Boosting and Convex Optimization

Gunnar Rätsch

Fraunhofer FIRST, Berlin

Max-Planck-Institute for biol. Cyb., Tübingen

http://ida.first.fhg.de/~raetsch

Overview

- Motivation
- Leveraging as Greedy Optimization
- Boosting and Large Margins
- Boosting vs SVMs
- Improving Robustness & Extensions

Motto: How can we explain boosting using convex optimization?

Electronic copy of the slides can be downloaded from my homepage:
http://ida.first.fhg.de/~raetsch

Understanding the Optimization Objective

Machine learning principle:

- minimize regularized empirical loss

$$L_R[f] = \frac{1}{N}\sum_{n=1}^{N} l(f(\mathbf{x}_n), y_n) + \mathbf{P}[f]$$

- then learning theory can give guarantees:

$$\int l(f(\mathbf{x}), y) dp(\mathbf{x}, y) \leq \frac{1}{N}\sum_{n=1}^{N} l(f(\mathbf{x}_n), y_n) + \mathcal{O}(\mathbf{P}[f])$$

$\Rightarrow$ important what loss function is minimized!

Without knowing the loss function, it is hard to understand what the algorithm actually does.

Understanding and Modifying Boosting

- What loss function is minimized by AdaBoost?
- How can the algorithm be changed to minimize other loss functions?
- How fast does it converge?
- What is the regularizer $\mathbf{P}$?
- How is AdaBoost connected to maximal margins & linear Programming?
- How does Boosting compare to Support Vector Machines?
- How can we design new algorithms?

2. Boosting as Greedy Optimization

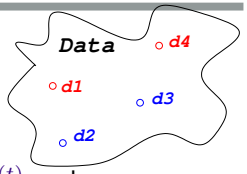


Hill Inlet, Whitsunday Island

Main Sources:

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

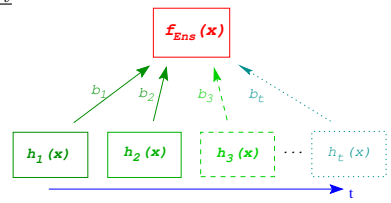
AdaBoost Algorithm



Input: N examples $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$
Initialize: $d_n^{(1)} = 1/N$ $(n = 1 \ldots N)$
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 training error $\epsilon_t = \sum_{n=1}^{N} d_n^{(t)} \mathrm{I}(y_n \neq h_t(\mathbf{x}_n))$
3. compute hypothesis weight $\alpha_t = \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$
4. update example distribution

$$d_n^{(t+1)} = \exp\left(-y_n \sum_{r=1}^{t} \alpha_r h_r(\mathbf{x}_n)\right)$$



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

Loss Function for AdaBoost

- AdaBoost stepwise minimizes a function

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

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

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

- 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 function $\mathrm{G} : \mathbf{R}^N \to \mathbf{R}$, 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

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

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

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

- 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)

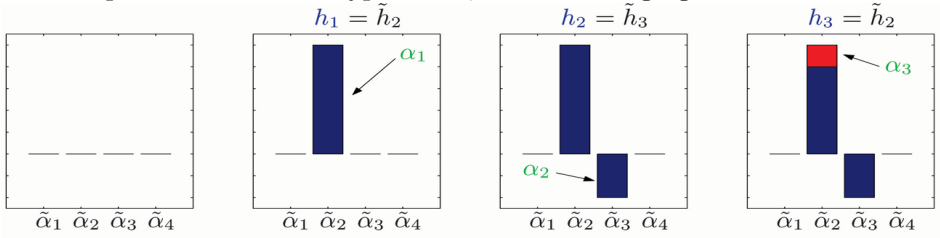
$$\mathrm{G}[f_{\boldsymbol{\alpha}}] = \sum_{n=1}^{N} \log\left(1 + \exp\{1 - y_n f_{\boldsymbol{\alpha}}(\mathbf{x}_n)\}\right) \quad d_n = \frac{\exp(1 - y_n f_{\boldsymbol{\alpha}}(\mathbf{x}_n))}{1 + \exp(1 - y_n f_{\boldsymbol{\alpha}}(\mathbf{x}_n))}$$

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 $\tilde{\alpha}$-coefficient space $\mathbf{R}^J$ to minimize $\mathrm{G}[f_{\tilde{\boldsymbol{\alpha}}}]$ stepwise.**
- 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:

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

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).

A 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

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

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

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

Dual Optimization – Leveraging

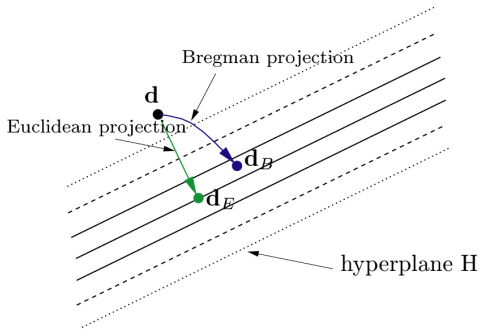
Define generalized "Distance" (Bregman Divergence):

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

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

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

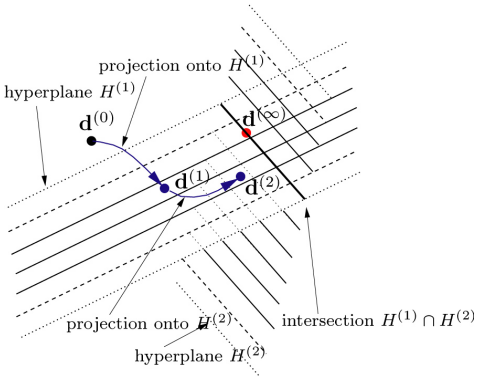
$$\text{with} \quad \sum_{n=1}^N d_n y_n h_t(\mathbf{x}_n) = 0$$



Bregman Algorithm and Leveraging

Bregman Algorithm:

– Iterate projections



Leads to Dual solution of:

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

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

Primal Problem:

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

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

Digestion

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")

3. Boosting and Large Margins

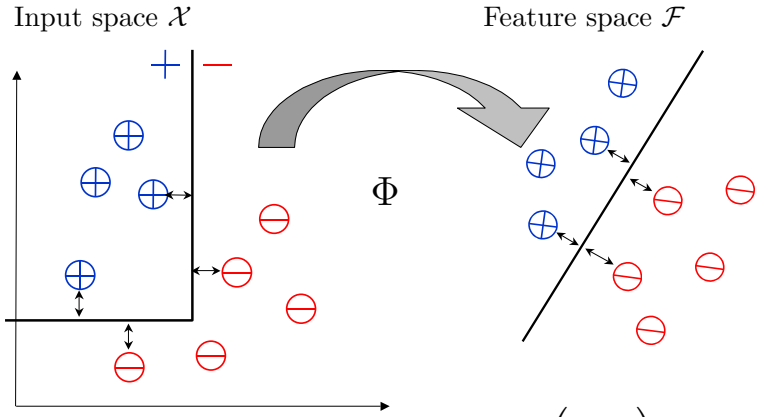


Hill Inlet, Whitsunday Island

Main Sources:

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

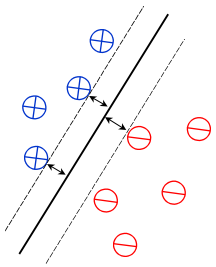
Large Margin and Linear Separation



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?

⇒ finds a solution with large, but not maximal margin

⇒ 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* 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

The AdaBoost$_\varrho$ Algorithm (Breiman 1997)

1. **Input:** $S = \langle(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\rangle$, No. Iterations 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 hypothesis $h_t : \mathbf{x} \mapsto [-1, 1]$
   (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 weights: $d_n^{t+1} = d_n^t \exp\left\{-\alpha_t y_n h_t(\mathbf{x}_n)\right\}/Z_t$, s.t. $\sum_{n=1}^{N} d_n^{t+1} = 1$.
4. **Output:** $f(\mathbf{x}) = \sum_{t=1}^{T} \frac{\alpha_t}{\sum_r \alpha_r} h_t(\mathbf{x})$

Digestion

Observe:

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

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?

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

- 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-M.
$\kappa_\beta(t) = \beta\exp(-t/\beta)$ $\rightsquigarrow$ 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-M.

$\kappa_\beta(t) = \beta \exp(-t/\beta)$ $\rightsquigarrow$ 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!**

4. Boosting vs SVMs



Hill Inlet, Whitsunday Island

Main Sources:

- Schapire, Freund, Bartlett & Lee 1998
- Rätsch, Mika, Schölkopf, Müller 2001

SVM's Mathematical Program

The SVM minimization of

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

reformulate as maximization of the margin ρ

$$\begin{aligned} \max_{\mathbf{w} \in \mathcal{F}_\Phi, \rho \in \mathbf{R}_+} \quad & \rho \\ \text{subject to} \quad & y_n \sum_{j=1}^{D} w_j \Phi_j(\mathbf{x}_n) \geq \rho \quad \text{for} \quad n = 1, \ldots, N \\ & \|\mathbf{w}\|_2 = 1, \end{aligned} \tag{3}$$

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

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

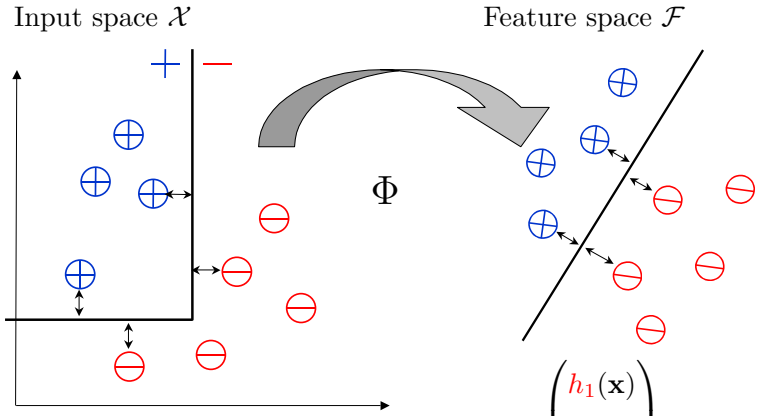
- base hypotheses h_t produced by the base learning algorithm.

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

$$\begin{aligned} \max_{\mathbf{w}\in\mathbf{R}^J, \rho\in\mathbf{R}_+} \quad & \rho \\ \text{subject to} \quad & y_n \sum_{j=1}^{J} w_j h_j(\mathbf{x}_n) \geq \rho \quad \text{for} \quad n = 1, \ldots, N \\ & \|\mathbf{w}\|_1 = 1, \end{aligned} \tag{4}$$

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

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

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

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 $r(\|\mathbf{w}\|_2) = \lambda\|\mathbf{w}\|_2$, $\lambda > 0$).

$$\min_{\mathbf{w}\in\mathcal{F}_\Phi} \sum_{n=1}^{N} c(y_n, \underbrace{f_{\mathbf{w}}(\mathbf{x}_n)}_{\langle \mathbf{w}, \Phi(\mathbf{x}_n)\rangle}) + r(\|\mathbf{w}\|_2) \tag{5}$$

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 (5) 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 Algorithms

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

$$\min_{\mathbf{w}\in\mathcal{F}_\mathcal{H}} \sum_{n=1}^{N} c(y_n, \underbrace{f_{\mathbf{w}}(\mathbf{x}_n)}_{\sum_{h\in\mathcal{H}} w_h h(\mathbf{x}_n)}) + r(\|\mathbf{w}\|_1) \tag{6}$$

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 (6) 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 → 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 learner
- large margins (for two-class classification)
- algorithms for solving convex optimization problems
  - reweighting is optimization trick
  - Barrier minimization (exponential) or column generation
- sparseness in "Feature Space" allows efficient algorithms

5. Improving Robustness & Extensions



Hill Inlet, Whitsunday Island

Main Sources:

- Rätsch et al. 1998, 2000, 2001

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 (Bennet et al., 2000). . .
- prove the convergence using tools from num. optimization

underlying opt. problem	Construct the new Algorithm using		
	AdaBoost	Barrier Meth.	CG-AdaBoost
LP-Classification	AdaBoost*		CG-AdaBoost
ν-LP-Classification	ν-Arc	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	+	+
. . .	. . .	. . .	. . .

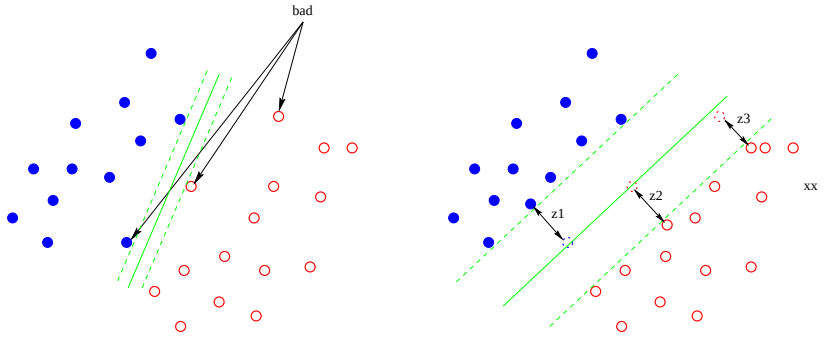
AdaBoost with Soft Margins

- 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



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 $\boldsymbol{\alpha}$:

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

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

Result: Empirical margin error is about ν

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

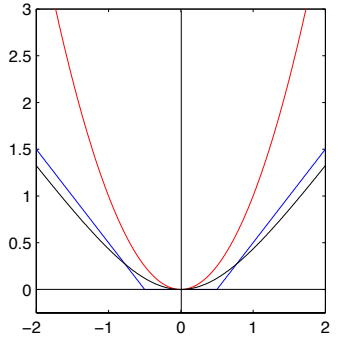
ν-LP-Regression

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

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

such that

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



Conclusion

Boosting ...

... is a special instance of coordinate descent methods

... can maximize the margin and solves a linear program

... has connections to barrier optimization

... is essentially the same as SVMs (different regularizer though)

... prefers sparse solutions

... can be easily extended to many other problem settings

- Soft-Margins
- Multi-Class
- Regression
- Novelty Detection
- (any linear algorithm)