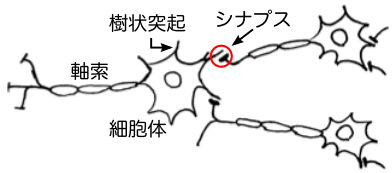
目次

★13　パターン認識と機械学習 (4) — ニューラルネットワーク

ニューラルネットワークは，元々は脳における情報処理のモデルとして提案されたものであるが，現在では機械学習の手法として幅広く応用されている．

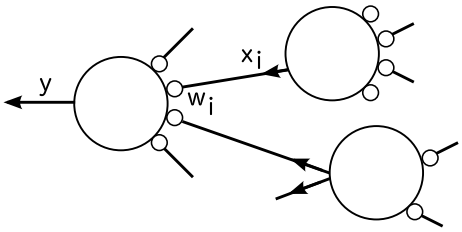
★13.1　ニューラルネットワークって？

★13.1.1　神経細胞とそのモデル



ヒトの脳には数百億の**ニューロン**（**神経細胞**, neuron）が存在している．ニューロンは信号を伝達する機能をもっており，多数のニューロンが相互につながりあって多様な情報処理を行なっている．生命維持から感情，思考にいたるまでの脳機能は，主にこれらニューロンの集団が担っているものと考えられている．

ニューロンのふるまいを思い切って単純化すると，次のようにまとめることができる．



1. 他のニューロンの出力を受け取る．ただし，そのまま受け取るのではなく，ニューロン間のつながりの強さに応じて重みづけされた値を受け取る．
2. それらの和を求める
3. その値に応じて自身の出力を決める

このようなニューロンのふるまいは，次式のようにモデル化できる (☆ 1)．

$$y = \sigma\left(\sum_{i=1}^{n} w_i x_i + \theta\right) \tag{1}$$

ただし， x_i はこのニューロンに信号を伝達する n 個のニューロンのうち i 番目のものの出力であり，w_i は x_i の**結合重み** (☆ 2) を表すパラメータである．また，θ は**しきい値** (☆ 3) と呼ばれるパラメータである．関数 $\sigma(s)$ は**活性化関数** (☆ 4) と呼ばれるものであり，以下に示す**ステップ関数** (☆ 5) や**シグモイド関数** (☆ 6)，**Rectified Linear 関数** (☆ 7) などがよく用いられる．ニューロンの入力と出力の関係は一般に非線形であり，これらはその性質をモデル化したものとなっている．

☆ 1) 1940 年代に McCulloch と Pitts が提案した．ただし，彼らのモデルではニューロンの出力は 0, 1 の二値である．

☆ 2) 結合重み: connection weight.

☆ 3) しきい値: threshold value.

☆ 4) 活性化関数: activation function.

☆ 5) ステップ関数: step function.

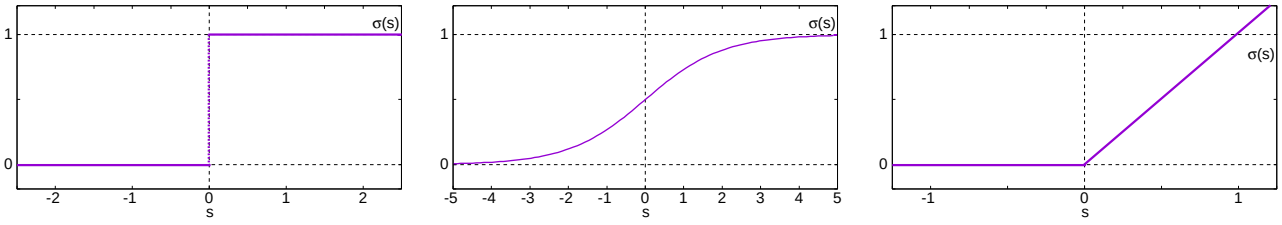
☆ 6) シグモイド関数: sigmoid function.

☆ 7) ランプ関数ということもある．

$$\text{ステップ関数} \quad \sigma(s) = \begin{cases} 1 & s \geq 0 \\ 0 & otherwise \end{cases} \tag{2}$$

$$\text{シグモイド関数} \quad \sigma(s) = \frac{1}{1+e^{-s}} \tag{3}$$

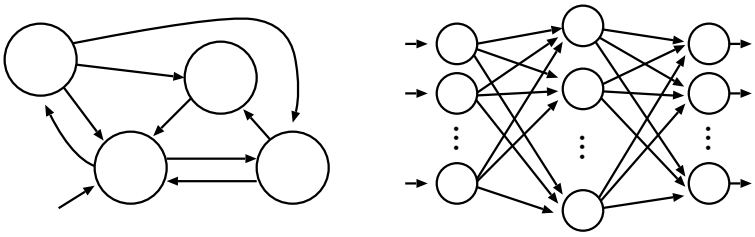
$$\text{Rectified Linear} \quad \sigma(s) = \begin{cases} s & s \geq 0 \\ 0 & otherwise \end{cases} \tag{4}$$



ステップ　　シグモイド　　Rectified Linear

★ 13.1.2　ニューラルネットワーク，多層パーセプトロン

上記のようなニューロンモデルを複数つなぎあわせたものを**ニューラルネットワーク** (☆ 8) という．様々なつながり方のものが考えられるが，図右のようにニューロンが層を成し，層間のニューロンのみに単方向のつながりがあるタイプのものを**多層パーセプトロン** (☆ 9) と呼ぶ．



☆ 8) ニューラルネットワーク: neural network，神経回路網とも．

☆ 9) 多層パーセプトロン: Multi-Layer Perceptron, MLP.

★ 13.1.3　深層学習

ニューラルネットワークに関しては数十年前から地道な研究が続けられてきたが，近年，その進歩とコンピュータの高性能化が相まって，いわゆる**人工知能** (☆ 10) の一種として急速に発展し，様々な分野で応用されるようになってきている (☆ 11)．多くの場合，上述の多層パーセプトロンのように階層的な構造をしたニューラルネットワークで，たくさんの層を積み重ねた（したがって学習するパラメータの数も非常に多い）ものが用いられる．このような多層のニューラルネットワークの学習は，**深層学習** (☆ 12) と呼ばれている．次頁以降でその概略を説明する．

☆ 10) 人工知能: Artificial Intelligence.

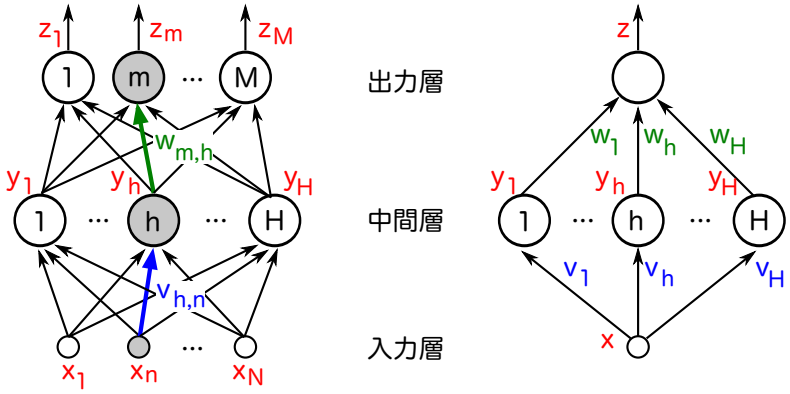
☆ 11) 画像認識，自動運転，音声認識，自動翻訳，将棋や囲碁でプロに勝つ AI，etc.

☆ 12) 深層学習: deep learning.

★ 13.2　多層パーセプトロン

★ 13.2.1　多層パーセプトロンとその学習

下図に，2 種類のニューラルネットを示す．左右の図のいずれも，大きめの丸一つが一つのニューロンを表している．左のものは入力が直接出力とつながった構造をしている．一方，右の方は，入力と出力の間に「隠れた」ニューロンの層（これを「隠れ層」または「中間層」という）を有している．一般に,「多層」パーセプトロンという時は，このような隠れ層を 1 層以上有するものを指す．



図左の多層パーセプトロン（以下 MLP と略記する）の入出力は，次のような式で表される．

$$y_h = \sigma\left(v_{h,0} + \sum_{d=1}^{D} v_{h,d} x_d\right) \qquad (h = 1, 2, \ldots, H) \tag{5}$$

$$z_m = \sigma\left(w_{m,0} + \sum_{h=1}^{H} w_{m,h} y_h\right) \qquad (m = 1, 2, \ldots, M) \tag{6}$$

ここで，y_h は隠れ層の h 番目のニューロンの値であり，$v_{h,d}$ はこのニューロンと入力の d 番目の要素との間の結合重みである．同様に，z_m は出力層の m 番目のニューロンの値であり，$w_{m,h}$ はこのニューロンと隠れ層の h 番目のニューロンとの間の結合重みである．関数 $\sigma(s)$ は前述の活性化関数である．

MLP の特徴は，上述のように隠れ層を有することである．隠れ層ニューロンの活性化関数にシグモイドのように非線形なものを用いると，MLP 全体の入出力も非線形関数となる．そのため，入力と出力の間に複雑な関係があるようなデータの場合でも，うまく学習できると期待される．

MLP においては，ニューロン間の結合の強さを表す値が，学習すべきパラメータとなる．式 (5) と (6) で表される MLP の場合，入力―隠れ層のニューロン間の結合を表す値 $v_{h,d}$ と，隠れ層ー出力層のニューロン間の結合を表す値 $w_{m,h}$ がパラメータである．それらの学習には，ロジスティック回帰の場合と同様に，勾配法／最急降下法を用いることが多い．目的関数としては，出力の正解と実際の出力との間の二乗誤差（最小二乗法で説明した誤差関数と同じ形のもの，次節も参照）や交差エントロピー（前回資料参照）などを用いることができる．いずれの場合も，上記パラメータに関する微分が計算できるので，パラメータを適当な初期値から逐次修正していく形の学習アルゴリズムを構成することができる．

★ 13.2.2　多層パーセプトロンの応用例

多層パーセプトロン (MLP) は，教師データと目的関数の与え方次第で，回帰／識別どちらの問題にも適用することができる．回帰の場合，通常は二乗誤差を目的関数とする．一方，識別の場合，交差エントロピーを用いることが多い．以下，回帰と識別それぞれの応用例を示す．

非線形回帰　最も単純な形の MLP として，上の図および式 (5) と (6) で $D = M = 1$ とした場合，つまり入力も出力も 1 つの場合を考える．出力層ニューロンの活性化関数は恒等関数（$\sigma(s) = s$）とする (☆ 13) このとき，入力 x に対するこの MLP の出力 z は，次式のようになる．

☆ 13) シグモイドと違って出力が $(0, 1)$ に限定されないので，この問題のような回帰／関数近似では出力の活性化関数としてよく用いられる．

$$y_h = s(v_{h,0} + v_{h,1}x) \qquad (h = 1, 2, \ldots, H) \tag{7}$$

$$z = \sum_{h=1}^{H} w_h y_h \tag{8}$$

この MLP に対して，入力と出力の正解のペア N 個から成る学習データ $\{(x_n, \tilde{z}_n)|n = 1, 2, \ldots, N\}$ を与え，x_n に対する MLP 出力 z_n と正解 $\tilde{z}_n$ との間の二乗誤差の和

$$E = \frac{1}{2}\sum_{n=1}^{N}(\tilde{z}_n - z_n)^2 \tag{9}$$

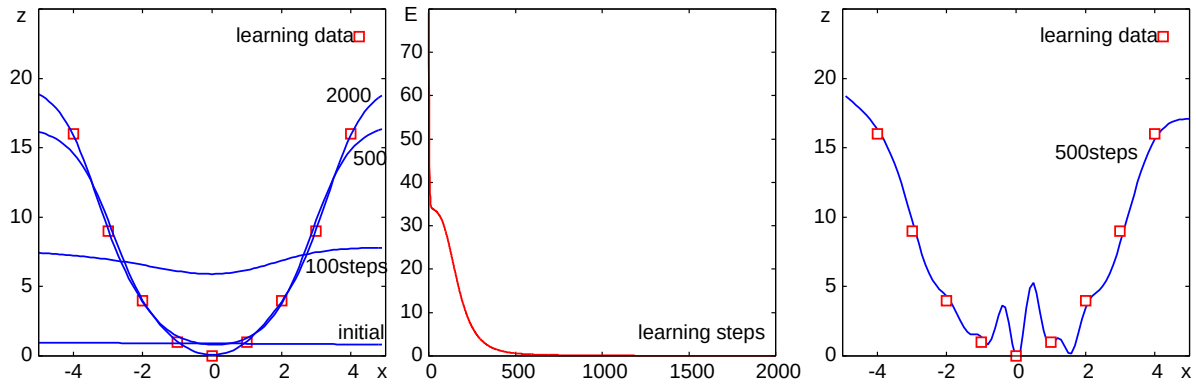
を目的関数として学習させる．これは，入出力がともに 1 変数の場合の回帰問題への MLP の適用例となっている．MLP を用いると，非線形な活性化関数をもった隠れ層のおかげで複雑な曲線を当てはめることができる．

以下に，実際に学習を行なった実験の結果を示す．二次関数 $z = x^2$ を近似することを目標に，学習データを $(-4, 16), (-3, 9), \ldots, (4, 16)$ の $N = 9$ 個とした．また，中間層のニューロン数は $H = 10$ と $H = 100$ の二通りとした．

下図左は，$H = 10$ の場合の学習過程における MLP 出力の変化を示している．学習が進むにつれてうまく学習データを近似できるようになっていることがわかる．このことは，下図中に示した二乗誤差 E の変化の様子からもわかる．

☆ 14) 過適合: overfitting.

一方，下図右は，$H = 100$ とした場合の学習結果の一例を示している．この場合，学習データに対する二乗誤差 E はほぼ 0 になっており，MLP 出力の曲線は全ての学習データ点を通っているが，その形は近似対象である放物線とはほど遠いものとなっている．このような現象は**過適合・過学習** (☆ 14) と呼ばれている．

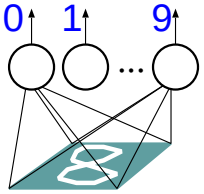


手書き数字の識別　多層パーセプトロン (MLP) を識別問題に用いる例として，MNIST 手書き数字データセット で実験を行ってみる．このデータセットは，28×28 画素の'0' から'9' までの手書き数字のグレイスケール画像 7 万枚（学習用 6 万枚+テスト用 1 万枚）から成るものであり，機械学習の練習問題としてよく用いられている．入力の次元数 $D = 28 \times 28 = 784$ であり，クラス数 $K = 10$ である．

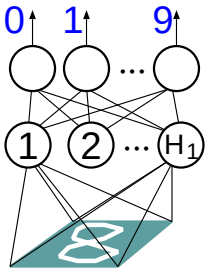
☆ 15) `http://yann.lecun.com/exdb/mnist/`

ここでは，ロジスティック回帰（logistic と表記，前回登場のものをクラス数が 3 以上に一般化したもの），それに隠れ層を 1 つ追加した MLP（MLP2 と表記），さらにもう 1 つ追加した MLP（MLP3）の 3 通り（下図参照）で実験を行った．隠れ層のニューロン数は $H_1 = 500$ (MLP2) および $H_1 = H_2 = 500$ (MLP3) とし，隠れ層ニューロンの活性化関数は全て Rectified Linear とした．

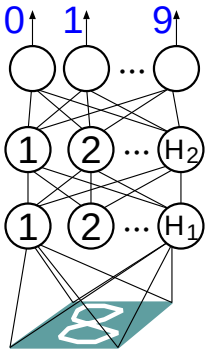
MNIST データの例



logistic



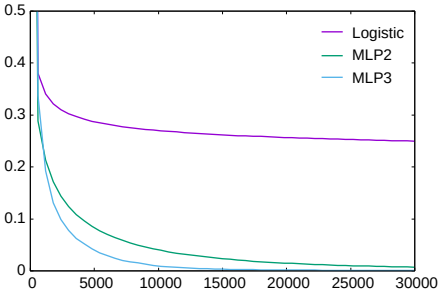
MLP2



MLP3

左下の表に，3 万回の学習（詳しい条件は説明を省略する）を行った後の MLP を用いて測った誤識別率（MLP が出力したクラスが誤りだった割合）を示す．ロジスティック回帰も MLP も，パラメータの初期値が異なれば異なる結果が得られるので，ここでは初期値を 5 通りずつ変えて得られた誤識別率の平均を示している．また，右下のグラフに学習の進行の様子の例を示す．横軸は学習回数，縦軸は学習データに対する交差エントロピーの平均である．

	logistic	MLP2	MLP3
学習データの誤識別率 [%]	6.9	0.036	0.0
テストデータの誤識別率 [%]	7.5	1.9	1.9



★14　パターン認識と機械学習 (5) — 教師なし学習

★14.1　教師なし学習とは

教師あり学習では，個々の学習データが「入力」と「それに対する出力の正解」のペアとして与えられていた．一方，**教師なし学習** (☆ 16) では，学習データとして「入力」のみが与えられる．教師なし学習の目的は，大量のデータが与えられたときに，そのデータのもつ規則性や構造を見い出し，有益な情報を抽出する処理を自動的に行うことである．**データ分析** (☆ 17) の手法の一種といえる．ここでは，様々な教師なし学習の問題設定のうち，**クラスタリング** (☆ 18) と**次元圧縮** (☆ 19) を取り上げる．

クラスタリングとは，大量のデータが与えられた時に，それらをいくつかの「塊」（クラスタ）に分類する手法である．データが D 次元実ベクトルの場合を考えると，N 個の D 次元ベクトルから成るデータ集合

$$\left\{\boldsymbol{x}_n \in \mathcal{R}^D | n = 1, 2, \ldots, N\right\} \tag{10}$$

が与えられた時に，これを $K \ll N$ 通りのクラスタに分類する．教師ありの識別問題と異なり，どのデータをどこに分類するかの正解は与えられないことに注意しよう．クラスタリングを行うと，大量のデータを要約したり，構造を見つけ出したりすることができる (☆ 20)．

一方，次元圧縮は,「データの数を減らす」かわりに「データの次元数を減らす」手法である．例えば，式 (10) のデータ集合の個々の要素を，その本質的な特徴をなるべく保ったまま，より低次元のベクトル $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_N \in \mathcal{R}^H$ $(H \leq D)$ に変換する．高次元の（つまりたくさんの変数から成る）データを，その性質をよく表す低次元の（少数の変数から成る）データに変換することで，データを分析しやすくしたり (☆ 21)，記憶容量や後の処理の計算量を減らしたりすることができる (☆ 22)．

☆ 16) 教師なし学習: unsupervised learning.

☆ 17) データ分析: data analysis.

☆ 18) クラスタリング: clustering

☆ 19) 次元圧縮: dimensionality reduction または dimension reduction. 次元削減とも．

☆ 20) 例えば，遺伝子データをクラスタリングして解析することで，生物種の間の類縁関係を推定（種 A と種 B は遺伝子配列が似てるので同じ種から進化した可能性が高い，とか）したりできる．

☆ 21) 例えば，20 科目の点数データ（20 次元）を 3 つの変数（3 次元）に変換して，学生の成績傾向を把握・分析しやすくするとか．

☆ 22) 以前登場した離散コサイン変換 (DCT) も，次元圧縮の用途に使える．

★14.2　K 平均法によるクラスタリング

クラスタリングの手法には様々なものがあるが，ここでは代表的なアルゴリズムとして，K **平均法** (☆ 23) を紹介する．K 平均法は，2 つのデータ $\boldsymbol{x}$ と $\boldsymbol{y}$ の間の非類似度 (☆ 24) を両者間のユークリッド距離 (☆ 25) $\|\boldsymbol{x} - \boldsymbol{y}\|$ で表せるようなデータに対して，それらを K 個のクラスタに分ける分け方を見つけるための教師なし学習のアルゴリズムである．クラスタ数 K はあらかじめ適当な方法で決めておかねばならない．K 平均法の手順は次のようなものである．

1. 各学習データ $\boldsymbol{x}_n (n = 1, 2, \ldots, N)$ をランダムにクラスタ C_1 から C_K までの K 個のクラスタのいずれかに割り当てる (☆ 26)．
2. クラスタ $C_k (k = 1, 2, \ldots, K)$ に割り当てられたデータの平均 $\boldsymbol{c}_k$ を求める．

$$\boldsymbol{c}_k = \frac{1}{|C_k|} \sum_{n:\boldsymbol{x}_n \in C_k} \boldsymbol{x}_n \tag{11}$$

これをクラスタ C_k の**セントロイド** (☆ 27)

☆ 23) K 平均法: K-means 法とも．

☆ 24) 距離が小さい方が類似度が大きいので,「非」類似度が距離に対応すると考えている．

☆ 25) $\boldsymbol{x}$ が D 次元ベクトル $\boldsymbol{x} = (x_1, x_2, \ldots, x_D)$ であり，$\boldsymbol{y}$ も同様の場合，$\|\boldsymbol{x} - \boldsymbol{y}\| = \sqrt{\sum_{d=1}^{D} (x_d - y_d)^2}$

☆ 26) 後述のように K 平均法の結果は初期値に依存するので，実用的には初期値の選び方を工夫したアルゴリズムが用いられることが多い．

☆ 27) セントロイド: centroid.

3. 各学習データ $\boldsymbol{x}_n (n = 1, 2, \ldots, N)$ を，セントロイドとの距離が最小となるクラスタに割り当て直す．例えば $\boldsymbol{x}_n$ に対して という．ここで，$|C_k|$ は集合 C_k の要素数，すなわちクラスタ C_k に割り当てられた学習データの数を表す．

$$k^* = \underset{k=1,2,\ldots,K}{\operatorname{argmin}} \|\boldsymbol{x}_n - \boldsymbol{c}_k\|^2 \tag{12}$$

であれば (☆ 28) (☆ 29)，$\boldsymbol{x}_n$ はクラスタ C_{k^*} に割り当てる．

4. 上のステップの結果があらかじめ定めておいた条件を満たしている（後述）ならば終了，さもなくば 2. へ戻る．

☆ 28) argmin の意味は次の例でわかるだろう．
$f(x) = (x-2)^2 + 3$ のとき，
$\min_x f(x) = 3$,
$\operatorname{argmin}_x f(x) = 2$.

☆ 29) ここでは距離の大小関係のみが問題なので，ユークリッド距離そのものではなく二乗した値で考えている（実際の計算では平方根が出てこない分その方が簡単だから）．

この学習の終了条件としては,「クラスタ割り当てに変化がなくなった」,「ステップ 2,3 の実行回数が一定に達した」などが用いられる．また，K 平均法では次式の E の値 (これはクラスタリングの「誤差」を表している) が学習ステップ毎に単調減少することが知られているので，この値の減少幅が一定より小さくなったら終了する，という方法もよく用いられる．

$$E = \sum_{k=1}^{k} \sum_{n:\boldsymbol{x}_n \in C_k} \|\boldsymbol{x}_n - \boldsymbol{c}_k\|^2 \tag{13}$$

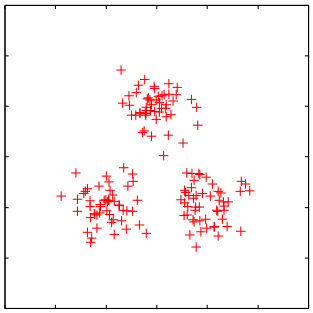
K 平均法の結果は，学習データに対するクラスタ割り当ての初期値に依存する．そのため，実際には初期値を変えて何度か K 平均法を実行し，上記の E の値が最も小さかった結果を採用する，というような方法がとられる．

上記の学習手続きによってクラスタセントロイドが推定できたら，式 (12) と同様の計算によって未知データの所属クラスタも決めることができる．例えば，ある未知データ $\boldsymbol{x}$ について，

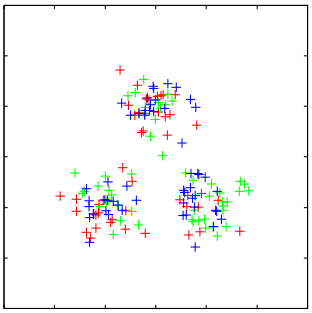
$$i = \underset{k=1,2,\ldots,K}{\operatorname{argmin}} \|\boldsymbol{x} - \boldsymbol{c}_k\|^2 \tag{14}$$

であれば，このデータの所属はクラスタ C_i とすればよい．

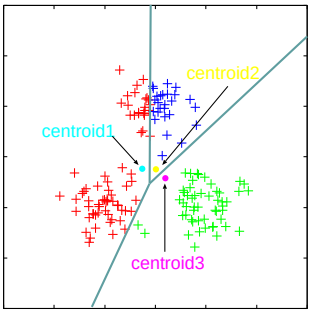
図 1 に，2 次元のデータに対して K 平均法を適用した結果を示す．また，図 2 に，猫の顔画像 131 枚 (画素数は 64×64) に K 平均法を適用して得られたセントロイドすなわちクラスタ毎の平均画像を示す．



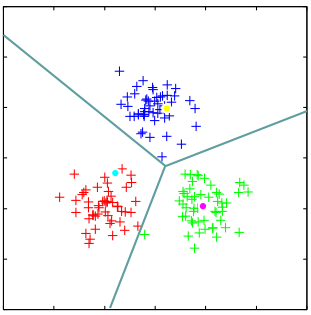
(1A) 学習データその 1



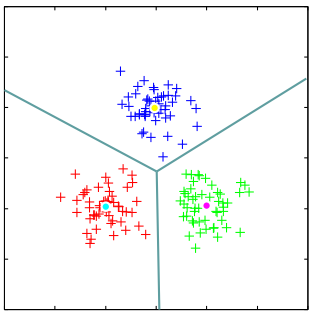
(1B) 初期値



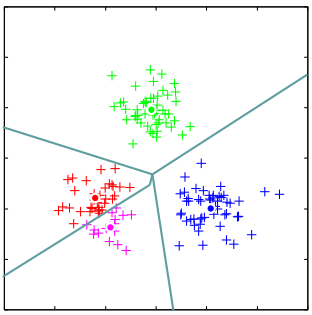
(1C) 学習 1 回



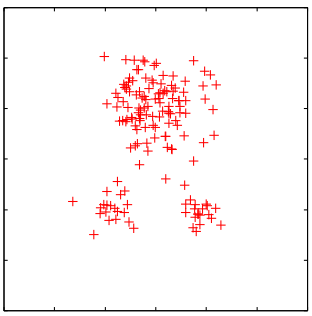
(1D) 学習 2 回



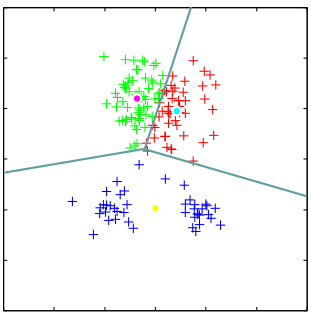
(1E) 学習 10 回



(2) $K = 4$ で学習 10 回



(3A) 学習データその 2



(3B) 学習 10 回

図 1: K 平均法による 2 次元データのクラスタリング．(1A) は学習データの例．(1B) から (1E) までは，(1A) のデータ対して $K = 3$ で K 平均法を適用した結果．(2) は，同じデータに $K = 4$ で K 平均法を適用した結果．(3A) と (3B) は，別の学習データとそのクラスタリング結果 ($K = 3$)．

図 2: 131 枚の猫画像に K 平均法を適用して得られたクラスタ平均画像 ($K = 5$)．