デジタルメディア処理2

担当: 井尻 敬

デジタルメディア処理２、2018（前期）

4/19　序論　　：イントロダクション，テクスチャ合成
4/26　特徴検出1　　：テンプレートマッチング、コーナー・エッジ検出
5/10　特徴検出2　　：DoG特徴量、SIFT特徴量、ハフ変換
5/17　領域分割　　：領域分割とは，閾値法，領域拡張法，動的輪郭モデル
5/24　領域分割　　：グラフカット，モーフォロジー処理，Marching cubes
5/31　パターン認識基礎1: パターン認識概論，サポートベクタマシン
6/07　パターン認識基礎2: ニューラルネットワーク、深層学習
6/14　パターン認識基礎3: 主成分分析とオートエンコーダ
6/21　筆記試験（50点満点)(*n*点以下の場合レポート出すかも)
6/28　プログラミング演習　1 (基礎的な課題40点, 発展的な課題 20点)
7/05　プログラミング演習　2
7/12　プログラミング演習　3
7/19　プログラミング演習　4
7/26　プログラミング演習　5

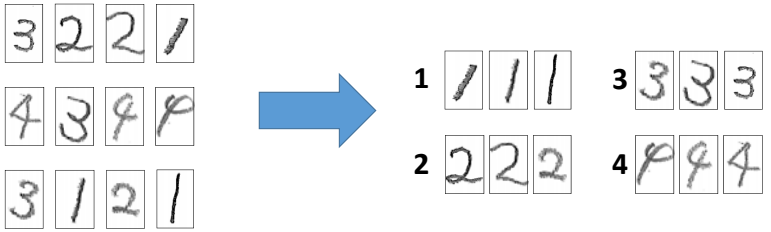
特徴点検出

- パターン認識概論（復習）
- パーセプトロン
- ニューラルネットワーク
- 深層学習

パターン認識

『データの中の規則性を自動的に見つけ出し、その規則性を使ってデータを異なるカテゴリに分類する処理』 *(PRML, C.M. Bishop)*

例) 手書き文字画像の認識



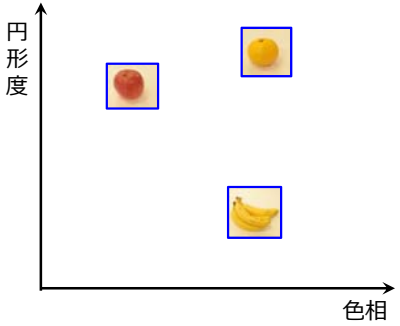
『写真を、リンゴ・バナナ・みかんの3クラスに分類せよ』

特徴抽出: 画像からクラスを良く分離する特徴量（数値データ）を抽出する

(1)平均色相と(2)円形度により、
入力画像を**2D空間**に配置できる

↑

特徴空間



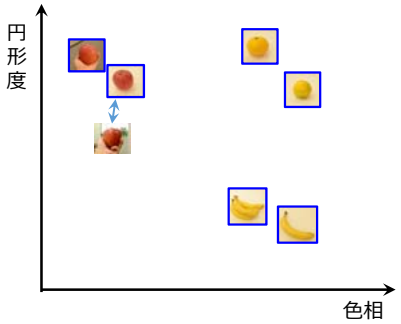
『写真を、リンゴ・バナナ・みかんの3クラスに分類せよ』

識別：特徴空間に入力画像を射影（配置）し、クラスIDを割り当てる

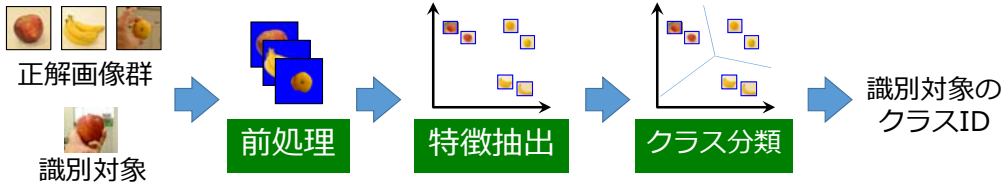
1. 正解画像を特徴空間に射影

2. 分類したい画像も特徴空間射影し距離が一番近い正解画像のIDを返す

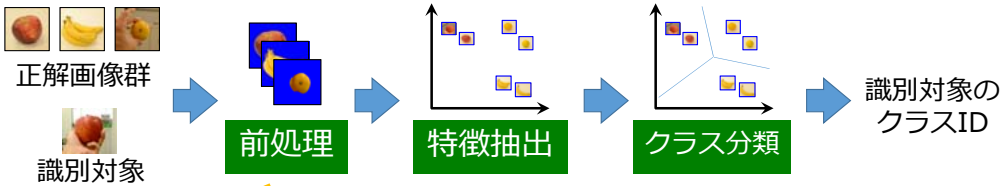
※ Nearest neighbor 法



クラス分類の一般的な処理手順

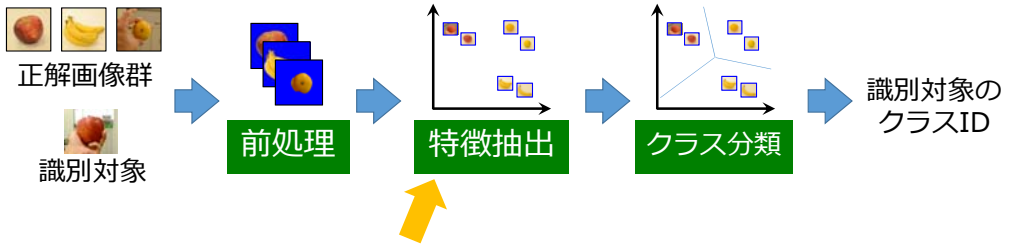


クラス分類の一般的な処理手順



特徴抽出のための前処理
データが画像ならば…
二値化、平滑化、先鋭化、特徴保存平滑化、など

クラス分類の一般的な処理手順



入力データ群に対し，同じクラスは近く・異なるクラス遠くなるような特徴空間にデータを射影する

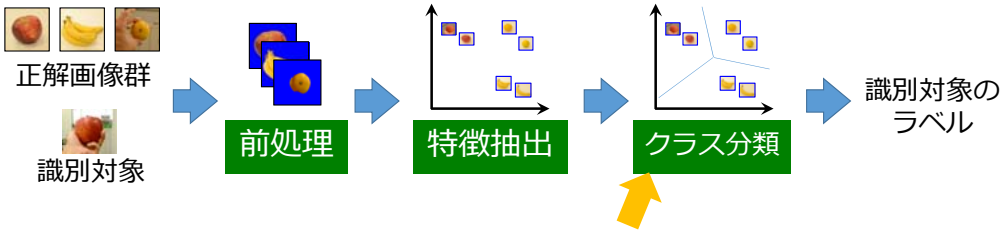
良い特徴空間を構築するには、知識・経験・試行錯誤が必要

画像認識：HLAC・SIFT・HoG特徴などが有名

※最近流行りの深層学習は特徴量の設計もデータから学習する

※深層学習の発展に伴い，人がデザインした特徴量は「Hand Craftな」特徴量と呼ばれる

クラス分類の一般的な処理手順



正解データ群を利用して特徴空間を分割する（訓練）

識別対象を特徴空間に射影し，上記の分割結果を用いてラベルを割り振る

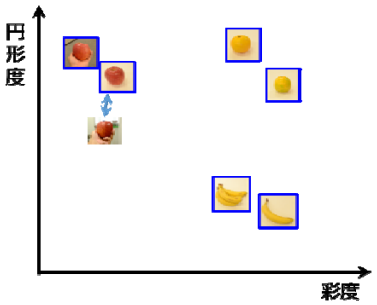
クラス分類の手法

K-Nearest Neighbor, ベイズ決定則, 決定木（random forests）, サポートベクタマシン
ニューラルネットワーク, etc…

準備：クラス識別でやりたいこと

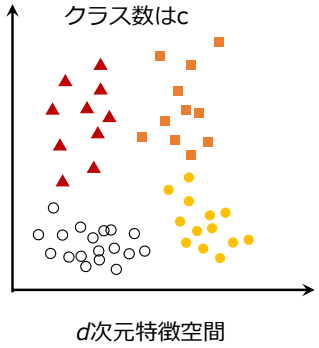
- りんご/バナナ/みかんの写真分類問題を考える
- 入力：N個の教師データ $(\mathbf{x}_i, \mathbf{b}_i),\ i = 1,2,\dots,N$
  - $\mathbf{x}_i \in \boldsymbol{R}^2$ は2次元の特徴ベクトル（色相，円形度）
  - $\mathbf{b}_i \in \boldsymbol{R}^3$ は3次元の教師信号
    - $\mathbf{x}_i$がりんごクラス → $\mathbf{b}_i = (1,0,0)$,
    - $\mathbf{x}_i$がバナナクラス → $\mathbf{b}_i = (0,1,0)$,
    - $\mathbf{x}_i$がみかんクラス → $\mathbf{b}_i = (0,0,1)$,
- 出力：関数 $g(\mathbf{x})$
  - 2次元の特徴ベクトル，3次元のベクトルを返す
  - 教師データを正しく分類できる
  - $g(\mathbf{x}_i) = (b_1, b_2, b_3)$
    - $\mathbf{x}_i$がりんごクラス → $b_1 > b_2,\ b_1 > b_3$
    - $\mathbf{x}_i$がバナナクラス → $b_2 > b_1,\ b_2 > b_3$
    - $\mathbf{x}_i$がみかんクラス → $b_3 > b_1,\ b_3 > b_2$

特徴ベクトルは２次元
クラス数は3



準備：クラス識別でやりたいこと

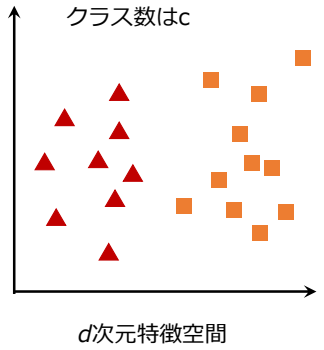
- 一般化すると…
- 入力：N個の教師データ $(\mathbf{x}_i, \mathbf{b}_i),\ i = 1,2,\dots,N$
  - $\mathbf{x}_i \in \boldsymbol{R}^d$ はd次元の特徴ベクトル（特徴1, 特徴2,…）
  - $\mathbf{b}_i \in \boldsymbol{R}^c$ はc次元の教師信号
    - $\mathbf{x}_i$がk番目のクラス
    - → $\mathbf{b}_i = (0,0,\dots,1,\dots,0)$, k次元成分だけ１
- 出力：関数 $\boldsymbol{g}(\mathbf{x})$
  - $\mathbf{x}_i \in \boldsymbol{R}^d,\ \boldsymbol{g}(\mathbf{x}) \in \boldsymbol{R}^c$
  - 教師データを正しく分類できる
  - $g(\mathbf{x}_i) = (b_1, b_2, \dots, b_c)$
    - $\mathbf{x}_i$がk番目のクラス
    - → $b_k > b_i \quad i \neq k$



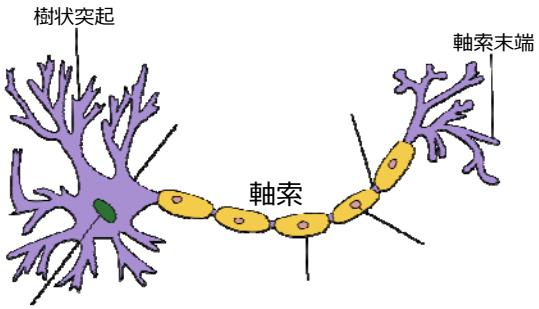
パーセプトロン

パーセプトロン：問題

- 以下の簡単な問題を考える
  - クラス数は２
  - 線形分離可能（超平面で分割可能）
- 教師データ：$(\mathbf{x}_i, b_i)$
  - $\mathbf{x}_i$ d次元ベクトル
  - $b_i = \begin{cases} 1 & \text{if } \mathbf{x}_i\text{がクラス1に属する} \\ 0 & \text{if } \mathbf{x}_i\text{がクラス2に属する} \end{cases}$



神経細胞（ニューロン）

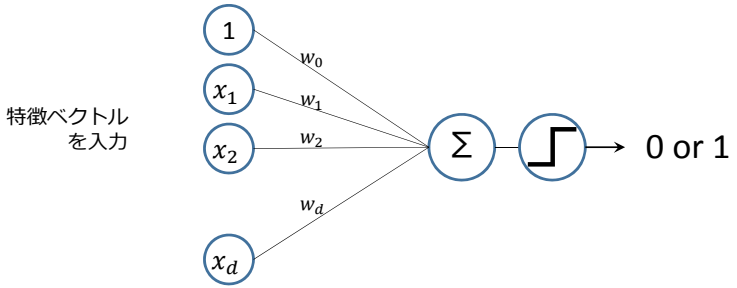


By Quasar Jarosz
[CC-BY-SA-3.0]

- 神経細胞：神経系を構成する単位
- 樹状突起：他の細胞から信号を受け取る（複数ある）
- 軸索末端：他の細胞へ信号を伝達する
- 入力される電気信号の総和がある閾値を超えると，軸索を通じて次の細胞へ信号を送る（発火）

※ これはニューラルネットの説明のときによくある解説で，だいぶ簡略化した説明です。

パーセプトロン：ニューロンの振る舞いをモデル化

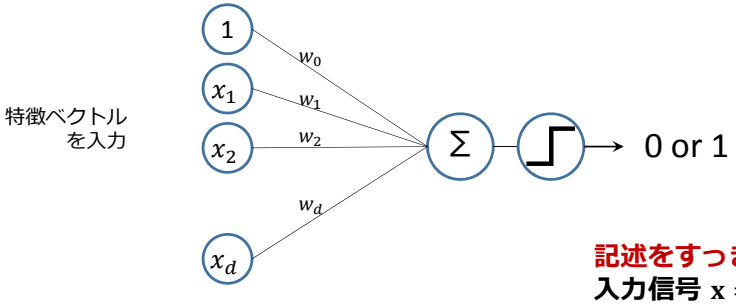


1. 入力信号 $\mathbf{x} = (x_1, \ldots, x_d)$ の重付け和を計算：$w_0 + \sum_i w_i x_i$
2. 総和を用いて閾値処理

$$f(\mathbf{x}) = \begin{cases} 1 & if\ w_0 + \sum_i w_i x_i \geq 0 \\ 0 & if\ w_0 + \sum_i w_i x_i < 0 \end{cases}$$

※教科書にはバイアス項w_0についての記載がないのですが、
ここでは『わかぱた』に順じてバイアス項を記載しています

パーセプトロン：ニューロンの振る舞いをモデル化



1. 入力信号 $\mathbf{x}$の重付け和を計算：$\mathbf{w}^T\mathbf{x} = \sum_i w_i x_i$
2. 総和を用いて閾値処理

$$f(\mathbf{x}) = \begin{cases} 1 & if\ \mathbf{w}^T\mathbf{x} \geq 0 \\ 0 & if\ \mathbf{w}^T\mathbf{x} < 0 \end{cases}$$

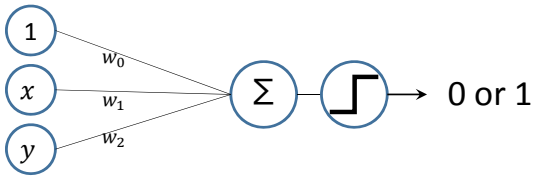
記述をすっきりさせました
入力信号 $\mathbf{x} = (1, x_1, \ldots, x_d)$
重み $\mathbf{w} = (w_0, w_1, \ldots, w_d)$

確認
重み w が未知です
大量の教師データを使ってよい
w を発見します

パーセプトロンの直感的な説明

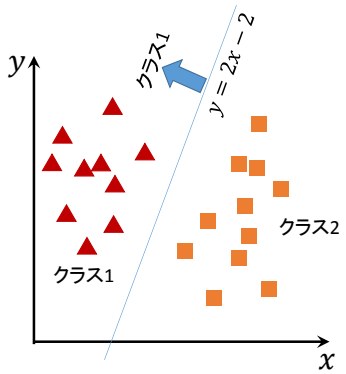
- 特徴空間が2次元の2クラス分類を考える
  - 教師データ $(\mathbf{x}_i, b_i)$
  - $\mathbf{x}_i = (x_i, y_i), b_i \in \{0,1\}, i = 1, \ldots N$
- 教師データから重み(w_0, w_1, w_2)を学習
- 重みを使ってクラスわけ

$$f(x,y) = \begin{cases} 1 & if\ w_0 + w_1 x + w_2 y \geq 0 \\ 0 & if\ w_0 + w_1 x + w_2 y < 0 \end{cases}$$



$(w_0, w_1, w_2) = (2, -2, 1)$のとき

$y \geq 2x - 2$ならばクラス1

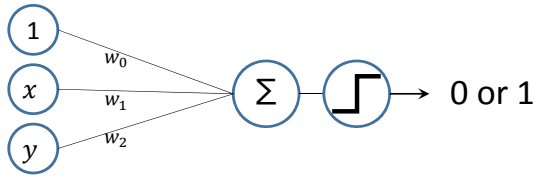


この(w_0, w_1, w_2)はよさそう
パーセプトロンは特徴空間を超平面で分割する手法

パーセプトロンの直感的な説明

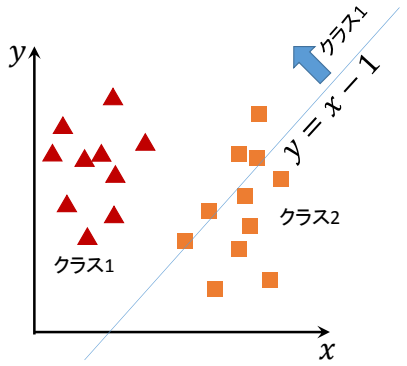
- 特徴空間が2次元の2クラス分類を考える
  - 教師データ $(\mathbf{x}_i, b_i)$
  - $\mathbf{x}_i = (x_i, y_i), b_i \in \{0,1\}, i = 1, \ldots N$
- 教師データから重み(w_0, w_1, w_2)を学習
- 重みを使ってクラスわけ

$$f(x,y) = \begin{cases} 1 & if\ w_0 + w_1 x + w_2 y \geq 0 \\ 0 & if\ w_0 + w_1 x + w_2 y < 0 \end{cases}$$



$(w_0, w_1, w_2) = (1, -1, 1)$のとき

$y \geq x - 1$ならばクラス1



教師データ▲ ■ を正しく分類できていないので
この(w_0, w_1, w_2)はだめ

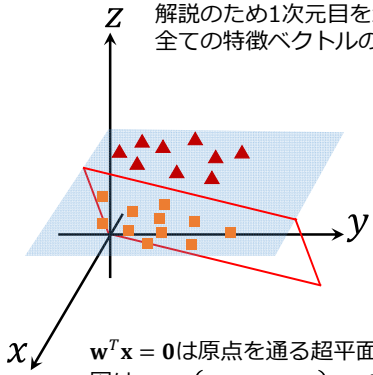
パーセプトロンの直感的な説明（内積表現）

内積で説明されること
も多いので少し解説します

- 特徴空間が2次元の2クラス分類を考える
  - 教師データ $(\mathbf{x}_i, b_i)$
  - $\mathbf{x}_i = (1, x_i, y_i)$ **※ここ3次元で表現**
- 重みベクトル $\mathbf{w} = (w_0, w_1, w_2)$を学習
- クラス分類

$$f(x,y) = \begin{cases} 1 & if\ \mathbf{w}^T\mathbf{x} \geq 0 \\ 0 & if\ \mathbf{w}^T\mathbf{x} < 0 \end{cases}$$

$\mathbf{w}$との内積が正とは、右図の特徴空間において$\mathbf{w}$の方向に位置する（$\mathbf{w}$との成す角が90度以下ということ）
つまり$\mathbf{w}$はクラス1らしさを表現する方向ベクトル



解説のため1次元目をz軸とした
全ての特徴ベクトルのz値は1

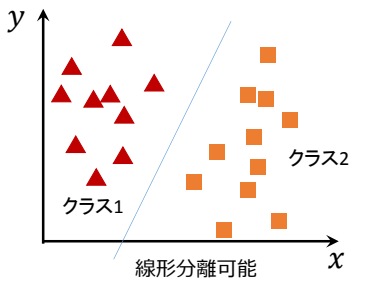
$\mathbf{w}^T\mathbf{x} = \mathbf{0}$は原点を通る超平面となる
図は$\mathbf{w} = (w_0, w_1, w_2) = (2, -2, 1)$のとき
$2z - 2x + y = 0$の平面

パーセプトロン

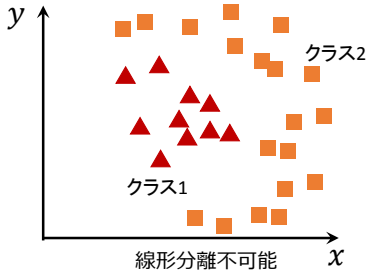
- 超平面で分割できないデータ群はどうするの？

→ パーセプトロンで扱えるのは線形分離可能な問題のみ

- どうやって重みを学習するの？



線形分離可能



線形分離不可能

パーセプトロン：重みの学習

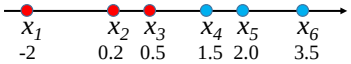
- 入力 :教師データ $(\mathbf{x}_i, b_i)$
  - $\mathbf{x}_i \in R^d, b_i \in \{0,1\}, i = 1, \ldots N$
- 出力：教師データを正しく分割する重み$\mathbf{w}$

- 学習アルゴリズム
  1. 重みベクトル適当（乱数など）に初期化 $\mathbf{w}$
  2. 教師データ群からデータをひとつ選ぶ $(\mathbf{x}, b)$
  3. 現在の重み$\mathbf{w}$で $\mathbf{x}$を識別し結果が誤っているとき

     $\mathbf{w} \leftarrow \mathbf{w} + \rho\mathbf{x}$　（クラス１に対して $\mathbf{w}^T\mathbf{x} < 0$とした）

     $\mathbf{w} \leftarrow \mathbf{w} - \rho\mathbf{x}$　（クラス2に対して $\mathbf{w}^T\mathbf{x} \geq 0$とした）
  4. 全教師データに対して2,3を繰り返す.
  5. 全教師データを正しく識別で来たら終了

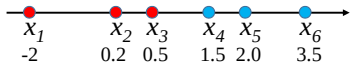
下の例を使ってこのアルゴリズムの動作を見ていきます
※『わかばたも参照』



教師データ

x_1 = -2.0, クラス2
x_2 = 0.2, クラス2
x_3 = 0.5, クラス2
x_4 = 1.5, クラス1
x_5 = 2.0, クラス1
x_6 = 3.5, クラス1

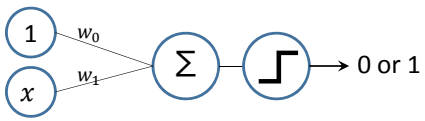
パーセプトロン：重みの学習



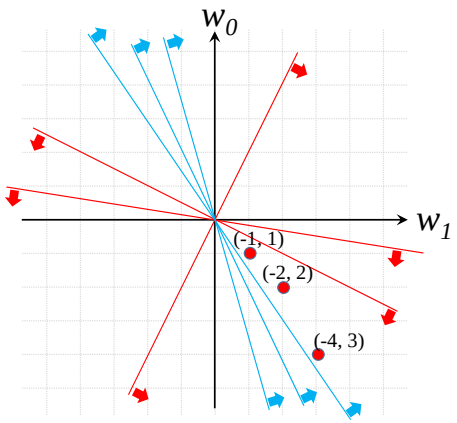
教師データ

x_1 = -2.0, クラス2
x_2 = 0.2, クラス2
x_3 = 0.5, クラス1
x_4 = 1.5, クラス1
x_5 = 2.0, クラス1
x_6 = 3.5, クラス1

$w_0 - 2.0\, w_1 < 0$
$w_0 + 0.2\, w_1 < 0$
$w_0 + 0.5\, w_1 < 0$
$w_0 + 1.5\, w_1 \geqq 0$
$w_0 + 2.0\, w_1 \geqq 0$
$w_0 + 3.5\, w_1 \geqq 0$



w_0が縦軸なので注意

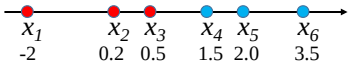


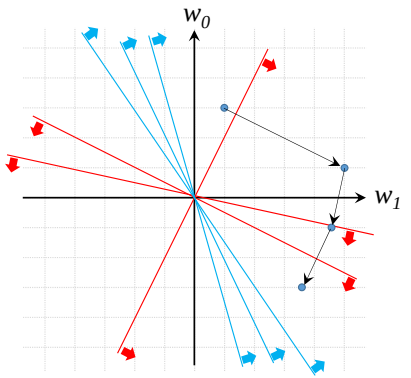
(w_0, w_1) = (-1, 1) , (-2, 2) , (-4, 3)
などは条件を満たす重み

1. ランダムに重みの初期値を決定
2. 以下を全教師データが識別できるまで繰り返す
3. 現在の重み$\mathbf{w}$で $\mathbf{x}$を識別し結果が誤っているとき

   $\mathbf{w} \leftarrow \mathbf{w} + \rho\mathbf{x}$　クラス１に対して $\mathbf{w}^T\mathbf{x} < 0$とした

   $\mathbf{w} \leftarrow \mathbf{w} - \rho\mathbf{x}$　クラス2に対して $\mathbf{w}^T\mathbf{x} \geq 0$とした



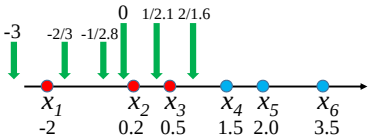


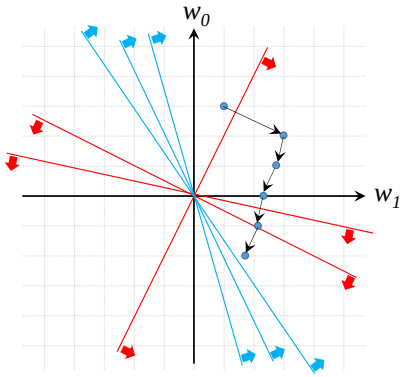
ρ = 2.0として
1. 初期値決定$\mathbf{w}$=(3,1)
x_1=-2.0 を確認 → 3+1.0*(-2.0) > 0 → $\mathbf{w}$=(3, 1.0) - 2*(1, -2.0)=(1, 5)
x_2= 0.2 を確認 → 1+5.0*(0.2) > 0 → $\mathbf{w}$=(1, 5.0) - 2*(1, 0.2)=(-1, 4.6)
x_3= 0.5 を確認 →-1+4.6*(0.5) > 0 → $\mathbf{w}$=(-1, 4.6) - 2*(1, 0.5)=(-3, 3.6)
x_4= 1.5 を確認 →-3+3.6*(1.5) > 0 OK $\mathbf{w}$=(-3, 3.6)
x_5= 2.0 を確認 →-3+3.6*(2.0) > 0 OK $\mathbf{w}$=(-3, 3.6)
x_6= 3.5 を確認 →-3+3.6*(3.5) > 0 OK $\mathbf{w}$=(-3, 3.6)
x_1=-2.0 を確認 →-3+3.6*(-2.0) > 0 OK $\mathbf{w}$=(-3, 3.6)
x_2= 0.2 を確認 →-3+3.6*(0.2) > 0 OK $\mathbf{w}$=(-3, 3.6)
x_3= 0.5 を確認 →-3+3.6*(0.5) > 0 OK $\mathbf{w}$=(-3, 3.6)

1. ランダムに重みの初期値を決定
2. 以下を全教師データが識別できるまで繰り返す
3. 現在の重み$\mathbf{w}$で $\mathbf{x}$を識別し結果が誤っているとき

   $\mathbf{w} \leftarrow \mathbf{w} + \rho\mathbf{x}$　クラス1に対して $\mathbf{w}^T\mathbf{x} < 0$とした

   $\mathbf{w} \leftarrow \mathbf{w} - \rho\mathbf{x}$　クラス2に対して $\mathbf{w}^T\mathbf{x} \geq 0$とした





ρ = 1.0として
初期値決定$\mathbf{w}$=(3,1)
x_1=-2.0 を確認 → 3+1.0*(-2.0) > 0 → $\mathbf{w}$=(3, 1.0) - 1*(1, -2.0)=(2, 3)
x_2= 0.2 を確認 → 2+3.0*(0.2) > 0 → $\mathbf{w}$=(2, 3.0) - 1*(1, 0.2)=(1, 2.8)
x_3= 0.5 を確認 → 1+2.8*(0.5) > 0 → $\mathbf{w}$=(1, 2.8) - 1*(1, 0.5)=(0, 2.3)
x_4= 1.5 を確認 → 0+2.3*(1.5) > 0 OK $\mathbf{w}$=(0, 2.3)
x_5= 2.0 を確認 → 0+2.3*(2.0) > 0 OK $\mathbf{w}$=(0, 2.3)
x_6= 3.5 を確認 → 0+2.3*(3.5) > 0 OK $\mathbf{w}$=(0, 2.3)
x_1=-2.0 を確認 → 0+2.3*(-2.0) < 0 OK $\mathbf{w}$=(0, 2.3)
x_2= 0.2 を確認 → 0+2.3*(0.2) > 0 → $\mathbf{w}$=(0, 2.3) - 1*(1, 0.2)=(-1, 2.1)
x_3= 0.5 を確認 →-1+2.1*(0.5) > 0 → $\mathbf{w}$=(-1, 2.1) - 1*(1, 0.5)=(-2, 1.6)

以降、全てを満たすので更新無し

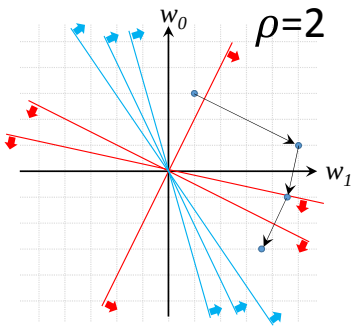
パーセプトロン：重みの学習

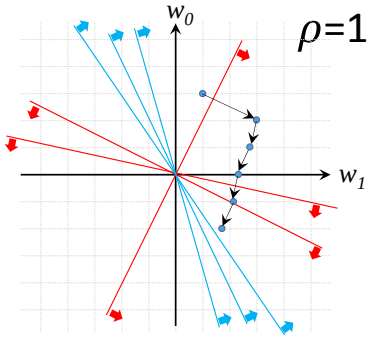
1. ランダムに重みの初期値を決定
2. 以下を全教師データが識別できるまで繰り返す
3. 現在の重み$\mathbf{w}$で $\mathbf{x}$を識別し結果が誤っているとき

   $\mathbf{w} \leftarrow \mathbf{w} + \rho\mathbf{x}$　クラス1に対して $\mathbf{w}^T\mathbf{x} < 0$とした

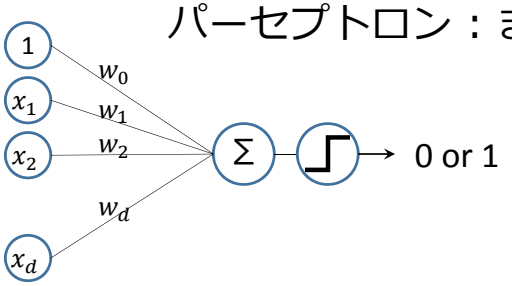
   $\mathbf{w} \leftarrow \mathbf{w} - \rho\mathbf{x}$　クラス2に対して $\mathbf{w}^T\mathbf{x} \geq 0$とした

- ひとつずつデータをチェックし、誤識別していたら重みを更新
- ρは更新の際の幅を示す





パーセプトロン：まとめ



$\mathbf{x} = (1, x_1, \ldots, x_d)$
$\mathbf{w} = (w_0, w_1, \ldots, w_d)$

パーセプトロンとはニューロンの振る舞いをモデル化した識別器

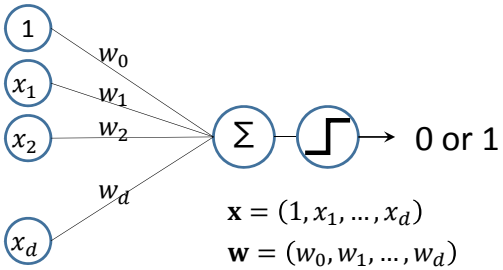
識別: 入力信号 $\mathbf{x}$ の重付け和(内積)を計算し閾値処 $f(\mathbf{x}) = \begin{cases} 1 & if \mathbf{w}^T\mathbf{x} \geq 0 \\ 0 & if \mathbf{w}^T\mathbf{x} < 0 \end{cases}$

学習: 教師データ$(\mathbf{x}_i, b_i)$を順番に利用し，重みベクトル$\mathbf{w}$を更新する

限界: 上記のモデルは2クラス分類 & 線形分離可能なときのみ有効

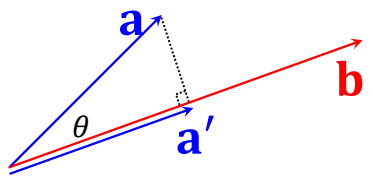
※学習過程を１Dの例を用いて解説しました．是非2Dや3Dの例についても考えてみてください

パーセプトロンの性質



$\mathbf{x} = (1, x_1, \ldots, x_d)$
$\mathbf{w} = (w_0, w_1, \ldots, w_d)$

内積



$\mathbf{a} \cdot \mathbf{b} = |\mathbf{a}||\mathbf{b}|\cos\theta$

- 特徴ベクトル$\mathbf{x}$と重みベクトル$\mathbf{w}$の内積が0以上なら1を出力
- 特徴ベクトル$\mathbf{x}$と重みベクトル$\mathbf{w}$が**似ていたら**1を出力

ニューラルネットワークへ

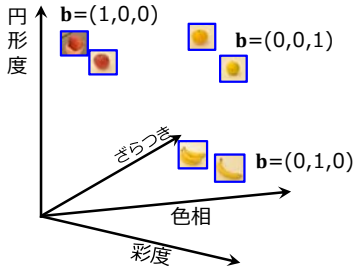
- ここでは，どのような仕組みで動作するかを中心に解説します．
- 学習法の詳細は『パターン認識の講義』を取るか，『分かりやすいパターン認識』を読むか，『Back propagation』で検索してください

- 井尻も試行錯誤してみたのですが、九州大学内田誠一先生の講義資料がとても分かりやすく，これよりうまい説明ができそうになかったのでそちらに習って解説します
- ↓内田先生の講義資料
- https://www.slideshare.net/SeiichiUchida/ss-71479583

解きたい問題

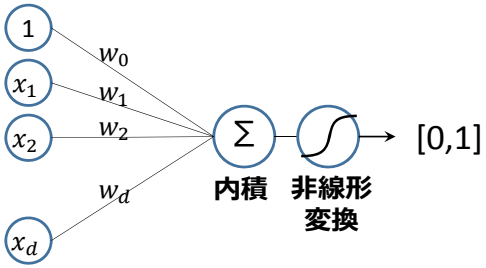
- 入力：N個の教師データ $(\mathbf{x}_i, \mathbf{b}_i),\ i = 1, \dots, N$
  - $\mathbf{x}_i \in \boldsymbol{R}^{\boldsymbol{d}}$ はd次元の特徴ベクトル
  - $\mathbf{b}_i \in \boldsymbol{R}^{c}$ はc次元の教師信号
  - $\mathbf{x}_i$がクラスk → $\mathbf{b}_i = (0, \dots, 1, \dots, 0)$ k成分だけ1
- 出力：関数 $\boldsymbol{g}(\mathbf{x})$
  - $\mathbf{x}_i \in \boldsymbol{R}^{\boldsymbol{d}}$, $\boldsymbol{g}(\mathbf{x}) \in \boldsymbol{R}^{c}$
  - 教師データを正しく分類できる
  - $g(\mathbf{x}_i) = (b_1, b_2, \dots, b_c)$
    - $\mathbf{x}_i$がk番目のクラス
    → $b_k > b_i \quad i \neq k$

例
- 果物写真の分類
  - 特徴ベクトルは4D
  →(円形度，彩度，色相，ざらつき)
  - クラス数は3
  →(りんご，バナナ，みかん)

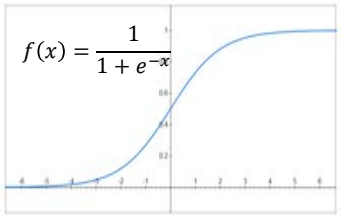


ニューラルネットワーク: ユニット

- ニューラルネットワークを構成する最小単位をユニット（又はニューロン）と呼ぶ
- ユニットは，重み係数$\mathbf{w}$と非線形関数fを持つ
- 入力信号$\mathbf{x}$を受け取り，係数との内積$\mathbf{w}^T\mathbf{x}$を計算し，非線形関数にかけて，[0,1]の実数値を返す
- **入力信号$\mathbf{x}$が係数$\mathbf{w}$と似ていると大きな値を返す**



非線形変換部分には、シグモイド関数などが利用される

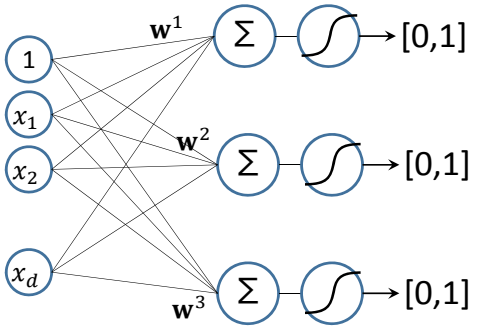


シグモイド関数

※非線形変換がないとこのユニットを直列につなぐうまみがあまりないです．さらに深く理解するとこの意味はわかるかと…

ニューラルネットワーク: ユニットを並列につなぐ

- ユニットは入力信号が重みと似ているときに大きな値をかえすので…
- 複数のユニットを並列に並べたら，複数クラスを扱えるのでは？



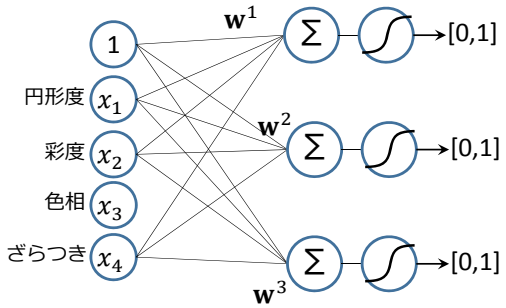
入力信号が$\mathbf{w}^1$に似てるとき大きい値に

入力信号が$\mathbf{w}^2$に似てるとき大きい値に

入力信号が$\mathbf{w}^3$に似てるとき大きい値に

ニューラルネットワーク: ユニットを並列につなぐ

- 果物写真の分類を考える
  - 特徴ベクトルは4D →(円形度，彩度，色相，ざらつき)
  - クラス数は3→(りんご，バナナ，みかん)



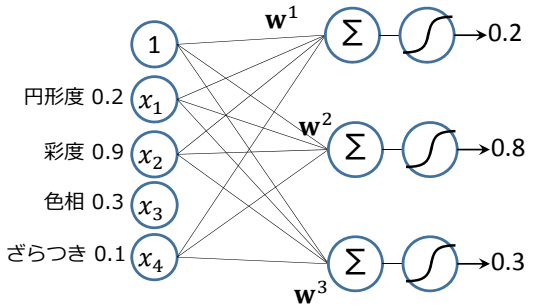
$\mathbf{w}^1$は**りんごの写真**が持つ特徴ベクトルに似せておく

$\mathbf{w}^2$は**バナナの写真**が持つ特徴ベクトルに似せておく

$\mathbf{w}^3$は**みかんの写真**が持つ特徴ベクトルに似せておく

ニューラルネットワーク: ユニットを並列につなぐ

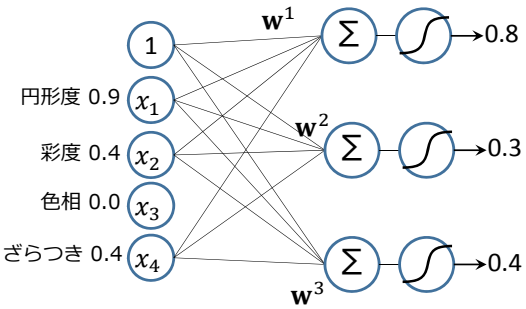
- 左のバナナの写真が入力されたとき…
- (円形度，彩度，色相，ざらつき) = (0.2, 0.9, 0.3, 0.1)
  - クラス数は3→(りんご，バナナ，みかん)



(0.2, 0.8, 0.3)が出力されたので
クラス２に分類しよう

ニューラルネットワーク: ユニットを並列につなぐ

- 左のりんごの写真が入力されたとき…
- (円形度，彩度，色相，ざらつき) = (0.9, 0.4, 0.0, 0.4)
  - クラス数は3→(りんご，バナナ，みかん)

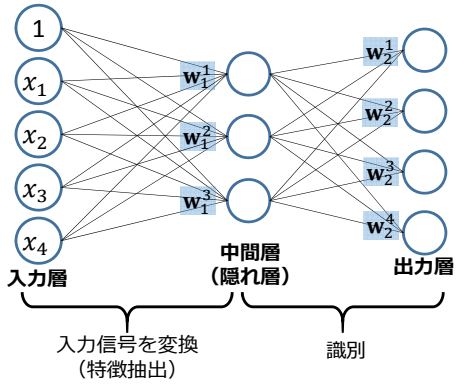


(0.8, 0.3, 0.4)が出力されたので
クラス1に分類しよう

ニューラルネットワーク: ユニットを直列につなぐ

- 入力層と出力層の間に中間層をはさみこむ
- 入力信号(特徴ベクトル)を識別しやすい形に変換してから識別する

  → 線形分離不可能な問題にも対応できる

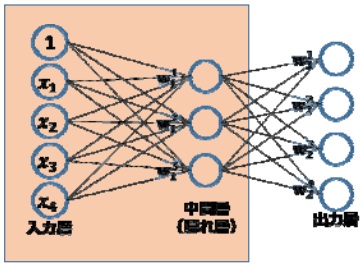


左図では非線形関数を省略（実際は計算）

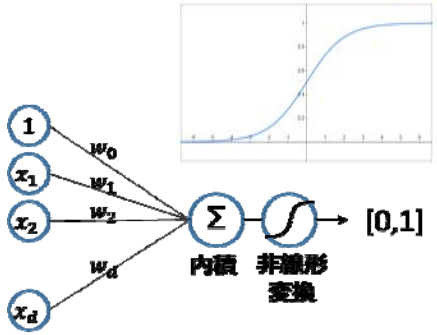
重み係数が未知数で，これを教師データから学習する

→左図では 5*3 + 3*4 = 27個の未知数

ニューラルネットワーク: 中間層の効果について



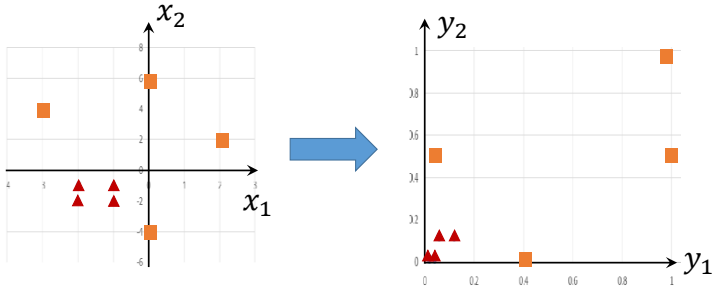
この部分にはどんな効果があるの？



各ユニットは

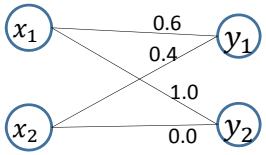
- 入力信号と重みの内積を計算し非線形変換
- 入力と重みが似ているほど大きな値を返す
- 非線形変換は大小を強調する，大きいものはより大きく小さいものはより小さく

ニューラルネットワーク: 中間層の効果について



Class 1
(-2,-2) → (0.18, 0.18)
(-1,-1) → (0.12, 0.12)
(-1,-2) → (0.06, 0.12)
(-2,-1) → (0.04, 0.02)

Class 2
(0,-4) → (0.04, 0.50)
(2, 2) → (0.98, 0.98)
(0, 6) → (0.99, 0.50)
(-3, 4) → (0.40 , 0.00)

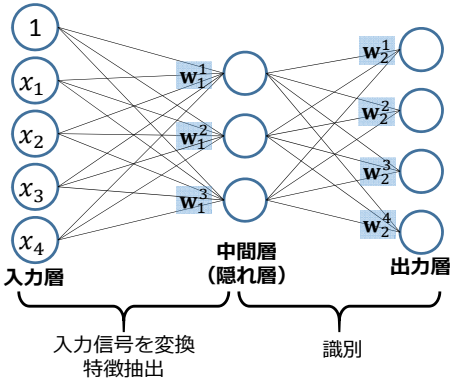


$\mathbf{w}_1 = (0.6, 0.4)$
$\mathbf{w}_2 = (1.0, 0.0)$
$f(x) = \frac{1}{1+\exp(-2x)}$

ユニットによる変換により線形分離不可能な分布が線形分離可能に

まとめ: ニューラルネットワーク

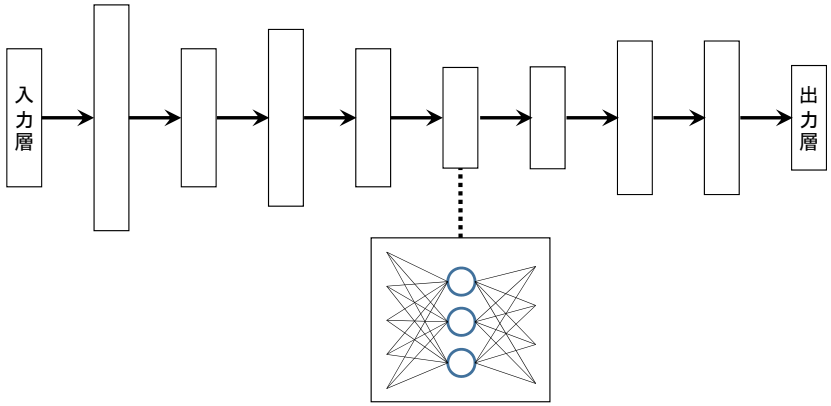
- 複数のユニット（ニューロン）を層状につないだネットワーク
- 入力層が特徴ベクトルを受け取り，出力層から識別結果が出力される



- 未知数が多いため，この学習には大量の教師データを必要とします
- 学習方法については『back propagation で検索』『パターン認識の講義を取る』『教科書を参照』してください

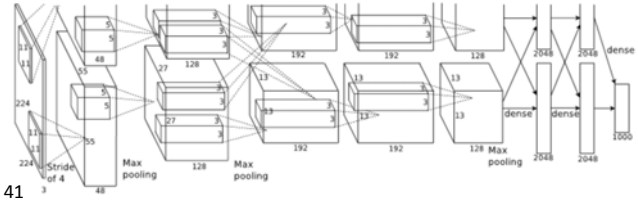
深層学習：Deep learning

- 7層 ~ 20層など，多層構造を持つニューラルネットワークのこと
- 特徴抽出を繰り返し，最後に識別を行なう（最近は違うものも多い）



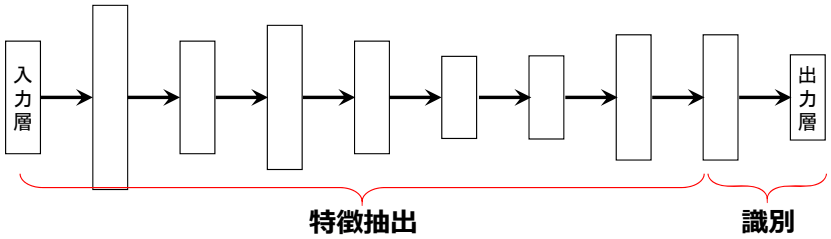
深層学習：Deep learning

- ニューラルネットワークの歴史的な背景
  - 1960年代: パーセプトロンが考案される → 線形分離不可能な問題を扱えないことが分かる
  - 1980年代: 多層パーセプトロンや誤差逆伝播法が考案される
    - [RUMELHART, HINTON, WILLIAMS, 1986] [Fukishima & Miyake Neocognitron, 1982.]
    →当時の計算機能力&データ量では，大規模なネットワークの学習に限界があった
    →2000年代の前半は冬の時代.
  - 2006年：多層NNにおける革新, Deep auto encoder の活用 [Hinton and Salakhutdinov]
  - 2012年：ILSVRC'12(画像識別コンペ)でDeep learningベースのAlexNetが圧勝



AlexNetの構造
画像は[Alex Krizhevsky et al. ImageNet Classification with Deep Convolutional Neural Networks]より

Deep learningの非常に簡単な説明



- 深く大規模なDeep Neural Networkを用いた識別器
- 出力層の直前まで特徴抽出を繰り返し，最後の層で識別を行なう
- End-to-endな構造：データから特徴抽出をせず生データから出力を得る
  - 画像データから特徴を抽出せず，画像データそのものを入力層に入れる
  - つまり，深層学習は特徴抽出自体と識別方法を学習する
  - 深層学習の流行後，従来の人が設計する特徴量を"hand-craft feature"と呼ぶことも

※画像自体を出力するようなDNNも存在する

 ※画像畳み込みをする層を持つConvolutional Neural Networkも有名に

→ 研究室で学ぶことになると思います

誤差逆伝搬
back propagation

代数の定義：簡単のため，入力層・中間層・出力層の3層から構成されるNNを考える

図の通り3個のユニットに特に注目する

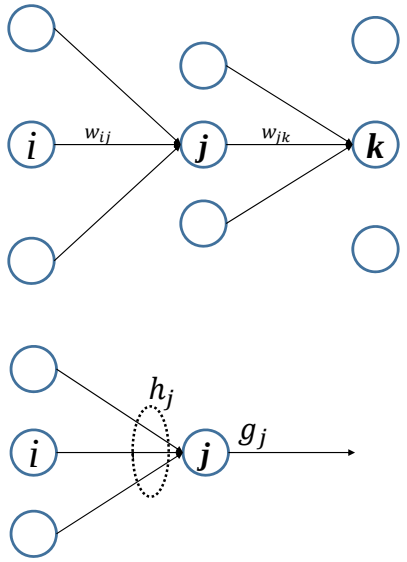
- 入力層の i 番目のユニット
- 中間層の j 番目のユニット
- 出力層の k 番目のユニット

ベクトル$\mathbf{x}$を入力層へ入力したときの各ユニットへの入出力を以下の通り定義する

- ユニットi への入力　$:h_i = x_i$
- ユニット j への入力　$: h_j = \sum_i w_{ij}\, g_i$
- ユニットk への入力　$: h_k = \sum_i w_{jk}\, g_j$
- ユニットi からの出力 : $g_i = x_i$
- ユニットj からの出力 : $g_j = f(h_j)$
- ユニットk からの出力 : $g_k = f(h_k)$

※ w_{ij} はユニットiとユニットj間の重み係数

※ fは微分可能な非線形関数



補足資料

誤差逆伝播法

教師データ(**x, t**)に対するNNの出力を g_kとする.
コスト関数 J を以下の通り定義する

$$J = \frac{1}{2}\sum_k (g_k - t_k)^2$$

重みを更新しJを最小化するのが目的.
最急降下法を適用する.

$$w_{ij} \leftarrow w_{ij} - \rho\frac{\partial J}{\partial w_{ij}}$$

この $\frac{\partial J}{\partial w_{ij}}$ が誤差を利用すると綺麗に表現できる

補足資料

誤差逆伝播法 - 出力層について

$$w_{jk} \leftarrow w_{jk} - \rho\frac{\partial J}{\partial w_{jk}}$$

$$J = \frac{1}{2}\sum_k (g_k - t_k)^2$$

$\frac{\partial J}{\partial w_{jk}}$ は以下の通り計算できる.

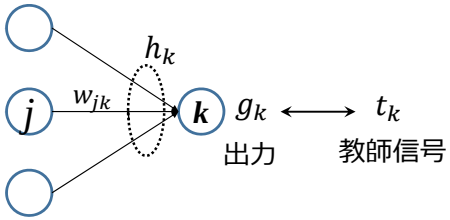
$$\frac{\partial J}{\partial w_{jk}} = \frac{\partial J}{\partial h_k}\frac{\partial h_k}{\partial w_{jk}} \quad \dots(1)$$

$$\frac{\partial h_k}{\partial w_{jk}} = \frac{\partial \sum_a w_{ak} g_a}{\partial w_{jk}} = g_j \quad \dots(2)$$

$$\frac{\partial J}{\partial h_k} = \epsilon_k = \frac{\partial J}{\partial g_k}\frac{\partial g_k}{\partial h_k} = (g_k - t_k)f'(h_k) \quad \dots(3)$$

式(1)に(2,3)を代入すると以下が得られる

$$\frac{\partial J}{\partial w_{jk}} = \epsilon_k g_j = (g_k - t_k)f'(h_k)g_j$$



$$\epsilon_k = \frac{\partial J}{\partial g_k} = (g_k - t_k)f'(h_k)$$

補足資料
目的関数JはLossと呼ばれ
このスライドとは違うものもよく利用される

誤差逆伝播法 – 中間層について

$$w_{ij} \leftarrow w_{ij} - \rho\frac{\partial J}{\partial w_{ij}}$$

$$J = \frac{1}{2}\sum_k (g_k - t_k)^2$$

$\frac{\partial J}{\partial w_{ij}}$ は以下の通り計算できる.

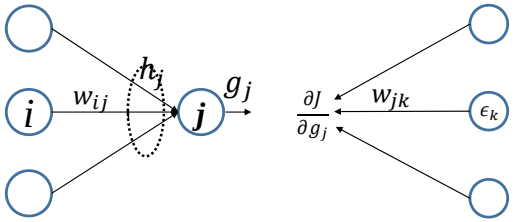
$$\frac{\partial J}{\partial w_{ij}} = \frac{\partial J}{\partial h_j}\frac{\partial h_j}{\partial w_{ij}} \quad \dots(1)$$

$$\frac{\partial h_j}{\partial w_{ij}} = \frac{\partial \sum_a w_{aj} g_a}{\partial w_{ij}} = g_i \quad \dots(2)$$

$$\frac{\partial J}{\partial h_j} = \epsilon_j = \frac{\partial J}{\partial g_j}\frac{\partial g_j}{\partial h_j} = \left(\sum_k \epsilon_k w_{jk}\right)f'(h_j) \quad \dots(3)$$

式(1)に(2,3)を代入すると以下が得られる

$$\frac{\partial J}{\partial w_{ij}} = \epsilon_j g_i = \left(\sum_k \epsilon_k w_{jk}\right)f'(h_j)g_i$$



$$\frac{\partial J}{\partial g_j} = \sum_k \frac{\partial J}{\partial h_k}\frac{\partial h_k}{\partial g_j} = \sum_k \epsilon_k w_{jk}$$

多変数関数の連鎖律を利用
(Jはh_kの関数で，h_kはg_jの関数)
$\epsilon_k = \frac{\partial J}{\partial h_k}$ と$\frac{\partial h_k}{\partial g_j} = w_{jk}$を利用した

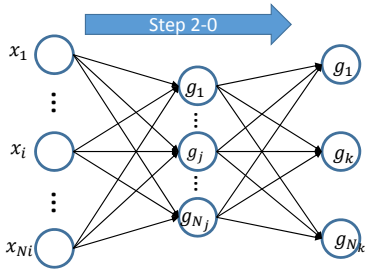
補足資料

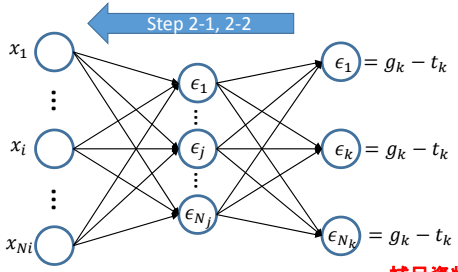
誤差逆伝播法

ひとつずつ（または一定数）教師データ(**x, t**)を読み込み，
誤差を利用して逐次的に重みw_{ij}を更新する手法

1. 重みは何かしらの方法（ランダムとか）で初期化する
2. ある教師データ(**x, t**)を読み込み…
   - 2-0. NNを計算し全層における出力 を得る
   - 2-1. 出力層の誤差を計算
     ユニットkの誤差は $\epsilon_k = (g_k - t_k)f'(h_k)$
   - 2-2. 中間層の誤差を計算
     ユニットjの誤差は $\epsilon_j = \left(\sum_k w_{jk}\epsilon_k\right)f'(h_j)$
   - 2-3. 重みを以下の通り更新
     $w_{ij} \leftarrow w_{ij} - \rho\epsilon_j g_i$
     $w_{jk} \leftarrow w_{jk} - \rho\epsilon_k g_j$

教師データをひとつピックアップし、
2-0 前進方向に出力を計算し
2-1, 2-2 逆方向に誤差を計算し
2-3 出力と誤差を用いて重みを更新





補足資料