PRML 第14章

宮澤　彬

総合研究大学院大学 博士前期 1 年

miyazawa-a@nii.ac.jp

September 26, 2015

はじめに

- このスライドの LuaLaTeX のソースコードは `https://github.com/pecorarista/documents` にあります.
- 教科書とは若干異なる表記をしている場合があります.
- 図は TikZ で描いたものを除き，Bishop (2006) のサポートページ `http://research.microsoft.com/en-us/um/people/cmbishop/prml/` または Murphy (2012) のサポートページ `http://www.cs.ubc.ca/~murphyk/MLbook/` から引用しています.
- 間違った記述を見つけた方は連絡ください.

今日やること

1. ベイズモデル平均化
2. コミッティ
3. ブースティング (AdaBoost)
   - 分類
   - ~~回帰~~
4. 木構造モデル (CART)
   - 回帰
   - 分類
5. ~~条件付き混合モデル~~

モデルの結合

複数のモデルの予測結果の平均を使って回帰を行うと，過学習が避けられて，より良い予測値が得られそう．また，分類の場合は多数決の結果を使うと同様のことが言えそう．

複数のモデルの出力を組み合わせて使う方法を**コミッティ**あるいは**合議** (committee) と呼ぶ．

モデルの結合

モデルの結合とベイズモデル平均化を混同しないように注意する.

まずはモデル結合の例として，混合ガウス分布の密度推定を考える．モデルは，データがどの混合要素から生成されたかを示す潜在変数 z を使って，同時確率 $p(x, z)$ で与えられる．観測変数 x の密度は

$$p(x) = \sum_{z} p(x, z) \tag{14.3}$$

$$= \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k) \tag{14.4}$$

となる．データ $X = (x_1, \ldots, x_N)$ に関する周辺分布は

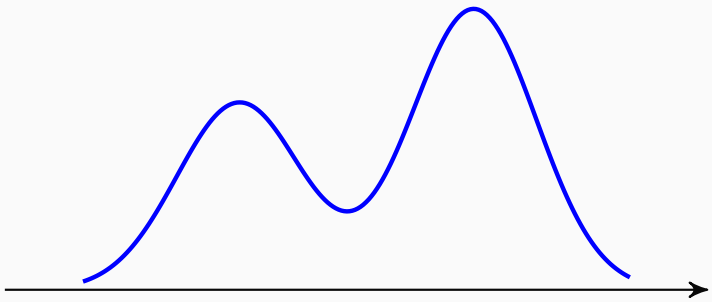
$$p(X) = \prod_{n=1}^{N} p(x_n) = \prod_{n=1}^{N} \left(\sum_{z_n} p(x_n, z_n) \right) \tag{14.5}$$

であり，各 x_n に対して z_n が存在していることがわかる.

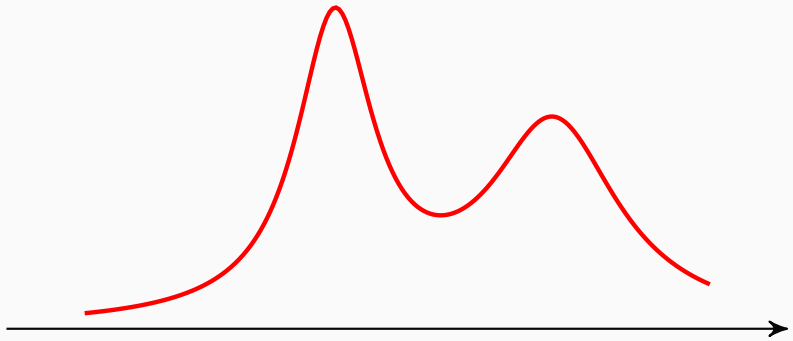
ベイズモデル平均化

次にベイズモデル平均化の例を示す．いくつかの異なるモデルがあって，$h = 1, \ldots, H$ で番号付けられているとする．例えば 1 つは混合ガウス分布で，もう 1 つは混合コーシー分布であるとする．

$$\sum_k \pi_k \mathcal{N}\left(x|\mu_k, \Sigma_k\right)$$



$$\sum_\ell \rho_\ell t_1\left(x|\mu_\ell, \Sigma_\ell\right)$$



ベイズモデル平均化

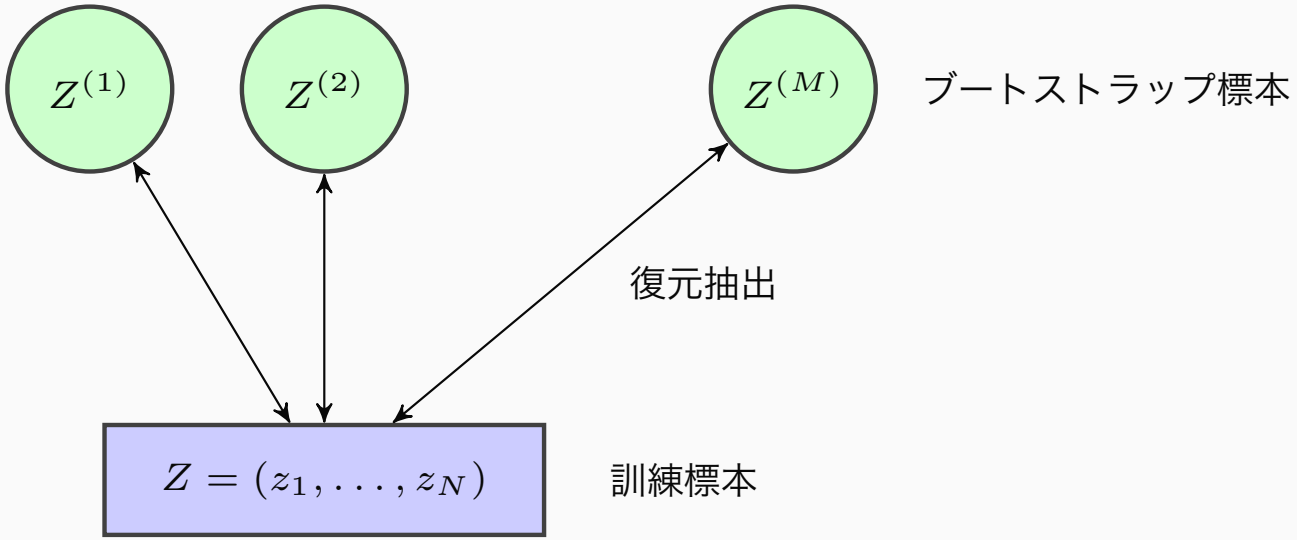
モデルに関する事前確率が $p(h)$ で与えられているならば，データに関する周辺分布は次式で求められる.

$$p(X) = \sum_{h=1}^{H} p(X|h)\, p(h) \tag{14.6}$$

モデル結合との違いは，データを生成しているのはモデル $h = 1, \ldots, H$ のどれか 1 つだということである.

コミッティ

コミッティを使う手法の例として，**バギング** (bagging; **b**ootstrap **agg**regation) を紹介する．まず訓練用データ $Z = (z_1, \ldots, z_N)$ から，ブートストラップ標本 $Z^{(1)}, \ldots, Z^{(M)}$ を作る．



コミッティ

バギングにおけるコミッティの予測値 $y_{\mathrm{COM}}(x)$ は，各ブートストラップ集合 $Z^{(1)}, \ldots, Z^{(M)}$ で学習したモデルの予測 $y_1(x), \ldots, y_m(x)$ を使って

$$y_{\mathrm{COM}}(x) = \frac{1}{M} \sum_{m=1}^{M} y_m(x) \tag{14.7}$$

となる.

これで実際に良い結果が得られるのか，誤差を評価して確認する.

バギングの誤差評価

予測したい回帰関数を $h(x)$ とし，各モデルの予測値 $y_m(x)$ が，$h(x)$ と加法的な誤差 $\varepsilon_m(x)$ を使って

$$y_m(x) = h(x) + \varepsilon_m(x) \tag{14.8}$$

と表せると仮定する．このとき，二乗誤差は

$$\mathbb{E}\left[(y_m(x) - h(x))^2\right] = \mathbb{E}\left[(\varepsilon_m(x))^2\right] \tag{14.9}$$

となるので，平均二乗誤差の期待値 E_{AV} は

$$\begin{aligned} E_{\mathrm{AV}} &:= \mathbb{E}\left[\frac{1}{M}\sum_{m=1}^{M}(y_m(x) - h(x))^2\right] \\ &= \frac{1}{M}\sum_{m=1}^{M}\mathbb{E}\left[(\varepsilon_m(x))^2\right] \end{aligned} \tag{14.10}$$

で求められる．

バギングの誤差評価

一方，コミッティによる出力について，誤差の期待値 E_{COM} は

$$E_{\mathrm{COM}} := \mathbb{E}\left[\left(\frac{1}{M}\sum_{m=1}^{M} y_m - h(x)\right)^2\right]$$
$$= \mathbb{E}\left[\left(\frac{1}{M}\sum_{m=1}^{M} \varepsilon_m(x)\right)^2\right] \tag{14.11}$$

と求められる．ベクトル $(1 \cdots 1)' \in \mathbb{R}^M$ と $(\varepsilon_1(x) \cdots \varepsilon_m(x))' \in \mathbb{R}^M$ に対して Cauchy-Schwarz の不等式を用いると

$$\left(\sum_{m=1}^{M} \varepsilon_m(x)\right)^2 \leq M \sum_{m=1}^{M} (\varepsilon_m(x))^2$$
$$E_{\mathrm{COM}} \leq E_{\mathrm{AV}}$$

が成り立つ．つまりコミッティを使う場合の誤差の期待値は，構成要素の誤差の期待値の和を超えない．

バギングの誤差評価

特に誤差の平均が 0 で無相関である，すなわち

$$\mathbb{E}\left[\varepsilon_m\left(x\right)\right] = 0 \tag{14.12}$$

$$\mathrm{cov}\left(\varepsilon_m\left(x\right), \varepsilon_\ell\left(x\right)\right) = \mathbb{E}\left[\varepsilon_m\left(x\right)\varepsilon_\ell\left(x\right)\right] = 0,\ m \neq \ell \tag{14.13}$$

が成り立つならば [1]

$$\begin{aligned} E_{\mathrm{COM}} &= \mathbb{E}\left[\left(\frac{1}{M}\sum_{m=1}^{M}\varepsilon_m\left(x\right)\right)^2\right] \\ &= \frac{1}{M^2}\mathbb{E}\left[\sum_{m=1}^{M}\left(\varepsilon_m\left(x\right)\right)^2\right] \\ &= \frac{1}{M}E_{\mathrm{AV}} \end{aligned} \tag{14.14}$$

となり，誤差の期待値が大幅に低減される．

1 同じようなモデルを，同じような訓練データで学習させているのだから，こんなことは期待できない．

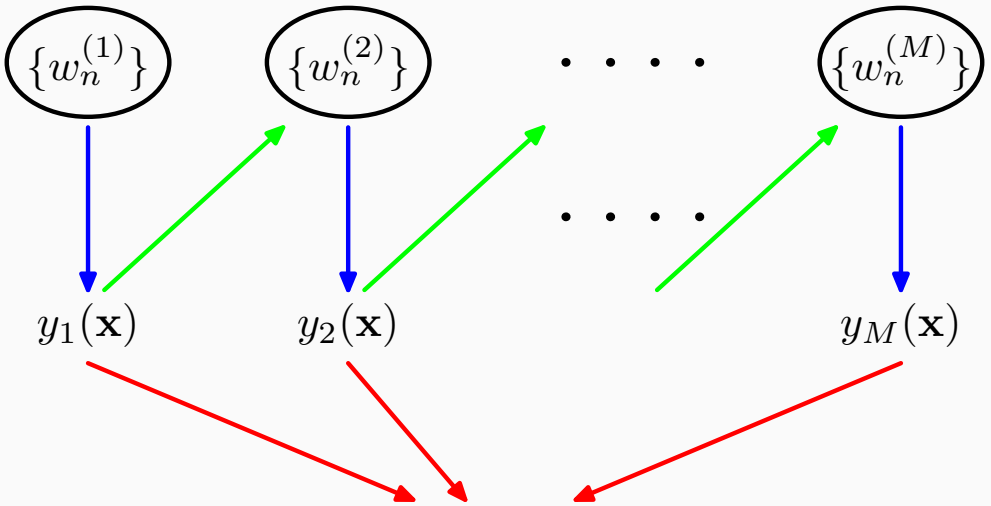
ブースティング

コミッティを使う手法の別な例として，**ブースティング** (boosting) と呼ばれる手法を紹介する．ブースティングは分類のためのアルゴリズムとして設計されたが，回帰にも拡張できる．とりあえず分類について考える．

コミッティを構成する個々の分類器は**ベース学習器** (base learner) または**弱学習器** (weak learner) と呼ばれる．ブースティングは，弱学習器を（並列的にではなく）逐次的に，重み付きデータを使って学習を進める方法である．重みの大きさは，それ以前に誤って分類されたデータに対して大きくなるように定める．

ブースティングの中で代表的な手法である **AdaBoost** (**ada**ptive **boost**ing) を紹介する．

AdaBoost



$$Y_M(\mathbf{x}) = \text{sign}\left(\sum_m^M \alpha_m y_m(\mathbf{x})\right)$$

for $m = 1, \ldots, N$

Initialize the data weighting coefficients

$$w_n^{(1)} := \frac{1}{N}$$

AdaBoost

for $m = 1, \ldots, M$

Fit a classifier $y_m(x)$ to the training data by minimizing

$$J_m := \sum_{n=1}^{N} w_n^{(m)} \mathbb{1}_{\{x \mid y_m(x) \neq t_n\}}(x_n). \tag{14.15}$$

Evaluate

$$\alpha_m := \log \frac{1 - \varepsilon_m}{\varepsilon_m} \tag{14.17}$$

where

$$\varepsilon_m := \frac{\sum_{n=1}^{N} w_n^{(m)} \mathbb{1}_{\{x \mid y_m(x) \neq t_n\}}(x_n)}{\sum_{n=1}^{N} w_n^{(m)}}. \tag{14.15}$$

Update the data weighting coefficients

$$w_n^{(m+1)} := w_n^{(m)} \exp\left(\alpha_m \mathbb{1}_{\{x \mid y_m(x_n) \neq t_n\}}(x_n)\right). \tag{14.18}$$

AdaBoost

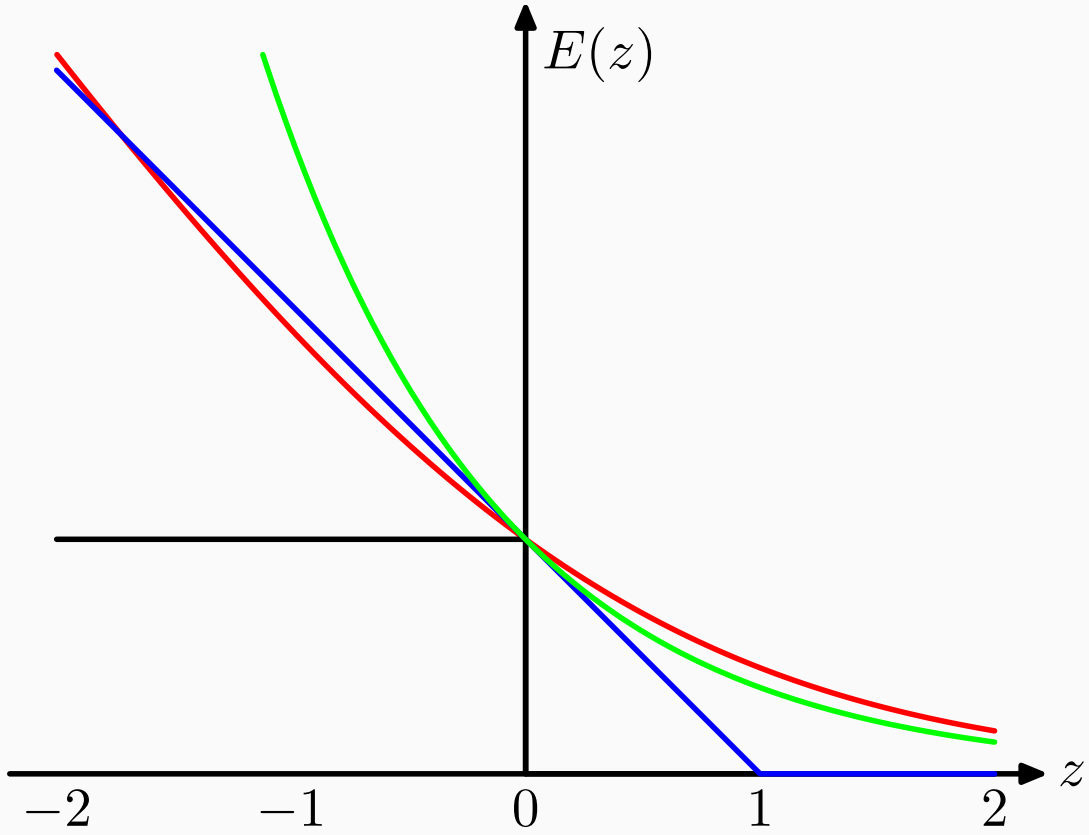
Make predictions using the final model, which is given by

$$Y_M(x) := \operatorname{sign}\left(\sum_{m=1}^{M} \alpha_m y_m(x)\right). \tag{14.19}$$

このアルゴリズム中に現れる式は，指数誤差の逐次的な最小化から導かれる.

指数誤差の最小化

目標値を $t \in \{-1, 1\}$，予測値を $y \in \mathbb{R}$ とし，これらの積 ty を z と置く．このとき指数誤差は $E(z) := \exp(-z)$ という形で与えられる．$E(z)$ は t と y が同符号 $(z > 0)$ なら 0 に近い正の値を，異符号 $(z < 0)$ ならば正の大きな値をとる．



指数誤差関数 交差エントロピー誤差 ヒンジ形誤差関数 誤分類誤差

指数誤差の最小化

以下の誤差 E を最小化したい.

$$E := \sum_{n=1}^{N} \exp\left(-t_n \sum_{\ell=1}^{m} \frac{1}{2}\alpha_\ell y_\ell\left(x_n\right)\right)$$

逐次的に学習していくので $\alpha_1, \ldots, \alpha_{m-1}$ と $y_1\left(x\right), \ldots, y_{m-1}\left(x\right)$ は所与として，α_m と $y_m\left(x\right)$ に関して E を最小化する．E を

$$\begin{aligned} E &= \sum_{n=1}^{N} \exp\left(-t_n \sum_{\ell=1}^{m-1} \frac{1}{2}\alpha_\ell y_\ell\left(x_n\right) - \frac{1}{2}t_n\alpha_m y_m\left(x_n\right)\right) \\ &= \sum_{n=1}^{N} w_n^{(m)} \exp\left(-\frac{1}{2}t_n\alpha_m y_m\left(x_n\right)\right) \end{aligned} \tag{14.22}$$

と書き換える．ここで以下のように置いた.

$$w_n^{(m)} := \exp\left(-t_n \sum_{\ell=1}^{m-1} \frac{1}{2}\alpha_\ell y_\ell\left(x_n\right)\right) \tag{14.22$'$}$$

指数誤差の最小化

$y_m : x \mapsto \{-1, 1\}$ によって正しく分類されたデータ点の添字集合を $\mathcal{T}_m$, 誤って分類されたデータの添字集合を $\mathcal{M}_m$ とする. このとき

$$
\begin{aligned}
E &= e^{-\alpha_m/2} \sum_{n \in \mathcal{T}_m} w_n^{(m)} + e^{\alpha_m/2} \sum_{n \in \mathcal{M}_m} w_n^{(m)} \\
&= \left(e^{\alpha_m/2} - e^{-\alpha_m/2} \right) \sum_{n=1}^{N} w_n^{(m)} \mathbb{1}_{\{x | y_m(x_n) \neq t_n\}} \left(x_n \right) \\
&\quad + e^{-\alpha_m/2} \sum_{n=1}^{N} w_n^{(m)}
\end{aligned} \tag{14.23}
$$

が成り立つ. これを $y_m\left(x\right)$ について最小化することは (14.15) を最小化することと同値である.

指数誤差の最小化

また α_m について微分して 0 と置くと

$$\left(e^{\alpha_m/2} + e^{-\alpha_m/2}\right) \sum_{n=1}^{N} w_n^{(m)} \mathbb{1}_{\{x|y_m(x_n)\neq t_n\}}(x_n) - e^{-\alpha_m/2} \sum_{n=1}^{N} w_n^{(m)} = 0$$

$$e^{\alpha_m} + 1 = \frac{\displaystyle\sum_{n=1}^{N} w_n^{(m)}}{\displaystyle\sum_{n=1}^{N} w_n^{(m)} \mathbb{1}_{\{x|y_m(x_n)\neq t_n\}}(x_n)}$$

となる．(14.15) と同じ置き換えを行うことにより

$$\alpha_m = \log \frac{1-\varepsilon_m}{\varepsilon_m}$$

を得る．これはアルゴリズム中の (14.17) に等しい．

指数誤差の最小化

重みは $(14.22')$ に従い次のように更新する.

$$w_n^{(m+1)} = w_n^{(m)} \exp\left(-\frac{1}{2} t_n \alpha_m y_m\left(x_n\right)\right) \tag{14.24}$$

ここで

$$t_n y_m\left(x_n\right) = 1 - 2\mathbb{1}_{\{x|y_m(x)\neq t_n\}}\left(x_n\right) \tag{14.25}$$

が成り立つことを使うと

$$\begin{aligned} w_n^{(m+1)} &= w_n^{(x)} \exp\left(-\frac{1}{2}\alpha_m\left(1 - 2\mathbb{1}_{\{x|y_m(x)\neq t_n\}}\left(x_n\right)\right)\right) \\ &= w_n^{(x)} \exp\left(-\frac{\alpha_m}{2}\right) \exp\left(\mathbb{1}_{\{x|y_m(x)\neq t_n\}}\left(x_n\right)\right) \end{aligned} \tag{14.26}$$

と書き換えられる. $\exp\left(-\alpha_m/2\right)$ はすべてのデータに共通なので無視すると (14.18) が得られる.

ブースティングのための誤差関数

以下の指数誤差を考える.

$$\mathbb{E}\left[\exp\left(-ty\left(x\right)\right)\right] = \sum_{t\in\{-1,1\}} \int \exp\left(-ty\left(x\right)\right) p\left(t|x\right) p\left(x\right) dx \qquad (14.27)$$

変分法を使って y について上式を最小化する.
$\varphi_t\left(y\right) := \exp\left(-ty\left(x\right)\right) p\left(t|x\right) p\left(x\right)$ とする. 停留点は以下のように求められる.

$$\sum_{t\in\{-1,1\}} D_y \varphi_t\left(y\right) = 0$$

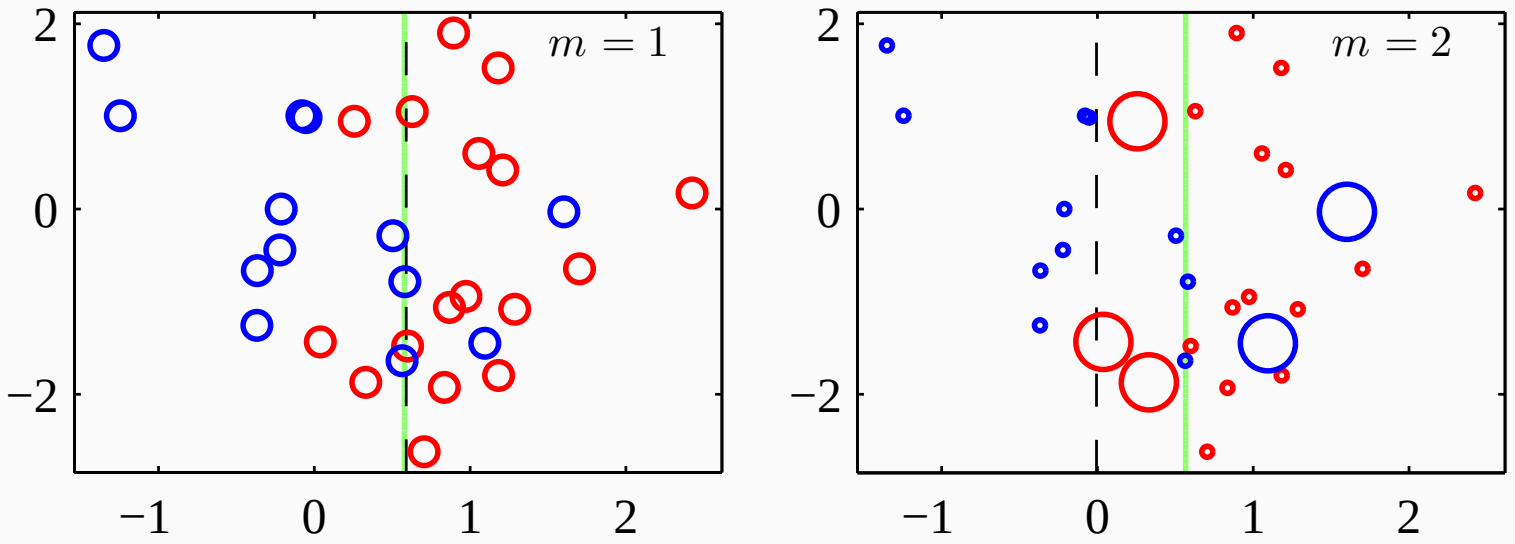
$$\exp\left(y\left(x\right)\right) p\left(t=-1|x\right) - \exp\left(-y\left(x\right)\right) p\left(t=1|x\right) = 0$$

$$y\left(x\right) = \frac{1}{2}\log\frac{p\left(t=1|x\right)}{p\left(t=-1|x\right)} \qquad (14.28)$$

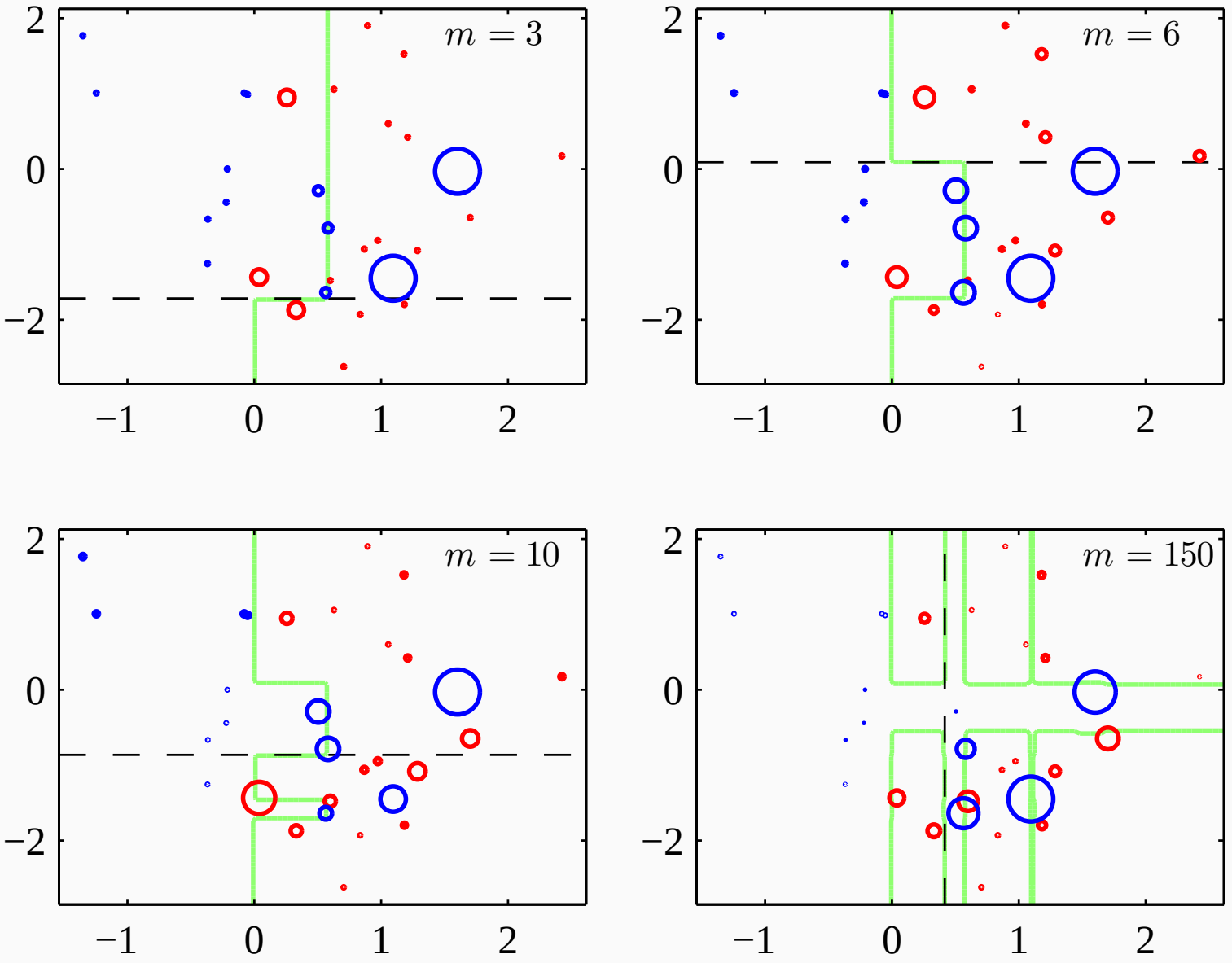
つまり AdaBoost は逐次的に $p\left(t=1|x\right)$ と $\left(t=-1|x\right)$ の比の対数の近似を求めている. これが最終的に (14.19) で符号関数を使って分類器を作った理由となっている.

AdaBoost の振る舞い

AdaBoost による分類の例を示す．図中の円は各データを表し，その半径は割り振られた重みを表している．



AdaBoostの振る舞い



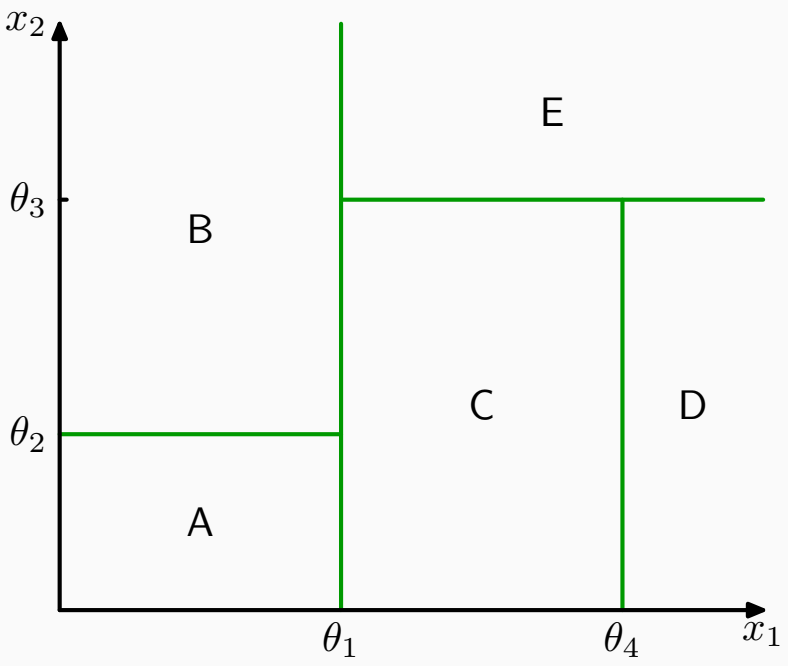
AdaBoost の拡張

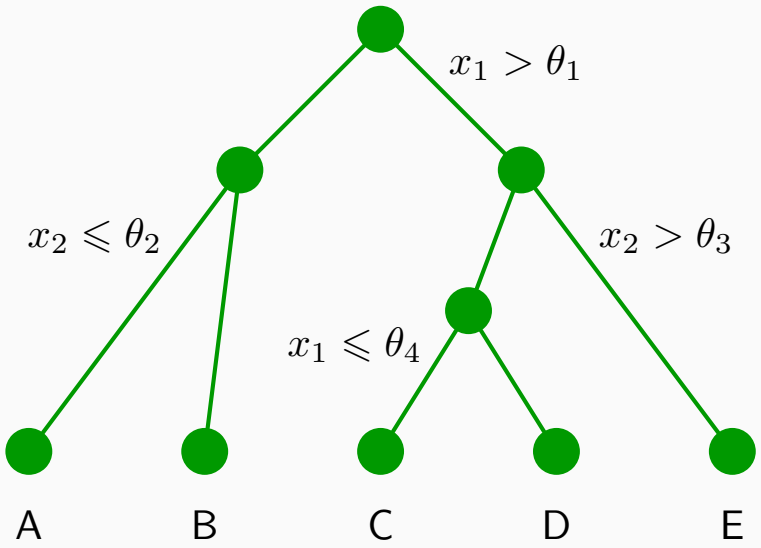
AdaBoost は Freund and Schapire (1996) で導入されたが，指数誤差の逐次的最小化としての解釈は Friedman et al. (2000) で与えられた．これ以降，誤差関数の変更により様々な拡張が行われるようになった．

木構造モデル

今までは複数の出力を合わせる方法を見てきた．今度は入力を複数のモデルの 1 つに振り分ける方法である木構造モデルを扱う．

木構造モデルのでは，入力空間を矩形 (cuboid) 領域に分けて，それぞれの領域に対応するモデルに予測をさせる．





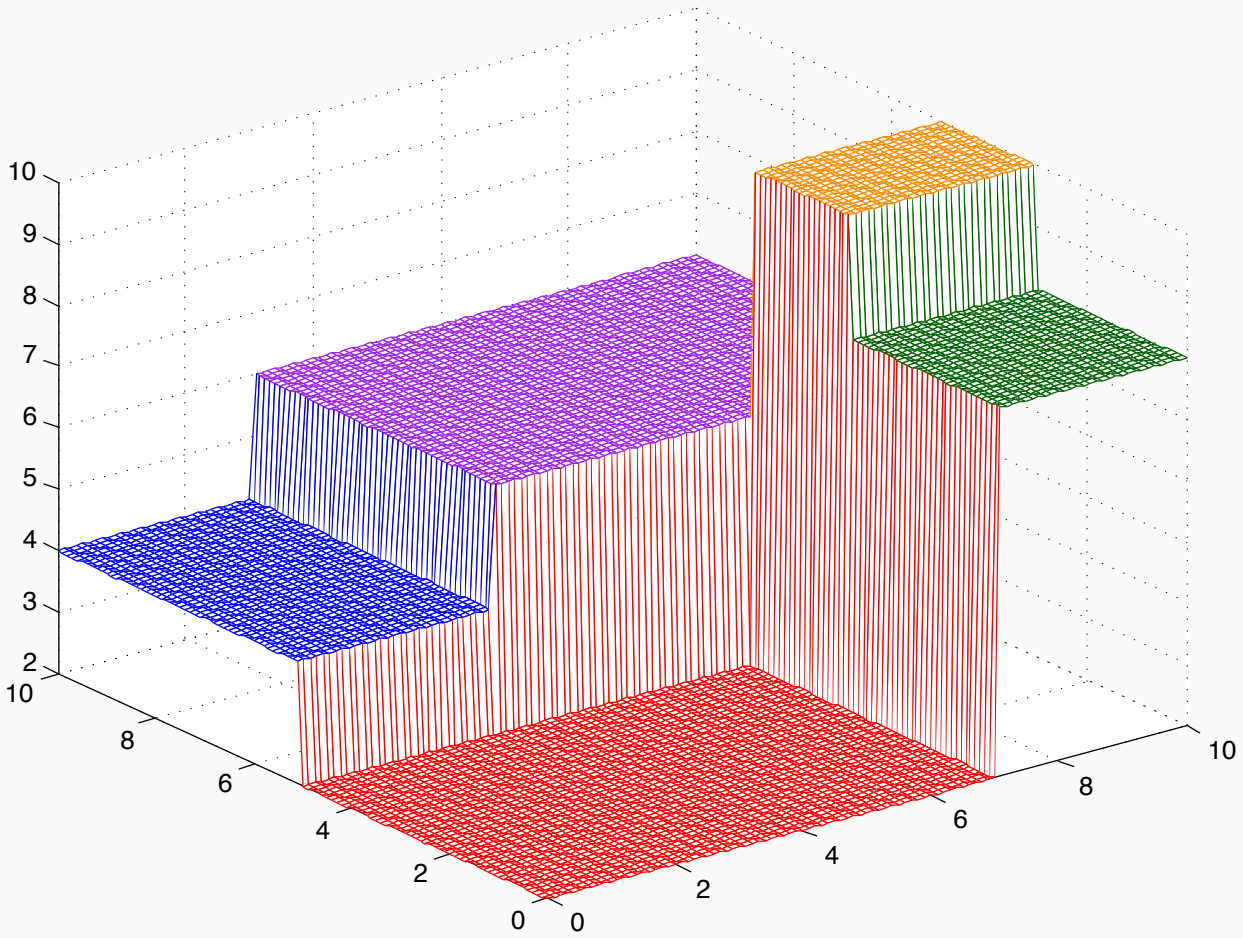
CART

以下では木構造モデルの 1 つである **CART** (classification and regression tree) を扱う．CART は名前の通り分類にも回帰にも使うことができるが，まずは回帰に絞って説明する．

入力と目標値の組 $\mathcal{D} = (\boldsymbol{x}_1, t_n), \ldots, (\boldsymbol{x}_N, t_N) \in \mathbb{R}^D \times \mathbb{R}$ が与えられているとする．入力に対する分割が $\mathcal{R}_1, \ldots, \mathcal{R}_M$ となっているとき，予測値を次のように定義する．

$$f(\boldsymbol{x}) := \sum_{m=1}^{M} c_m \mathbb{1}_{\mathcal{R}_m}(\boldsymbol{x})$$

CART



分割が与えられているときの最適な予測値

各領域で二乗和誤差を最小化する予測値を求めよう.
$\mathcal{I}_m := \{n\,;\, \boldsymbol{x}_n \in \mathcal{R}_m\}$ と表すと誤差は以下のようになる.

$$\begin{aligned}\sum_{n=1}^{N} \left(t_n - f\left(\boldsymbol{x}_n\right)\right)^2 &= \sum_{n=1}^{N} \left(t_n - \sum_{m=1}^{M} c_m \mathbb{1}_{\mathcal{R}_m}\left(\boldsymbol{x}_n\right)\right)^2 \\ &= \sum_{m=1}^{M} \sum_{n \in \mathcal{I}_m} \left(c_m - t_n\right)^2\end{aligned}$$

各 c_m について微分して 0 と置くと，最適な予測値は

$$\bar{c}_m = \frac{1}{|\mathcal{I}_m|} \sum_{n \in \mathcal{I}_m} t_n$$

となることが分かる. ただし二乗和誤差を最小化するような分割を決めようとすると計算量が大きくなる. そこで**貪欲法**を使って木の構造を定める.

分割基準

各回の領域の分割基準を考える．変数を表す添字を $j \in \{1, \ldots, D\}$，閾値を θ として，２つの領域 $\mathcal{R}_1(j, \theta)$ と $\mathcal{R}_2(j, \theta)$ を次で定める．

$$\mathcal{R}_1(j, \theta) = \{\boldsymbol{x}\,;\, x_j \leq \theta\}, \quad \mathcal{R}_2(j, \theta) = \{\boldsymbol{x}\,;\, x_j > \theta\}$$

$\mathcal{I}_i(j, \theta) := \{n\,;\, \boldsymbol{x}_n \in \mathcal{R}_i(j, \theta)\}$ とする．分割の基準となる (j, θ) は次を解いて得られる．

$$\min_{j \in \{1,\ldots,D\}} \min_{\theta} \left\{ \min_{c_1} \sum_{n \in \mathcal{I}_1(j,\theta)} (c_1 - t_n)^2 + \min_{c_2} \sum_{n \in \mathcal{I}_2(j,\theta)} (c_2 - t_n)^2 \right\}.$$

内側の和を最小化する c_1 と c_2 は，それぞれ

$$\bar{c}_1 = \frac{1}{|\mathcal{I}_1(j, \theta)|} \sum_{n \in \mathcal{I}_1(j,\theta)} t_n, \quad \bar{c}_2 = \frac{1}{|\mathcal{I}_2(j, \theta)|} \sum_{n \in \mathcal{I}_2(j,\theta)} t_n$$

となる．あとは各 j について最適な θ を計算し，そのあとで最適な j を求めればよい．

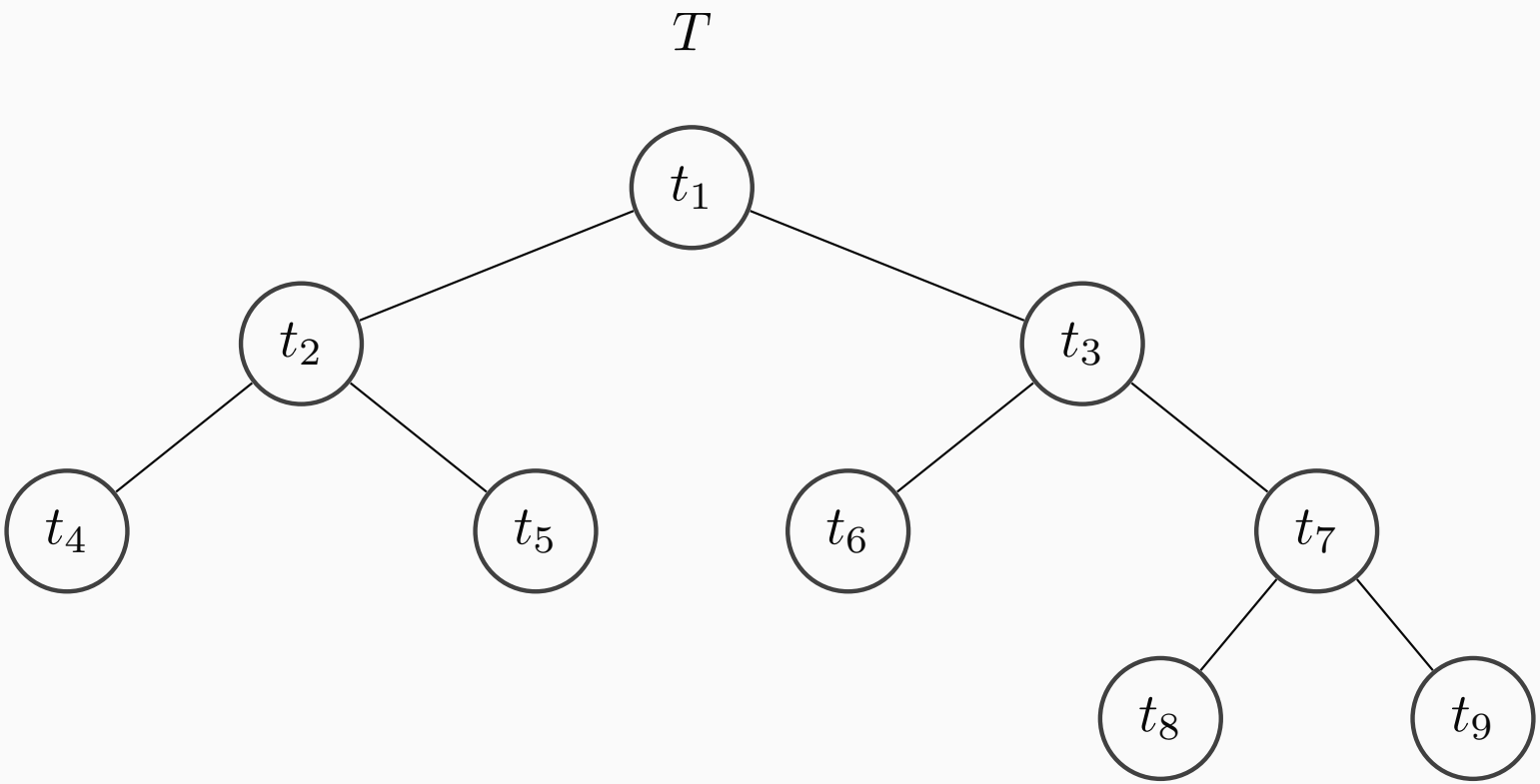
停止基準

次に停止基準について考えなければならない.

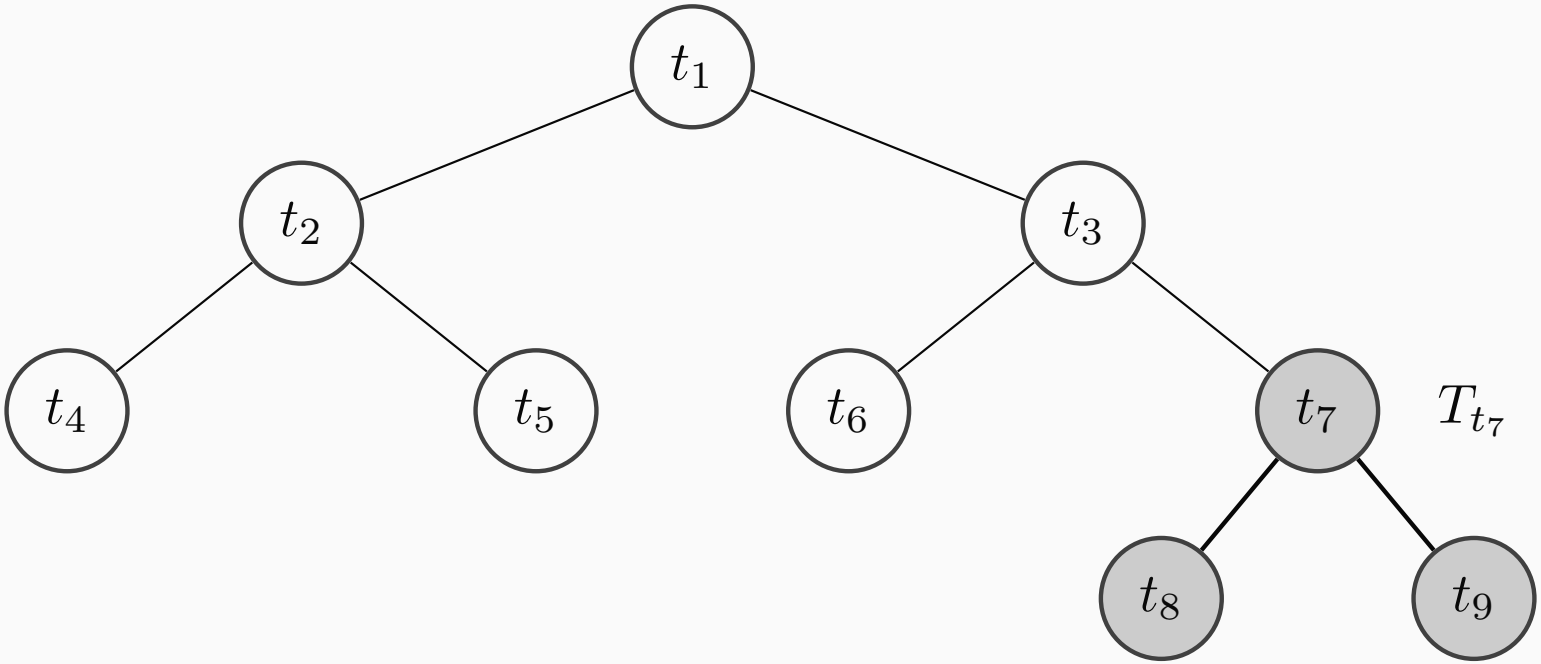
単純に考えると誤差の減少幅が小さくなったときに止める方法が良さそうであるが，分割を続けていると大きな誤差の減少が見られることが経験的に知られている.

そこで，先に大きな木を作ってからその木の枝を刈って適当な木を得ることにする.

枝刈り

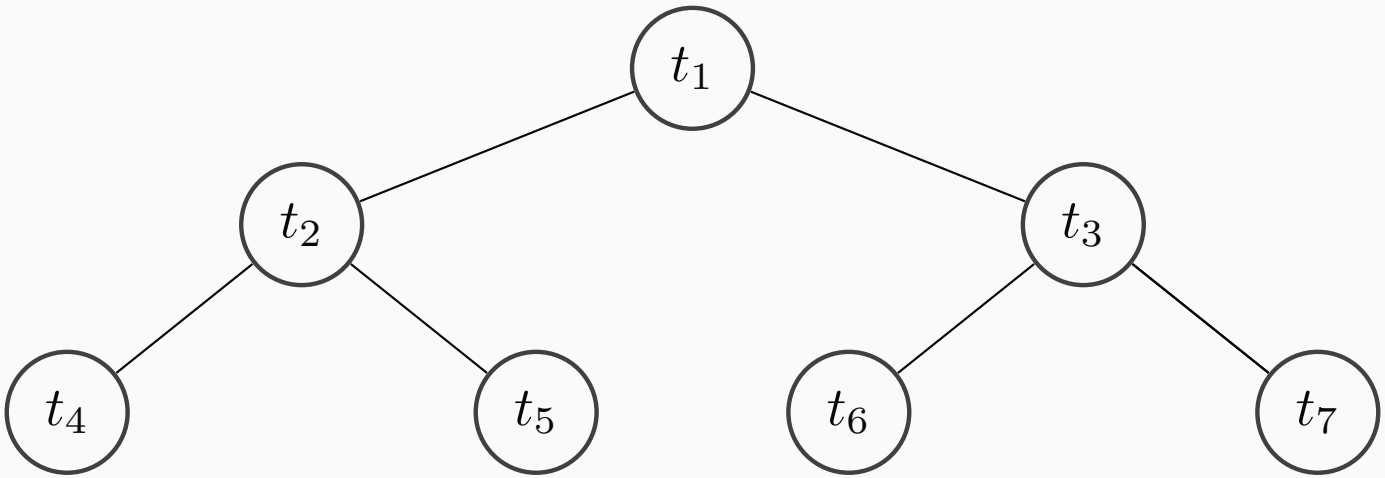


枝刈り



枝刈り

$T - T_{t_7}$



t_8 に対応する領域と t_9 に対応する領域を併合するので t_7 は残る.

枝刈りの基準

切り落とす位置は，次の **error-complexity measure**

$$R_\alpha(T) = R(T) + \alpha \left|\widetilde{T}\right|$$

が一番大きく減少するような節点 t を選ぶ．ただし $\widetilde{T}$ は木 T の葉全体の集合，$R(T)$ は誤差

$$R(T) := \sum_{m=1}^{\left|\widetilde{T}\right|} \sum_{n \in \mathcal{I}_m} (\bar{c}_m - t_n)^2$$

である．$\alpha \geq 0$ は枝の数に対する罰則を調整するパラメータである．この基準では，刈っても（通常大きくなってしまう）誤差があまり大きくならず，葉の数が減るような枝刈りが優先して行われる．

枝刈りの基準

α を動かすと次のようなことがわかる.

- $\alpha = 0$ のとき
  各データに 1 つの葉を対応させるような木が最適になる.
- α が大きいとき
  根 t_1 だけから成る木が最適になる.

ちょうどいい木はこの間にある.

枝刈り

部分木 T_t と節点 t から下を刈ってできる木 $(\{t\}, \emptyset)$ の error-complexity measure はそれぞれ

$$R_\alpha(T_t) = R(T_t) + \alpha \left|\widetilde{T}_t\right|,$$

$$R_\alpha((\{t\}, \emptyset)) = R((\{t\}, \emptyset)) + \alpha$$

である．$R_\alpha(T_t) > R_\alpha((\{t\}, \emptyset))$，すなわち

$$\alpha < \frac{R(T_t) - R((\{t\}, \emptyset))}{\left|\widetilde{T}_t\right| - 1}$$

が成り立つならば刈り取ったほうがよい．この右辺を刈り取る効果の指標とし

$$g(t; T) := \frac{R(T_t) - R((\{t\}, \emptyset))}{\left|\widetilde{T}_t\right| - 1}$$

と定める．

枝刈り

次の**最弱リンク枝刈り** (weakest link pruning) と呼ばれるアルゴリズムにより $T^0 \succ \cdots \succ T^J = (\{t_1\}, \emptyset)$ と狭義単調増加列 $\alpha_0 < \cdots < \alpha_J$ を得る.

1: $i \leftarrow 0$

2: **while** $\left|\widetilde{T^i}\right| > 1$

3: $\quad \alpha_i \leftarrow \min_{t \in V(T^i) \setminus \widetilde{T^i}} g\left(t\,;\, T^i\right)$

4: $\quad \mathcal{T}^i \leftarrow \arg\min_{t \in V(T^i) \setminus \widetilde{T^i}} g\left(t\,;\, T^i\right)$

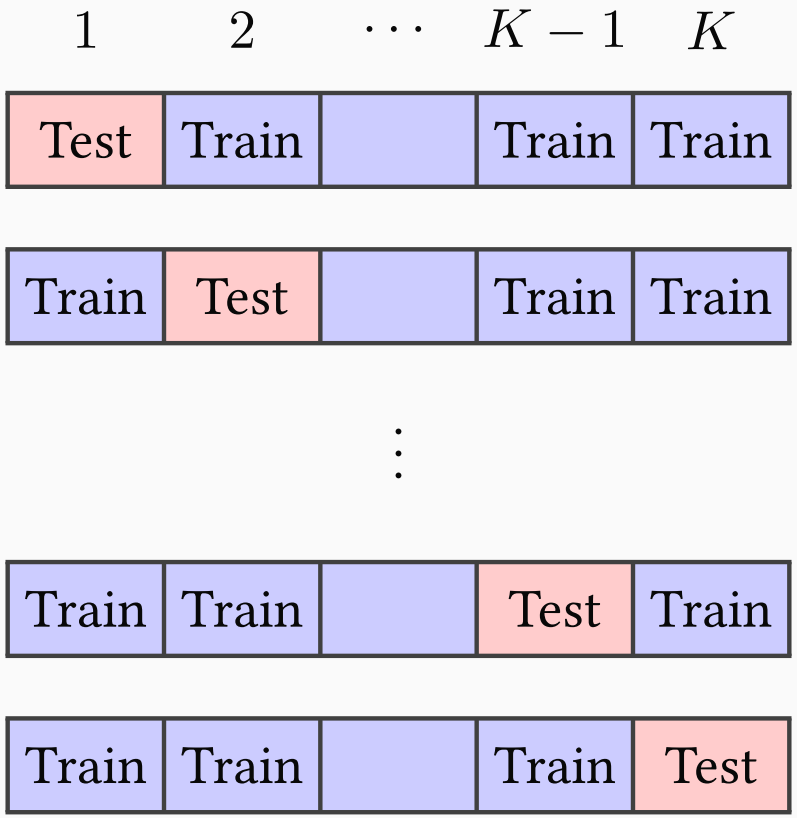
5: $\quad$ **for** $t \in \mathcal{T}^i$

6: $\quad\quad T^i \leftarrow T^i - T^i_t$

7: $\quad i \leftarrow i + 1$

交差確認

最終的にやらなければならないことは，得られた列 $\left\langle T^i \right\rangle_{i=0}^{J}$ の中から適当な T^i 選ぶことである．そのための方法としては交差確認 (cross validation) が使われる．



交差確認

手順は以下のようになる.

1: データ $\mathcal{D}$ を使って，木の列 $\left\langle T^i \right\rangle_{i=0}^{J}$ とパラメータの列 $\left\langle \alpha_i \right\rangle_{i=0}^{J}$ を作る.

2: データ $\mathcal{D}$ を $\mathcal{D} = \coprod_{k=1}^{K} \mathcal{D}_k$ と分割する．極力，各 $|\mathcal{D}_k|$ が同じ大きさになるようにする.

3: 各 $\mathcal{D}^{(k)} := \mathcal{D} \backslash \mathcal{D}_k$ を使い，木の列 $\left\langle T^{(k)i} \right\rangle_{i=0}^{i_k}$ とパラメータの列 $\left\langle \alpha_i^{(k)} \right\rangle_{i=0}^{i_k}$ を作る.

交差確認

4: 各 $\alpha_i' := \sqrt{\alpha_i \alpha_{i+1}}$ に対して $R_{\alpha_i'}(T)$ を最小化するような $T^{(1)}(\alpha_i'), \ldots, T^{(K)}(\alpha_i')$ とそれらに対応する予測 $y_i^{(1)}, \ldots, y_i^{(K)}$ を求める．ここで $\alpha \in \left[\alpha_i^{(k)}, \alpha_{i+1}^{(k)}\right)$ ならば $T^{(k)}(\alpha) = T^{(k)i}$ となることに注意する．

5: 今までの結果から以下を求めることができる．

$$R^{\mathrm{CV}}\left(T^i\right) = \frac{1}{N} \sum_{k=1}^{K} \sum_{n \in \{n' \,;\, (\boldsymbol{x}_n, t_n) \in \mathcal{D}_k\}} \left(t_n - y_i^{(k)}(\boldsymbol{x}_n)\right)^2$$

6: 誤差が一番小さい木を $T^{**} := \arg\min_{T^i} R\left(T^i\right)$ とする．

交差確認

7：標準誤差 (standard error) SE を求める.

$$\mathrm{SE}\left(R^{\mathrm{CV}}\left(T^i\right)\right) := s\left(T^i\right)/\sqrt{N},$$

$$s\left(T^i\right) := \sqrt{\frac{1}{N}\sum_{n=1}^{N}\left(\left(t_n - y_i^{(\kappa(n))}\left(x_n\right)\right)^2 - R^{\mathrm{CV}}\left(T^i\right)\right)^2},$$

$$\kappa\left(n\right) = \sum_{k=1}^{K} k\mathbb{1}_{\mathcal{D}_k}\left(\left(x_n, t_n\right)\right)$$

8：以下を満たす T の中で最小の木 T^* を最終的な結果とする.

$$R^{\mathrm{CV}}\left(T\right) \leq R^{\mathrm{CV}}\left(T^{**}\right) + 1 \times \mathrm{SE}\left(T^{**}\right)$$

このヒューリスティックな決め方は **1 SE rule** と呼ばれる.

交差確認

Breiman et al. (1984) の TABLE 3.3

i	$\widetilde{T}^i$	$R\left(T^i\right)$	$R^{\mathrm{CV}}\left(T^i\right) \pm \mathrm{SE}\left(T^i\right)$
1	31	.17	$.30 \pm .03$
2^{**}	23	.19	$\mathbf{.27 \pm .03}$
3	17	.22	$.30 \pm .03$
4	15	.23	$.30 \pm .03$
5	14	.24	$.31 \pm .03$
$\mathbf{6}^{*}$	10	.29	$.30 \pm .03$
7	9	.32	$.41 \pm .04$
8	7	.41	$.51 \pm .04$
9	6	.46	$.53 \pm .04$
10	5	.53	$.61 \pm .04$
11	2	.75	$.75 \pm .03$
12	1	.86	$.86 \pm .03$

CART による分類

次に分類を扱う．手順は回帰とほぼ同様なので，分割の基準だけ扱う．

$\mathcal{C}_1, \ldots, \mathcal{C}_K$ の K クラスに分類する問題を解きたいとする．節点 t を通って流れていくデータの数を $N(t)$ とする．またそれらのうちで $\mathcal{C}_k$ に属するものの数を $N_k(t)$ と表す．このとき

$$p(t|\mathcal{C}_k) = \frac{N_k(t)}{N_k},$$

$$p(\mathcal{C}_k, t) = p(\mathcal{C}_k)\, p(t|\mathcal{C}_k) = \frac{N_k}{N} \frac{N_k(t)}{N_k} = \frac{N_k(t)}{N},$$

$$p(t) = \sum_{k=1}^{K} p(\mathcal{C}_k, t) = \frac{N(t)}{N}$$

である．よって事後確率 $p(\mathcal{C}_k|t)$ は次のように求められる．

$$p(\mathcal{C}_k|t) = \frac{p(\mathcal{C}_k, t)}{p(t)} = \frac{N_k(t)}{N(t)}$$

CART による分類

分割の基準には**不純度関数** (impurity function) というものを使う．不純度関数とは，以下の条件を満たす関数 $\varphi : \Delta^M \to \mathbb{R}$ のことである.

1. $\varphi(p)$ は $p = (1/M, \cdots, 1/M)$ のときに限り最大となる.
2. $\varphi(p)$ は $p = (0, \cdots, 0, \overset{i}{\underset{\smile}{1}}, 0, \cdots, 0)$ のときに限り最小となる.
3. $\varphi(p_1, \ldots, \overset{i}{\underset{\smile}{p_i}}, \ldots, \overset{j}{\underset{\smile}{p_j}}, \ldots, p_M) = \varphi(p_1, \ldots, \overset{i}{\underset{\smile}{p_j}}, \ldots, \overset{j}{\underset{\smile}{p_i}}, \ldots, p_M)$.

不純度

不純度関数を φ とすると，任意の節点 t の不純度 $I(t)$ は

$$I(t) := \varphi\left(p\left(\mathcal{C}_1|t\right), \ldots, p\left(\mathcal{C}_K|t\right)\right)$$

で定められる.

節点 t の子 t_L と t_R について，t からそれらに流れるデータの割合を

$$p_L = \frac{p\left(t_L\right)}{p\left(t\right)}, \quad p_R = \frac{p\left(t_R\right)}{p\left(t\right)}$$

とすると，t において基準 s で分割したときの不純度の減少は

$$\Delta I(s,t) = I(t) - p_R I\left(t_R\right) - p_L I\left(t_L\right)$$

と表される．不純度の減少が最大になるような分割を行っていけばよい.

不純度関数の例

- 誤分類率

$$I(t) = 1 - \max_k p(\mathcal{C}_k|t)$$

- 交差エントロピー

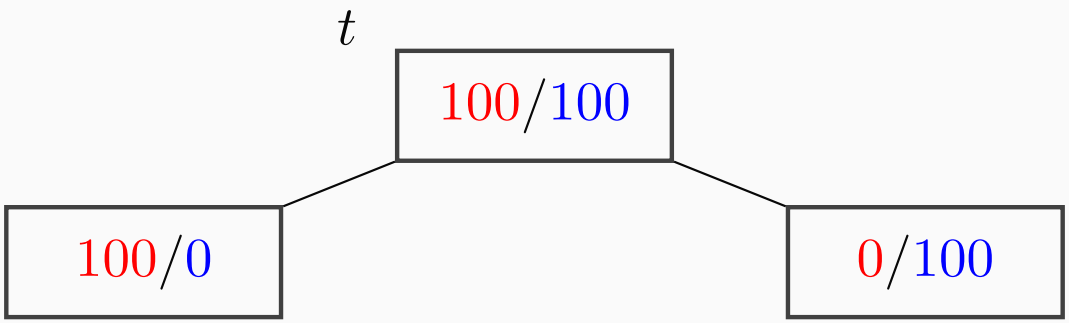
$$I(t) = -\sum_{k=1}^{K} p(\mathcal{C}_k|t) \log p(\mathcal{C}_k|t)$$

- ジニ指数 (Gini index)

$$I(t) = \sum_{k=1}^{K} p(\mathcal{C}_k|t) (1 - p(\mathcal{C}_k|t))$$

不純度の数値例

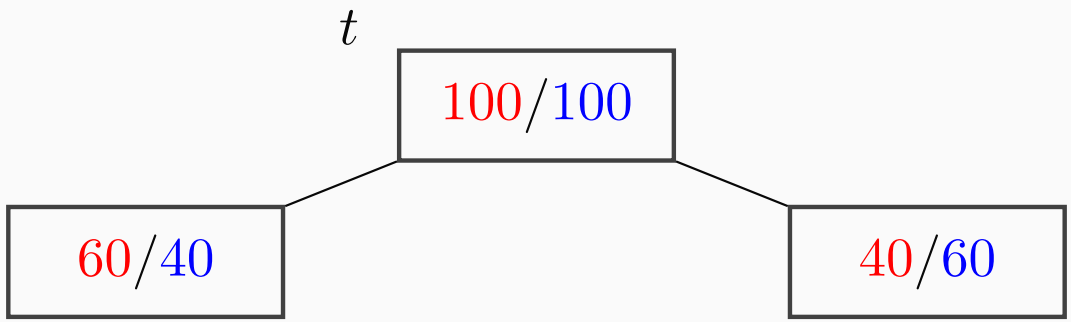
2 クラス分類において，I として交差エントロピーを使う例.



$$
\begin{aligned}
\Delta I(s,t) &= I(t) - p_L I(t_L) - p_R I(t_R) \\
&= -\frac{1}{2}\log\frac{1}{2} - \frac{1}{2}\log\frac{1}{2} - 2\cdot\frac{1}{2}(-1\log 1 - 0\log 0) \\
&= \log 2
\end{aligned}
$$

不純度の数値例

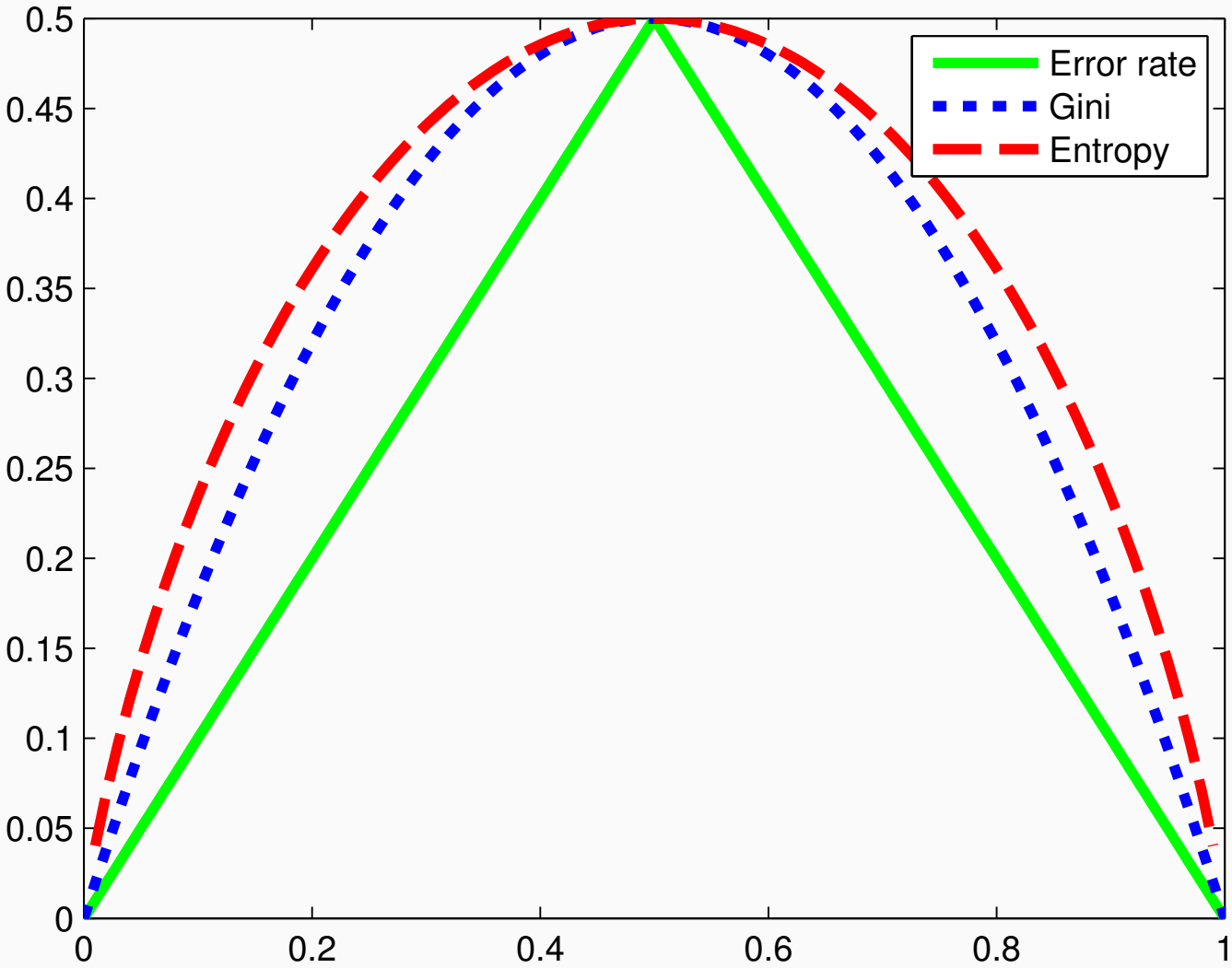
別な分割 s' を使う例.



$$\begin{aligned}\Delta I(s', t) &= I(t) - p_L I(t_L) - p_R I(t_R) \\ &= \log 2 - 2 \cdot \frac{1}{2}\left(-\frac{3}{5}\log\frac{3}{5} - \frac{2}{5}\log\frac{2}{5}\right) < \Delta I(s, t)\end{aligned}$$

前の分割 s のほうが好ましい.

不純度関数の比較


0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
0.2
0.4
0.6
0.8
1
Error rate
Gini
Entropy

木構造モデルの問題点

- 軸に合わせて分割しているので，軸と平行でない方向にうまく境界が引けない.
- 入力空間をハードに（ただ 1 つのモデルが対応するように）分割してしまう.
- 通常，回帰では滑らかな関数での近似するが，このモデルでは境界で不連続な，領域ごとの定数予測になってしまう.

参考文献

Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer.

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. (1984). Classification and regression trees. CRC press.

Freund, Y. and Schapire, R. E. (1996). Experiments with a new boosting algorithm.

Friedman, J., Hastie, T., Tibshirani, R., et al. (2000). Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). The annals of statistics.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). The Elements of Statistical Learning: data mining, inference and prediction. Springer, second edition.

Murphy, K. P. (2012). Machine Learning: A Probabilistic Perspective.

平井有三 (2012). はじめてのパターン認識. 森北出版.