# 情報意味論（12）Boosting

慶應義塾大学理工学部
櫻井 彰人

## 競馬で当てるには？

- 予想屋（ではなく専門家に）訊く
- 仮定:
  - 専門家であっても、極めて正確な予測規則を作成することはできない
  - けれども、どんな事例であっても、それを聞けば、ランダム以上の予測をする予測規則を作成することはできる
- よく当たる予測規則を作る方法はあるか？

## アイデア

- 専門家に経験則を作ってもらい、それを集める（統合する）。
- 統合方法その１
  - 一気に作ってもらい、例えば、多数決を取る
- 統合方法その２
  - ある人の経験則を使ってみる。
  - その人の経験則が失敗する事例を集め、別の人の経験則を適用する
  - そして、、、

実は、これがうまくいくのです。おまけに、専門家でなくても弱学習アルゴリズム “weak” learning algorithm でよい

## 課題

- （教えを請うときには）どのレースを選べばよいのか?
  - 前の人が失敗したレースを選ぶのだが、その中でも
  - 最も難しいレースに集中する
    (それまでの経験則では最も外れているレースのこと)
- これらの経験則をどう統合すれば、一つの予測規則にできるのか?
  - 経験則の（重み付き）多数決をとる

ただ、学習事例を人によって変えてしまったので、何か工夫が必要そうな気がする。

## ベイズ最適な分類器: 例

補足（復習含む）

- 3個の仮説からなる空間を考える

$$P(h_1 \mid D) = 0.4;\ P(h_2 \mid D) = 0.3;\ \ P(h_3 \mid D) = 0.3 \quad \Rightarrow h_{MAP} = h_1$$

- 新しい未知の事例に対し、次を仮定しよう
  $h_1(x) = 1$　　$h_2(x) = 0$　　$h_3(x) = 0$
- この場合,
  $P(f(x) = 1) = 0.4$　　$P(f(x) = 0) = 0.6$　　しかし　$h_{map}(x) = 1$
- （最もありうる仮説ではなく）最もありうる分類結果を、すべての仮説の予測を結合して、得たい。
- 各仮説には事後確率による重みづけをすればよさそう
  (うまくいくにはいくつかの仮定が必要)

## ベイズ最適な分類器: 例(2)

- Vを可能な分類結果としよう

$$P(v_j \mid D) = \sum_{h_i \in H} P(v_j, h_i \mid D)$$
$$= \sum_{h_i \in H} P(v_j \mid h_i, D) P(h_i \mid D) = \sum_{h_i \in H} P(v_j \mid h_i) P(h_i \mid D)$$

- ベイズ最適な分類:

$$v = \operatorname{argmax}_{v_j \in V} P(v_j \mid D) = \operatorname{argmax}_{v_j \in V} \sum_{h_i \in H} P(v_j \mid h_i) P(h_i \mid D)$$

- 先ほどの例では:

$$P(1 \mid D) = \sum_{h_i \in H} P(1 \mid h_i) P(h_i \mid D) = 1 \times 0.4 + 0 \times 0.3 + 0 \times 0.3 = 0.4$$
$$P(0 \mid D) = \sum_{h_i \in H} P(0 \mid h_i) P(h_i \mid D) = 0 \times 0.4 + 1 \times 0.3 + 1 \times 0.3 = 0.6$$

- 最適な予測は、勿論、0.

## 背後にある仮定 (1)

- これはいつもうまくいくのであろうか?
- 考えるヒント: 仮説の線形結合である
- (医師の診断だとしよう)仮に、何人かが同じ医局だとしたら? 出身が同じ、インターン先が同じ、、、
- 仮に、何人かは、webサイトでしかも同じ医師の意見に基づくものであったら?
- 一般に、医師間(仮説間)に無視できない相関があると、それは、相互に依存する冗長性があることを意味する
- こうした意見は、過剰に重みづけすることになる
- ベイズ最適は、仮説空間に関する周辺化に見える

$$v = \operatorname{argmax}_{v_j \in V} P(v_j \mid D) = \operatorname{argmax}_{v_j \in V} \sum_{h_i \in H} P(v_j \mid h_i) P(h_i \mid D)$$

## 背後にある仮説(2)

- うまくいくとはどういうことであろうか?
- |D| が無限に増大するとき、ベイズ最適な分類結果は最良な答えに収束すべきである。そうなるか?

- |D| $\to \infty$ となる時、荷重ベクトル w の動きを考えてみよう
- "最良な答え" は他の何よりもよいということ
- 仮に、同点はないものとしよう(そうすれば最良が存在する)
- 最良の w は、一つの1(最良のh用)を除いて、全部0.
- 一般に、これは起こるのか? なぜ? 起こるようにするには、どうしたらよいのか?

## ベイズ最適な分類器

$$v = \operatorname{argmax}_{v_j \in V} P(v_j \mid D) = \operatorname{argmax}_{v_j \in V} \sum_{h_i \in H} P(v_j \mid h_i) P(h_i \mid D)$$

- 追加的情報がない場合、ベイズ最適以上に良くはできない.
- ベイズ最適な分類器は、一般に、仮説空間 H の要素ではない(!)
- ベイズ最適分類器は、「独立性」(冗長性がないこと)に関し、強い仮説「仮説の誤り間に相関はない」をおいている
- 「誤りに相関はない」 – 一種のナイーブベイズ。ただし、仮説空間 H で。
- もう一つの強い仮定: ある h∈H は正しい; "agnostic"な学習ではない.
- 専門の統合(combining expertise); 専門家の線形結合(アンサンブル)を見出す

Agnostic: 目的関数に何に仮定も設けない

## Gibbs 分類器

- Bayes 最適分類器は訓練コストが高い
- すべての h∈H について事後確率を計算する必要がある
- 空間 H 上で現在の事後分布に従い、訓練と分類を行う
- 訓練:
  - H 上のある事前分布を仮定する
  - (更新される)この分布に従い、ある仮説 h を選ぶ
  - 仮説 h に基づいて分類する
  - 仮説 h の事後確率を更新する
  - 繰り返す
- 訓練データを何回も; 複数個の仮説 h を一度に引いて更新することも可能
- 確率が不当に高い h が(相対的に)多く引かれる
  - 訓練誤りが多いと、その h の事後確率が下がる.
    正規化を通じて、他の仮説の事後確率が上がる
  - より正確な仮説が引かれ事後確率が上がる傾向にある
- 収束する
- 最悪時の期待誤り率は、Bayes最適分類器のそれの2倍以下

## Bagging: Bootstrap AGGregatING

- 分散の減少を狙う
- 訓練データによる性能のばらつきが大きい分類器がある
  - 統計的信頼度が低い過剰適応・過学習(overfitting) をする
  - データ中のもっともらしい(本当でない)パターンを発見する
- 多数の分類器を "平均" しよう
- Bootstrap: データのリサンプリング(再標本化)
  - 複数の訓練データ集合を生成する
    - もとの訓練データをリサンプル(再標本化)する
    - 復元抽出である
  - 得られたデータ集合はそれぞれ異なる「もっともらしい」パターンを持つ
- 複数個の分類器を学習する
  - 「もっともらしい」パターンは相関しない
  - 根底にある真のパターンは多くのデータ集合に共通であろう
- 分類器の結合: 新テストサンプルのラベルは、分類器の多数決で決める

## Bagging

- ロジスティック回帰も overfit しうる. 例えば、
  - 線形分離可能な場合
  - 非常に急峻な関数の確率分布でフィットする
- Baggingを考えてみよう
- 多くの属性（次元）があるとき
  - ある次元で急峻になることは珍しくなかろう
  - しかし、システマティックではない（偶然の産物であろう）
  - であれば、平均することにより、その悪影響を減少させることができる
- 訓練したしかし「ランダムな」分類器の集合を生成する
  - 時に、リサンプリングさえ必要ではない – 反復アルゴリズム
  - リサンプリングにより訓練データ集合の情報が減少しよう
  - 正しく行えば, それの影響は小さい
  - 時にデータの順序を入れ替えるだけで十分なこともある
  - そうすれば、訓練データにある情報や証拠が減少することはない
- 決定「切り株」（stumps）
  - 決定木、しかし深さ1レベル（分岐1回）
  - 時には、少数のレベルを用いる。特徴間にあるある非線形性をとらえるため
- 決定株の bagging
  - しばしば、結構うまくいく
  - 最初に試すべきもののーつ

## Boosting: 弱い学習器を強くする

- 弱い学習器（ weak learner ）:
  - 重み付訓練データが与えられれば、ある仮説を生成する
  - それは高い確率で
  - 少なくともランダムな推定よりは「少し」よく正確である,
  - どのようなデータ分布に対しても.
- 訓練データ集合 Z と仮説空間 H が与えられたとする
- 弱い学習器の線形結合を学習する
- 各繰り返しにおいて, 新しい仮説(分類器) $h_i \in H$ を追加し,
- 重み付 Z に対する分類器の性能に従って $h_i$ に重みを付ける
- 新 h は、同じ Z（しかし再重みづけ：難しい $z_j$ は重く）で訓練
- 注：2つの重み – 訓練データの重みと弱い学習器の重み
- 各分類器の重み付投票により分類
- 「強い」学習器が作れる：任意に高い精度にできる

## Boosting

補足終わり

- 一種の「メタ」学習アルゴリズム
- どんな「弱い」学習器も「boost」できる
- Boosting によれば欲しいだけの高い精度が得られる
- 訓練データが完全に分類できたとしても、その後訓練を継続することにより、性能の向上が図れる
- 過学習（overfit）しないように見える
- 外れ値（outlier）やノイズに、過剰に敏感になりうる
- ポピュラーかつ実用的なのが AdaBoost (adaptive boosting)

## Boosting

- boosting = 複数個の低精度の経験則を高精度な予測規則に変換する一般的方法
- 機械学習では:
  - 弱(weak)学習アルゴリズム（誤差 $\leq 1/2-\gamma$ なる仮説（分類規則）を常に見出すことができる）が与えられたとき
  - boosting アルゴリズムは、誤差 $\leq \varepsilon$ なる単一の仮説を構成することができる（ことが証明できる）
  - 理論によれば、しばしば、汎化能力はよい

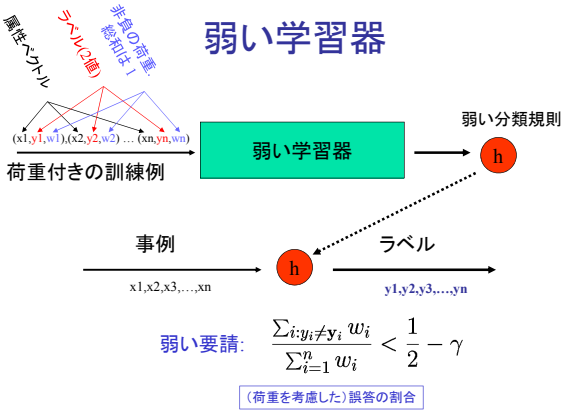
## 目次

- boosting 入門 (AdaBoost)
- 訓練誤差の解析
- マージンの理論に基づく、汎化誤差の検討
- 結果例

以下のスライドは、主に、下記論文に基づく
Robert E. Schapire. **The boosting approach to machine learning: An overview.** In D. D. Denison, M. H. Hansen, C. Holmes, B. Mallick, B. Yu, editors, *Nonlinear Estimation and Classification*. Springer, 2003.

Robert E. Schapire, Yoav Freund, Peter Bartlett and Wee Sun Lee. **Boosting the margin: A new explanation for the effectiveness of voting methods**. *The Annals of Statistics*, 26(5):1651-1686, 1998.

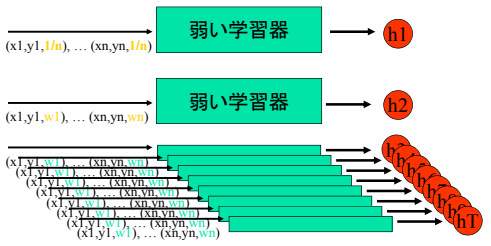
## 弱い学習器




弱い要請: $\frac{\sum_{i:y_i \neq \mathbf{y}_i} w_i}{\sum_{i=1}^{n} w_i} < \frac{1}{2} - \gamma$

（荷重を考慮した）誤答の割合

## ブースティングの過程

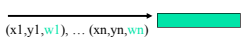



最終規則: Sign[ α1 h1 + α2 h2 + … + αT hT ]

## AdaBoost [Freund & Schapire '97]

- 2値ラベル y = -1,+1
- 出力: Sgn[$\Sigma_t \alpha_t h_t(x)$] （これは多数決）
  （微妙な違いに注意）
- margin(x,y) = y [$\Sigma_t \alpha_t h_t(x)$]
- 次の値を最小化するように $h_t$ と $\alpha_t$ を選ぶ（まず $h_t$ を選び、次に $\alpha_t$ を選ぶ）

  $\Sigma_{(x,y)}$ exp (-margin(x,y))
  = $\Sigma_{(x,y)}$ exp (- y [$\Sigma_t \alpha_t h_t(x)$])
  （陽に解けるので、$\alpha_t$ の最小化問題の解の計算は簡単）




$h_t$ はどう決めるのか？
$w_t$=$D_t$ と学習器で決める
$w_t$ はどう決めるのか？
$h_{t-1}$ の誤りから決める

## AdaBoost の計算手順

- $D_1(i) = 1/m$
- 学習により $D_t(\cdot) \Rightarrow h_t(\cdot)$

学習器に対する要請では、誤り率が0.5未満だが、最良のものを返す方がよいと考えられる。

- $\varepsilon_t = \Pr_{i\sim D_t}[h_t(x_i) \neq y_i] = \sum_{i:h_t(x_i)\neq y_i} D_t(i)$

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) > 0$$

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t}\cdot\begin{cases} e^{-\alpha_t} & \text{if } y_i = h_t(x_i) \\ e^{\alpha_t} & \text{if } y_i \neq h_t(x_i) \end{cases}$$

ただし、 $Z_t$ は $1 = \sum_{i=1}^{m} D_{t+1}(i)$ となるように定める

- $H_{\text{final}}(x) = \operatorname{sgn}\left(\sum_t \alpha_t h_t(x)\right)$

Yoav Freund and Robert E. Schapire.
A decision-theoretic generalization of on-line learning and an application to boosting.
Journal of Computer and System Sciences, 55(1):119-139, August 1997.

## Adaboost の主な性質

- あてずっぽう（正解確率1/2）に対する、弱い学習器の正解率差（正値）: $\gamma_1,\gamma_2,..,\gamma_T$ . その時 最終規則の 訓練誤差 は高々

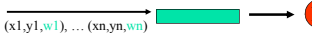
$$\exp\left(-2\sum_{t=1}^{T}\gamma_t^2\right) = \exp\left(-2\sum_{t=1}^{T}(1/2-\varepsilon_t)^2\right)$$

訓練誤りを計算するときの、訓練データの荷重は、初期荷重
（変更した荷重値は、あくまでも、学習のためであるから）

## 再掲: AdaBoost [Freund & Schapire '97]

- 2値ラベル y = -1,+1
- 出力: Sgn[$\Sigma_t \alpha_t h_t(x)$]
- margin(x,y) = y [$\Sigma_t \alpha_t h_t(x)$]
- 次の値を最小化するように $h_t$ と $\alpha_t$ を選ぶ（まず $h_t$ を選び、次に $\alpha_t$ を選ぶ）

  $\Sigma_{(x,y)}$ exp (-margin(x,y))
  = $\Sigma_{(x,y)}$ exp (- y [$\Sigma_t \alpha_t h_t(x)$])
  （陽に解けるので、$\alpha_t$ の最小化問題の解の計算は簡単）




$h_t$ はどう決めるのか？
$w_t$=$D_t$ と学習器が決める
$w_t$ はどう決めるのか？
$h_{t-1}$ の誤りから決める

## 最急降下法としての Adaboost

- 探索する、分類器の空間: "弱い仮説" の線形和のなす空間 $\Sigma_t \alpha_t h_t(x)$
- 当初の目標: 誤り数最小の超平面を見つける
  $\Sigma_{(x,y)}$ ( 1 - y Sgn[$\Sigma_t \alpha_t h_t(x)$] )/2
  = $\Sigma_{(x,y)}$ ( 1 + Sgn[ - y $\Sigma_t \alpha_t h_t(x)$] )/2
  - NP-hard な問題であることが知られている (d を当該空間の次元とするとき、d の多項式時間で動作するアルゴリズムが存在しないだろう)
- 妥協案: 指数損失関数で（誤り数関数を）代用して、軸毎の最急降下を用いる.
  $\Sigma_{(x,y)}$ exp (- y [$\Sigma_t \alpha_t h_t(x)$])

## 最小化: 定式化

- 判別関数の損失: $L(F(\cdot)) = \frac{1}{m}\sum_{i=1}^{m} \exp(-y_i F(x_i))$
- Adaboost の判別関数:

$$f(x) = \sum_t \alpha_t h_t(x) \qquad H_{\text{final}}(x) = \operatorname{sgn} f(x)$$

- $f(x)$ に新たに仮説 $h(x)$ を加えた関数

$$f(x) + ch(x)$$

の損失 $L(f(\cdot) + ch(\cdot))$ を最小化する $c$ を求めよう

## 損失関数最小化: 式変形

$$\begin{aligned}
&L(f(\cdot) + ch(\cdot)) \\
&= \frac{1}{m}\sum_{i=1}^{m} \exp(-y_i(f(x_i) + ch(x_i))) \\
&= \frac{1}{m}\sum_{i=1}^{m} \exp(-y_i f(x_i)) \exp(-y_i ch(x_i)) \\
&= L(f(\cdot))\sum_{i=1}^{m} \frac{\exp(-y_i f(x_i))}{\tilde{Z}} \exp(-y_i ch(x_i)) \\
&= L(f(\cdot))\sum_{i=1}^{m} \tilde{D}(i) \exp(-y_i ch(x_i)) \\
&= L(f(\cdot))\left(\sum_{i:y_i=h(x_i)} \tilde{D}(i)\exp(-y_i ch(x_i)) + \sum_{i:y_i\neq h(x_i)} \tilde{D}(i)\exp(-y_i ch(x_i))\right) \\
&= L(f(\cdot))\left(\exp(-c)\sum_{i:y_i=h(x_i)} \tilde{D}(i) + \exp(c)\sum_{i:y_i\neq h(x_i)} \tilde{D}(i)\right) \\
&= L(f(\cdot))(\exp(-c)(1-\varepsilon) + \exp(c)\varepsilon)
\end{aligned}$$

$$\tilde{D}(i) = \frac{\exp(-y_i f(x_i))}{\tilde{Z}}$$

$$\tilde{Z} = \sum_{i=1}^{m} \exp(-y_i f(x_i))$$

$$\varepsilon = \sum_{i:y_i\neq h(x_i)} \tilde{D}(i)$$

$$L(f(\cdot)) = \frac{1}{m}\sum_{i=1}^{m} \exp(-y_i f(x_i)) = \frac{\tilde{Z}}{m}$$

なお、 $L(ch(\cdot)) = \exp(-c)(1-\varepsilon) + \exp(c)\varepsilon$ とおく

## 損失関数最小化

$$\varepsilon = \sum_{i:y_i\neq h(x_i)} \tilde{D}(i)$$

$$\tilde{D}(i) = \frac{\exp(-y_i f(x_i))}{\tilde{Z}}$$

$$\tilde{Z} = \sum_{i=1}^{m} \exp(-y_i f(x_i))$$

- 導関数

$$\begin{aligned}
\frac{d}{dc}L(f(\cdot) + ch(\cdot)) &= \frac{d}{dc}L(f(\cdot))(\exp(-c)(1-\varepsilon) + \exp(c)\varepsilon) \\
&= L(f(\cdot))(-\exp(-c)(1-\varepsilon) + \exp(c)\varepsilon) \\
&= L(f(\cdot))\exp(-c)\varepsilon(-(1-\varepsilon)/\varepsilon + \exp(2c))
\end{aligned}$$

より、 $c = \frac{1}{2}\ln\frac{1-\varepsilon}{\varepsilon}$ のとき $L(f(\cdot) + ch(\cdot)) = L(f(\cdot))\,2\sqrt{(1-\varepsilon)\varepsilon}$

$L(ch(\cdot)) = 2\sqrt{(1-\varepsilon)\varepsilon}$

$h$ はなんでもよいのだが、$c > 0,\ 2\sqrt{(1-\varepsilon)\varepsilon} < 1,\ i.e.\ \varepsilon < 1/2$ となるべし

- すなわち

$$f(x) = \sum_t \alpha_t h_t(x)$$

$$\tilde{D}(i) = \frac{\exp(-y_i f(x_i))}{\tilde{Z}}, \quad \text{where } \tilde{Z} = \sum_{i=1}^{m} \exp(-y_i f(x_i))$$

$$\varepsilon = \sum_{i:y_i\neq h(x_i)} \tilde{D}(i)$$

$$\alpha = \frac{1}{2}\ln\frac{1-\varepsilon}{\varepsilon}$$

$$f_{new}(x) = \sum_t \alpha_t h_t(x) + \alpha h(x)$$

$c<0$ なら $-h$ を用いる. $\varepsilon = 1/2$ はダメ

$h$ に自由度があるとはいえ、損失関数 $L$ がより小さくなるためには、$\varepsilon$が小さい $h$ の方がよい

## 逐次的アルゴリズムに変換(1)

$$f(x) = \sum_t \alpha_t h_t(x)$$

$$\tilde{D}(i) = \frac{\exp(-y_i f(x_i))}{\tilde{Z}},$$

$$\tilde{Z} = \sum_{i=1}^{m} \exp(-y_i f(x_i))$$

$$\varepsilon = \sum_{i:y_i\neq h(x_i)} \tilde{D}(i)$$

$$\alpha = \frac{1}{2}\ln\frac{1-\varepsilon}{\varepsilon}$$

$$f_{new}(x) = \sum_t \alpha_t h_t(x) + \alpha h(x)$$

$$f_{t-1}(x) = \sum_{s=1}^{t-1} \alpha_s h_s(x)$$

$$D_t(i) = \frac{\exp(-y_i f_{t-1}(x_i))}{\tilde{Z}},$$

where $1 = \sum_{i=1}^{m} D_t(i)$

$D_t(\cdot) \Rightarrow h_t(\cdot)$

$$\varepsilon_t = \sum_{i:y_i\neq h_t(x_i)} D_t(i)$$

$$\alpha_t = \frac{1}{2}\ln\frac{1-\varepsilon_t}{\varepsilon_t}$$

$$f_t(x) = \sum_{s=1}^{t} \alpha_s h_s(x)$$

$$\frac{L(f_t(\cdot))}{L(f_{t-1}(\cdot))} = 2\sqrt{(1-\varepsilon_t)\varepsilon_t}$$

$h_t$ に自由度があるとはいえ、$D_t$ によって定まる $\varepsilon_t$ がより小さくなる方がよい。すなわち、$D_t$ を参照しながら、$h_t$ を定めるべし

$D_t(\cdot) \Rightarrow h_t(\cdot)$

$$\varepsilon_t = \sum_{i:y_i\neq h_t(x_i)} D_t(i)$$

$$\alpha_t = \frac{1}{2}\ln\frac{1-\varepsilon_t}{\varepsilon_t}$$

$$f_t(x) = \sum_{s=1}^{t} \alpha_s h_s(x)$$

$$D_{t+1}(i) = \frac{\exp(-y_i f_t(x_i))}{\tilde{Z}_t},$$

where $1 = \sum_{i=1}^{m} D_{t+1}(i)$

## 逐次的アルゴリズムに変換(2)

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

学習により $D_t(\cdot) \Rightarrow h_t(\cdot)$

$$\varepsilon_t = \sum_{i:h_t(x_i)\neq y_i} D_t(i) = \Pr_{i\sim D_t}[h_t(x_i) \neq y_i]$$

$$\alpha_t = \frac{1}{2}\ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) > 0$$

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t}\cdot\begin{cases} e^{-\alpha_t} & \text{if } y_i = h_t(x_i) \\ e^{\alpha_t} & \text{if } y_i \neq h_t(x_i) \end{cases}$$

$$1 = \sum_{i=1}^{m} D_{t+1}(i)$$

$$H_{\text{final}}(x) = \operatorname{sgn}\left(\sum_t \alpha_t h_t(x)\right)$$

$D_t(\cdot) \Rightarrow h_t(\cdot)$

$$\varepsilon_t = \sum_{i:y_i\neq h_t(x_i)} D_t(i)$$

$$\alpha_t = \frac{1}{2}\ln\frac{1-\varepsilon_t}{\varepsilon_t}$$

$$f_t(x) = \sum_{s=1}^{t} \alpha_s h_s(x)$$

$$D_{t+1}(i) = \frac{\exp(-y_i f_t(x_i))}{\tilde{Z}_t},$$

where $1 = \sum_{i=1}^{m} D_{t+1}(i)$

$$\exp(-y_i f_t(x_i)) = \exp(-y_i f_{t-1}(x_i))\cdot\exp(-y_i \alpha_t h_t(x_i))$$

$$D_{t+1}(i) = \frac{\tilde{Z}_{t-1} D_t(i)\cdot\exp(-y_i\cdot\alpha_t\cdot h_t(x_i))}{\tilde{Z}_t} = \frac{D_t(i)}{\tilde{Z}_t/\tilde{Z}_{t-1}}\cdot\exp(-y_i\cdot\alpha_t\cdot h_t(x_i))$$

## 訓練誤差

- 定理 [Freund and Schapire '97]:

$\varepsilon_t$ を $\frac{1}{2} - \gamma_t$ と書く. i.e. $\gamma_t = \frac{1}{2} - \varepsilon_t$

この時 $\text{training error}(H_{\text{final}}) \le \exp\left(-2\sum_t \gamma_t^2\right)$

従って、もし $\forall t: \gamma_t \ge \gamma > 0$ なら

$$\text{training error}(H_{\text{final}}) \le \exp\left(-2\gamma^2 T\right)$$

訓練誤差は、初期分布（一様分布）で考えている（それが与えられた問題だから）

- 注: AdaBoost は adaptive:
  - $\gamma$ や $T$ を事前に知っている必要はない

## 証明（レポート課題）

$$\mathrm{Error}_{\mathrm{Train}}(H_{\mathrm{final}}) = \frac{1}{m}\left(\sum_{i=1}^{m}(1 - y_i H_{\mathrm{final}}(x_i))/2\right)$$

$$= \leq = = \leq = \exp\left(-2\sum \gamma_t^2\right)$$

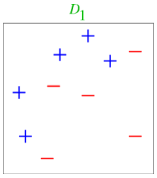
$$L(F(\cdot)) = \frac{1}{m}\sum_{i=1}^{m}\exp(-y_i F(x_i))$$

$$\frac{L(f_t(\cdot))}{L(f_{t-1}(\cdot))} = 2\sqrt{(1-\varepsilon_t)\varepsilon_t} = \sqrt{1-4\gamma_t^2}$$

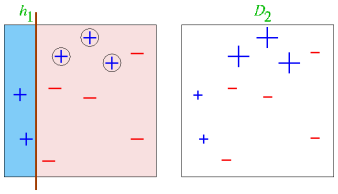
$$L(f_1(\cdot)) = \sqrt{1-4\gamma_1^2}$$

$$\exp(-x) \geq 1 - x \text{ for } x \geq 0$$

## トイ

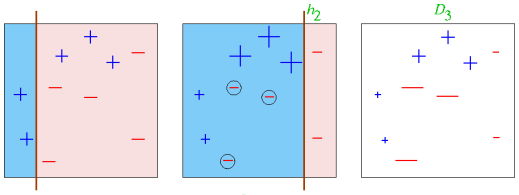



## 第一巡目



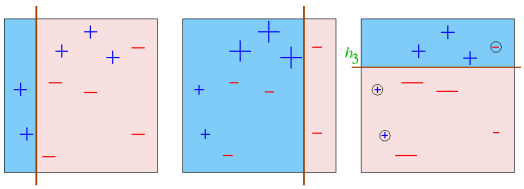

$\varepsilon_1 = 0.30$
$\alpha_1 = 0.42$

## 第二巡目



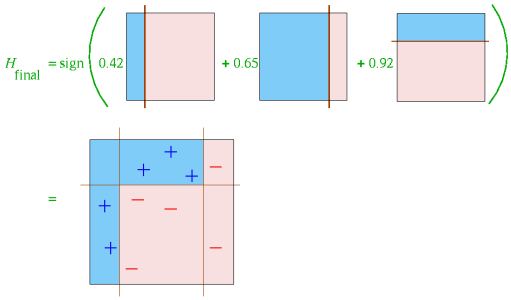

$\varepsilon_2 = 0.21$
$\alpha_2 = 0.65$

## 第三巡目




$\varepsilon_3 = 0.14$
$\alpha_3 = 0.92$

## 最終仮説



## Boosting Applet

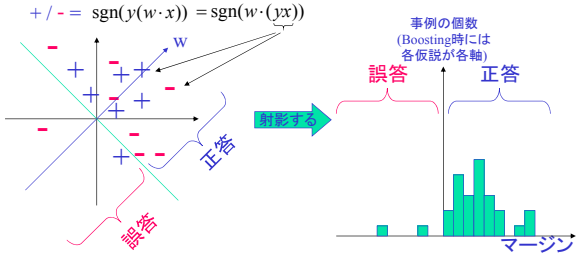
http://www.cse.ucsd.edu/~yfreund/adaboost/index.html

## マージンというもの

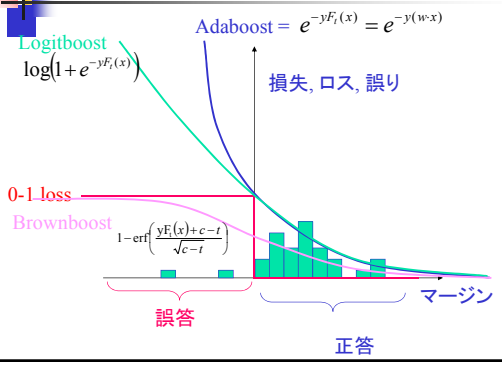
$x, w \in R^n; y \in \{-1,+1\}$　　予測 = $\mathrm{sgn}(w \cdot x)$

マージン = $y(w \cdot x)$

$+ / - = \mathrm{sgn}(y(w \cdot x)) = \mathrm{sgn}(w \cdot (yx))$




## 損失関数




## Boosting の形式化

- 所与の訓練データ集合 $X=\{(x_1,y_1),\ldots,(x_m,y_m)\}$
- $y_i \in \{-1,+1\}$ 事例 $x_i \in X$ に対する正しいラベル
- for $t = 1,\ldots,T$:
  - $\{1,\ldots,m\}$ 上の分布 $D_t$ を作成する
  - 弱仮説を見出す

    $h_t : X \to \{-1,+1\}$

    ただし $D_t$ 上で小さい誤差 $\varepsilon_t$ あり

    $\varepsilon_t = \Pr_{i \sim D_t}[h_t(x_i) \neq y_i] = \sum_{i:h_t(x_i) \neq y_i} D_t(i)$
- 最終仮説 $H_{\text{final}}$ を出力

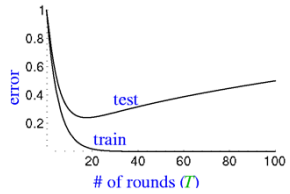
## 一度に一軸ごと

- Adaboost は指数損失関数に対して 最急降下法 を適用する
- 繰り返し一度につき, 一軸 ("弱い学習器")追加.
- 2進分類器 における弱学習= あてずっぽうよりちょっとよい学習器.
  - 回帰における弱学習 – 未解明.
- 事例に対する荷重 を用いて、弱学習器に降下方向を教える
- これによって実際に 計算 できるようになる

## 良い弱学習器とは?

- 弱学習器(達)は、
- 属性・ラベル間のありうる関係のほとんどと(弱い)相関が表現できるように、十分に柔軟でなければならない.
- 荷重つき訓練誤差を最小化する仮説の空間が全探索ができるくらい十分に小さくあるべき.
- 過学習とならないよう小さくあるべき.
- ラベルの予測値が非常に効率よく計算できるべき.
- "狭い専門家" であってよい – 入力空間の小さい部分空間内でのみ予測を行い, それ以外では 予測を控える (出力 0)としてよい

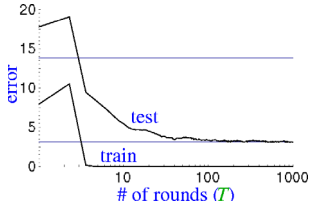
## 汎化誤差の解析




通常の期待 or 予想:

- 訓練誤差は、継続して、低下する(0になるかも)
- $H_{\text{final}}$ が複雑になりすぎると、テスト誤差は、増大する (オッカムの剃刀)

## ある実験結果 [Schapire et al. 98]

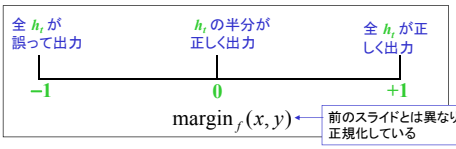



(boosting on C4.5 on "letter" dataset)

- 1,000 巡以降でもテスト誤差は増加しない
  - (C4.5を用いているため) ノード数の合計 ~2,000,000
- 訓練誤差が0となった後も、テスト誤差は減少を続ける
- オッカムの剃刀のいう、単純な規則がよいというのは、誤り

http://www.cs.princeton.edu/courses/archive/fall05/cos402/readings/boost-slides.pdf

## (正規化)マージンからみると




前のスライドとは異なり、正規化している

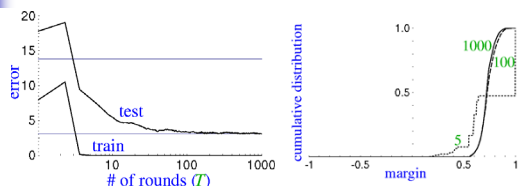
アイデア: 分類の信頼度 (マージン) を考えよう:

- まず下記に注意

$$H_{\text{final}}(x) = \text{sgn}(f(x)) \qquad \frac{f(x)}{\sum_t |\alpha_t|} = \frac{\sum_t \alpha_t h_t(x)}{\sum_t \alpha_t} \in [-1,1]$$

- 定義: $(x,y)$ のマージン: $\text{margin}_f(x,y) = \dfrac{y \cdot f(x)}{\sum_t |\alpha_t|}$

## マージンの累積分布 [Schapire et al. 98]




| epoch | 5 | 100 | 1000 |
|---|---|---|---|
| training error | 0.0 | 0.0 | 0.0 |
| test error | 8.4 | 3.3 | 3.1 |
| %margins≤0.5 | 7.7 | 0.0 | 0.0 |
| Minimum margin | 0.14 | 0.52 | 0.55 |

## Boosting はマージンを最大化する

- 次の損失関数を最小化することであった

$$\sum_i e^{-y_i f(x_i)} = \sum_i e^{-y_i \sum_t \alpha_t h_t(x_i)} = \sum_i e^{-\text{margin}_f(x_i,y_i) \sum_t \alpha_t}$$

$(x_i,y_i)$ のマージンに比例

## マージンに基づく解析

汎化誤差を訓練事例のマージンの関数で抑える:

$$\text{error} = \Pr[\text{margin}_f(x,y) \le 0]$$

θ が大きくとれれば、これは小さくなる

上手く学習してこのような学習サンプルがないように(またはは少ないように)する。

$$\le \hat{\Pr}[\text{margin}_f(x,y) \le \theta] + \tilde{O}\left(\sqrt{\frac{\text{VC}(H)}{m\theta^2}}\right)$$

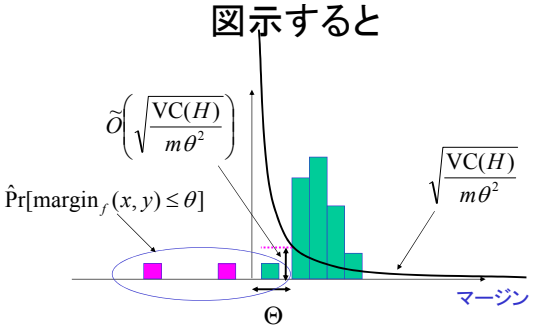
(Pr はデータ空間で、P^r は訓練データ上)
(Hが有限なら VC(H) ~ log |H| )
(任意のθ>0 に対し、訓練事例分布上確率 1-δで成立)

$$O\left(\sqrt{\frac{d\log^2(m/d)}{m\theta^2} + \frac{\log(1/\delta)}{m}}\right)$$

- 訓練事例マージン大 ⇒ θが大きくとれる
- 上界は学習エポック数に依存しない
- boosting は、マージンが最小の事例に着目し、当該事例のマージンを増加させようとする

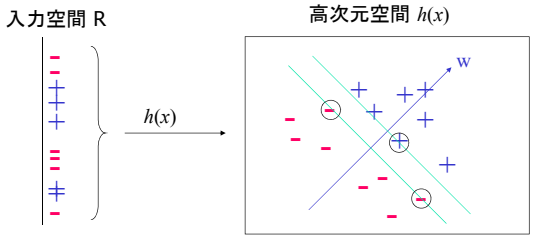
Robert E. Schapire, Yoav Freund, Peter Bartlett and Wee Sun Lee. **Boosting the margin: A new explanation for the effectiveness of voting methods.** *The Annals of Statistics*, 26(5):1651-1686, 1998.

## 図示すると




## SVM との関係

SVM: $x$ を高次元空間に写像して、線形分離する




## SVM との関係（続）

$$H(x) = \begin{cases} +1 & \text{if } 2x^5 - 5x^2 + x > 10 \\ -1 & \text{otherwise} \end{cases}$$

$$\vec{h}(x) = (1, x, x^2, x^3, x^4, x^5)$$
$$\vec{\alpha} = (-10, 1, -5, 0, 0, 2)$$

$$H(x) = \begin{cases} +1 & \text{if } \vec{\alpha} \cdot \vec{h}(x) > 0 \\ -1 & \text{otherwise} \end{cases}$$

## SVM との関係

- どちらもマージンを最大化する:

$$\theta \equiv \max_w \min_i \frac{y_i(\vec{\alpha} \cdot \vec{h}(x_i))}{\| \vec{\alpha} \|}$$

- SVM: $\| \vec{\alpha} \|_2$ ユークリッドノルム ($L_2$)
- AdaBoost: $\| \vec{\alpha} \|_1$ マンハッタンノルム ($L_1$)

- 最適化や PAC による上界と関係がでてくる

[Freund et al '98]

## AdaBoost の実用的価値

- かなり速い
- 単純かつ容易にプログラムできる
- チューニングパラメータは一個だけ ($T$)
- 事前知識不要
- 融通性: どんな分類器とも組合せ可能 (ニューラルネット, C4.5, …)
- 有効性が証明済み (弱学習器の存在は仮定する)
  - 発想の転換: 目標は、単に、random guessing よりよい仮説を見つければよいだけ
- はずれ値も見つける

## 注意点はある

- 性能は、データと当該弱学習器に依存
- AdaBoost が失敗するのは
  - 弱学習器が複雑すぎる (過学習)
  - 弱学習器が弱すぎる ($\gamma_t \to 0$ となるのが速すぎる)

どうしても、実際にはこうなってしまう

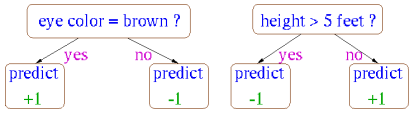
$$\text{training error}(H_{\text{final}}) \le \exp\left(-2\sum \gamma_t^2\right)$$

    - 学習不足
    - マージンが小 → 過学習
- 経験的には, AdaBoost はノイズの影響を受けやすいように思われる

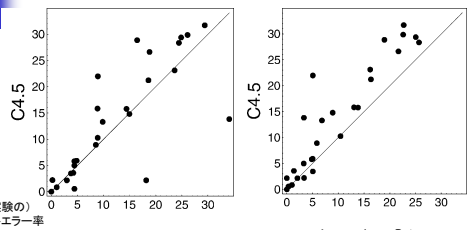
## UCI ベンチマーク

比較

- C4.5 (Quinlan の決定木学習)
- Decision Stumps (切株. ノード一個)




## UCI 結果[Schapire et al. 98]




（多数回実験の）平均テストエラー率

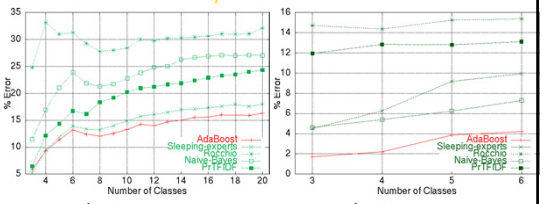
boosting Stumps　　boosting C4.5

Figure 3: Comparison of C4.5 versus boosting stumps and boosting C4.5 on a set of 27 benchmark problems as reported by Freund and Schapire [30]. Each point in each scatterplot shows the test error rate of the two competing algorithms on a single benchmark. The y-coordinate of each point gives the test error rate (in percent) of C4.5 on the given benchmark, and the x-coordinate gives the error rate of boosting stumps (left plot) or boosting C4.5 (right plot). All error rates have been averaged over multiple runs.

## テキスト分類[Schapire&Singer 00]

- Decision stumps: 単語や短句の存在/不存在. 例:

*"If the word Obama appears in the document predict document is about politics"*




データ: AP　　データ: Reuters

## 他の比較 [Quinlan, '96]

| | C4.5 | Bagged C4.5 vs C4.5 | | | Boosted C4.5 vs C4.5 | | | Boosting vs Bagging | |
|---|---|---|---|---|---|---|---|---|---|
| | err (%) | err (%) | w-l | ratio | err (%) | w-l | ratio | w-l | ratio |
| anneal | 7.67 | 6.25 | 10-0 | .814 | 4.73 | 10-0 | .617 | 10-0 | .758 |
| audiology | 22.12 | 19.29 | 9-0 | .872 | 15.71 | 10-0 | .710 | 10-0 | .814 |
| auto | 17.66 | 19.66 | 2-8 | 1.113 | 15.22 | 9-1 | .862 | 9-1 | .774 |
| breast-w | 5.28 | 4.23 | 9-0 | .802 | 4.09 | 9-0 | .775 | 7-2 | .966 |
| chess | 8.55 | 8.33 | 6-2 | .975 | 4.59 | 10-0 | .537 | 10-0 | .551 |
| colic | 14.92 | 15.19 | 0-6 | 1.018 | 18.83 | 0-10 | 1.262 | 0-10 | 1.240 |
| credit-a | 14.70 | 14.13 | 8-2 | .962 | 15.64 | 1-9 | 1.064 | 0-10 | 1.107 |
| credit-g | 28.44 | 25.81 | 10-0 | .908 | 29.14 | 2-8 | 1.025 | 0-10 | 1.129 |
| diabetes | 25.39 | 23.63 | 9-1 | .931 | 28.18 | 0-10 | 1.110 | 0-10 | 1.192 |
| glass | 32.48 | 27.01 | 10-0 | .832 | 23.55 | 10-0 | .725 | 9-1 | .872 |
| heart-c | 22.94 | 21.52 | 7-2 | .938 | 21.39 | 8-0 | .932 | 5-4 | .994 |
| heart-h | 21.53 | 20.31 | 8-1 | .943 | 21.05 | 5-4 | .978 | 3-6 | 1.037 |
| hepatitis | 20.39 | 18.52 | 9-0 | .908 | 17.68 | 10-0 | .867 | 6-1 | .955 |
| hypo | .48 | .45 | 7-2 | .928 | .36 | 9-1 | .746 | 9-1 | .804 |
| iris | 4.80 | 5.13 | 2-6 | 1.069 | 6.53 | 0-10 | 1.361 | 0-8 | 1.273 |
| labor | 19.12 | 14.39 | 10-0 | .752 | 13.86 | 9-1 | .725 | 5-3 | .963 |
| letter | 11.99 | 7.51 | 10-0 | .626 | 4.66 | 10-0 | .389 | 10-0 | .621 |
| lymphography | 21.69 | 20.41 | 8-2 | .941 | 17.43 | 10-0 | .804 | 10-0 | .854 |
| phoneme | 19.44 | 18.73 | 10-0 | .964 | 16.36 | 10-0 | .842 | 10-0 | .873 |
| segment | 3.21 | 2.74 | 9-1 | .853 | 1.87 | 10-0 | .583 | 10-0 | .684 |
| sick | 1.34 | 1.22 | 7-1 | .907 | 1.05 | 10-0 | .781 | 9-1 | .861 |
| sonar | 25.62 | 23.80 | 7-1 | .929 | 19.62 | 10-0 | .766 | 10-0 | .824 |
| soybean | 7.73 | 7.58 | 6-3 | .981 | 7.16 | 8-2 | .926 | 8-1 | .944 |
| splice | 5.91 | 5.58 | 9-1 | .943 | 5.43 | 9-0 | .919 | 6-4 | .974 |
| vehicle | 27.09 | 25.54 | 10-0 | .943 | 22.72 | 10-0 | .839 | 10-0 | .889 |
| vote | 5.06 | 4.37 | 9-0 | .864 | 3.29 | 3-6 | 1.046 | 1-9 | 1.211 |
| waveform | 27.33 | 19.77 | 10-0 | .723 | 18.53 | 10-0 | .678 | 8-2 | .938 |
| *average* | *15.66* | *14.11* | | *.905* | *13.36* | | *.847* | | *.930* |

| Name | Cases | Classes | Attributes | |
|---|---|---|---|---|
| | | | Cont | Discr |
| anneal | 898 | 6 | 9 | 29 |
| audiology | 226 | 6 | – | 69 |
| auto | 205 | 6 | 15 | 10 |
| breast-w | 699 | 2 | 9 | – |
| chess | 551 | 2 | – | 39 |
| colic | 368 | 2 | 10 | 12 |
| credit-a | 690 | 2 | 6 | 9 |
| credit-g | 1,000 | 2 | 7 | 13 |
| diabetes | 768 | 2 | 8 | – |
| glass | 214 | 6 | 9 | – |
| heart-c | 303 | 2 | 8 | 5 |
| heart-h | 294 | 2 | 8 | 5 |
| hepatitis | 155 | 2 | 6 | 13 |
| hypo | 3,772 | 5 | 7 | 22 |
| iris | 150 | 3 | 4 | – |
| labor | 57 | 2 | 8 | 8 |
| letter | 20,000 | 26 | 16 | – |
| lymph | 148 | 4 | – | 18 |
| phoneme | 5,438 | 47 | – | 7 |
| segment | 2,310 | 7 | 19 | – |
| sick | 3,772 | 2 | 7 | 22 |
| sonar | 208 | 2 | 60 | – |
| soybean | 683 | 19 | – | 35 |
| splice | 3,190 | 3 | – | 62 |
| vehicle | 846 | 4 | 18 | – |
| vote | 435 | 2 | – | 16 |
| waveform | 300 | 3 | 21 | – |

**Table 1:** Comparison of C4.5 and its bagged and boosted versions.

## まとめ

- boosting は分類課題に有用
  - 豊富な理論に裏付けられる
  - 実験的にも、パフォーマンスの良さが確認ずみ
  - しばしば (いつも、ではない) 過学習しにくい
  - 応用事例多い
- しかし
  - (得られた)分類器は遅い
  - 結果は、分かりにくい
  - ノイズに敏感なことあり