13.1 統計的パターン認識の概要

画像内の対象物が何であるかを判定することを画像認識と呼び，近年多くの研究者がこの課題に取り組んでいる．画像認識において，最も基本的な問題として，画像中に含まれる個々の対象物に対し，それらが何であるかを自動的に判定することが挙げられる．例えば，胸部X線画像を構成している個々の要素に対して，それが肋骨であるか，血管であるか，あるいは腫瘤であるかを自動的に判定する，といった問題である．本章では，このような判定を自動化するのに役立つパターン認識 (pattern recognition) 技術の基礎と医用画像応用において有用と思われる手法を紹介する．

パターン認識とは，観測されたパターン，またはサンプル (sample) をあらかじめ定められたクラス (class) やカテゴリー (category) と呼ばれる概念のうちの一つに対応させる処理を意味する [1]．例えば，手書き文字認識では，手書き文字 (パターン，サンプル) を観測したときに，その文字名 (クラス，カテゴリー) を決定することが目的となる．このパターン認識を計算機で実現するには，次のような構成とするのが一般的である．まず，パターンを計算機に入力する（観測）．次に，そのパターンに対して，位置や大きさを揃えたりノイズを除去したりする前処理 (preprocessing) を施す．その後，パターンから認識に有効な特徴を抽出するための特徴抽出 (feature extraction) を行う．最後に，その特徴を予め用意しておいた識別器 (classifier) に入力し，観測したパターンが属すであろうクラスを決定するための識別 (classification) を行う．上記のステップのうち，パターンの入力，前処理，特徴抽出の部分は認識したい対象によって異なるため，本章では主に識別部について述べることにする．

識別部では，属するクラスが未知である未知サンプル（test sample）を観測したとき，そのクラスを決定することになるが，この決定のためのルールを人手で設計することは簡単ではない．例えば腫瘍の自動判定を考えた場合，一口に腫瘤といってもサイズ，形状，濃淡など，さまざまなパターンが存在するためである．そこで，クラスが既知のサンプルである訓練サンプル (training sample) を多数収集し，それらの持つ統計的な性質を利用して任意のパターンとクラスとの対応付けのためのルールを自動的に設計する統計的パターン認識 (statistical pattern recognition) が用いられることが多い．統計的パターン認識は世界中で盛んに研究が行われており，多くの教科書が出版されている (例えば [1]〜[7] など)．

統計的パターン認識では，任意のパターン $\boldsymbol{x}$ は d 次元ユークリッド空間 $\mathbb{R}^d$ の部分集合であるパターン空間 (pattern space) $\mathcal{D}$ に属すると仮定し，$\boldsymbol{x}$ を d 個の実数の組である d 次元実ベクトル

$$\boldsymbol{x} = (x_1, \cdots, x_d)^\top \in \mathcal{D} \in \mathbb{R}^d \tag{1.13.1}$$

によって表現する．ここで，$\top$ は転置を表す．以後，パターンは列ベクトルで統一する．そして，この $\boldsymbol{x}$ を連続的な確率変数として扱えば，$\boldsymbol{x}$ の分布を表す確率密度関数 (probability density function) $p(\boldsymbol{x}) \geq 0$ を定義できる．$p(\boldsymbol{x})$ は確率密度関数であるから

$$\int_{\mathcal{D}} p(\boldsymbol{x}) d\boldsymbol{x} = 1 \tag{1.13.2}$$

を満たす．直感的には $p(\boldsymbol{x})$ の値が大きければ大きいほど，その $\boldsymbol{x}$ はより観測されやすいパターンであるといえるが，$p(\boldsymbol{x})$ がどのような形であるのかを事前に知ることは (特に d

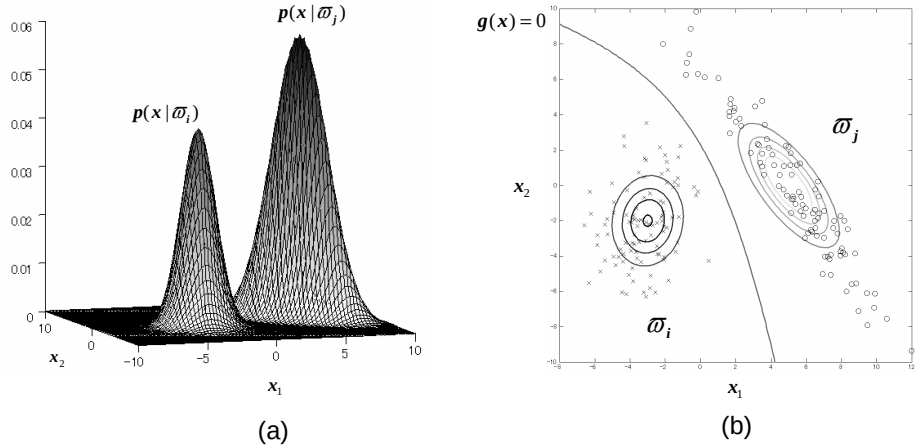

図 1.13.1 (a)：$d = 2$ の場合の二つのクラス ω_i と ω_j の確率密度関数 $p(\boldsymbol{x}|\omega_i)$ と $p(\boldsymbol{x}|\omega_j)$ の例．(b)：各クラスの事前確率と確率密度関数に従って生起したサンプルの例．$\times$ が ω_i に属するサンプル，$\circ$ が ω_j に属するサンプルを表す．同じ値となる尤度が等高線により表現されている．二つのクラスの間を通る曲線は，識別関数を $g_j(\boldsymbol{x}) = P(\omega_j)p(\boldsymbol{x}|\omega_j)$ とした場合の決定境界 $g(\boldsymbol{x}) = 0$ である．

が大きい場合) 非常に困難である．なお，$p(\boldsymbol{x})$ の値は確率そのものではないので，確率と区別するためにこの値を尤度 (likehood) とよぶことにする．今，c 個のクラス $\omega_1, ..., \omega_c$ に対するパターン認識を考えた場合，この確率密度関数はクラスごとに存在すると考えるのが自然である．そこで，ω_j におけるパターン $\boldsymbol{x}$ の分布を表す確率密度関数を $p(\boldsymbol{x}|\omega_j)$ と書くことにする（図 1.13.1 (a) に一例を示す）．

統計的パターン認識では，パターン $\boldsymbol{x}$ を連続的な確率変数として扱うのと同様に，c 個のクラス $\omega_1, ..., \omega_c$ を離散的な確率変数として扱う．ここでは，その確率を $P(\omega_j) \geq 0$ で表すことにする．$P(\omega_j)$ は離散的な確率なので，総和は 1 となる．

$$\sum_{j=1}^{c} P(\omega_j) = 1 \tag{1.13.3}$$

この $P(\omega_j)$ は，パターン $\boldsymbol{x}$ を観測する前のクラスに関する確率のため，クラスの事前確率 (a prior probability) とよぶ．統計的パターン認識では，これらの未知の確率密度関数と事前確率に従って生起した総数 n 個の訓練サンプル $\boldsymbol{x}_1, ..., \boldsymbol{x}_n \in \mathbb{R}^d$ を用いて識別のためのルールである識別則 (decision rule) を定めることが目標となる．なお，各訓練サンプルがどのクラスから観測されたかを表す変数をラベル (label) とよび，特に第 i 訓練サンプルに対するラベルを $y_i \in \{\omega_1, \cdots, \omega_c\}$ で表すことにする．

識別では，未知サンプル $\boldsymbol{x}$ を c 個のクラスのうちいずれか一つに対応付けることになるが，このときクラスを判定するのに使用する $\boldsymbol{x}$ の関数を識別関数 (discriminant function) とよび $g(\boldsymbol{x})$ で表す．特に，クラス ω_j の判定に用いる識別関数を $g_j(\boldsymbol{x})$ で表す．この識別関数を用いてクラスを判定する規則が識別則であり，例えば，$g_j(\boldsymbol{x})$ が最大となるクラスへ

$\boldsymbol{x}$ を分類する識別則は

$$j^* = \arg \max_{j=1,\ldots,c} \{g_j(\boldsymbol{x})\} \Rightarrow \boldsymbol{x} \in \omega_{j^*} \tag{1.13.4}$$

と書ける．ここで arg max とは最大値を与える引数を意味する．すなわち，上記の場合，$g_j(\boldsymbol{x})$ が最大となる j が j^* であり，その引数に対応するクラス ω_{j^*} に $\boldsymbol{x}$ を識別することを意味している．識別関数と識別則の組を識別器とよぶ．また，$g(\boldsymbol{x}) = g_i(\boldsymbol{x}) - g_j(\boldsymbol{x}) = 0$ を満たす集合を，クラス ω_i と ω_j の決定境界 (decision boundary) とよぶ（図 1.13.1 (b) に一例を示す）．統計的パターン認識では，識別器を有限の訓練サンプルを用いて設計するが，このとき未知サンプルを誤ったクラスへ識別する確率ができるだけ小さい識別器，すなわち汎化能力 (generalization ability) の高い識別器を設計することが目標となる．

なお，統計的パターン認識では，観測された訓練サンプルが独立同一分布 (independent and identically distributed, i.i.d.) に従うことが暗に仮定されている．i.i.d. に従う訓練サンプルとは，次の手順を何度も繰り返して収集されたものを意味する．

1. クラスを $P(\omega_j)$ にしたがってランダムに選ぶ.
2. 選ばれたクラスが ω_k だった場合，ω_k に属する訓練サンプル，すなわち $y_i = \omega_k$ となる $\boldsymbol{x}_i$ を確率密度関数 $p(\boldsymbol{x}_i|\omega_k)$ に従ってランダムに取り出す.

実際には，$p(\omega)$ や $p(\boldsymbol{x}_i|\omega_k)$ は未知であり，これらを直接利用して i.i.d. に従うパターンを収集することはできないが，可能な限り i.i.d. に従うよう網羅的にパターンを収集することは重要である．

13.2 パターン認識の具体例

ここで，手書き数字認識を例として，具体的にパターン認識の問題を考えてみよう．図 1.13.2 (a) に本章で用いる手書き数字の一例を示す．これらは複数の人物が書いた手書き数字をスキャナで取り込み（観測），一つの数字が 16×16 ピクセルの濃淡画像となるよう前処理を施して作成されたものである．このようにして作成した 9298 個の手書き数字画像と，それぞれに対応するラベルを用いてパターン認識の実験を行う．そこで，9298 個の手書き数字画像をランダムに二つのグループに分け，一方を訓練サンプルとし，残りを未知サンプルとした．したがって，訓練サンプルと未知サンプルはそれぞれ 4649 個となる．手書き数字認識の問題は，未知サンプルを 0 から 9 のクラスのうち，どれか一つに分類する問題であるのでクラス数は $c = 10$ となる．

パターン認識では特徴抽出を経てサンプル（パターン）を d 次元実ベクトルで表す必要があるが，ここでは画像の画素値を並べたものを特徴として考える．すなわち，$16 \times 16 = 256$ 個の画素値を左上から順番に縦に並べた 256 次元のベクトルを $\boldsymbol{x}$ とするが，各画素値は 0（黒）から 255（白）の 256 階調の離散値で与えられているので，ここでは $\boldsymbol{x}$ の成分を以下のように規格化して実ベクトルに変換する．

$$x_i \leftarrow \frac{x_i - m}{\sigma}, \quad (i = 1, ..., d) \tag{1.13.5}$$

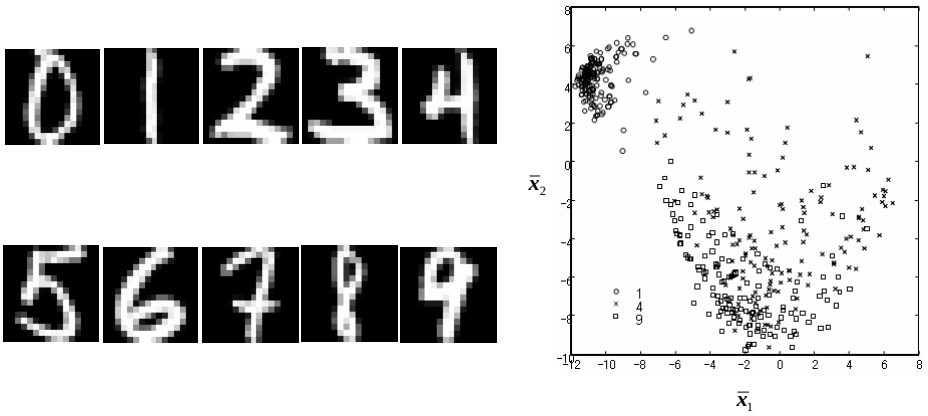

図 1.13.2 (a) : 手書き数字パターンの例．(b) : 手書き文字，1，4，9 の二次元空間での分布．

ここで mとσ は，それぞれ以下の式で与えられる $\boldsymbol{x}$ の成分の平均値と標準偏差である．

$$m = \frac{1}{d}\sum_{i=1}^{d} x_i, \quad \sigma = \sqrt{\frac{1}{d}\sum_{i=1}^{d}(x_i - m)^2} \tag{1.13.6}$$

上記のようにして得られたサンプルは，256 次元のパターン空間上の 1 点として表現できるが，それらがどのように分布しているかを直接観察することはできない．そこで，訓練サンプルに主成分分析 (principal component analysis, PCA) を適用して 256 次元のサンプルを二次元に圧縮して観察することにする．PCA による次元の圧縮は以下の手順で行う．まず全訓練サンプルから $d \times d$ の標本共分散行列

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}})(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}})^\top \tag{1.13.7}$$

を求める．ここで $\hat{\boldsymbol{\mu}} = \sum_{i=1}^{n} \boldsymbol{x}_i/n$ である．次に，標本共分散行列を固有値分解し，値の大きい上位二つの固有値に対応する d 次元の固有ベクトル $\boldsymbol{u}_1, \boldsymbol{u}_2$ を求める．この二つの固有ベクトルの張る二次元アフィン部分空間へ任意のパターン $\boldsymbol{x}$ を正射影したとき，二次元空間上の座標値は $(\bar{x}_1, \bar{x}_2) = (\boldsymbol{u}_1^\top(\boldsymbol{x} - \hat{\boldsymbol{\mu}}), \boldsymbol{u}_2^\top(\boldsymbol{x} - \hat{\boldsymbol{\mu}}))$ として与えられる．これにより，d 次元空間における大まかな訓練サンプルの分布を二次元空間で観察することができる．

図 1.13.2 (b) は PCA により手書き数字の 1，4，9 の訓練サンプルの分布を二次元平面で表したものである．見やすいように各クラスのサンプル数を 150 にしている．図から，同じクラスに属するサンプルは塊となって分布しているのがわかる．このような塊をクラスタ (cluster) とよぶ．特に，数字の 1 は，他のクラスから離れた場所に凝集している．このため，数字の 1 を識別することは他のクラスに比べて容易であると予想できる．一方，4 と 9 はお互いの形状が似ていることから，分布が広い範囲で重なっていることがわかる．したがって，これらを正しく識別することは難しいと予想される．

表 1.13.1 最小距離識別法を用いたときの混同行列.

class	0	1	2	3	4	5	6	7	8	9	total
0	680	0	2	1	5	7	72	1	17	1	786
1	0	642	0	0	1	0	2	0	2	0	647
2	10	1	377	15	11	6	6	9	19	0	454
3	5	0	2	369	2	19	0	4	15	2	418
4	4	15	7	0	356	1	9	1	8	42	443
5	10	0	3	18	10	286	8	0	15	5	355
6	32	7	9	0	3	3	355	0	5	0	414
7	2	3	1	0	7	0	0	348	2	39	402
8	4	9	5	15	5	7	0	1	273	12	331
9	0	10	0	1	45	0	0	25	2	316	399
total	747	687	406	419	445	329	453	389	358	417	4649

上記の予想がどの程度正しいかを，実際に識別を行って検証してみよう．ここでは識別器として後述する最小距離識別法を用いた．表 1.13.1 に未知サンプルに対する混同行列 (confusion matrix) を示す．混同行列は，ω_i に属するパターンが ω_j に分類された個数を第 i 行 j 列の要素に持つ行列である．例えば，数字の 0 が正しく分類された個数は 680 個であり，6 と誤分類された個数は 72 個であることがわかる．この行列の各行の総和は，各クラスに属する未知サンプル数を表す．一方，各列の総和は，各クラスへ分類された未知サンプル数の総和となる．表から，PCA の観察で予想された通り，数字の 1 はほとんどが正しく識別されており，数字の 4 と 9 は互いに誤識別を起こしていることがわかる．

ところで認識率 (recognition rate)，または識別率 (classification accracy) とは正しく識別されたサンプル数をサンプルの総数で割って百分率で表したものである．言い換えれば，識別率は混同行列の対角要素の総和をサンプルの総数で割ることで求められる．したがって，今回の実験における未知サンプルに対する識別率は表 1.13.1 から 86.1% であると計算できる．一方，誤識別率，またはエラー率 (error rate) とは 100− 識別率（%）で与えられるもので，今回の実験では未知サンプルに対するエラー率は 13.9% となる．なお，サンプル数に偏りがあったり，クラスによって識別の難しさが大きく異なったりする場合には，識別率を各クラスごとに求め，それらの平均値を識別率とする場合もある．

このように，実際のパターン認識では，真の確率密度関数 $p(\omega_j|\boldsymbol{x})$ と事前確率 $P(\omega_j)$ が未知であるため実験的に未知サンプルに対する識別率を求めることで，識別器の汎化性能を評価することが多い．しかし，もし $p(\omega_j|\boldsymbol{x})$ と $P(\omega_j)$ が既知であった場合，識別率を最大（エラー率を最小）にする識別則はどのようなものになるのであろうか？次節ではそのような識別則について考える．

13.3 最大事後確率則

ここでは $P(\omega_j)$ と $p(\omega_j|\boldsymbol{x})$ が既知であると仮定する．このとき，これらの間には

$$p(\boldsymbol{x}) = \sum_{j=1}^{c} P(\omega_j)p(\boldsymbol{x}|\omega_j) \tag{1.13.8}$$

の関係が成り立つ．さらに，$\boldsymbol{x}$ を観測したとき，それがクラス ω_j から観測される確率を $P(\omega_j|\boldsymbol{x})$ で表す．この $P(\omega_j|\boldsymbol{x})$ は事後確率 (a posteriori probability) とよばれるもので，パターン $\boldsymbol{x}$ を観測した後に，それがクラス ω_j に属している確率を表している．$P(\omega_j|\boldsymbol{x})$ も確率であるので $\sum_{j=1}^{c} P(\omega_j|\boldsymbol{x}) = 1$ を満たす．

ここで，ω_j と $\boldsymbol{x}$ が同時に生起する分布を表す同時分布を $p(\omega_j, \boldsymbol{x})$ と書くとすると，$p(\omega_j, \boldsymbol{x})$ と条件付き確率密度関数の間には

$$p(\omega_j, \boldsymbol{x}) = P(\omega_j|\boldsymbol{x})p(\boldsymbol{x}) = P(\omega_j)p(\boldsymbol{x}|\omega_j) \tag{1.13.9}$$

の関係が成り立つ．ここから次のベイズの公式 (Bayes formula) を導くことができる．

$$P(\omega_j|\boldsymbol{x}) = \frac{P(\omega_j)p(\boldsymbol{x}|\omega_j)}{p(\boldsymbol{x})} \tag{1.13.10}$$

この公式は事前確率 $P(\omega_j)$ を事後確率 $P(\omega_j|\boldsymbol{x})$ へ変換できることを示している．$P(\omega_j|\boldsymbol{x})$ はパターン $\boldsymbol{x}$ を観測したとき，それがクラス ω_j に属する確からしさを表しているので，これが最大のクラスを識別結果とするのは自然である．すなわち識別則は

$$\max_{j=1,\ldots,c} \{P(\omega_j|\boldsymbol{x})\} = P(\omega_k|\boldsymbol{x}) \Rightarrow \boldsymbol{x} \in \omega_k \tag{1.13.11}$$

となる．この識別則を最大事後確率則 (maximum a posteriori probability rule) とよぶ．この識別則は，パターンが誤って分類される確率を最小にする最小誤識別則と，損失がクラスで一定の場合のベイズ決定則と一致する [7]．すなわち，$P(\omega_j)$ と $p(\boldsymbol{x}|\omega_j)$ が既知である場合には，この識別則により理論上，最大の識別率を達成することができる．

13.4 パラメトリック学習とノンパラメトリック学習

最大事後確率則は識別率を最大にすることから，すべてのパターン認識問題に最大事後確率則を適用すれば良いと思うかもしれない．しかし，最大事後確率則を用いるには事前確率 $P(\omega_j)$ と確率密度関数 $p(\boldsymbol{x}|\omega_j)$ を予め知る必要があるが，一般には知りえないものである．そこで，これらを訓練サンプルから推定することを考える．このうち事前確率 $P(\omega_j)$ は，各クラスで $1/c$ の均等として推定する，またはクラス ω_j に属する訓練サンプルの数が n_j の場合，訓練サンプル数の比 n_j/n によって推定することが多い．一方で，$p(\boldsymbol{x}|\omega_j)$ を精度よく推定するためには訓練サンプルを大量に集める必要があったり，推定のための計算量が多くなったりするなど，それほど容易なことではない．そこで，$p(\boldsymbol{x}|\omega_j)$ を直接推定するのではなく，簡単なモデル，例えばガウス分布を $p(\boldsymbol{x}|\omega_j)$ の替わりとして用いるアプローチが考えられる．この場合はモデルの持つパラメータ (ガウス分布の場合は平均

と共分散行列) を訓練サンプルを用いて推定することが主な問題となるが，これは最尤推定 (maximum-likelihood estimation) やベイズ推定 (Bayesian parameter estimation) により達成される．なお，推定されたパラメータで記述されたモデルを用いて識別器を構成することをパラメトリックな学習 (parametric learning) とよぶ [1]．一方，モデルを用いず，訓練サンプルから直接識別器を構築する方法をノンパラメトリックな学習 (nonparametric learning) とよぶ．このようなものとしては，後述する最近傍決定則，サポートベクトルマシンや Adaboost などが挙げられる．

13.5 パラメトリックな学習の例

ここではガウス分布をモデルとしたパラメトリックな学習について考える．多次元ガウス分布は図 1.13.1(a) に示したような釣鐘型の形状を持つ分布であり，d 次元のパターン $\boldsymbol{x}$ に対するクラス ω_j におけるガウス分布は次式で与えられる．

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}_j|^{1/2}} \exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_j)^\top \boldsymbol{\Sigma}_j^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_j)\right\} \tag{1.13.12}$$

ここで $\boldsymbol{\mu}_j$ は $d \times 1$ の母平均 (population mean)，$\boldsymbol{\Sigma}_j$ は $d \times d$ の母共分散行列 (population covariance matrix) である．$|\boldsymbol{\Sigma}_j|$ と $\boldsymbol{\Sigma}_j^{-1}$ はそれぞれ $\boldsymbol{\Sigma}_j$ の行列式と逆行列を表す．パラメトリックな学習を行うためには，$\boldsymbol{\mu}_j$ と $\boldsymbol{\Sigma}_j$ を訓練サンプルから推定する必要があるが，ここでは最尤推定とよばれる方法によって推定する方法を示す．

13.5.1 最尤推定

クラス ω_j に属する n_j 個の訓練サンプルを $\mathcal{X}_j = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_{n_j}\}$ とし，これらが i.i.d. に従っているとする．最尤推定では，$\mathcal{X}_j$ が生起しやすい $\boldsymbol{\mu}_j$ と $\boldsymbol{\Sigma}_j$ を求めるために，以下のような尤度関数 (likelihood function) と呼ばれる，パラメータに関する関数を用いる．

$$L(\mathcal{X}_j|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) = \prod_{i=1}^{n_j} \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \tag{1.13.13}$$

最尤推定では，この関数の値が最大となるようなパラメータをクラス ω_j のモデルのパラメータとする．ただし，ガウス分布の場合には，尤度関数の対数をとった

$$\ln L(\mathcal{X}_j|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) = \sum_{i=1}^{n_j} \ln \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \tag{1.13.14}$$

である対数尤度 (log likelihood) で計算する方が簡単であり，求まるパラメータも同じなのでここからは対数尤度で考えることにする．なお，ln はネイピア数 $e = 2.7182\cdots$ を底とする対数（自然対数）である．

尤度関数が最大となる必要条件は，各パラメータで尤度関数を偏微分したものがゼロとなることである．したがって，式 (1.13.14) の対数尤度を最大にするパラメータは

$$\sum_{i=1}^{n_j} \frac{\partial}{\partial \boldsymbol{\mu}_j} \ln L(\boldsymbol{x}_i|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) = \mathbf{0}, \quad \sum_{i=1}^{n_j} \frac{\partial}{\partial \boldsymbol{\Sigma}_j} \ln L(\boldsymbol{x}_i|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) = \mathbf{O} \tag{1.13.15}$$

を解くことによって求めることができる．ここで $\mathbf{0}$ は d 次元のゼロベクトル，$\mathbf{O}$ は $d \times d$ のゼロ行列である．これを実際に解くと最尤推定量が

$$\hat{\boldsymbol{\mu}}_j = \frac{1}{n_j} \sum_{i=1}^{n_j} \boldsymbol{x}_i \tag{1.13.16}$$

$$\hat{\boldsymbol{\Sigma}}_j = \frac{1}{n_j} \sum_{i=1}^{n_j} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_j)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_j)^\top \tag{1.13.17}$$

として得られる．すなわち，クラス ω_j における $p(\boldsymbol{x}|\omega_j)$ を，ガウス分布を使って推定したものを $\hat{p}(\boldsymbol{x}|\omega_j)$ とすると，それは ω_j に属する訓練サンプルから求まる標本平均 $\hat{\boldsymbol{\mu}}_j$ と標本共分散行列 $\hat{\boldsymbol{\Sigma}}_j$ によって

$$\hat{p}(\boldsymbol{x}|\omega_j) = \frac{1}{(2\pi)^{d/2}|\hat{\boldsymbol{\Sigma}}_j|^{1/2}} \exp\left\{-\frac{1}{2}(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_j)^\top \hat{\boldsymbol{\Sigma}}_j^{-1}(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_j)\right\} \tag{1.13.18}$$

として与えられる．

上記はガウス分布をモデルにした場合の例であるが，他のモデルであっても，それが確率密度関数であれば同様の手順で最尤推定量を求めることが可能である．

13.5.2 $\hat{p}(\boldsymbol{x}|\omega_j)$ を利用した最大事後確率則

ここで，式 (1.13.18) で与えられる $\hat{p}(\boldsymbol{x}|\omega_j)$ を利用して，最大事後確率則によりパターン $\boldsymbol{x}$ を識別することを考える．事前確率の推定値を訓練サンプル数の比 $\hat{P}(\omega_j) = n_j/n$ とした場合，事後確率の推定値はベイズの公式から

$$\hat{P}(\omega_j|\boldsymbol{x}) = \frac{\hat{P}(\omega_j)\hat{p}(\boldsymbol{x}|\omega_j)}{\hat{p}(\boldsymbol{x})} \tag{1.13.19}$$

で与えられる．ここで $\hat{p}(\boldsymbol{x}) = \sum_{j=1}^{c} \hat{P}(\omega_j)\hat{p}(\boldsymbol{x}|\omega_j)$ である．この事後確率の推定値を用いて $\boldsymbol{x}$ を識別するには，$\hat{p}(\boldsymbol{x})$ がクラスに依らないことに注意すると，ω_j の識別関数を

$$g_j(\boldsymbol{x}) \stackrel{\text{def}}{=} \hat{P}(\omega_j)\hat{p}(\boldsymbol{x}|\omega_j) \tag{1.13.20}$$

として，これが最大となるクラスに $\boldsymbol{x}$ を識別することと同じである．ところで，右辺の対数をとっても $g_j(\boldsymbol{x})$ の大小関係は変わらないので，簡略のため以後は対数をとった

$$g_j(\boldsymbol{x}) = \ln\{\hat{P}(\omega_j)\hat{p}(\boldsymbol{x}|\omega_j)\} = \ln \hat{P}(\omega_j) + \ln \hat{p}(\boldsymbol{x}|\omega_j) \tag{1.13.21}$$

について考える．式 (1.13.21) に式 (1.13.18) を代入し，クラスに依存しない項を省くと

$$g_j(\boldsymbol{x}) = -\frac{1}{2}\left\{(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_j)^\top \hat{\boldsymbol{\Sigma}}_j^{-1}(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_j) + \ln|\hat{\boldsymbol{\Sigma}}_j| - 2\ln \hat{P}(\omega_j)\right\} \tag{1.13.22}$$

となり，$\boldsymbol{x}$ に関して 2 次の関数となる．したがって，この識別関数を用いた場合の決定境界は図 1.13.1(b) に示したような二次超曲面で与えられる．なお，逆行列 $\hat{\boldsymbol{\Sigma}}_j^{-1}$ や行列式 $|\hat{\boldsymbol{\Sigma}}_j|$

がうまく求められない場合には，$d \times d$ の単位行列 $\mathbf{I}$ と，正の値 γ を用いて $\hat{\mathbf{\Sigma}}_j + \gamma\mathbf{I}$ を $\hat{\mathbf{\Sigma}}_j$ の代わりに用いることが多い．このような方法を正則化 (regularization) とよぶ．なお，上式の括弧内に含まれる

$$D_m(\boldsymbol{x}, \hat{\boldsymbol{\mu}}_j) \stackrel{\text{def}}{=} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_j)^\top \hat{\mathbf{\Sigma}}_j^{-1} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_j) \tag{1.13.23}$$

はマハラノビス距離 (Mahalanobis distance) とよばれるもので，しばしばユークリッド距離

$$D(\boldsymbol{x}, \hat{\boldsymbol{\mu}}_j) \stackrel{\text{def}}{=} \sqrt{(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_j)^\top (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_j)} \tag{1.13.24}$$

の代わりにパターン間の非類似度を測る尺度として用いられる．この場合，$\boldsymbol{x}$ は $D_m(\boldsymbol{x}, \hat{\boldsymbol{\mu}}_j)$ が最小となるクラスに属すると識別される．

ここで，ガウス分布を用いたモデルにおいて，各クラスの共分散行列 $\mathbf{\Sigma}_j$ が，全て等しい行列 $\mathbf{\Sigma}_T$ で与えられる場合を考えよう．このとき，クラス ω_j における対数尤度は

$$\ln L(\mathcal{X}_j | \boldsymbol{\mu}_j, \mathbf{\Sigma}_T) = \sum_{j=1}^{c} \sum_{\boldsymbol{x}_i \in \mathcal{X}_j} \ln \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{\mu}_j, \mathbf{\Sigma}_T) \tag{1.13.25}$$

として与えられる．最尤推定によりパラメータを推定すると，平均の推定量は式 (1.13.16) の $\hat{\boldsymbol{\mu}}_j$ で与えられるが，$\mathbf{\Sigma}_T$ の推定量は

$$\hat{\mathbf{\Sigma}}_T = \frac{1}{n} \sum_{j=1}^{c} \sum_{\boldsymbol{x}_i \in \mathcal{X}_j} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_j)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_j)^\top = \sum_{j=1}^{c} \hat{P}(\omega_j) \hat{\mathbf{\Sigma}}_j \tag{1.13.26}$$

で与えられることがわかる．この行列は全共分散行列 (total covariance matrix) とよばれる．この $\hat{\mathbf{\Sigma}}_T$ と $\hat{\boldsymbol{\mu}}_j$ をパラメータとして持つガウス分布を $\hat{p}(\boldsymbol{x}|\omega_j)$ として用いた場合，式 (1.13.21) の識別関数は（クラスに依存しない項を省くことで）

$$g_j(\boldsymbol{x}) = \boldsymbol{x}^\top \hat{\mathbf{\Sigma}}_T^{-1} \hat{\boldsymbol{\mu}}_j - \frac{1}{2} \hat{\boldsymbol{\mu}}_j^\top \hat{\mathbf{\Sigma}}_T^{-1} \hat{\boldsymbol{\mu}}_j + \ln \hat{P}(\omega_j) \tag{1.13.27}$$

となる．すなわち，$\boldsymbol{x}$ に関して 1 次の関数となり，決定境界は超平面（直線や平面）となる．このような識別関数を線形識別関数 (linear discriminant function) とよび，線形識別関数を用いてパターンを識別する方法を線形識別法とよぶ．ここで二つのクラス ω_i と ω_j を式 (1.13.27) の識別関数を用いて分離することを考える．その場合の決定境界は

$$g(\boldsymbol{x}) = g_i(\boldsymbol{x}) - g_j(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b = 0 \tag{1.13.28}$$

と与えられる．ただし

$$\boldsymbol{w} = \hat{\mathbf{\Sigma}}_T^{-1} (\hat{\boldsymbol{\mu}}_i - \hat{\boldsymbol{\mu}}_j), \quad b = -\frac{1}{2} (\hat{\boldsymbol{\mu}}_i - \hat{\boldsymbol{\mu}}_j)^\top \hat{\mathbf{\Sigma}}_T^{-1} (\hat{\boldsymbol{\mu}}_i - \hat{\boldsymbol{\mu}}_j) + \ln \frac{n_i}{n_j} \tag{1.13.29}$$

である．このようにして二つのクラスの決定境界を決める方法を，フィッシャーの線形判別分析 (Fisher's linear discriminant analysis) とよぶ．

最後に，全てのクラスで共分散行列が $d \times d$ の単位行列 $\mathbf{I}$ で与えられるガウス分布をモデルとして用いる場合を考える．この場合，推定すべきパラメータは $\boldsymbol{\mu}_j$ であり，それは最

尤推定により式 (1.13.16) の $\hat{\boldsymbol{\mu}}_j$ で与えられる．さらに事前確率が全てのクラスで等しいと仮定した場合には，式 (1.13.21) の識別関数は（クラスに依存しない項を省くことで）

$$g_j(\boldsymbol{x}) = 2\hat{\boldsymbol{\mu}}_j^\top \boldsymbol{x} - \|\hat{\boldsymbol{\mu}}_j\|^2 \tag{1.13.30}$$

となる．この方法が，最小距離識別法 (minimum distance method) であり，先述の実験で用いたものである．最小距離識別法も線形識別関数の一種であるので，二つのクラス ω_i と ω_j の決定境界 $g(\boldsymbol{x})$ は

$$\boldsymbol{w} = (\hat{\boldsymbol{\mu}}_i - \hat{\boldsymbol{\mu}}_j), \quad b = -\frac{1}{2}\left(\|\hat{\boldsymbol{\mu}}_i\|^2 - \|\hat{\boldsymbol{\mu}}_j\|^2\right) \tag{1.13.31}$$

とおけば，$g(\boldsymbol{x}) = g_i(\boldsymbol{x}) - g_j(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b = 0$ として表現できる．

13.5.3 混合分布

ここまでは $\hat{p}(\boldsymbol{x}|\omega_j)$ が単一のガウス分布に従う場合を考えてきた．この場合，各クラスのパターンが一つの山を持つ単峰性 (unimodal) の確率密度関数に従って生起していれば高い識別率が期待できるが，複数の山を持つ多峰性 (multimodal) の場合には，うまく認識できないことが予想される．そこで，多峰性を持つ分布を複数のガウス分布の線形結合 (係数をかけて足し合わせたもの) であるガウス混合モデル (Gaussian mixture model) で表すことを考え，これを訓練サンプルから最尤推定で推定することを考える．ただし，この場合の推定量は解析に求めることができないため，反復的なアルゴリズムである EM アルゴリズム (expectation maximization algorithm) により推定する方法を示す．なお，ここではガウス混合モデルに対する EM アルゴリズムを示すが，EM アルゴリズムは汎用的なパラメータ推定手法であり，ガウス分布以外のモデルであっても，それが確率密度関数であれば同様の手順でパラメータを推定することが可能である．

ガウス混合モデルは，K 個のガウス分布とそれらを線形結合するための係数 $\pi_1, ..., \pi_K$ を用いて定義される．すなわち，K 個のガウス分布のうち，j 番目のものが式 (1.13.12) で与えられ，それに対する係数を π_j とするとガウス混合モデルは

$$\mathcal{M}(\boldsymbol{x}|\boldsymbol{\theta}) \stackrel{\text{def}}{=} \sum_{j=1}^{K} \pi_j \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \tag{1.13.32}$$

によって定義される．ここで，$\boldsymbol{\theta}$ はガウス混合モデルに関する全てのパラメータを表す．

$$\boldsymbol{\theta} = (\pi_1, ..., \pi_K, \boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_K, \boldsymbol{\Sigma}_1, ..., \boldsymbol{\Sigma}_K) \tag{1.13.33}$$

これらのパラメータを n 個の訓練サンプル $\mathcal{X} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\}$ を用いて推定する場合，尤度関数は

$$L(\mathcal{X}|\boldsymbol{\theta}) = \prod_{i=1}^{n} \mathcal{M}(\boldsymbol{x}_i|\boldsymbol{\theta}) \tag{1.13.34}$$

となる．ただし，$\mathcal{M}(\boldsymbol{x}|\boldsymbol{\theta})$ が確率密度関数となるためには，各係数が 0 以上で総和が 1 となるようにしなければならない．

$$\pi_j \geq 0 \ (j = 1, ..., K), \quad \sum_{j=1}^{K} \pi_j = 1 \tag{1.13.35}$$

そこで，この条件を満たしながら尤度関数の局値を求めることで $\boldsymbol{\theta}$ を推定することになるが，ここでは訓練サンプルが各々のガウス分布に所属する確率 (所属確率) を利用して推定する方法を示す．

サンプル $\boldsymbol{x}_i$ が j 番目のガウス分布に所属する確率を $w_i^{(j)}$ とする．ただし，この値は予め知ることはできないので，適当に（乱数などを用いて）初期化する．ここで $w_i^{(j)}$ は確率なので

$$w_i^{(j)} \geq 0 \ (i = 1, ..., n; \ j = 1, ..., K), \quad \sum_{j=1}^{K} w_i^{(j)} = 1 \tag{1.13.36}$$

を満たすように決める．この所属確率が定まれば，j 番目のガウス分布に含まれるサンプルの個数 n_j は，$w_i^{(j)}$ を用いることで確率的に定まり，その総和はサンプルの総数 n と一致する．

$$n_j = \sum_{i=1}^{n} w_i^{(j)}, \ \sum_{j=1}^{K} n_j = n \tag{1.13.37}$$

この n_j の大きさに比例するように係数 π_j を $\pi_j = n_j/n$ と定める．さらに，$w_i^{(j)}$ を用いることで，j 番目のガウス分布のパラメータ $\boldsymbol{\mu}_j$ と $\boldsymbol{\Sigma}_j$ は

$$\hat{\boldsymbol{\mu}}_j = \frac{1}{n_j} \sum_{i=1}^{n} w_i^{(j)} \boldsymbol{x}_i, \quad \hat{\boldsymbol{\Sigma}}_j = \frac{1}{n_j} \sum_{i=1}^{n} w_i^{(j)} (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_j)(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_j)^\top \tag{1.13.38}$$

と推定でき，これらを用いることで，j 番目のガウス分布は

$$p_j(\boldsymbol{x}) = \frac{1}{(2\pi)^{d/2} |\hat{\boldsymbol{\Sigma}}_j|^{1/2}} \exp\left\{ -\frac{1}{2} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_j)^\top \hat{\boldsymbol{\Sigma}}_j^{-1} (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_j) \right\} \tag{1.13.39}$$

で表すことができる．この $p_j(\boldsymbol{x})$ を用いることで，$\boldsymbol{x}_i$ の第 j ガウス分布への所属確率は以下のように更新できる．

$$w_i^{(j)} = \frac{\pi_j p_j(\boldsymbol{x}_i)}{\sum_{l=1}^{K} \pi_l p_l(\boldsymbol{x}_i)} \tag{1.13.40}$$

更新前と更新後の $w_i^{(j)}$ に変化がなければアルゴリズムを終了し，そうでなければ式 (1.13.37) に戻って，同様の計算を再び行う．このアルゴリズムは反復のたびに対数尤度が増加すること，およびアルゴリズムは必ず収束することが証明されている．ただし，このアルゴリズムは初期値によって解が変化することと，分布数の K をユーザが経験的に決めるか，モデル選択 (model selection) などによって決定する必要がある点に注意されたい．

13.6 教師なし学習 — クラスタリング

以上はクラス毎に多峰性の確率密度関数の推定分布 $\hat{p}(\boldsymbol{x}|\omega_j)$ について考えていたが，ラベルが付与されていないサンプル集合にこのアルゴリズムを適用すれば，サンプル集合を K 個のグループ（クラスタ）に自動的に分類することが可能である．このように，ラベルを用いないでサンプルを分類することをクラスタリング (clustering) と呼び，データの要約や検索などでよく用いられる．代表的なクラスタリング手法としては，K-平均法 (K-means) が挙げられる．これは，EM アルゴリズムと異なり，サンプルのクラスタへの所属が 0 か 1 かの二値で与えられるハードクラスタリング (hard clustering) の一種である．最近では，画像認識における Bog-of-words [8] の作成や，任意形状のクラスタを抽出できるスペクトラルクラスタリング (spectral clustering) [9] において利用されることが多い．K-平均法では，以下の手順で n 個のサンプルを K 個のクラスタに分割する．

K-means 法のアルゴリズム

Step1: 各サンプル $\boldsymbol{x}_i$ がどのクラスタに属しているかという状態を表すクラスタラベル y_i $(i = 1, ..., n)$ をランダムに 1 から K までの自然数により設定する．

Step2: 各クラスタで，そのクラスタに属するサンプルの平均を求める．

$$\hat{\boldsymbol{\mu}}_j \leftarrow \frac{1}{n_j} \sum_{i:y_i=j} \boldsymbol{x}_i \quad (j = 1, .., K)$$

ただし，n_j は第 j クラスタに属するサンプルの個数であり，$\sum_{i:y_i=j}$ は $y_i = j$ となる i に関する和を表す．

Step3: $\boldsymbol{x}_i$ と K 個の平均とのユークリッド距離

$$D(\boldsymbol{x}_i, \hat{\boldsymbol{\mu}}_j) = \sqrt{(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_j)^\top (\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_j)}$$

を計算し，値が最小となるクラスタに $\boldsymbol{x}_i$ を割り当てる．

$$y_i \leftarrow \arg\min_j D(\boldsymbol{x}_i, \hat{\boldsymbol{\mu}}_j)$$

Step4: 全ての y_i が変化しなくなったら終了．そうでなければ **Step2** へ．

13.7 ノンパラメトリックな学習の例

ここでは，モデルを用いずに訓練サンプルから直接，決定境界を決めるノンパラメトリックな学習の例として，k 最近傍決定則，パーセプトロンの一種であるサポートベクタマシン，弱識別器を組合わせて強い識別器を構築する Adaboost について概説する．

13.7.1 k 近傍決定則

k 近傍決定則 (k-nearest neighbor rule, kNN) では，未知サンプル $\boldsymbol{x}$ と第 i 訓練サンプル $\boldsymbol{x}_i$ とのユークリッド距離を $D(\boldsymbol{x}, \boldsymbol{x}_i) = \sqrt{(\boldsymbol{x} - \boldsymbol{x}_i)^\top (\boldsymbol{x} - \boldsymbol{x}_i)}$ としたとき，$D(\boldsymbol{x}, \boldsymbol{x}_i)$ が小さい上位 k 個の訓練サンプルのラベルのうち，最も数の多いクラスを $\boldsymbol{x}$ の識別結果として出力する．すなわち，上位 k 個のラベルのうち，ラベルが y_j である個数を k_j とすれば，kNN の識別関数は

$$g_j(\boldsymbol{x}) = \frac{k_j}{k} \tag{1.13.41}$$

であり，この値が最大となるクラスを $\boldsymbol{x}$ の識別結果とする．これは k 最近傍密度推定法を利用した事後確率の推定値に基づく最大事後確率則である [3, 7]．なお，$k = 1$ のときは最近傍決定則 (nearest negihbor rule) とよばれる．kNN は実装が容易であり，パターンの分布が線形分離が困難 (直線や平面で分離することが難しい場合) であっても，しばしば高い識別率を示す．しかし，多くの訓練サンプルをそのまま記憶する必要があり，次元数 d が大きい場合には距離計算に非常に時間がかかるといった難点がある．そのため，近似探索や PCA などの次元削減と組合わせて用いられることが多い．

13.7.2 サポートベクタマシン

サポートベクタマシン (support vector machine, SVM) [10] は線形識別法の一種である．すなわち，二つのクラスを分離するための識別関数を次のように $\boldsymbol{x}$ の一次式で表す．

$$g(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b \tag{1.13.42}$$

ここで，$\boldsymbol{w} = (w_1, \cdots, w_d)^\top$ は重みベクトル (weight vector)，b はバイアス (bias) と呼ばれる．直感的に言えば，線形識別法は $\boldsymbol{w}$ と直交する超平面を b だけ並行移動した $g(\boldsymbol{x}) = 0$ を決定境界として利用する方法である．ここで便宜上，二つのクラスを ω_- と ω_+ とし，この二つのクラスに属する n 個の訓練サンプルのラベルを $y_i \in \{-1, +1\}$ で表すことにする．$\boldsymbol{w}$ と b が定まれば，未知サンプル $\boldsymbol{x}$ は $g(\boldsymbol{x})$ の符号に基づいて識別できる．

$$g(\boldsymbol{x}) = \begin{cases} \omega_+ & \text{if } g(\boldsymbol{x}) > 0 \\ \omega_- & \text{if } g(\boldsymbol{x}) < 0 \end{cases} \tag{1.13.43}$$

ここで，二つのクラス ω_+ と ω_- に属する訓練サンプルが，$g(\boldsymbol{x}) = 0$ によって完全に分離できる (線形分離可能) と仮定しよう．この場合，二つのクラスを完全に分離することのできる決定境界はいくつも存在するが，SVM ではそれらのうち，二つのクラスのちょうど真ん中を通るような決定境界を選ぶ．具体的には，決定境界に最も近い訓練サンプルを各々のクラスから選び，これらの間の距離の半分をマージン (margin) と呼んで，このマージンが最大となるように $\boldsymbol{w}$ と b を決める．このとき，各々のクラスで最も決定境界に近い訓練サンプルが，決定境界と平行な直線である $g(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b = \pm 1$ 上に存在するように $\boldsymbol{w}$ と b を決めよ，という新たな制約を加えると，マージンは $1/\|\boldsymbol{w}\|$ と簡単に計算でき

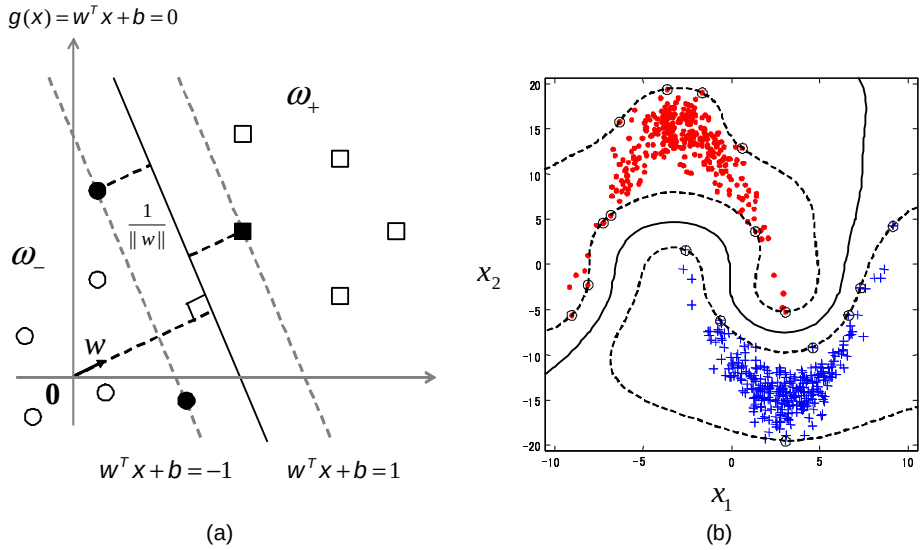

図 1.13.3 (a): 二次元空間での SVM の概念図．決定境界 $g(\boldsymbol{x}) = 0$ に最も近い訓練サンプルが黒塗りで表されている．黒塗りのサンプルが $\boldsymbol{w}^\top \boldsymbol{x} + b = \pm 1$ 上に存在するとすれば，マージンは $1/\|\boldsymbol{w}\|$ となる．(b): ガウスカーネルを用いた SVM の例．実線が決定境界であり，点線が高次元空間における直線 $\boldsymbol{w}^\top \boldsymbol{x} + b = \pm 1$ に対応する．カーネル法によりパターンの分布を超平面で分離できない場合でも識別が容易となる．

るようになる (図 1.13.3(a) を参照)．したがって，SVM の決定境界を決める問題は，制約を考慮しながらマージンを最大化する最適化問題として以下のように書ける．

$$\begin{aligned} &\min_{\boldsymbol{w}} \quad \|\boldsymbol{w}\|^2 \\ &\text{subject to} \quad y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1, \quad (i = 1, ..., n) \end{aligned} \tag{1.13.44}$$

ここで，subject to とは「〜の条件のもとで」という意味である．$1/\|\boldsymbol{w}\|$ を最大化するには $\|\boldsymbol{w}\|$ (の二乗) を最小化すればよいことに注意しよう．この最適化問題を解いて得られた $\boldsymbol{w}$ と b を用いれば，二つのクラスが線形分離可能の場合，全ての訓練サンプルに対して $y_i(\boldsymbol{w}\top \boldsymbol{x}_i + b) \geq 1$ が満たされる．すなわち，全ての訓練サンプルは正しく識別される．

しかし，実際のパターン認識問題では，訓練サンプルが線形分離可能でない場合が多い．そこで，SVM ではソフトマージン (soft margin) というテクニックにより線形分離可能でない場合にも対応できるようにしている．直感的に言えば，クラスの境界付近に存在する訓練サンプルに対して，$y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1$ を満たすように $\boldsymbol{w}$ と b を決定するのは難しいので，そのような訓練サンプルに関しては変数 $\xi_i \geq 0$ を用いて $y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1 - \xi_i$ と制約を緩めてやる．ただし，制約を緩めるための ξ_i の総和 $\sum_{i=1}^n \xi_i$ がなるべく小さくなるように $\boldsymbol{w}$ と b を決める．以上をまとめると，ソフトマージンを用いた SVM による決定境界

を求める問題は次のように書ける．

$$\begin{aligned} \min_{\boldsymbol{w}} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n}\xi_i \\ \text{subject to} \quad & y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1 - \xi_i, \quad (i = 1, ..., n) \end{aligned} \tag{1.13.45}$$

ここで C は制約を緩める度合いを調節するパラメータであり，実験的に決定する．

ところで，上記のような制約条件が課せられた最大・最小化問題は，ラグランジュ乗数法を用いると簡単な問題に変形できることが多い．そこで，ラグランジュ乗数を $\boldsymbol{\alpha} = (\alpha_1 \cdots \alpha_n)^\top, \alpha_i\ (\geq 0)$ としてラグランジュ乗数法に基づき式を変形していくと，式 (1.13.45) の最適化問題は以下のような $\boldsymbol{\alpha}$ に関する最大化問題となる．

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^\top \boldsymbol{x}_j \\ \text{subject to} \quad & 0 \leq \alpha_i \leq C, \quad \sum_{i=1}^{n}\alpha_i y_i = 0 \end{aligned} \tag{1.13.46}$$

この最適化は凸二次計画問題 (convex quadratic programming problem) と呼ばれるもので，数値計算ソフトに含まれているライブラリを利用すれば最適解を得ることができる．得られた最適解 $\boldsymbol{\alpha}$ の成分のうち，$\alpha_i \neq 0$ に対応する訓練サンプルのことをサポートベクトル (support vector) とよぶ．全てのサポートベクトルの集合を $\mathcal{S}$ とすると，サポートベクトルにより $\boldsymbol{w}$ は

$$\boldsymbol{w} = \sum_{i\in\mathcal{S}} y_i\alpha_i\boldsymbol{x}_i \tag{1.13.47}$$

で与えられる．一方，b は，$\mathcal{S}$ のうち任意のサポートベクトルを $\boldsymbol{x}_s$，そのラベルを y_s とすると，$\boldsymbol{w}$ が式 (1.13.47) によって与えられることに注意すれば

$$b = -\boldsymbol{w}^\top \boldsymbol{x}_s + y_s = \sum_{i\in\mathcal{S}} y_i\alpha_i\boldsymbol{x}_i^\top \boldsymbol{x}_s - y_s \tag{1.13.48}$$

によって計算できる．SVM では，このようにして求めた $\boldsymbol{w}$ と b を用いて式 (1.13.43) により未知サンプルを分類する．

SVM は，kNN と異なり，全ての訓練サンプルを記憶する必要はなく，問題によっては非常に少ない訓練サンプル（サポートベクトル）を記憶するだけで済む場合がある．また，後述するカーネル法を併用すれば，複雑な決定境界を求めることができ，結果として高い汎化性能を実現することが可能である．このような理由から，SVM は近年のパターン認識問題において非常に良く使われる識別法の一つとなっている．

なお，SVM は二クラスを分類する識別法であるが，これを $c > 2$ の多クラスへ拡張するには，クラス毎に識別関数を作成し，その値が最大となるクラスへ識別することで実現できる．すなわち，クラス ω_j に属する訓練サンプルのラベルを $y_i = +1$，それ以外のクラスに属する全ての訓練サンプルのラベルを $y_i = -1$ として SVM により ω_j に対する識別関数 $g_j(\boldsymbol{x})$ を求める．このようにして c 個の識別関数をクラス毎に準備した後に，未知サンプル $\boldsymbol{x}$ を $g_j(\boldsymbol{x})$ が最大となるクラスへ分類する．

13.7.3 カーネル法

SVM は線形識別法であるので，その決定境界は超平面となるが，実際のパターン認識問題ではクラスを平面で分離できない場合もある．そのような場合には，異なるクラスのパターンができるだけ分離するように特徴を抽出したり，高次元空間に写像したりすることは良く行われる．ここでは高次元空間への写像を介してパターン認識を行うことを容易にするカーネル法 (kernel method) について考える．まず，高次元空間への非線形写像を $\Phi(\cdot)$ とし，これにより $\boldsymbol{x}$ を

$$\boldsymbol{x} \mapsto \Phi(\boldsymbol{x}) \tag{1.13.49}$$

として $\hat{d}$ 次元空間に写像する．ここで $\hat{d}$ は元のベクトルの次元数 d よりもはるかに大きく，無限大でも良いとする．もし，識別関数が線形識別法のように，パターンの内積項のみで構成される場合には，カーネル法によりパターン $\Phi(\boldsymbol{x})$ を直接求めなくても，写像先の空間でパターンを識別することが可能であり，それは高次元空間に写像されたパターン間の内積 $\Phi(\boldsymbol{x}_i)^\top \Phi(\boldsymbol{x}_j)$ を，d 次元パターン間のカーネル関数 (kernel function) の値 $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ で置き換えることで実現できる．

$$\Phi(\boldsymbol{x}_i)^\top \Phi(\boldsymbol{x}_j) = K(\boldsymbol{x}_i, \boldsymbol{x}_j) \tag{1.13.50}$$

これにより，高次元空間への写像を介した SVM の最適化問題は

$$\begin{aligned} &\max_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) \\ &\text{subject to} \quad 0 \le \alpha_i \le C, \quad \sum_{i=1}^{n} \alpha_i y_i = 0 \end{aligned} \tag{1.13.51}$$

のように書ける．もちろん他の線形識別法も高次元空間への写像を介した識別を実現でき，例えば最小距離識別法の識別関数も

$$g_j(\Phi(\boldsymbol{x})) = -\frac{2}{n_j} \sum_{i=1}^{n_j} K(\boldsymbol{x}, \boldsymbol{x}_i) + \frac{1}{n_j^2} \sum_{k=1}^{n_j} \sum_{l=1}^{n_j} K(\boldsymbol{x}_k, \boldsymbol{x}_l) \tag{1.13.52}$$

と書ける．このようにすれば，超平面で分離が困難なパターンを高い識別率で識別できる場合がある (図 1.13.3(b) を参照)．なお，カーネル関数としては代表的なものとして，以下の多項式カーネルやガウスカーネル

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i^\top \boldsymbol{x}_j + 1)^p, \quad K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\beta \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2\right) \tag{1.13.53}$$

などが挙げられるが，カーネル関数が半正定値性を満たしていれば，独自のカーネル関数の設計も可能である [11].

13.7.4 Adaboost

Adaboost [12] とは，単体では識別能力の低い識別器である弱識別器 (weak learner) を利用して，汎化性能の高い識別器を構築するためのアルゴリズムの名称である．Adaboost

では二クラスの識別について考えるので，SVM と同様に，それぞれのクラスを ω_- と ω_+ で表し，各クラスに属する総数 n 個の訓練サンプルのラベルを $y_i \in \{-1, +1\}$ とする．さらに，Adaboost では重み付けされた訓練サンプルを用いて複数の弱識別器を設計する．そこで，総数 T 個の弱識別器のうち，t 番目 $(t = 1, ..., T)$ の弱識別器を設計する際の訓練サンプルに対する重みを $w_1^{(t)}, ..., w_n^{(t)}$ で表すことにする．

Adaboost では，弱識別器を重み付き訓練サンプルを利用して複数構築する必要がある．そこで，総数 T の弱識別器のうち t 番目 $(t = 1, ..., T)$ のものを

$$h_t(\boldsymbol{x}) = \begin{cases} +1 & \boldsymbol{x} \text{ を } \omega_+ \text{ と識別した場合} \\ -1 & \boldsymbol{x} \text{ を } \omega_- \text{ と識別した場合} \end{cases} \tag{1.13.54}$$

とする．すなわち，$h_t(\boldsymbol{x})$ は入力 $\boldsymbol{x}$ の識別結果を $+1$ か -1 で出力する関数である．ここで，弱識別器としてどのようなものを用いるのか考える必要があるが，これは識別率が 50% より大きいものであればどのようなものでも構わない．ただし，弱識別器としてもともと識別率が高いものを採用してしまうと，Adaboost による性能向上が期待できなくなるので注意が必要である．このような理由から，Adaboost では弱識別器として決定株 (decision stump) を用いる場合が多いが，ここでは理解しやすいものとして，各クラスに属する訓練サンプルの加重平均と $\boldsymbol{x}$ とのユークリッド距離に基づく識別を弱識別器とした場合を考える．すなわち，t 番目の弱識別器を

$$h_t(\boldsymbol{x}) = \begin{cases} +1 & \text{if } D(\boldsymbol{x}, \boldsymbol{\mu}_+^{(t)}) < D(\boldsymbol{x}, \boldsymbol{\mu}_-^{(t)}) \\ -1 & \text{otherwise} \end{cases} \tag{1.13.55}$$

とした場合を考える．ここで $\boldsymbol{\mu}_+^{(t)}$ と $\boldsymbol{\mu}_-^{(t)}$ は，訓練サンプルと対応する重み $w_i^{(t)}$ によってクラス毎に計算される加重平均である．

$$\boldsymbol{\mu}_+^{(t)} = \frac{1}{\sum_{i \in \omega_+} w_i^{(t)}} \sum_{i \in \omega_+} w_i^{(t)} \boldsymbol{x}_i, \quad \boldsymbol{\mu}_-^{(t)} = \frac{1}{\sum_{i \in \omega_-} w_i^{(t)}} \sum_{i \in \omega_-} w_i^{(t)} \boldsymbol{x}_i \tag{1.13.56}$$

これらの加重平均は重みを調節することで変化するため，この重みを各弱識別器で異なるように設定すれば，複数の弱識別器を構築することができる．Adaboost ではこの重みを訓練サンプルに対するエラー率に基づいて以下のようにして決める．まず，重みの初期値を，全ての訓練サンプルに対して均等に $w_i^{(1)} = 1$ と設定する．次に，加重平均 (はじめは全ての訓練サンプルの重みは均等なので単純な標本平均) を計算し，式 (1.13.55) により訓練サンプルを識別した場合のエラー率を計算する．そして，エラー率から信頼度と呼ばれる値を計算し，その値を用いて，誤識別を起こした訓練サンプルの重みを大きくし，正しく識別された訓練サンプルの重みを小さくする．このようにして更新した重みを次の弱識別器の加重平均の計算に利用すれば，誤識別を起こした訓練サンプルを反映した重心を求めることができる．このような加重平均の更新と信頼度の計算を次々と繰り返すことで T 個の弱識別器を作ったとする．Adaboost では T 個の $h_t(\boldsymbol{x})$ と，その弱識別器に対応する信頼度を α_t としたとき，未知サンプル $\boldsymbol{x}$ のクラスを次のような重み付き多数決により決定する．

$$g(\boldsymbol{x}) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x}) \right) \tag{1.13.57}$$

ここで，sign は出力の符号を返す関数である．すなわち，重み付き多数決が 0 より小さい場合は -1 を，そうでなければ $+1$ を返す．

上記の手続きをアルゴリズムとしてまとめれば以下のように書ける．

加重平均との距離に基づく弱識別器を用いた場合の Adaboost のアルゴリズム

初期化：訓練サンプルに対する重みを $w_i^{(1)} = 1/n$ と初期化

$t = 1, ..., T$ に対して以下の 1 から 4 を繰り返す

1. **加重平均の計算**：重み $w_1^{(t)}, ..., w_n^{(t)}$ を使って式 (1.13.56) で加重平均を計算
2. **エラー率の計算**：式 (1.13.55) で訓練サンプルを識別し，誤って識別された訓練サンプルの重みの総和 ϵ_t を計算
3. **信頼度の計算**：$\alpha_t = \frac{1}{2} \ln \left(\frac{1-\epsilon_t}{\epsilon_t} \right)$ を計算する．$\alpha_t = 0$ の場合は繰り返しを終了し，それ以外は以下の 4 へ進む
4. **重みの更新**：

$$w_i^{(t+1)} = \frac{w_i^{(t)} \exp(-\alpha_t y_i h_t(\boldsymbol{x}_i))}{\sum_{l=1}^{n} w_l^{(t)} \exp(-\alpha_t y_l h_t(\boldsymbol{x}_l))}$$

識別：T 個の弱識別器を使って $\boldsymbol{x}$ を $g(\boldsymbol{x}) = \text{sign} \left(\sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x}) \right)$ に基づき識別

Adaboost のアルゴリズムは $F(\boldsymbol{x}_i) = \sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x}_i)$ としたときの損失

$$L(F) = \frac{1}{n} \sum_{i=1}^{n} \exp(-y_i F(\boldsymbol{x}_i)) \tag{1.13.58}$$

を逐次に最適化するアルゴリズムと見なすことができ，それはマージン最大化と関連が深いことが知られている [4]．なお，Adaboost を特徴選択に利用することで，高い精度で画像から顔を検出する方法も提案されている [13]．なお，Adaboost は二クラスに対する識別器であるので，多クラス識別を行う場合には，SVM と同じように，各クラス毎に識別関数を求めて，その値が最大となるクラスへ未知サンプルを識別する．

13.8 手書き数字パターンに対する識別率

最後に，これまで述べた識別法を **13.2** の手書き数字パターンに適用した場合の未知サンプルに対する識別率を調べた．表 1.13.2 に各手法で得られた識別率とパラメータを示す．ここで，パラメータとは正則化の γ，SVM の C，ガウスカーネルの β，kNN の k，Adaboost の T を意味する．実験では，これらのパラメータを訓練サンプルを用いた交差確認法 (cross validation, CV) により決定した．交差確認法では，まず訓練サンプルを K 個のグループ

表 1.13.2 各識別法による識別率とパラメータ.

識別法	識別率 [%]
最大事後確率則 ($\gamma = 5$)	93.5
マハラノビス距離 ($\gamma = 0.01$)	95.4
線形判別分析 ($\gamma = 0.01$)	92.7
最小距離識別法	86.1
線形 SVM ($C = 1$)	94.0
kNN ($k = 1$)	97.5
ガウスカーネルを用いた SVM ($\beta = 0.0068, C = 1$)	97.9
ガウスカーネルを用いた最小距離識別法 ($\beta = 0.0068$)	89.1
Adaboost (加重平均，$T = 100$)	90.8

にランダムに分け，その一つを検証サンプルとして用い，残りを訓練サンプルとして検証サンプルに対する識別率をパラメータを変化させながら求める．これを K 回，検証サンプルを変えて行い，識別率の平均が最も良いパラメータを選択する．今回は，$K = 5$ とした 5CV により各識別器のパラメータを決定した．

13.8.1 パラメトリックな手法の識別率

ここではパラメトリックな手法の識別率について考える．まず最大事後確率則では，各クラスの確率密度関数を一つのガウス分布で表し，パラメータの平均と共分散行列を最尤推定で推定した．識別率はマハラノビス距離や線形判別分析（式 (1.13.27) が最大となるクラスへ $\boldsymbol{x}$ を識別）とほとんど同じか，若干劣ることがわかる．線形判別分析は各クラスで同じ共分散行列を持つと仮定した場合の最大事後確率則であり，直感的にはこのような仮定を用いない方が識別には有利と思われるが，現実のパターン認識問題では必ずしもそのようにならないことがこの例からわかる．

パラメトリックな識別法の利点としては，モデルによってパターンの分布を説明できること，事後確率の推定値と損失を用いてリスクの異なる識別（例えば病気の診断）を容易に実現できること，ノンパラメトリックな手法よりも少ないメモリ容量で高速に未知サンプルを識別できること，などが挙げられる．

13.8.2 ノンパラメトリックな手法の識別率

ここではノンパラメトリックな手法の識別率について考える．表から，kNN や SVM が高い識別率を達成しているのがわかる．特に，線形 SVM は他の線形識別法よりも高い識別率を達成しており，カーネル法を併用することで，さらに高い識別率が達成できることがわかる．kNN はすべての訓練サンプルを記憶しておく必要があり，識別に時間もかかるが実装は非常に簡単である．Adaboost も，最小距離識別法と比べ識別率が高いことから，複数の弱識別器を用いることが識別率の向上に役立つことがわかる．

以上のように，実際のパターン認識では，真の事前確率や確率密度関数が未知であるため，どの識別法が高い識別率を達成するのかは実験で調べる必要がある．今回の実験ではノンパラメトリックな手法で高い識別率が達成されたが，これがすべてのパターンに対して成り立つわけではないことに注意しよう．もし，パラメトリックな手法で十分高い識別率を達成できるのであれば，ノンパラメトリックな手法を用いる特段の理由はないであろう．一方，パターンの分布に対する知識や仮定を持ち合わせていない場合には，ノンパラメトリックな手法を試すべきであろう．いずれにしても，所望のパターン認識システムを実現するためには，さまざまな特徴や識別法を試してみることが肝要である．

関連図書

[1] 石井健一郎, 上田修功, 前田英作, 村瀬 洋: “わかりやすいパターン認識,” オーム社 (1998)

[2] K. Fukunaga, “Introduction to statistical pattern recognition,” 2nd edition, Academic Press (1990)

[3] R.O. Duda, P.E. Hart, and D.G. Stork, “Pattern classification,” 2nd edition, John Wiley & Sons (2001)

[4] 麻生英樹，津田宏治，村田 昇: “パターン認識と学習の統計学,” 岩波書店 (2003)

[5] C.M. Bishop(著)，元田浩，栗田多喜夫，樋口知之，松本裕治，村田 昇 (編), “パターン認識と機械学習 (上) : ベイズ理論による統計的予測,” シュプリンガージャパン，(2007)

[6] C.M. Bishop(著)，元田浩，栗田多喜夫，樋口知之，松本裕治，村田 昇 (編), “パターン認識と機械学習 (下) : ベイズ理論による統計的予測,” シュプリンガージャパン，(2008)

[7] 杉山将，“生成モデルに基づくパターン認識,” オーム社 (2009)

[8] G. Csurka, C.R. Dance, L. Fan, J. Willamowski, and C. Bray, “Visual categorization with bags of keypoints,” Proc. of ECCV Workshop on Statistical Learning in Computer Vision, pp. 1–22 (2004)

[9] C. Ding and H. Zha, “Spectral Clustering, Ordering and Ranking: Statistical Learning with Matrix Factorizations,” Springer, (2008)

[10] V. Vapnik, “Statistical learning theory,” John Wiley & Sons (1998)

[11] 渡辺澄夫，萩原克幸，赤穂昭太郎，本村陽一，福水健次，岡田真人，青柳美輝，“学習システムの理論と実現,” 森北出版 (2005)

[12] Y. Freund and R.E. Schapire, “A decision-theoretic generalization of on-line learning and an application to boosting,” J. of Computer and System Sciences, vol. 55, no. 1, pp. 119–139 (1997)

[13] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” Proc. of CVPR, vol. 1, pp. 511–518 (2001)