RITSUMEIKAN

機械学習 第７回 サポートベクトルマシン

立命館大学 情報理工学部

福森 隆寛

Beyond Borders

講義スケジュール

（第1～4回、第14回）　（第5～13回、第15回）

◘ 担当教員：村上 陽平先生・福森 隆寛

1	機械学習とは、機械学習の分類
2	機械学習の基本的な手順
3	識別（１）
4	識別（２）
5	識別（３）
6	回帰
7	サポートベクトルマシン
8	ニューラルネットワーク

9	深層学習
10	アンサンブル学習
11	モデル推定
12	パターンマイニング
13	系列データの識別
14	強化学習
15	半教師あり学習

◘ 担当教員：叶 昕辰先生（第16回の講義を担当）

講義スケジュール

担当教員１：福森（第1回～第15回）

1	機械学習とは、機械学習の分類
2	機械学習の基本的な手順
3	識別（１）
4	識別（２）
5	識別（３）
6	回帰
7	**サポートベクトルマシン**
8	ニューラルネットワーク

9	深層学習
10	アンサンブル学習
11	モデル推定
12	パターンマイニング
13	系列データの識別
14	半教師あり学習
15	強化学習

担当教員２：叶昕辰先生（第16回の講義を担当）

今回の講義内容

- 取り扱う問題の定義
- サポートベクトルマシン
  - ハードマージン
  - ソフトマージン
  - カーネル関数
- 演習問題

取り扱う問題の定義：教師あり・識別問題

- カテゴリデータ、または数値データからなる特徴ベクトルを入力して、それをクラス分けする識別器を作る

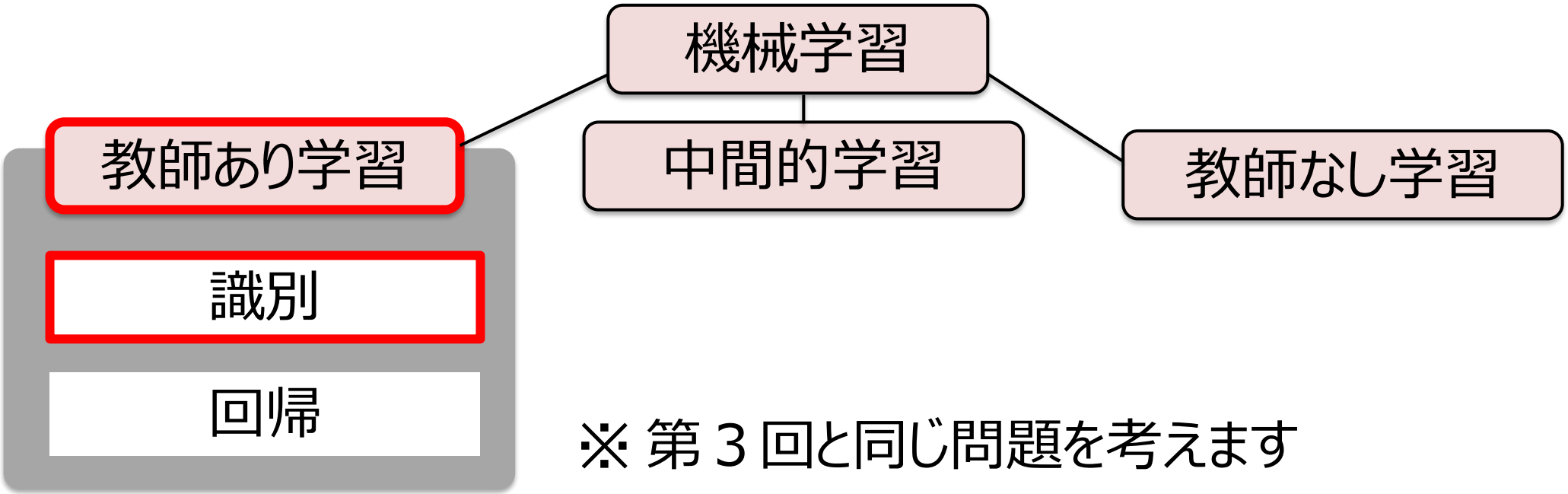
※ 教師あり学習の識別問題での学習データは、以下のペアで構成される

$\{\boldsymbol{x}_i, y_i\}, \quad i = 1,2,\ldots,N$

入力データの特徴ベクトル（カテゴリデータ/数値データ） ← $\boldsymbol{x}_i$

y_i → **カテゴリ形式の正解情報 →「クラス」と呼ぶ**

N → 学習データの総数

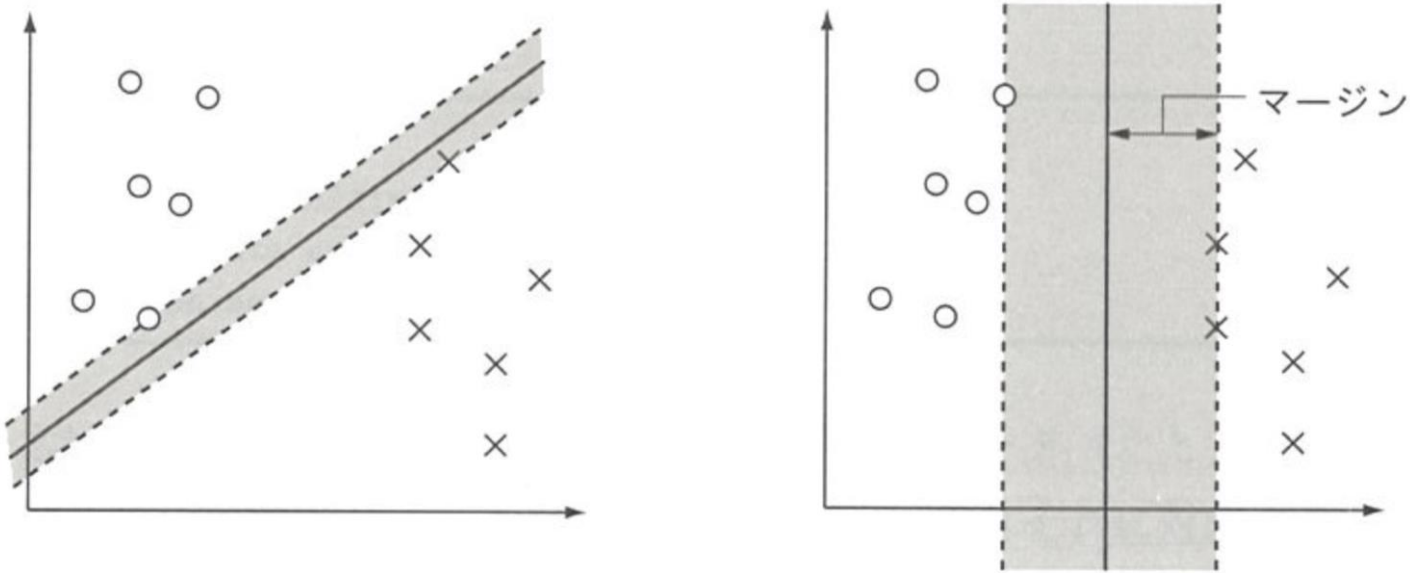


※ 第３回と同じ問題を考えます

サポートベクトルマシン

- 下図のような特徴空間上で超平面によって分離できる学習データを考える
  - 正解率100%で識別できる識別面が無数(むすう)に存在
  - **マージン**（識別面と最も近いデータとの距離）に基づいて識別面を決定（→汎化能力の高さを定量的(ていりょう)に表現）

マージンの小さい識別面



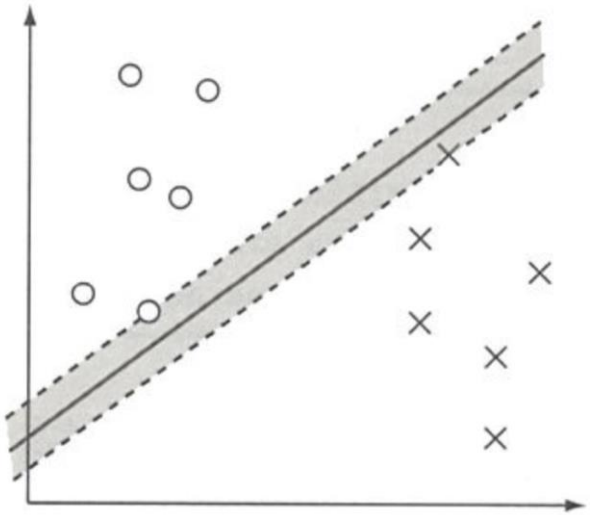
マージンの大きい識別面

学習データと識別面とのマージン

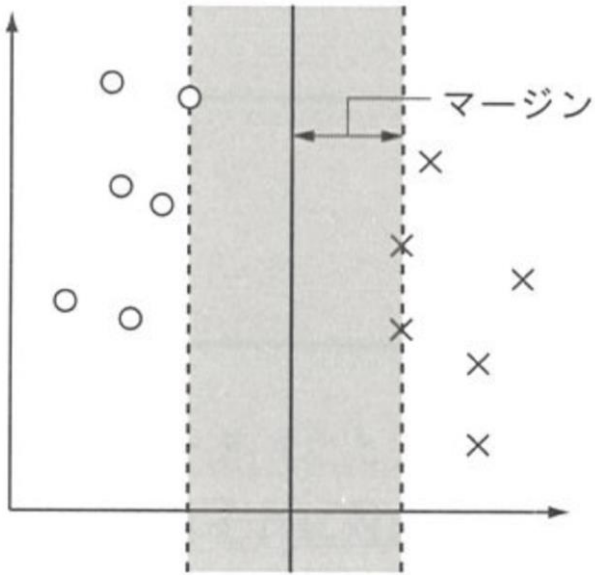
サポートベクトルマシン

ロ サポートベクトルマシン（support vector machine: SVM）

- 学習データからのマージンが最大となる識別面（一般には識別超平面）を求める手法
- マージンが広(ひろ)いと、学習データと識別面の間に未知データが入る余地(よち)がある



マージンの小さい識別面



マージンの大きい識別面

学習データから
少しずれた
未知データが
別クラスに
識別されることが
少なくなる

サポートベクトルマシン：定式化

- マージンが最大となる識別面を求める方法を考える

- 前提条件
  - 学習データが**線形分離可能**な状況を仮定
  - 数値特徴に対して正解情報の付いたデータを使用

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

  - 2値分類問題に限定
    - 正解情報y_i：正例 → $y_i = 1$、負例 → $y_i = -1$

サポートベクトルマシン：定式化

- 特徴空間上での識別面（超平面を仮定）
  - 識別面：$\boldsymbol{w} \cdot \boldsymbol{x} + w_0 = 0$　※ $\boldsymbol{w}$、$\boldsymbol{x}$： d次元ベクトル
  - 正例：$\boldsymbol{w} \cdot \boldsymbol{x} + w_0 > 0$
  - 負例：$\boldsymbol{w} \cdot \boldsymbol{x} + w_0 < 0$

- 点と直線の距離の公式（こうしき）より
  i番目のデータ$\boldsymbol{x}_i$と、この識別面との距離 $\mathrm{Dist}(\boldsymbol{x}_i)$

$$\mathrm{Dist}(\boldsymbol{x}_i) = \frac{|\boldsymbol{w} \cdot \boldsymbol{x}_i + w_0|}{\|\boldsymbol{w}\|}$$

サポートベクトルマシン：定式化

- 識別面 $\boldsymbol{w} \cdot \boldsymbol{x} + w_0 = 0$ の両辺（りょうへん）を定数倍しても表す識別面は変わらない

- 識別面に最も近いデータを識別面の式に代入したときその絶対値が1になるように係数$\boldsymbol{w}$とw_0を調整

$$\min_{i=1,\ldots,N} |\boldsymbol{w} \cdot \boldsymbol{x}_i + w_0| = 1$$

この状況での学習パターンと識別面との最小距離は

$$\min_{i=1,\ldots,N} \mathrm{Dist}(\boldsymbol{x}_i) = \min_{i=1,\ldots,N} \frac{|\boldsymbol{w} \cdot \boldsymbol{x}_i + w_0|}{\|\boldsymbol{w}\|} = \frac{1}{\|\boldsymbol{w}\|}$$

この最小距離は「マージン」を表している

サポートベクトルマシン：定式化

$$\min_{i=1,\ldots,N} \mathrm{Dist}(\boldsymbol{x}_i) = \min_{i=1,\ldots,N} \frac{|\boldsymbol{w}\cdot\boldsymbol{x}_i + w_0|}{\|\boldsymbol{w}\|} = \frac{1}{\|\boldsymbol{w}\|}$$

このマージンを最大にする識別面を求める問題は、
$\|\boldsymbol{w}\|$を最小化する問題となる

- これを$\|\boldsymbol{w}\|^2$の最小化問題に置き換える
  - 普通(ふつう)は$\boldsymbol{w} = \boldsymbol{0}$が最小解だが、これでは識別面にならない
  - 識別面として全学習データを識別できるという条件を追加

$$y_i \cdot (\boldsymbol{w}\cdot\boldsymbol{x}_i + w_0) \geq 1, \quad i = 1, \ldots N$$

- $y_i = 1$または$y_i = -1$としていたので、
  正例・負例両方の制約を一つの式で表現

サポートベクトルマシン：定式化

- マージンを最大にする識別面を求める問題の最終的な定式化

$$\min \frac{1}{2} \|\boldsymbol{w}\|^2$$

条件： $y_i \cdot (\boldsymbol{w} \cdot \boldsymbol{x}_i + w_0) \geq 1, \quad i = 1, \ldots, N$

  - 後(のち)ほど、微分を利用して極値を求めて最小解を導くので上式のように乗数(じょうすう) $\frac{1}{2}$ を付けておく

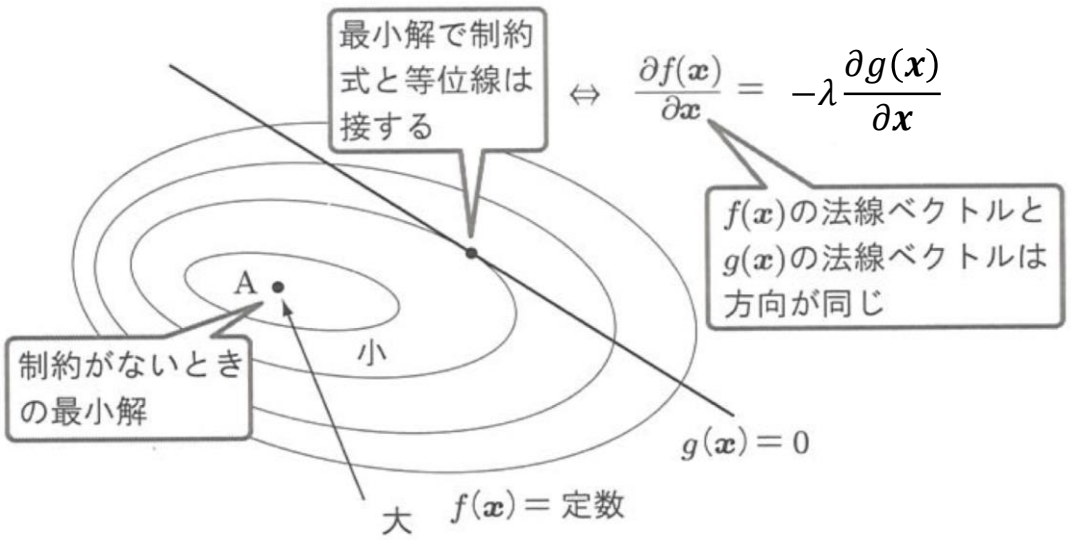
サポートベクトルマシン：識別面の計算

- 前スライドで定式化した問題を
  ラグランジュの未定(みてい)乗数法を用いて解決
- **ラグランジュ未定乗数法**
  - $g(\boldsymbol{x}) = 0$ の条件下で$f(\boldsymbol{x})$の最小値（または最大値）を計算する手法
  - ラグランジュ関数 $L(\boldsymbol{x}, \lambda) = f(\boldsymbol{x}) - \lambda g(\boldsymbol{x})$を導入して、この関数の極値を求める問題に置き換える
    - λ：ラグランジュ乗数
  - この$\boldsymbol{x}$に関する偏微分を0とすると、下式が得られる

$$\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}} = -\lambda \frac{\partial g(\boldsymbol{x})}{\partial \boldsymbol{x}}$$

サポートベクトルマシン：識別面の計算

- 制約がない状況での最小解は、下図の点A
- 直線 $g(\boldsymbol{x}) = 0$の上で、$f(\boldsymbol{x})$が最低となる点
  → $f(\boldsymbol{x})$の等位線と直線 $g(\boldsymbol{x}) = 0$が接する点



この等位線と直線が接すること

||

それらの法線ベクトルが一致すること

$$\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}} = -\lambda \frac{\partial g(\boldsymbol{x})}{\partial \boldsymbol{x}}$$

サポートベクトルマシン：識別面の計算

（つづき）

$$\min \frac{1}{2}\|\boldsymbol{w}\|^2 \quad \text{条件：} \ y_i \cdot (\boldsymbol{w} \cdot \boldsymbol{x}_i + w_0) \geq 1, \quad i = 1, \ldots, N$$

の制約付きの最小化問題は、ラグランジュ乗数α_iを導入して、関数Lの最小値を求める問題に置き換える

$$L(\boldsymbol{w}, w_0, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{N} \alpha_i \{y_i \cdot (\boldsymbol{w} \cdot \boldsymbol{x}_i + w_0) - 1\}$$

最小値では、Lの勾配が0になるので、以下の式が成立（せいりつ）

$$\frac{\partial L}{\partial w_0} = 0 \Rightarrow \sum_{i=1}^{N} \alpha_i y_i = 0 \qquad \frac{\partial L}{\partial \boldsymbol{w}} = 0 \Rightarrow \boldsymbol{w} = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i$$

サポートベクトルマシン：識別面の計算

（つづき）

$$L(\boldsymbol{w}, w_0, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{N} \alpha_i \{y_i \cdot (\boldsymbol{w} \cdot \boldsymbol{x}_i + w_0) - 1\}$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0, \qquad \boldsymbol{w} = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i$$

これらの式を整理すると

$$L(\boldsymbol{w}, w_0, \boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j = L_{\mathrm{dual}}(\boldsymbol{\alpha})$$

「$L(\boldsymbol{w}, w_0, \boldsymbol{\alpha})$を最小にする問題」は「上に凸（とつ）の関数である$L_{\mathrm{dual}}(\boldsymbol{\alpha})$を最大にする$\boldsymbol{\alpha}$を計算する問題」と等価（とうか）

サポートベクトルマシン：識別面の計算

- $L_{\mathrm{dual}}(\boldsymbol{\alpha})$が最大となる$\boldsymbol{\alpha}$を計算すると
  - $\alpha_i \neq 0$：サポートベクトルに対応するもののみ
  - $\alpha_i = 0$：サポートベクトル以外
  - サポートベクトル：識別面の計算に寄与する学習データ

- マージンを最大にする識別面の重み

$$\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i\,, \qquad w_0 = -\frac{1}{2}(\boldsymbol{w}\cdot\boldsymbol{x}_+ + \boldsymbol{w}\cdot\boldsymbol{x}_-)$$

  - α_i： $L_{\mathrm{dual}}(\boldsymbol{\alpha})$が最大となる$\boldsymbol{\alpha}$の$i$番目（ばんめ）の要素
  - $\boldsymbol{x}_+$、$\boldsymbol{x}_-$：正例、負例に属するサポートベクトル

サポートベクトルマシン：ソフトマージン

- 実際(じっさい)は、線形分離不可能な学習データが多い
  - ある程度の誤識別のデータを許容(きょよう)して、
    それらが識別面からあまり離れていない識別面を選択
  - ソフトマージンの条件を導入
- **ハードマージン**：全データを正しく識別できる条件

$$y_i \cdot (\boldsymbol{w} \cdot \boldsymbol{x}_i + w_0) \geq 1, \quad i = 1, \ldots, N$$

- **ソフトマージン**：ハードマージンを弱(よわ)める変数ξ_iを導入
  - $\xi_i (\geq 0)$：i番目のデータが制約を満たしていない程度で、
    小さい方が良い

$$y_i \cdot (\boldsymbol{w} \cdot \boldsymbol{x}_i + w_0) \geq 1 - \xi_i, \quad i = 1, \ldots, N$$

サポートベクトルマシン：ソフトマージン

- SVMのマージン最大化問題にξ_iを導入

$$\min\left(\frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}\xi_i\right)$$

$$y_i \cdot (\boldsymbol{w}\cdot\boldsymbol{x}_i + w_0) \geq 1 - \xi_i, \quad i = 1, \ldots, N, \quad \xi_i \geq 0$$

- 上式のCは制約を満たさないデータを、どの程度の重みで最適化に組み込むかを決定する定数
  - Cが大きい
    - ハードマージンの問題設定に近づき、複雑な識別面になる
  - Cが小さい
    - 誤りをほぼ無視する振る舞いで、比較的単純な識別面になる

サポートベクトルマシン：ソフトマージン

- 前スライドをラグランジュの未定乗数法で解くと、
  ハードマージンでの識別面と同じ式が導出され、

$$L_{\mathrm{dual}}(\boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j$$

  が最大となる$\boldsymbol{\alpha}$を計算する

ただし、ソフトマージンでは $0 \leq \alpha_i \leq C$ の制約がつく

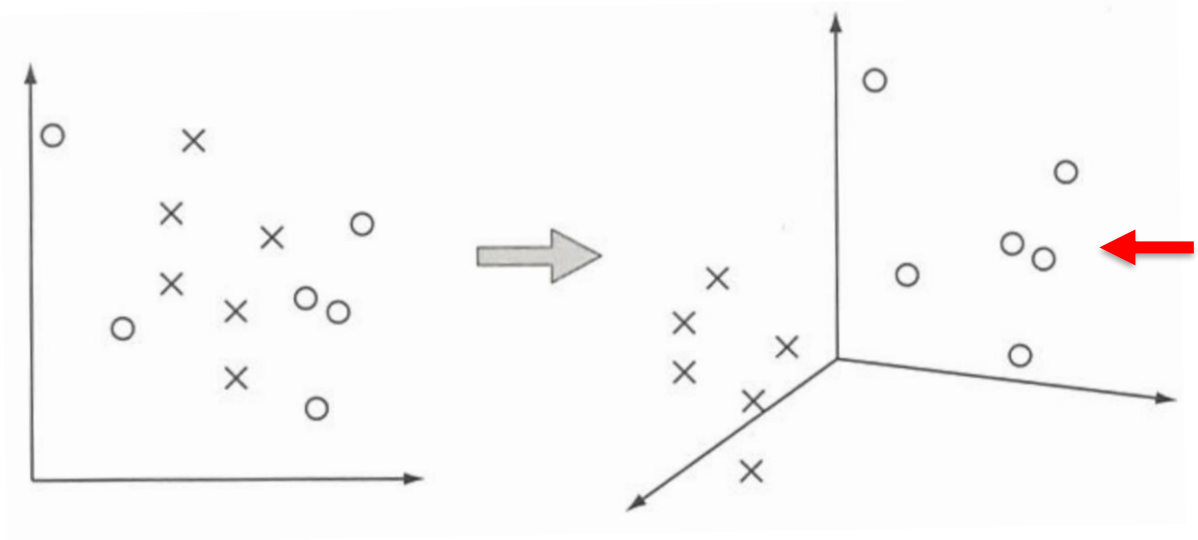
演習問題7-1（10分間）

- SVMは2クラスのデータに対する分類器である。
  ここで、3クラス以上のデータに対する多クラス
  識別問題にSVMを適用するには、
  どうすれば良いか考えよ

サポートベクトルマシン：カーネル関数

- 信頼できる識別面が得られそうにない場合を考える
- 特徴空間の次元数dが大きい場合
  - データが高次元空間上に疎らに分布することになるので、線形識別面が存在する可能性が高くなる

ソフトマージンを
使っても
良い識別面が
得られなさそう



高次元空間に
写像すると、
線形分離の
可能性が高まる

低次元の特徴ベクトルを高次元空間に写像し、
高次元空間上でSVMを使って識別超平面を求める方法を考える

サポートベクトルマシン：カーネル関数

- d次元の特徴空間に対して
  元の空間におけるデータ間の距離関係を保存しながら高次元に非線形に写像する
  - 高次元空間上での線形識別器の性能は
    元の空間での複雑な非線形識別器の性能に相当する
  - ただし、識別に無関係な特徴を持ち込(も こ)むと、データが
    無意味な方向に疎(こわ)らに分布し、元の分布の性質が
    壊されやすくなる
- 元の空間におけるデータ間の距離関係を
  保存するような非線形写像が見つかるか？

サポートベクトルマシン：カーネル関数

- 元の特徴空間上の2点$\boldsymbol{x}, \boldsymbol{x}'$の距離に基づいて定義される類似度(るいじど)関数$K(\boldsymbol{x}, \boldsymbol{x}')$を考える
  - 2点$\boldsymbol{x}, \boldsymbol{x}'$が近いほど、$K(\boldsymbol{x}, \boldsymbol{x}')$は大きな値になる

- $K(\boldsymbol{x}, \boldsymbol{x}')$が、半正定値性などの条件を満たすと

$$K(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{\phi}(\boldsymbol{x})^T \boldsymbol{\phi}(\boldsymbol{x}')$$

のように、2点$\boldsymbol{x}, \boldsymbol{x}'$から求まる高次元ベクトルの内積(ないせき)によって、類似度関数の値を計算できる
  - この関数を**カーネル関数**と呼ぶ

サポートベクトルマシン：カーネル関数

カーネル関数の例

多項式カーネル

$$K(\boldsymbol{x}, \boldsymbol{x}') = (\boldsymbol{x}^T \boldsymbol{x}' + 1)^p$$

※ p：自然数

ガウシアンカーネル関数

$$K(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\gamma \|\boldsymbol{x} - \boldsymbol{x}'\|^2)$$

- γ：カーネル関数の広がりを表すパラメータσ^2の逆数（ぎゃくすう） $\left(\frac{1}{\sigma^2}\right)$
- γが大きい場合は、複雑な識別面が形成される
  - カーネル関数が大きな値をとる範囲が狭（せま）くなるので、識別面の形成（けいせい）に関与（かんよ）するデータが近傍のものに限（かぎ）られる

サポートベクトルマシン：カーネル関数

- 写像後の空間での識別関数$g(\boldsymbol{\phi}(\boldsymbol{x}))$

$$g(\boldsymbol{\phi}(\boldsymbol{x})) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) + w_0$$

- ここでSVMを適用すると、$\boldsymbol{w}$は

$$\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{\phi}(\boldsymbol{x}_i)$$

となるので、識別関数$g(\boldsymbol{\phi}(\boldsymbol{x}))$は

$$g(\boldsymbol{\phi}(\boldsymbol{x})) = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{\phi}(\boldsymbol{x}_i)^T \boldsymbol{\phi}(\boldsymbol{x}) + w_0$$

$$= \sum_{i=1}^{N} \alpha_i y_i K(\boldsymbol{x}, \boldsymbol{x}_i) + w_0$$

サポートベクトルマシン：カーネル関数

- これまでと同様に、学習の問題も

$$\sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

  が最大となる$\boldsymbol{\alpha}$を計算すれば良い

- カーネル関数Kさえ定(さだ)まれば、
  識別面を得られるのが重要なポイント

- **カーネルトリック**
  - 複雑な非線形写像を求める操作を避(さ)ける方法
  - これがSVMが色々(いろいろ)な応用に使われてきた理由

サポートベクトルマシン：カーネル関数

- カーネル法
  - 入力データを高次元空間に写像しながら、計算上は明示的に高次元空間を考えずに識別面を構成できる

- 非線形写像（と）で線形分離可能な高次元空間にデータを飛ばして、マージン最大化基準（ひじょう きょうりょく）で信頼できる識別面を求めるSVMは非常に強力
  - 文書分類やバイオインフォマティックスなどの様々な分野（特に高次元識別問題）で利用

演習問題7-2（10分間）

- カーネル関数の1つである多項式（たこうしき）カーネルは、以下の式で与えられる

$$K(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{\phi}(\boldsymbol{x})^T \boldsymbol{\phi}(\boldsymbol{x}') = (\boldsymbol{x}^T \boldsymbol{x}' + 1)^p$$

- 2次元特徴ベクトルを $\boldsymbol{x} = (x_1, x_2)$、$p = 2$として $\boldsymbol{\phi}(\boldsymbol{x})$を求めよ

補足資料

サポートベクトルマシンのマージンを最大とする識別面の計算にて

$$L(\boldsymbol{w}, w_0, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{N} \alpha_i \{y_i \cdot (\boldsymbol{w} \cdot \boldsymbol{x}_i + w_0) - 1\}$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0, \qquad \boldsymbol{w} = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i$$

から

$$L(\boldsymbol{w}, w_0, \boldsymbol{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j = L_{\mathrm{dual}}(\boldsymbol{\alpha})$$

を求める

補足資料

$$L(\boldsymbol{w}, w_0, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{N} \alpha_i \{y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + w_0) - 1\}$$

$$= \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{N} \alpha_i y_i (\boldsymbol{w} \cdot \boldsymbol{x}_i + w_0) + \sum_{i=1}^{N} \alpha_i$$

$$= \frac{1}{2}\|\boldsymbol{w}\|^2 - \boldsymbol{w} \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i - w_0 \sum_{i=1}^{N} \alpha_i y_i + \sum_{i=1}^{N} \alpha_i$$

$$= \frac{1}{2}\|\boldsymbol{w}\|^2 - \boldsymbol{w} \cdot \boldsymbol{w} - w_0 \cdot 0 + \sum_{i=1}^{N} \alpha_i$$

$$= \frac{1}{2}\|\boldsymbol{w}\|^2 - \|\boldsymbol{w}\|^2 + \sum_{i=1}^{N} \alpha_i = -\frac{1}{2}\|\boldsymbol{w}\|^2 + \sum_{i=1}^{N} \alpha_i$$

$$= -\frac{1}{2}\left\|\sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i\right\|^2 + \sum_{i=1}^{N} \alpha_i$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

$$\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i$$

$$\boldsymbol{w} = \sum_{i=1}^{N} \alpha_i y_i \boldsymbol{x}_i$$

補足資料

（つづき）

$$L(\boldsymbol{w}, w_0, \boldsymbol{\alpha}) = -\frac{1}{2}\left\|\sum_{i=1}^{N}\alpha_i y_i \boldsymbol{x}_i\right\|^2 + \sum_{i=1}^{N}\alpha_i$$

$$= -\frac{1}{2}\left(\sum_{i=1}^{N}\alpha_i y_i \boldsymbol{x}_i\right)^T\left(\sum_{j=1}^{N}\alpha_j y_j \boldsymbol{x}_j\right) + \sum_{i=1}^{N}\alpha_i$$

$$= -\frac{1}{2}(\alpha_1 y_1 \boldsymbol{x}_1^T + \cdots + \alpha_N y_N \boldsymbol{x}_N^T)\left(\sum_{j=1}^{N}\alpha_j y_j \boldsymbol{x}_j\right) + \sum_{i=1}^{N}\alpha_i$$

$$= -\frac{1}{2}\left(\alpha_1 y_1 \boldsymbol{x}_1^T \sum_{j=1}^{N}\alpha_j y_j \boldsymbol{x}_j + \cdots + \alpha_N y_N \boldsymbol{x}_N^T \sum_{j=1}^{N}\alpha_j y_j \boldsymbol{x}_j\right) + \sum_{i=1}^{N}\alpha_i$$

$$= -\frac{1}{2}\left(\sum_{i=1}^{N}\alpha_i y_i \boldsymbol{x}_i^T \sum_{j=1}^{N}\alpha_j y_j \boldsymbol{x}_j\right) + \sum_{i=1}^{N}\alpha_i = -\frac{1}{2}\left(\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j\right) + \sum_{i=1}^{N}\alpha_i$$