which could in principle be used for SVM learning.

$\Rightarrow$ Any hypothesis set $\mathcal{H}$ spans a feature space $\mathcal{F}$.

$\Rightarrow$ For any feature space $\mathcal{F}$ (spanned by Φ), the corresponding hypothesis set $\mathcal{H}$ can be readily constructed by

$$h_j = P_j[\Phi].$$

About Norms



- ℓ_2–norm (SVMs) $\Rightarrow$ Euclidean distance between hyperplane and examples is maximized
- using an arbitrary ℓ_p–norm constraint on the weight vector leads to maximizing the ℓ_q–distance between hyperplane and training points, where $\frac{1}{q} + \frac{1}{p} = 1$.
- Boosting-type algorithms maximize minimum ℓ_∞–distance of the training points to the hyperplane

What happens in the long run?

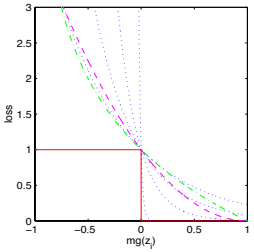
- Explicit expression for $d_n^{(t+1)}$:

$$d_n^{(t+1)} = \frac{\exp\left\{-\rho_n(\boldsymbol{\alpha}^{(t)})\right\}^{\|\boldsymbol{\alpha}^{(t)}\|}}{\sum_{n'=1}^{N}\exp\left\{-\rho_{n'}(\boldsymbol{\alpha}^{(t)})\right\}^{\|\boldsymbol{\alpha}^{(t)}\|}}$$

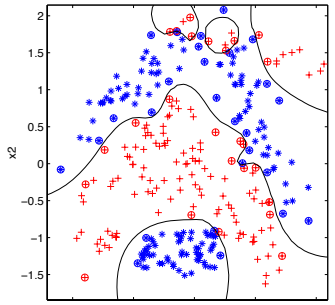
  $\rightsquigarrow$ **Soft-Max Function** with parameter $\|\boldsymbol{\alpha}^{(t)}\|_1$

- $\|\boldsymbol{\alpha}\|_1$ will increase monotonically ($\sim$ linear)

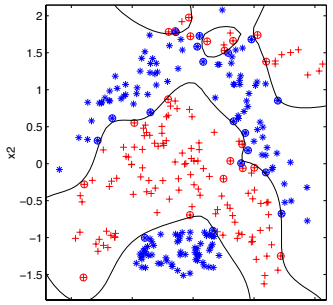
$\rightsquigarrow$ the d's concentrate on a **few** difficult patterns
$\rightarrow$ **"Support Patters"**



Support Patterns vs. Support Vectors



AdaBoost's decision line



SVM's decision line

These decision lines are for a low noise case with similar generalization errors. In AdaBoost, RBF networks with 13 centers were used.

Boosting vs. SVMs I

- SVMs

$$R[f] \leq R_{emp}[f] + \mathcal{O}\left(\sqrt{\frac{\log\left(N\theta^2\right)}{\theta^2 N} + \frac{\log(1/\eta)}{N}}\right).$$

- Boosting

$$R[f] \leq R^{\theta}_{emp}[f] + \mathcal{O}\left(\sqrt{\frac{d\log^2\left(\frac{N}{d}\right)}{\theta^2 N} + \frac{\log(1/\delta)}{N}}\right)$$

- **independent of the dimensionality of the space!**

Representer Theorem for Kernel Algorithms

Let c be an arbitrary cost function and r be a strictly monotonically increasing function on $[0,\infty)$ (for instance $r(\|\mathbf{w}\|_2) = \lambda\|\mathbf{w}\|_2$, $\lambda > 0$).

$$\min_{\mathbf{w}\in\mathcal{F}_\Phi} \sum_{n=1}^{N} c(y_n, \underbrace{f_{\mathbf{w}}(\mathbf{x}_n)}_{\langle \mathbf{w}, \Phi(\mathbf{x}_n)\rangle}) + r(\|\mathbf{w}\|_2) \qquad (4)$$

Then for any N examples $(\mathbf{x}_1, y_1), \ldots, (x_N, y_N)$ and any mapping Φ, then **any solution** $\mathbf{w}^*$ of (4) can be expressed by

$$\mathbf{w}^* = \sum_{n=1}^{N} \alpha_n \Phi(\mathbf{x}_n)$$

for some $\alpha_n \in \mathbf{R}$, $n = 1, \ldots, N$.
Only ℓ_2-norm regularizers have this property!

Representer Theorem for Boosting Algs



Let c be an arbitrary cost function and r be a monotonically increasing function on $[0, \infty)$.

$$\min_{\mathbf{w} \in \mathcal{F}_{\mathcal{H}}} \sum_{n=1}^{N} c(y_n, \underbrace{f_{\mathbf{w}}(\mathbf{x}_n)}_{\sum_{h \in \mathcal{H}} w_h h(\mathbf{x}_n)}) + r(\|\mathbf{w}\|_1) \qquad (5)$$

Then for any N example $(\mathbf{x}_1, y_1), \ldots, (x_N, y_N)$ and any compact hypothesis set $\mathcal{H}$, there **exist solutions** $f_{\mathbf{w}^*}$ of (5) that can be expressed by

$$f_{\mathbf{w}^*}(\mathbf{x}) = \sum_{n=1}^{N} \alpha_n h_n(\mathbf{x}).$$

Most of the w's can be made zero!
The ℓ_1-norm is the only convex regularizer that has this property!

Boosting vs. SVMs II



- Boosting, in contrast to SVMs, performs the computation explicitly in feature space.
- using the ℓ_1–norm instead of the ℓ_2–norm this becomes feasible $\rightarrow$ sparse solutions
- few hypotheses/dimensions $h_j = P_j[\Phi]$ needed to express the solution
- Boosting considers only the most important dimensions in feature space (at most N)

Ingredients of Boosting



- combinations of "weak hypotheses"
- large margins (for two-class classification)
- algorithms for solving convex optimization problems
  - re-weighting is optimization trick
  - Barrier minimization (exponential) or column generation
- sparseness in "Feature Space" allows efficient algorithms

8. Improving Robustness & Extensions



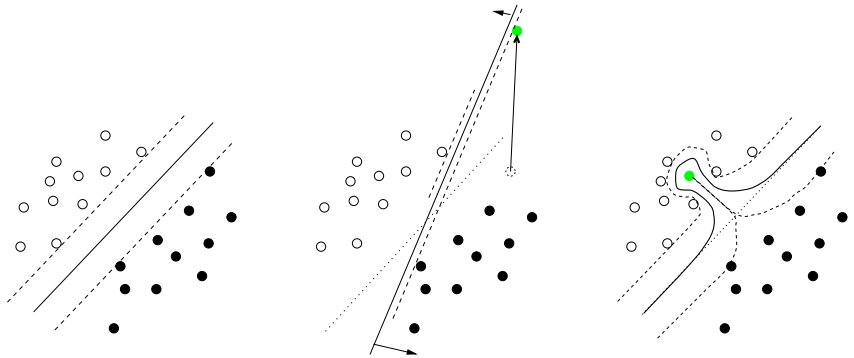


Hill Inlet, Whitsunday Island

Main Sources:

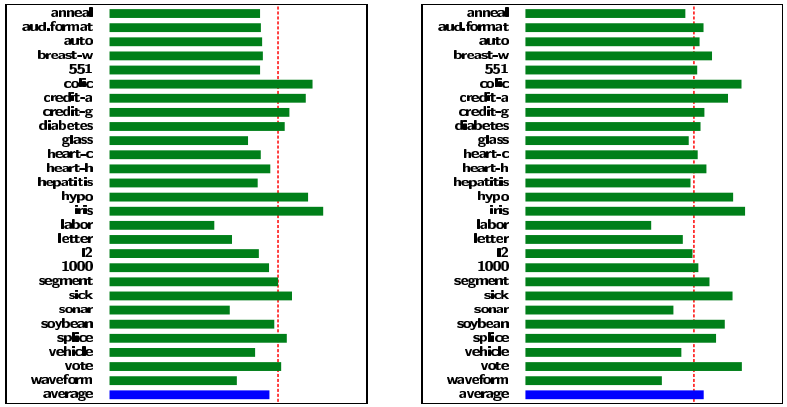
- Rätsch et al. 1998, 2000, 2001

Hard-Margin Classification



- The problem of finding a maximum margin "hyper-plane" on reliable data (left), data with outlier (middle) and a mislabeled pattern (right). The hard margin implies **noise sensitivity**.

Experiments on noisy data



Comparison on UCI repository data (Quinlan (1998)): Left 5% and right 10% label noise

How to get regularized versions of Boosting?

- Under certain conditions we want to ignore
  - Outliers and
  - mislabeled patterns

$\Rightarrow$ Change AdaBoost's Error Function

$\Rightarrow$ AdaBoost$_{Reg}$ (Rätsch et al., 1998)

$\Rightarrow$ DOOM (Mason et al., 1998)

$\Rightarrow$ Logit-Boost (Friedman et al., 1998)

$\Rightarrow$ Brown-Boost (Freund, 1999)

$\Rightarrow$ **Soft Margin** (like in SVM's)

- Extend the underlying LP to soft margins

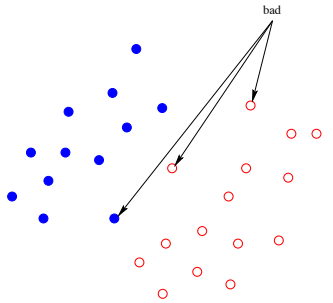
  $\Rightarrow$ ν-AdaBoost (Rätsch et al., 2000)

AdaBoost with Soft Margins

- Define a Soft Margin

$$\tilde{\rho}_n(\boldsymbol{\alpha}) = \rho_n(\boldsymbol{\alpha}) + \zeta_n,$$

  - where ζ_n is the amount of uncertainty in example $(\mathbf{x}_n, y_n)$
- Illustration



xx

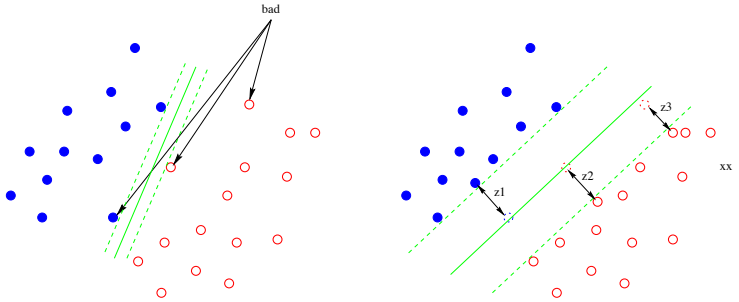
AdaBoost with Soft Margins



- Define a Soft Margin

$$\tilde{\rho}_n(\boldsymbol{\alpha}) = \rho_n(\boldsymbol{\alpha}) + \zeta_n$$

  - where ζ_n is the amount of uncertainty in pattern $\mathbf{z}_n$
- Illustration



AdaBoost with Soft Margins



- Once we have defined the uncertainty measure ζ_n, we can easily get a new regularized Boosting algorithm.

  $\Rightarrow$ Improve the Error Function by plugging-in the Soft Margin

$$\tilde{G}(\boldsymbol{\alpha}) = \sum_{n=1}^{N} \exp\left\{-\|\boldsymbol{\alpha}\|_1 \tilde{\rho}_n(\boldsymbol{\alpha})\right\}$$

$$d_n^{t+1} = \frac{\partial \tilde{G}(\boldsymbol{\alpha})}{\partial f_{\boldsymbol{\alpha}}(\mathbf{x}_n))}$$

$$\alpha_t = \operatorname*{argmin}_{\alpha_t \geq 0} \tilde{G}(\boldsymbol{\alpha})$$

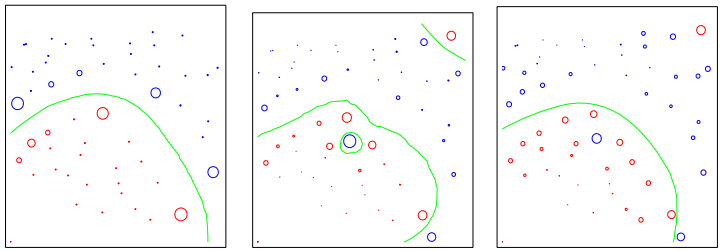
AdaBoost$_{Reg}$ – Reducing the Influence



- How can we know which patterns are unreliable?

  AdaBoost focuses on difficult-to-learn patterns by assigning high pattern weights d_n that we can exploit. Hence, we define the Influence of a pattern

$$\mu_n^t = \sum_{r=1}^{t} \frac{\alpha_r}{\|\boldsymbol{\alpha}\|_1} d_n^r \qquad \zeta_n = C(\mu_n^t)^2$$



AdaBoost without "Noise" **AdaBoost with "Noise"** **AdaBoost$_{Reg}$ with "Noise"**

AdaBoost$_{Reg}$



Positive:

- first algorithm that addresses overfitting in Boosting
- much improved results

Negative:

- Modification on algorithmic level
- hard to analyze
  - which optimization problem is solved
  - no generalization results

Idea:

- go back to beginning and redesign optimization problem
- use convergence results for leveraging
- apply margin bounds

ν-LP-Classification



Input: - N examples $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$
- hypothesis set $\mathcal{H}$ with J elements

- Introduce Slack-Variables ξ_n to the margin constraints
- Solve the Linear Program to find the hypotheses weights $\boldsymbol{\alpha}$:

$$\text{Maximize } \nu\rho - \frac{1}{N}\sum_{n=1}^{N}\xi_n \text{ under the constraints}$$

- $\rho_n(\boldsymbol{\alpha}) \geq \rho - \xi_n \qquad n = 1, \ldots, N$
- $\xi_n \geq 0 \qquad n = 1, \ldots, N$
- $\alpha_j \geq 0 \qquad j = 1, \ldots, J$
- $\|\boldsymbol{\alpha}\|_1 = 1$

Result: Empirical margin error is about ν (so-called ν-trick)

Margin Bounds (Schapire, Freund, Bartlett & Lee 1998)



For any $f \in \mathrm{co}_T(\mathcal{H})$ and $\theta \in (0, 1]$, with probability $1 - \delta$

$$L(f) \leq \underbrace{\hat{L}_N^{\theta}(f)}_{\approx\nu} + \frac{c_1}{\theta}\sqrt{\frac{d(\mathcal{H})}{N}} + c_2\sqrt{\frac{\log\frac{1}{\delta}}{N}}$$

Observe:

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

How to get regularized versions of Boosting?



- Extend/define the underlying optimization problem
  - . . . e.g. by incorporating some regularization . . .
- (re-)construct a new algorithm
  - . . . e.g. using Barrier (Rätsch et al., 2000) or Column-Generation techniques (Bennett et al., 2000). . .
- prove the convergence using tools from optimization

underlying opt. problem	Construct the new Algorithm using		
	AdaBoost	Barrier Meth.	CG-AdaBoost
LP-Classification	AdaBoost*		CG-AdaBoost
ν-LP-Classification	RoBoost	C-Bar-Lev	ν-CG-AdaBoost
ν-LP-Regression	+	ε-Boost	ν-CG-Regression
ν-LP One-class	-	One-Lev	1-Class CG
Multi-class LP	AdaBoost.M2	. . .	+
. . .	. . .	. . .	. . .

Barrier Optimization



Constraint Optimization:

$$\min_{\boldsymbol{\theta}} \; g(\boldsymbol{\theta}) \quad \text{s.t. } c_n(\boldsymbol{\theta}) \geq 0, \; n = 1, \ldots, N \qquad \min_{\boldsymbol{\alpha},\rho} \; -\rho \quad \text{s.t. } \rho_n(\boldsymbol{\alpha}) - \rho \geq 0, \; n = 1, \ldots, N$$

$$E_\beta(\boldsymbol{\theta}) = g(\boldsymbol{\theta}) + \sum_{n=1}^{N}\kappa_\beta(c_n(\boldsymbol{\theta})) \qquad E_\beta(\boldsymbol{\alpha}, \rho) = -\rho + \sum_{n=1}^{N}\kappa_\beta(\rho_N(\boldsymbol{\alpha}) - \rho)$$

Convergence: $\lim_{\beta\to 0} \operatorname{argmin}_{\boldsymbol{\theta}} E_\beta(\boldsymbol{\theta}) = \boldsymbol{\theta}^*$ $\rightsquigarrow$ solution to problem

Barrierfunctions: $\kappa_\beta(t) = -\beta\log(t)$ $\rightsquigarrow$ Log-Barrier $\rightsquigarrow$ Interior Point
$\kappa_\beta(t) = \beta\exp(-t/\beta)$ $\rightsquigarrow$ Exp-Barrier $\rightsquigarrow$ Boosting

ν-LP vs C-LP



- Minimize $-\nu\rho + \frac{1}{N}\sum_{n=1}^{N}\xi_n$ under the constraints

$$\begin{aligned}
&\rho_n(\boldsymbol{\alpha}) \geq \rho - \xi_n \qquad n = 1,\ldots,N\\
&\xi_n \geq 0 \qquad n = 1,\ldots,N\\
&\alpha_j \geq 0 \qquad j = 1,\ldots,J\\
&\textstyle\sum_j \alpha_j = 1
\end{aligned}$$

- Minimize $\sum_j \alpha_j + \frac{1}{N}\sum_{n=1}^{N}\xi_n$ under the constraints

$$\begin{aligned}
&\rho_n(\boldsymbol{\alpha}) \geq 1 - \xi_n \qquad n = 1,\ldots,N\\
&\xi_n \geq 0 \qquad n = 1,\ldots,N\\
&\alpha_j \geq 0 \qquad j = 1,\ldots,J
\end{aligned}$$

C-LP with Barriers I



- Minimize $\sum_j \alpha_j + \frac{1}{N}\sum_{n=1}^{N}\xi_n$ under the constraints

$$\begin{aligned}
&\rho_n(\boldsymbol{\alpha}) \geq 1 - \xi_n \qquad n = 1,\ldots,N\\
&\xi_n \geq 0 \qquad n = 1,\ldots,N\\
&\alpha_j \geq 0 \qquad j = 1,\ldots,J
\end{aligned}$$

- Barrier objective s.t. $\boldsymbol{\alpha} \geq \mathbf{0}$

$$\begin{aligned}
E_\beta(\boldsymbol{\alpha},\boldsymbol{\xi}) \;=\; &\sum_j \alpha_j + \frac{1}{N}\sum_{n=1}^{N}\xi_n +\\
&+\beta\sum_n \exp\left(-\frac{\xi_n}{\beta}\right)\\
&+\beta\sum_n \exp\left(-\frac{\rho_n(\boldsymbol{\alpha}) - 1 + \xi_n}{\beta}\right)
\end{aligned}$$

C-LP with Barriers II



- Barrier objective s.t. $\boldsymbol{\alpha} \geq \mathbf{0}$

$$\begin{aligned}
E_\beta(\boldsymbol{\alpha},\boldsymbol{\xi}) \;=\; &\sum_j \alpha_j + \frac{1}{N}\sum_{n=1}^{N}\xi_n +\\
&+\beta\sum_n \exp\left(-\frac{\xi_n}{\beta}\right)\\
&+\beta\sum_n \exp\left(-\frac{\rho_n(\boldsymbol{\alpha}) - 1 + \xi_n}{\beta}\right)
\end{aligned}$$

- Determine optimal $\xi_n \equiv \xi_n(\boldsymbol{\alpha},\beta)$:

$$\xi_n(\boldsymbol{\alpha},\beta) = \beta\log\left(\frac{1-\rho_n(\boldsymbol{\alpha})}{\beta}\right)$$

- Plug-in

$$E_\beta(\boldsymbol{\alpha}) \;=\; \sum_j \alpha_j + \beta\frac{1}{N}\sum_{n=1}^{N}\log\left[1+\exp\left(\frac{1-\rho_n(\boldsymbol{\alpha})}{\beta}\right)\right] + \text{const.}$$

C-LP with Barriers III



- Barrier objective s.t. $\boldsymbol{\alpha} \geq \mathbf{0}$

$$E_\beta(\boldsymbol{\alpha}) \;=\; \sum_j \alpha_j + \beta\frac{1}{N}\sum_{n=1}^{N}\log\left[1+\exp\left(\frac{1-\rho_n(\boldsymbol{\alpha})}{\beta}\right)\right] + \text{const.}$$

- Has general form:

$$\min_{\boldsymbol{\alpha}\geq\mathbf{0}} G_\beta(\boldsymbol{\alpha}) + \boldsymbol{\kappa}^\top\boldsymbol{\alpha}$$

$\Rightarrow$ use general Leveraging Scheme to solve optimization problem

$\Rightarrow$ reduce barrier parameter during optimization

$\Rightarrow$ show convergence

Resulting Algorithm



1. **Input:** N examples $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, T, C
2. **Initialize:** $f^{(0)} \equiv 0$, $\beta^{(1)} = \beta_{\text{start}}$, $d_n^{(1)} = [\exp(1/\beta^{(1)}) + 1]^{-1}$,
3. **Do for** $t = 1, \ldots, T$
   (a) Train weak learner on $\{S, \mathbf{d}^{(t)}\}$ and obtain $h_t : \mathbf{x} \mapsto [-1, 1]$
   (b) Let $\gamma_r = \sum_{n=1}^{N} y_n d_n^{(t)} h_r(\mathbf{x}_n)$ and $\underline{\alpha}_r = \sum_{s=1}^{t} \alpha_r \mathcal{I}(h_s = h_r)$ for $r = 1, \ldots, t$
   (c) $r^* = \operatorname{argmin}_{i \in \mathcal{I}} \gamma_s$, where $\mathcal{I} = \{i \mid i \in \{1, \ldots, t-1\} \text{ and } \underline{\alpha}_i > 0\}$.
   (d) **if** $\gamma_t - C < C - \tilde{\gamma}_{r^*}$ **then** $h_t = h_{r^*}$ and $\underline{\alpha}_t = \underline{\alpha}_{r^*}$ **else** $\underline{\alpha}_t = 0$
   (e) Find hypothesis coefficient by solving $\alpha_t = \operatorname{argmin}_{\alpha_t \geq -\underline{\alpha}_t} \mathrm{F}_{\beta^{(t)}}(\boldsymbol{\alpha}^{(t)})$
   (f) Let $f^{(t)} = f^{(t-1)} + \alpha_t h_t$.
   (g) Update weights $d_n^{(t+1)} = \dfrac{\exp([1 - y_n f^{(t)}(\mathbf{x}_n)]/\beta^{(t)})}{1 + \exp([1 - y_n f^{(t)}(\mathbf{x}_n)]/\beta^{(t)})}$.
   (h) **if** $\left|\sum_{n=1}^{N} d_n^{(t)} h_t(\mathbf{x}_n) - C\right| < \beta^{(t)}$, **then** $\beta^{(t+1)} := \texttt{next}(\beta^{(t)})$ **else** $\beta^{(t+1)} = \beta^{(t)}$

ν-LP-Regression



Input: - N examples $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$
- Hypotheses $h_1, \ldots, h_J$
- Regularization parameters C, ν

$$\text{Minimize } \frac{1}{N}\|\boldsymbol{\alpha}\|_1 + \frac{C}{N}\sum_{n=1}^{N}\xi_n + C\nu\epsilon \text{ such that}$$

$$* \; |f_{\boldsymbol{\alpha}}(\mathbf{x}_n) - y_n| \leq \epsilon + \xi_n$$
$$* \; \xi_n \geq 0$$
$$* \; \epsilon \geq 0$$

9. Applications of Boosting





Hill Inlet, Whitsunday Island

Main Sources:

- Schapire 2002
- Meir & Rätsch 2003

Practical Issues

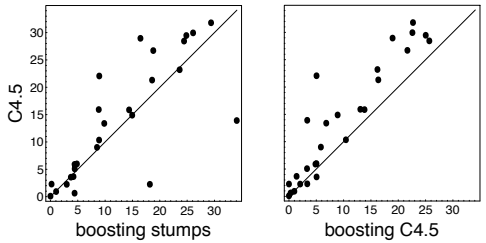


Advantages:	**Difficulties:**
A general meta-algorithm - use any 'reasonable' weak learner	Not clear how to incorporate prior knowledge effectively
Single parameter to be tuned (# iterations) - in principle	Regularization often essential - best strategy unclear
Fast and easy to program	The best choice 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

	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

⋆ Highly competitive with state-of-the-art SVM classifier

Source: Rätsch *et al.* 2001

Other Applications



Some examples:

Text classification	Schapire and Singer - Used stumps with normalized term frequency and multi-class encoding
OCR	Schwenk and Bengio (neural networks)
Natural language Proc.	Collins; Haruno, Shirai and Ooyama
Image retrieval	Thieu and Viola
Medical diagnosis	Merle *et al.*
Fraud Detection	Rätsch & Müller 2001
Drug Discovery	Rätsch, Demiriz, Bennett 2002
Elect. Power Monitoring	Onoda, Rätsch & Müller 2000

Fuller list: Schapire's 2002, Meir & Rätsch 2003 review