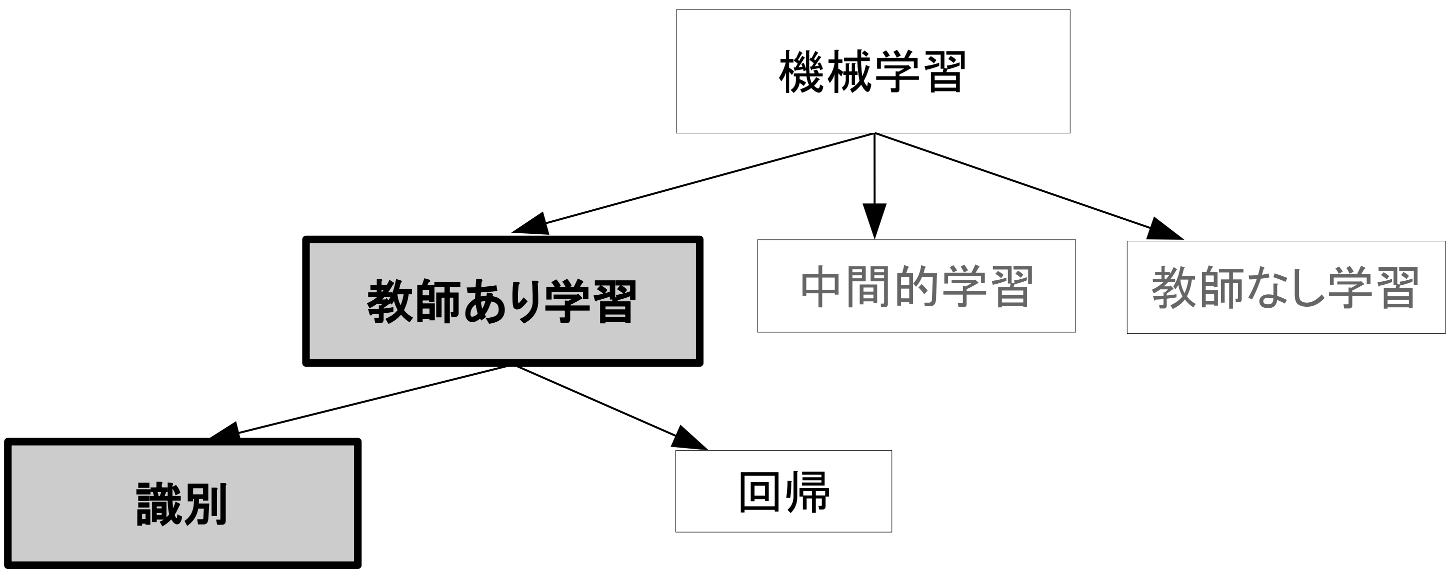
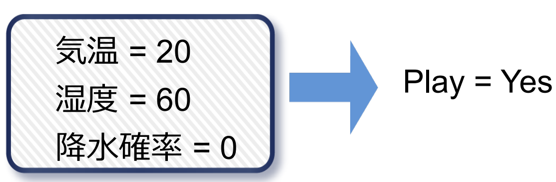
# 5. 識別　ー生成モデルと識別モデルー

- 問題設定
  - 教師あり学習
  - 数値入力　→　カテゴリ出力




- 数値特徴



# 5.1 数値特徴に対する「教師あり・識別」問題の定義

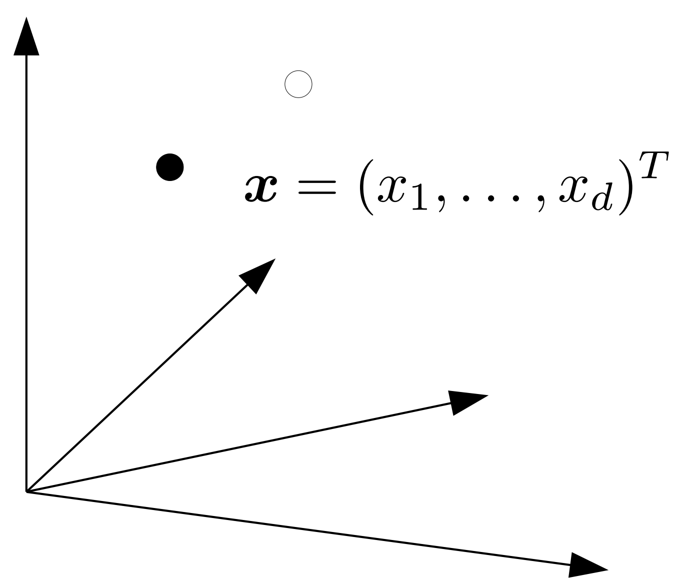
- 教師あり学習のデータ
  - 特徴ベクトル $\boldsymbol{x}$ と正解情報 $y$ のペア

$$\{(\boldsymbol{x}_i, y_i)\}, \quad i = 1 \ldots N$$

  - $\boldsymbol{x}$ は次元数 $d$ の固定長ベクトル、 $y$ はカテゴリ

$$\boldsymbol{x}_i = (x_{i1}, \ldots, x_{id})^T$$

  - $\boldsymbol{x}$ は $d$ 次元空間（特徴空間）上の点と見なせる

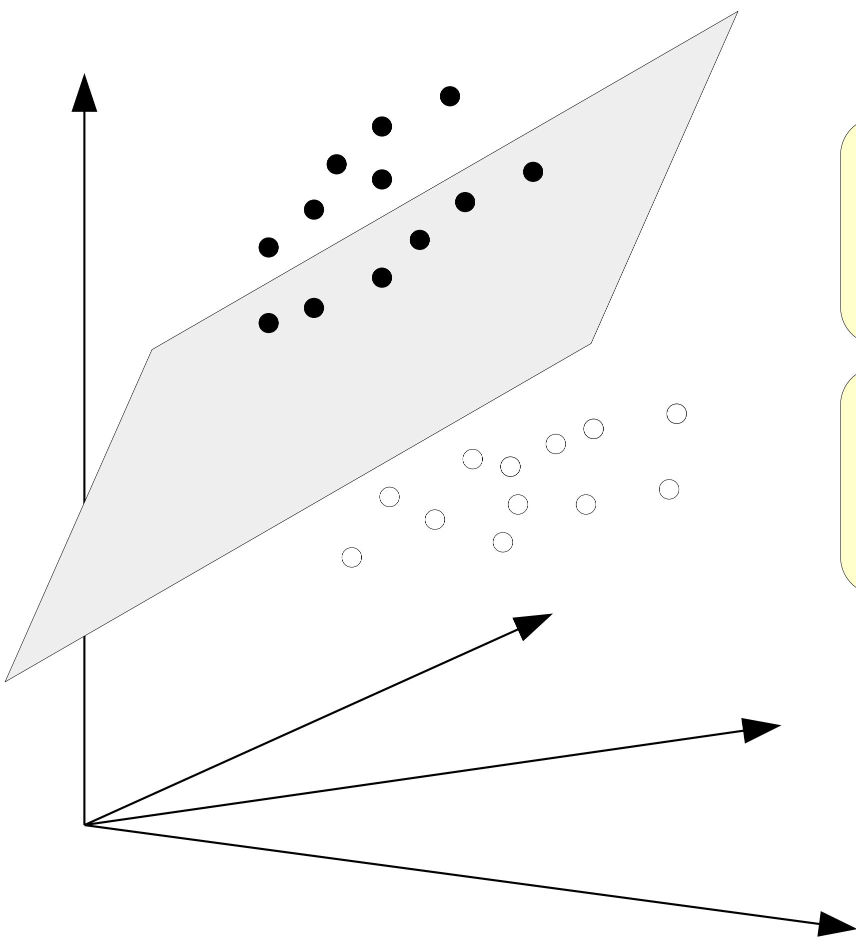



● $y$ = positive （正例）

○ $y$ = negative （負例）

# 5.1 数値特徴に対する「教師あり・識別」問題の定義

- 数値特徴に対する識別問題＝境界面の設定
  - 各クラスの確率分布を求めることで、結果として境界面（等確率となる点の集合）が定まる場合も含む



クラスが比較的きれいに分離している
→ 小数のパラメータで各クラスを表現可能
⇒ **統計的手法（第５章）**

クラス境界が複雑
高次元へマッピング⇒ **SVM （第 7 章）**
非線形識別面 ⇒ **ニューラルネット（第 8 章）**

# 4.1 統計的識別とは

復習

- 最大事後確率則による識別

$$C_{MAP} = \arg\max_i P(\omega_i|\boldsymbol{x})$$

$\boldsymbol{x}$ ：特徴ベクトル

$\omega_i \quad (1 \leq i \leq c)$ ：クラス

  - データから直接的にこの確率を求めるのは難しい

- ベイズの定理 $$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

$$C_{MAP} = \arg\max_i P(\omega_i|\boldsymbol{x})$$

$$= \arg\max_i \frac{P(\boldsymbol{x}|\omega_i)P(\omega_i)}{P(\boldsymbol{x})}$$

ベイズの定理

$$= \arg\max_i \underbrace{P(\boldsymbol{x}|\omega_i)}_{\text{尤度}}\underbrace{P(\omega_i)}_{\text{事前確率}}$$

上式の分母は
判定に寄与しない

# 4.1 統計的識別とは

復習

- 事前確率 $P(\omega_i)$
  - 特徴ベクトル $\boldsymbol{x}$ を観測する前の、各クラスの起こりやすさ
- 事前確率の最尤推定

$$P(\omega_i) = \frac{n_i}{N}$$

$N$: 全データ数、 $n_i$: クラス $\omega_i$ のデータ数

# 4.2.2 ナイーブベイス識別

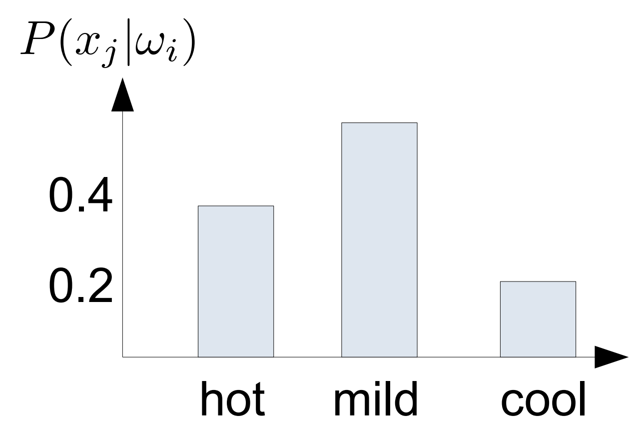
復習

- 尤度計算におけるナイーブベイズの近似
  - すべての特徴が独立であると仮定

$$P(\boldsymbol{x}|\omega_i) = P(x_1, \ldots, x_d|\omega_i)$$
$$\approx \prod_{j=1}^{d} P(x_j|\omega_i)$$

$$C_{NB} = \arg\max_i P(\omega_i) \prod_{j=1}^{d} P(x_j|\omega_i)$$

- $x_j$がカテゴリ特徴のとき
  - $y = \omega_i$のデータから、カテゴリ分布を最尤推定

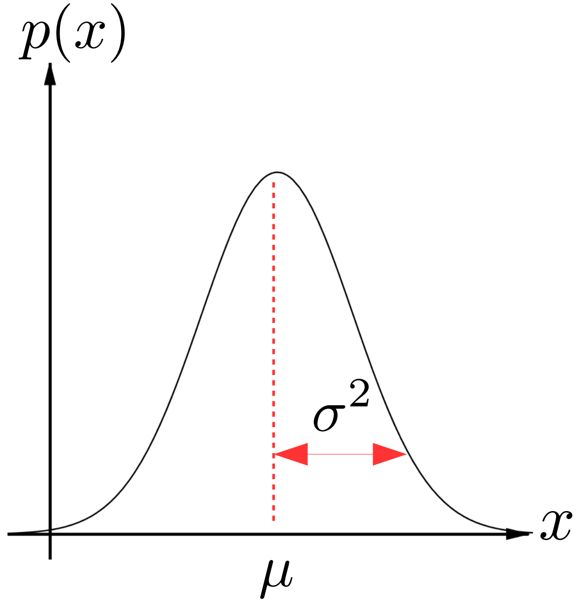


# 5.2 生成モデル
## 5.2.1 数値特徴に対するナイーブベイズ識別

- 確率密度関数 $p(x_j|\omega_i)$ の推定
  - 関数をクラス毎に求めるので $\omega_i$ は省略
  - 関数の形は正規分布を仮定

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

  - データの対数尤度を最大とする
    平均 $\mu$ と分散 $\sigma^2$ を求める



# 正規分布とは

- 離散型二項分布の例
  - n 枚のコインを投げた時の、表の枚数の度数
    - n=1　1 1
    - n=2　1 2 1
    - n=3　1 3 3 1
    - n=4　1 4 6 4 1
    - n=5　1 5 10 10 5 1

    …
  - n→∞　の時が正規分布
    - さまざまな要因が重なって生じた数値がこの分布になる

    例）テストの点数（ n 問からなる問題の結果）

    身長（遺伝、食事、運動、睡眠、 … ）

## 5.2.1 数値特徴に対するナイーブベイズ識別

- データの対数尤度（最大化したい）

$$\mathcal{L}(D) = \log P(D|\mu, \sigma^2) = \sum_{i=1}^{N} \log P(\boldsymbol{x}_i|\mu, \sigma^2)$$

- 正規分布の式を当てはめる

$$\mathcal{L}(D) = -\frac{N}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{N}(x_i - \mu)^2$$

- $\mu$ で偏微分して 0 とおく

$$\frac{1}{\sigma^2}\sum_{i=1}^{N}(x_i - \mu) = 0 \Rightarrow \mu = \frac{1}{N}\sum_{i=1}^{N} x_i$$

- $\sigma^2$ で偏微分して 0 とおく

$$-\frac{N}{2}\frac{1}{\sigma^2} + \frac{1}{2(\sigma^2)^2}\sum_{i=1}^{N}(x_i - \mu)^2 = 0 \Rightarrow \sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2$$

求める分布の平均はデータの平均、分散はデータの分散というごく当たり前の結果

## 5.2.2 生成モデルの考え方

- 事後確率を求めるにあたって、同時確率を求めている
  - データが生成される様子をモデル化していると見ることも出来る
    - 事前確率に基づいてクラスを選ぶ
    - そのもとで、特徴ベクトルを出力する

$$P(\omega_i|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\omega_i)P(\omega_i)}{p(\boldsymbol{x})} = \frac{p(\omega_i, \boldsymbol{x})}{p(\boldsymbol{x})}$$

事後確率を求めるより、難しい問題を解いているのではないか？

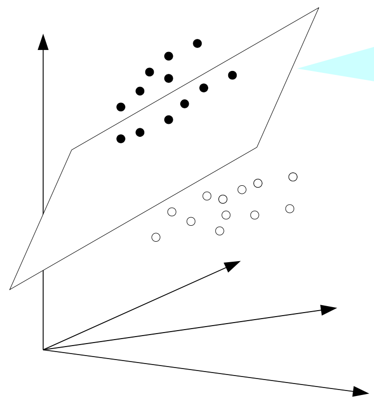
# 5.3 識別モデル

- 識別関数法
  - 確率の枠組みにはとらわれず、 $f_P(\boldsymbol{x}) > f_N(\boldsymbol{x})$ ならば $\boldsymbol{x}$ を positive と判定する関数を推定する
  - 2 クラス問題なら $f(\boldsymbol{x}) = f_P(\boldsymbol{x}) - f_N(\boldsymbol{x})$ の正負で判定すればよい（ $f(\boldsymbol{x}) = 0$ が識別面）
  - 単層パーセプトロン

最も単純な識別関数法の実現

    - 識別関数として 1 次式（＝直線・平面）を仮定

$$f(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x}$$



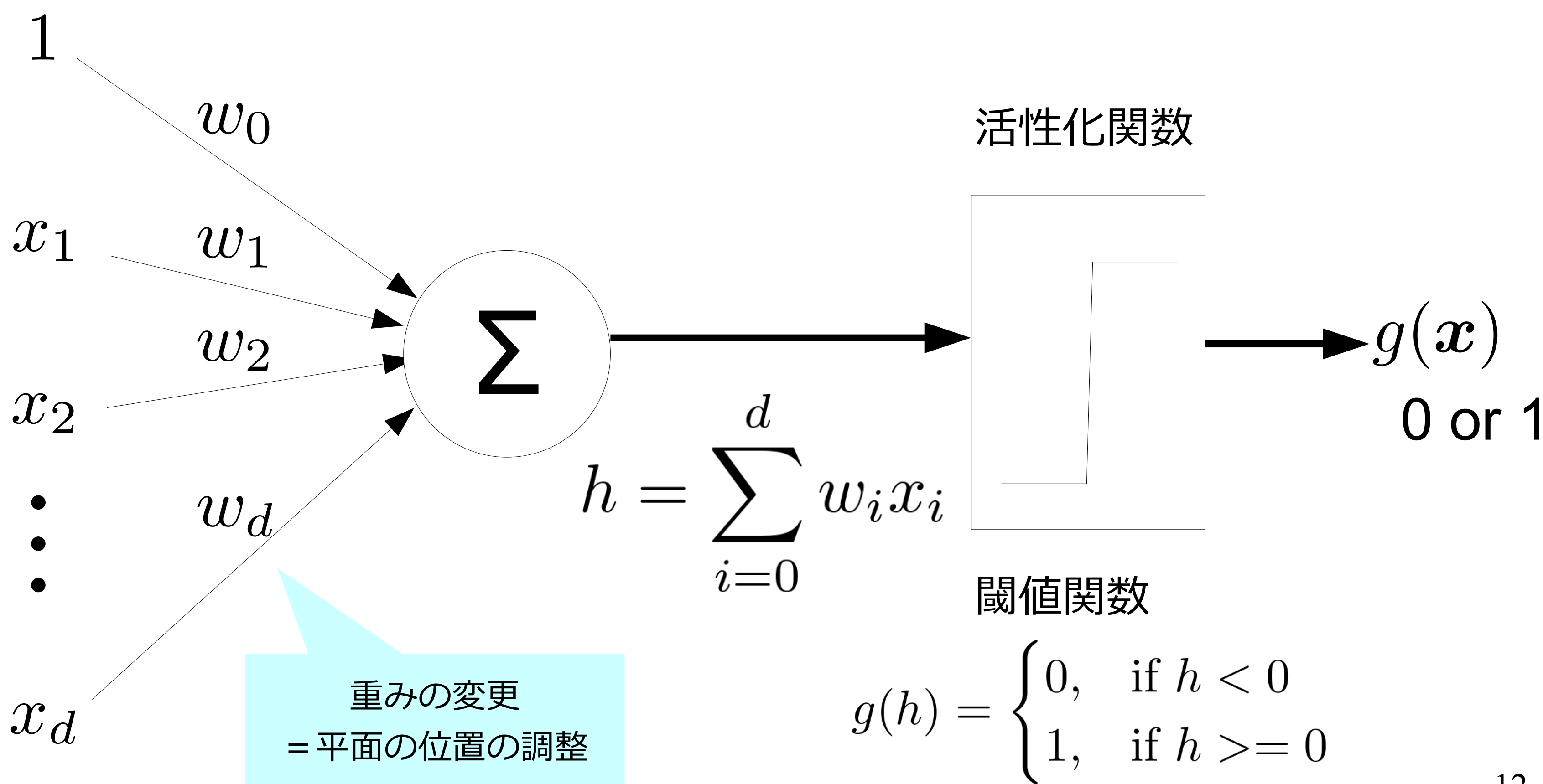
この平面を
求めている
ことになる

# 5.3.1 誤り訂正学習

- 単層パーセプトロンの定義

以後、 $\boldsymbol{w}$ は $w_0$ を含む

  - $\boldsymbol{w}^T\boldsymbol{x} = 0$ という特徴空間上の超平面を表現



# 5.3.1 誤り訂正学習

- パーセプトロンの学習規則

1. 重み $\boldsymbol{w}$ の初期値を適当に決める
2. 学習データからひとつ $\boldsymbol{x}$ を選び、 $g(\boldsymbol{x})$ を計算
3. 誤識別が起きたときのみ、 $\boldsymbol{w}$ を修正する

$$\boldsymbol{w}' = \boldsymbol{w} + \rho \boldsymbol{x}$$ （ positive のデータを negative と誤ったとき）

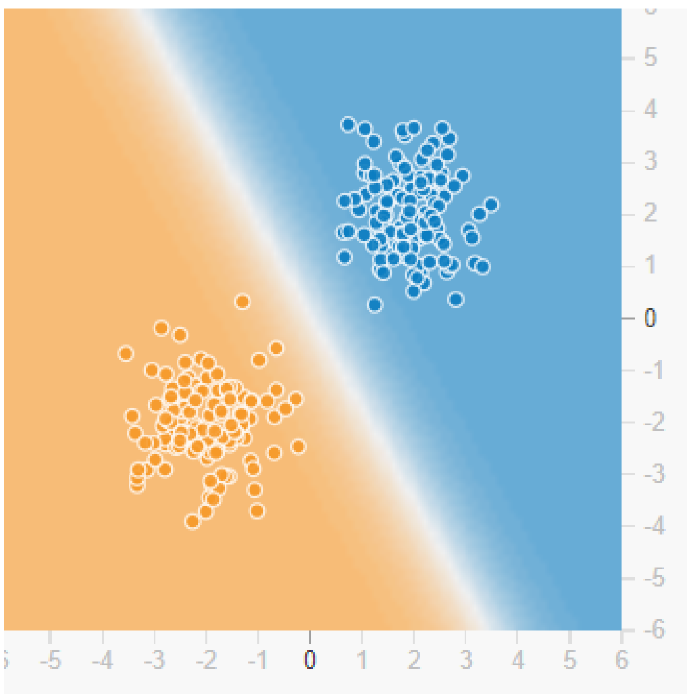
$$\boldsymbol{w}' = \boldsymbol{w} - \rho \boldsymbol{x}$$ （ negative のデータを positive と誤ったとき）

学習係数

4. 2,3 をすべての学習データについて繰り返す
5. すべて正しく識別できたら終了。そうでなければ 2 へ

# 5.3.1 誤り訂正学習

- パーセプトロンの学習規則の適用範囲
  - データが線形分離可能な場合はパーセプトロンの学習規則で学習可能




  - 線形分離不可能な場合は終了しない

# 5.3.3 識別モデルの考え方

- 事後確率を直接求める

正例で大きな値

負例で大きな値

あまり関係ない

$$\boldsymbol{x} = (x_1, \quad x_2, \quad x_3, \quad \ldots, \quad x_d)$$

$w_1$ 正の大きな値

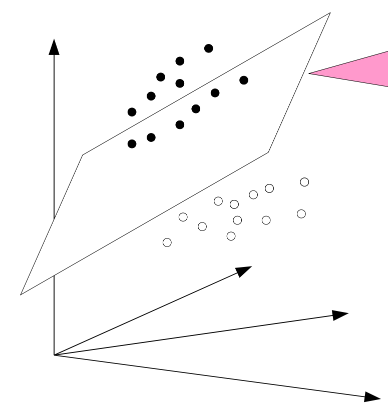
$w_2$ 負の大きな値

$w_3$ 小さな値

正例／負例の閾値

$$w_1 x_1 + w_2 x_2 + w_3 x_3 + \cdots + w_d x_d + w_0$$

この値が正なら正例、
負なら負例となるように
重み $\boldsymbol{w}$ を学習する



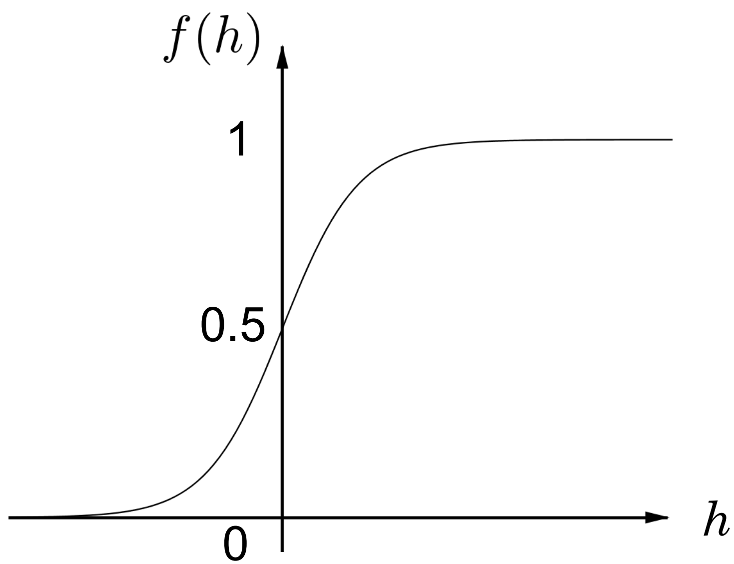
この平面を求めていることになる

**確率と対応づけるには？**

# 5.3.4 ロジスティック識別

- ロジスティック識別
  - 入力が正例である確率

$$P(Positive \mid \boldsymbol{x}) = \frac{1}{1 + \exp(-\boldsymbol{w}^T \boldsymbol{x})}$$




シグモイド関数 $f(h) = \frac{1}{1 + e^{-h}}$

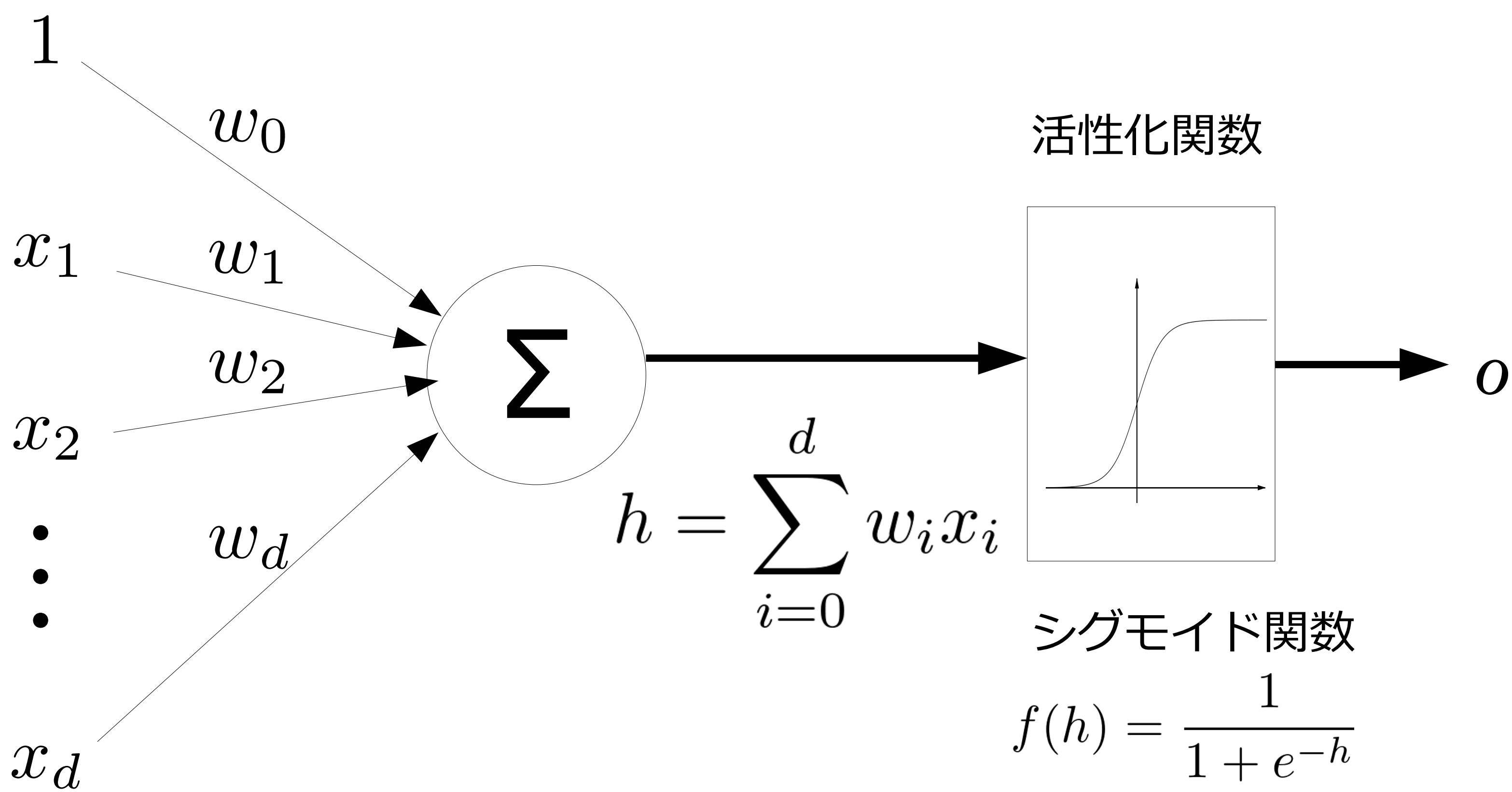
シグモイド関数の利点

- -∞ ～ +∞ の値域を持つものを、順序を変えずに 0 ～ 1 にマッピングできる
- 微分形が簡単な式になる

$f'(h) = f(h)(1\text{-}f(h))$

# 5.3.4 ロジスティック識別

- ロジスティック識別の計算ユニット
  - $\boldsymbol{w}^T\boldsymbol{x}$ の値を計算後、シグモイド関数で非線形変換



# 5.3.4 ロジスティック識別

- 最適化対象：モデルの対数尤度の反数（最小化）

$$
\begin{aligned}
E(\boldsymbol{w}) &= -\log P(D|\boldsymbol{w}) \\
&= -\log \prod_{\boldsymbol{x}_i \in D} o_i^{y_i}(1-o_i)^{(1-y_i)} \\
&= -\sum_{\boldsymbol{x}_i \in D} \{y_i \log o_i + (1-y_i)\log(1-o_i)\}
\end{aligned}
$$

$$
o = P(positive \mid \boldsymbol{x})
$$

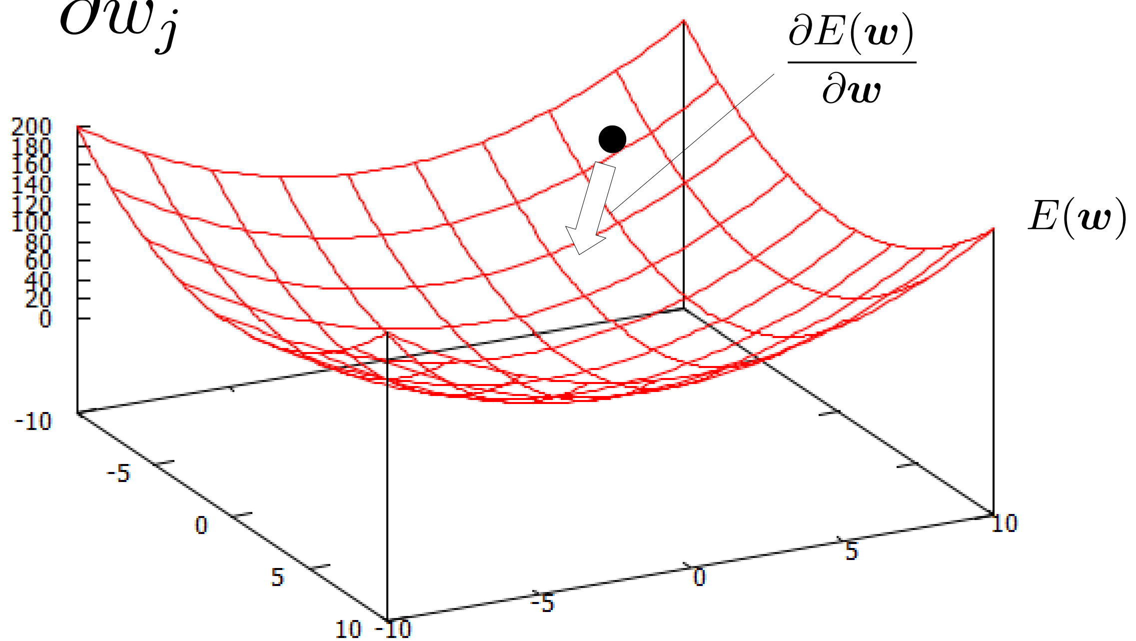
$$
y = 0 \ \ or \ \ 1
$$

正解ラベル

# 5.3.4 ロジスティック識別

- $E(\boldsymbol{w})$ を最急勾配法で最小化
    - 適当な初期値 $\boldsymbol{w}$ を選ぶ
    - $\boldsymbol{w}$ を $E(\boldsymbol{w})$ の勾配の逆方向に少しずつ修正

$$w_j \leftarrow w_j - \eta \frac{\partial E(\boldsymbol{w})}{\partial w_j}$$



# 5.3.4 ロジスティック識別

- 重み更新量の計算

$$\frac{\partial E(\boldsymbol{w})}{\partial w_j} = \sum_{\boldsymbol{x}_i \in D} \frac{\partial E(\boldsymbol{w})}{\partial o_i} \cdot \frac{\partial o_i}{\partial w_j}$$

$E(\boldsymbol{w})$ を $o_i$ で偏微分

$$= \sum_{\boldsymbol{x}_i \in D} \left(\frac{y_i}{o_i} - \frac{1-y_i}{1-o_i}\right) o_i(1-o_i) x_{ij}$$

シグモイド関数の微分

$w_j$ の係数

$$= \sum_{\boldsymbol{x}_i \in D} (y_i - o_i) x_{ij}$$

- 重みの更新式

$$w_j \leftarrow w_j - \eta \sum_{\boldsymbol{x}_i \in D} (y_i - o_i) x_{ij}$$

## 5.3.5 確率的最急勾配法

- 最急勾配法の問題点
  - 全データに対して誤差を計算するので、データ数が多い場合、重み更新に時間がかかる
- 確率的最急勾配法
  - 個々のデータの計算結果に基づき重みを更新
  - データが来る毎に学習するオンライン学習が可能
- ミニバッチ法
  - 数十～数百程度のデータで誤差を計算し、修正方向を決める方法

# 学習のシミュレーションサイト

https://playground.tensorflow.org/