テーマ３：計算論的学習理論とブースティング技法

本日の講義の目的

- 計算論的学習理論（とくに PAC 学習）速成コース

3.1. 確率的近似学習（PAC 学習）の諸定義

学習の対象：命題論理関数（論理関数）

真理値の組から真理値への関数である（このテーマでは真を $+1$，偽を -1 と表わす）．n 変数の論理関数は，n 個の ± 1 の組を ± 1 への対応させる対応表と考えてもよい．論理関数は，命題論理式で定義することもある．

論理関数の入力に相当するデータを入力例といい，それに対する結果を判別値とかラベルという．命題論理関数の学習は 2 値判別の学習ともいう．

ここでは f_* を目標関数を表わす記号とする．一般に，生徒には未知のものを $*$ 印を付けて表わすことにする．

与えらえれるデータ：事例，事例集合

事例は入力例（真理値の組）と判別値の対である．なお，$\{-1,+1\}^n$ で n 個の ± 1 の組の全体を表わす．つまり，$\{-1,+1\}^n$ は n 変数論理関数の定義域である．

事例集合は正確には事例の列である．つまり，同じ事例が重複して現われることもあるし，事例の順序が重要な場合もある．一般に，事例の順序が異なる場合には，異なった事例集合とみなすことにする．

データに対する確率分布：事例生起確率

各事例が生成される確率のこと．この事例生起確率をもとに仮説の近似度（誤判別確率）も測る．

学習の目標：確率的近似学習

(1) 学習アルゴリズムが出力すのが仮説である．仮説 h に対する（現在仮定している事例生起確率のもとでの）誤判別確率とは，次のような確率である．

$$\text{誤判別確率} \;=\; \Pr_{\boldsymbol{x}:D_*}\left[\, f_*(\boldsymbol{x}) \neq h(\boldsymbol{x}) \,\right]$$

記法： 事例生起確率には記号 D_* を用いる．本来は事例（つまり，関数の入出力の対）の発生確率であるが，出力は目標関数によって一意に決まるので，入力例に対する確率と同一視する．たとえば $D_*((-1,-1,+1)) = 0.2$ などと記述し，入力例 $(-1,-1,+1)$ が事例として現われる確率が 0.2 であることを表わす．

また,「入力 $\boldsymbol{x}$ が D_* に従う確率で発生するときに『$\cdots \boldsymbol{x} \cdots$』が起きる確率を，$\Pr_{\boldsymbol{x}:D_*}[\cdots \boldsymbol{x} \cdots]$ と記述する．さらに D_* に従って生成された m 個の事例からなる集合 S に対する確率を，$\Pr_{S:(D_*)^m}[\cdots S \cdots]$ と記述する．

(2) 学習アルゴリズムの実行に際し，アルゴリズムには，命題変数の数 n，ならびに精度パラメータ ε, δ $(0 < \varepsilon, \delta \leq 1)$ が与えられる．ε は誤判別の許容範囲を，δ はアルゴリズムの信頼度の許容範囲を各々指定するパラメータである．この n, ε, δ の 3 つを，確率的近似学習の学習パラメータと呼ぶ.

(3) 以上のような設定のもとで，次の条件を満たす学習アルゴリズムを，目標関数クラス $\mathcal{C}$ の確率的近似学習アルゴリズム（PAC 学習アルゴリズム，PAC learning algorithm）という.

$$\begin{array}{l} \text{任意の学習パラメータ } n, \varepsilon, \delta \text{ に対し，} \\ \text{任意の } n \text{ 変数論理関数 } f_* \in \mathcal{C} \text{ に対し，そして} \\ \text{事例の全体 } X \text{ 上の任意の確率分布 } D_* \text{ に対し，} \\ \Pr\limits_{S:(D_*)^m} \left[\begin{array}{c} \text{事例集合 } S \text{ 対し，学習アルゴリズムが} \\ S \text{ から求めた仮説 } h \text{ が以下を満たす} \\ (*) \quad \Pr\limits_{\boldsymbol{x}:D_*} [\ f_*(\boldsymbol{x}) \neq h(\boldsymbol{x})\ ] \ \leq \ \varepsilon \end{array} \right] \ \geq \ 1 - \delta. \end{array}$$

なお，条件 $(*)$ を満たす仮説を（目標関数の）ε-近似という.

3.2. PAC 学習の例，PAC 学習の基本定理

例: 節式の学習 節式（または論理和式）（英語では disjunctive expression）とは，リテラル (literal) （論理変数またはその否定）の論理和からなる式のことである．たとえば，$x_2 \vee x_3 \vee x_4 \vee x_6$ や $x_1 \vee x_3 \vee \overline{x_5}$ のような式である．こうした節の集合を DISJ と記述する．この節式を PAC 学習するアルゴリズムを以下に示す.（事例集合は本来，アルゴリズムの要求に対して，分布 D_* に基づいて生成して与えられるが，簡単のため，すでに与えられたものとしてアルゴリズムを記述する.）

Algorithm `DISJ-Learner`
目標関数： n 変数の節式 f_*;
入力： 学習パラメータ n, ε, δ;
出力： n 変数の節式;
事例： $S = (\ (\boldsymbol{a}_1, b_1), ..., (\boldsymbol{a}_m, b_m)\ )$，ただし，各 $1 \leq i \leq m$ に対し，
$\boldsymbol{a}_i = (a_{i,1}, ..., a_{i,n}) \in \{-1, +1\}^n$, $b_i = f_*(a_{i,1}, ..., a_{i,n})$ である;
手順：
式 t を $t = x_1 \vee \overline{x_1} \vee x_2 \vee \overline{x_2} \vee \cdots \vee x_n \vee \overline{x_n}$ とする;
for each $(a_{i,1}, ..., a_{i,n}, b_i) \in S$ **do** {
if $b_i = -1$ **then** {
各 $a_{i,j}$ に対し，$a_{i,j} = +1$ ならば x_j を，
$a_{i,j} = -1$ ならば $\overline{x_j}$ を，それぞれ t から除く; } }
得られた t を出力;
end.

このアルゴリズムは，各事例 $(\boldsymbol{a}_i, b_i)$ に対し，$b_i = +1$ ならば何もしない．一方，$b_i = -1$ のときは $t(\boldsymbol{a}_i) = +1$ の原因となるリテラルを式 t から取り除く．このように得られた式 t は（事例集合に対する）無矛盾仮説になっている．

定理 3.1. アルゴリズム `DISJ-Learner` が要求する事例数 m を，学習パラメータ n, ε, δ に対し

$$m \ \geq \ \frac{1}{\varepsilon} \ln \frac{1}{\delta} + \frac{n \ln 3}{\varepsilon}$$

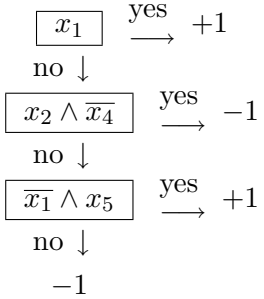
となるように定める（ln は自然対数）．このときアルゴリズムは，n, ε, δ に対し PAC アルゴリズムの条件を満たす．

例： 決定リストの学習

右図のような論理関数の表現方法を決定リストという．一般に決定リストは，単項式と真理値の組の列

$$(\ (t_1, b_1), (t_2, b_2), ..., (t_d, b_d), b\ )$$

で定義される．この各単項式を（決定リストの）分岐という．たとえば右図の決定リストは次のように表わされる．



$$(\ (x_1, +1), (x_2 \wedge \overline{x_4}, -1), (\overline{x_1} \wedge x_5, +1), -1\ )$$

分岐となる単項式を構成するリテラルの数（の最大値）を決定リストの幅と呼ぶ．一方，分岐の数を決定リストの深さという．この例は幅が 2，深さが 3 の決定リストである．

ここでは，幅が 2 の決定リストを対象とする．このように制限しても，かなりの表現力を持つため実際に利用できる場面は少なくない．この幅 2 の決定リストのクラスを 2-DL と表わす．この 2-DL の学習アルゴリズムを次のページに示す．このアルゴリズムも単項式の学習アルゴリズムと同様，与えられた事例集合に無矛盾な仮説（決定リスト）を出力する．

定理 3.2. アルゴリズム `2-DL-Learner` が要求する事例数 m を，学習パラメータ n, ε, δ に対し

$$m \ \geq \ \frac{1}{\varepsilon} \ln \frac{1}{\delta} + \frac{4n^2(6 + \ln n)}{\varepsilon}$$

となるように定める．このときアルゴリズムは，n, ε, δ に対し確率的学習アルゴリズムの条件を満たす．

Algorithm `2-DL-Learner`

目標関数： n 変数，幅 2 の決定リスト f_*；

入力： 学習パラメータ n, ε, δ；

出力： n 変数，幅 2 の決定リスト；

事例： $S = \{(\boldsymbol{a}_1, b_1), ..., (\boldsymbol{a}_m, b_m)\}$（単項式の学習と同様）；

手順：

$S_+ \leftarrow \{\ \boldsymbol{a}_i \,|\, b_i = +1\ \}$;（正事例の入力例の集合）

$S_- \leftarrow \{\ \boldsymbol{a}_i \,|\, b_i = -1\ \}$;（負事例の入力例の集合）

$\ell \leftarrow$ 空列;

while $S_+ \neq \emptyset \wedge S_- \neq \emptyset$ **do** {

（以下，$T(t)$ は t を真にする入力例の集合で $\|T(t)\|$ は要素数）

幅 2 以下の単項式 t で $T(t) \cap S_- = \emptyset$ となるものの中で

$\|T(t)\|$ が最大となるものを t_+ とする（無ければ未定義）;

幅 2 以下の単項式 t で $T(t) \cap S_+ = \emptyset$ となるものの中で

$\|T(t)\|$ が最大となるものを t_- とする（無ければ未定義）;

（t_+, t_- のどちらかは必ず定義される）

if t_- が未定義 $\vee\ \|T(t_+)\| \geq \|T(t_-)\|$ **then** {

ℓ の末尾に $(t_+, +1)$ を加える;　$S_+ \leftarrow S_+ - T(t_+)$; }

else {　（つまり，t_+ が未定義 $\vee\ \|T(t_+)\| < \|T(t_-)\|$）

ℓ の末尾に $(t_-, -1)$ を加える;　$S_- \leftarrow S_- - T(t_-)$; }

}（ここまで **while** の範囲）

if $S_+ \neq \emptyset$　**then** ℓ の末尾に $+1$ を加える;（$S_- = \emptyset$ のはず）

else　ℓ の末尾に -1 を加える;（$S_+ = \emptyset$ のはず）

得られた ℓ を出力;

end.

こうしたアルゴリズムの設計には,「オッカムの定理」と呼ばれる PAC 学習アルゴリズム構成の指針となる定理がある.

定理 3.3. 目標関数クラス $\mathcal{C}$ に対し，無矛盾仮説生成アルゴリズム L を考える．このアルゴリズムが，n 変数の目標関数から生成される事例数 m の事例集合に対して出力する仮説クラスを $\mathcal{H}_{n,m}$ とし，その要素数を $M(n,m)$ とする.

このとき，任意に与えられた学習パラメータ n, ε, δ に対し，事例数 m を

$$m \ \geq\ \frac{1}{\varepsilon} \ln \frac{1}{\delta} + \frac{\ln M(n,m)}{\varepsilon}$$

を満たすように設定できれば，アルゴリズム L は確率的学習アルゴリズムの条件を満たす.

参考文献

[1] 金森敬文，畑埜晃平，渡辺治，ブースティング，森北出版，2006.