今回からは 7 章「疎な解を持つカーネルマシン」に入っていく。先の章で学んだ非線形カーネル関数を用いた学習アルゴリズムとの違いについて以下の表 1 で整理しておきたい。表にもある通り「一部」の訓練データ

適用手法	非線形カーネル関数	疎な解を持つカーネルマシン
計算すべき学習データ	全ての訓練データ対	一部の訓練データ
計算負荷	大	小

表 1　非線形カーネルを用いた学習アルゴリズムと疎な解を持つカーネルマシンの違い

に対してカーネル関数を計算することで、入力に対する予測が可能となるアルゴリズムについて学んでいく。この手法を活用することは、先の章で学んだ手法について全データを計算しなければならないという制約をクリアできることに他ならない[*1]。

特に、本書ではカーネルマシンの代表的な手法である SVM；Support Vector Machine について学んでいくことにする。この手法はクラス分類、回帰などの分野で用いられることが多い。

1　最大マージン分類器

簡単な例として二値分類問題を考える。議論をシンプルにするため全てのデータ点は線形分離可能であると仮定する。このとき、我々がやりたいことはデータ点を完全に分離できるような境界線を定めることである。そのためには、マージンと呼ばれる量を最大化することが必要となることが知られている。このマージンに関する最適化問題を解くことで、SVM の解が疎な解を持つということを導くことが本セクションの目標となる。

本セクションのアウトラインは次の通りである。

・今回解きたい問題を設定する
・マージンという量を定義する
・マージンを構成するパラメタを動かして最大化する（＝最適化問題を考える）
・最適化問題を解くための計算テクニックを抑える
・簡単な場合について解の意味を解釈し SVM が疎な解を持つことを導く

上記アウトラインに沿って議論を進めていく。そして最後に次回の展望を簡単に述べることにする。

【問題設定】

二値分類問題を考えるために次の線形モデルを導入する。

$$y(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) + b \tag{1}$$

$\phi(\boldsymbol{x})$ は特徴空間変換関数（＝特徴量）、b はバイアスパラメタである。訓練データは N 組の入力 $\boldsymbol{x}_n$ と目標値 t_n とで構成される。ただし n は 1 から N の値をとる。訓練データは特徴空間で線形分離可能とする。入力 $\boldsymbol{x}_n$ は定義した線形モデル 1 によって $t_n y(\boldsymbol{x}_n) > 0$ という条件で分離される。

このときクラスを正確に分離できて、かつ汎化誤差が最も小さくなるような解 $\boldsymbol{w}$ や b を求めたい。そのためにマージンという量を定義することで議論を進めていく。

【マージンの定義】

[*1] 計算をラクしている以上、デメリットがあると当然予想されますが、教科書にはまだ明示されていないという認識です。この後出てくるのでしょうか。

ここでマージンという量を定義する（See 図 7.1)。マージンとは「分類境界と最も近くのデータ点までの（最短）距離」として定義される。このマージンを最大化するような分類境界を求める手法が SVM である。このとき分類境界の位置は一部の近くの点によってのみ定まることになる。そして、これらの一部の点のことをサポートベクトルと呼んでいる。

ここで、マージンを最大化することで分類境界の最適解が求められる理由を簡潔に説明する。ただし厳密な議論ではないことに注意されたい[*2]。もう少し考える時間をくれ。

【マージン最大化の定式化】

今回最適化を施す量を定式化しよう。今我々はマージンを最大化するようなパラメタを求めたかったので、まずはマージンをモデル 1 内の量を用いて表すことにしよう。点と超平面の公式[*3]から、点 $\boldsymbol{x}_n$ と分類境界 $y(\boldsymbol{x}_n)$ までの距離は次のようになる。

$$\begin{aligned}\frac{|\ y(\boldsymbol{x}_n)\ |}{\|\boldsymbol{w}\|} &= \frac{t_n y(\boldsymbol{x}_n)}{\|\boldsymbol{w}\|}\ (\because t_n y(\boldsymbol{x}_n) > 0) \\ &= \frac{t_n(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_n)+b)}{\|\boldsymbol{w}\|}\ (\because \text{式 } 1)\end{aligned} \tag{2}$$

式 2 は今対象としているマージンであり、したがい我々はこのマージンを最大化するようなパラメタ $\boldsymbol{w}$ と b を求めるという最適化問題を考えてやれば良い。これを式に表すと次のようになる。

$$\arg\max_{\boldsymbol{w},b}\{\frac{1}{\|\boldsymbol{w}\|}\min_n\left[t_n(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_n)+b)\right]\} \tag{3}$$

この最適化問題を直接解くのは複雑（らしい）なので、より簡単な形へ変形することを考える。まずパラメタを κ 倍してスケール変換すると式 3 内について次のような形を得る。

$$\frac{1}{\|\kappa\boldsymbol{w}\|}\left[t_n(\kappa\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_n)+\kappa b)\right] \tag{4}$$

最も境界に近い点 x_n について κ を適当に調整したうえで $\kappa\boldsymbol{w}\to\boldsymbol{w}$ と置き直すことで

$$t_n(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_n)+b)=1 \tag{5}$$

を成立させることができる。また、このスケーリングにおいては最も境界に近い点を含む全てのデータ点について次のような不等式が成り立つ。

$$t_n(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_n)+b)\geq 1 \tag{6}$$

ただし $n=1,\ldots,N$ である。このスケーリングによって今回の最適化問題は次のような形で定式化し直すことができる。

$$\arg\max_{\boldsymbol{w},b}\frac{1}{\|\kappa\boldsymbol{w}\|}\sim\arg\max_{\boldsymbol{w},b}\frac{1}{\|\boldsymbol{w}\|} \tag{7}$$

これはさらに $\|\boldsymbol{w}\|^2$ を最小化する問題へと帰着することができる。計算の都合上 $\frac{1}{2}$ をスケーリング係数としてつけてやると結局

$$\arg\min_{\boldsymbol{w},b}\frac{1}{2}\|\boldsymbol{w}\|^2 \tag{8}$$

[*2] なぜマージンを最大化すれば良いのかという議論についてはサブセクション 7.1.5 で厳密に取り扱うらしい。

[*3] 4 章レジュメ【210206 輪講.pdf】の頁 1 下部を参照。

を制約式 6 のもとで考えてやれば良いことになる。なお制約式の中には b が含まれているため、考えるべきパラメタから除外しないように注意してほしい。制約付き最小化問題を解くための定石としてラグランジュの未定乗数法があり、次のステップではこの手法を適用していくことになる。

【最適化計算実行のテクニック】

ラグランジュの未定乗数法を用いて先のステップで定式化した最適化問題を解く。今、未定乗数を $a_n \geq 0$ とするとラグランジュ関数 $L(\boldsymbol{w}, b, \boldsymbol{a})$ は次のようになる。

$$L(\boldsymbol{w}, b, \boldsymbol{a}) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{n=1}^{N} a_n\{t_n(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_n) + b) - 1\} \tag{9}$$

ここで $\boldsymbol{a} = (a_1, \ldots, a_N)$ である。式 9 をパラメタ $\boldsymbol{w}$ と b のそれぞれで偏微分しゼロとおくと次の関係式が導かれる。

$$\boldsymbol{w} = \sum_{n=1}^{N} a_n t_n \boldsymbol{\phi}(\boldsymbol{x}_n) \tag{10}$$

$$0 = \sum_{n=1}^{N} a_n t_n \tag{11}$$

これらをラグランジュ関数の式 9 に代入して $\boldsymbol{w}$ と b を消去する。この操作で新しく定義されるラグランジュ関数を $\tilde{L}(\boldsymbol{a})$ と置き直す。

$$\begin{aligned}
\tilde{L}(\boldsymbol{a}) &= \frac{1}{2}\sum_{n=1}^{N} a_n t_n \boldsymbol{\phi}^T(\boldsymbol{x}_n) \sum_{m=1}^{M} a_m t_m \boldsymbol{\phi}(\boldsymbol{x}_m) - \sum_{n=1}^{N} a_n t_n(\boldsymbol{w}^T\boldsymbol{\phi}(\boldsymbol{x}_n) + b) + \sum_{n=1}^{N} a_n \ (\because \text{式 10}) \\
&= \sum_{n=1}^{N} a_n + \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{M} a_n a_m t_n t_m \boldsymbol{\phi}^T(\boldsymbol{x}_n)\boldsymbol{\phi}(\boldsymbol{x}_m) - \sum_{n=1}^{N} \boldsymbol{w}^T a_n t_n \boldsymbol{\phi}(\boldsymbol{x}_n) \ (\because \text{式 10}) \\
&= \sum_{n=1}^{N} a_n + \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{M} a_n a_m t_n t_m \boldsymbol{\phi}^T(\boldsymbol{x}_n)\boldsymbol{\phi}(\boldsymbol{x}_m) - \sum_{n=1}^{N}\sum_{m=1}^{M} a_m t_m \boldsymbol{\phi}^T(\boldsymbol{x}_m) a_n t_n \boldsymbol{\phi}(\boldsymbol{x}_n) \ (\because \text{式 10}) \\
&= \sum_{n=1}^{N} a_n + \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{M} a_n a_m t_n t_m \boldsymbol{\phi}^T(\boldsymbol{x}_n)\boldsymbol{\phi}(\boldsymbol{x}_m) - \sum_{n=1}^{N}\sum_{m=1}^{M} a_n a_m t_n t_m \boldsymbol{\phi}^T(\boldsymbol{x}_n)\boldsymbol{\phi}(\boldsymbol{x}_m) \\
&= \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{M} a_n a_m t_n t_m \boldsymbol{\phi}^T(\boldsymbol{x}_n)\boldsymbol{\phi}(\boldsymbol{x}_m) \\
&= \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{M} a_n a_m t_n t_m k(\boldsymbol{x}_n, \boldsymbol{x}_m)
\end{aligned} \tag{12}$$

ここでカーネル関数は $k(\boldsymbol{x}_n, \boldsymbol{x}_m) = \boldsymbol{\phi}^T(\boldsymbol{x}_n)\boldsymbol{\phi}(\boldsymbol{x}_m)$ と定義される。式 12 を制約条件のもとで解くことで $\boldsymbol{a}$ が求められるが、具体的な解き方については 7 章で議論を行う。ここではひとまず解が得られたと仮定して話を進めていく。

ちなみに、このカーネル関数を用いることで入力 $\boldsymbol{x}$ に対する予測値 $y(\boldsymbol{x})$ は式 1 から次のように書ける。

$$\begin{aligned}
y(\boldsymbol{x}) &= \sum_{n=1}^{N} a_n t_n \boldsymbol{\phi}^T(\boldsymbol{x}_n)\boldsymbol{\phi}(\boldsymbol{x}) + b \ (\because \text{式 10}) \\
&= \sum_{n=1}^{N} a_n t_n k(\boldsymbol{x}_n, \boldsymbol{x}) + b
\end{aligned} \tag{13}$$

この式 13 より $a_n = 0$ となるようなデータ点は予測値 $y(\boldsymbol{x})$ に影響を与えないことがわかる。一方で $a_n \neq 0$ となるようなデータ点が予測値に影響を与えることになり、このようなデータ点をサポートベクトルと呼んでいる。したがって、いったん学習を終えてしまえばサポートベクトルとなっているデータ点以外の情報は捨ててしまっても良い。これが SVM の利点である。

なお、a_n に対する条件は KKT 条件を考えることで出てくることが知られている。それらの条件を書き下すと (7.14-16) のようになる。不等号の向きについて、もう少し考える時間をくれ。そして、これらの式から全てのデータ点について $a_n = 0$ または $t_n y(\boldsymbol{x}_n) = 1$ が成り立つことがわかる。

議論を解を定める問題に戻す。最適解 $\boldsymbol{a}$ が得られたとして、つぎにバイアスパラメタ b を求めていく。任意のサポートベクトル $\boldsymbol{x}_n$ は式 5、つまり $t_n y(\boldsymbol{x}_n) = 1$ を満たす。ここに式 13 を適用してやると次の表式が得られる。

$$t_n \left(\sum_{m \in S} a_m t_m k(\boldsymbol{x}_n, \boldsymbol{x}_m) + b \right) = 1 \tag{14}$$

ここで S はサポートベクトルの添字からなる集合である。全ての点について和を取っていないのは、先ほども議論したようにサポートベクトルでないデータ点は予測値に影響を与えないので予め弾いていることによる。式 14 を解いてやることでバイアス b を求めることができる。実際の計算では数値計算による誤差の影響を小さくするために次のような形に変形してから計算を実行することになる。変形にあたっては式 14 の両辺に t_n を掛けてからサポートベクトルのデータ点について総和を取ってやる操作を施しておく。

$$\begin{aligned}
\sum_{n \in S} t_n t_n \left(\sum_{m \in S} a_m t_m k(\boldsymbol{x}_n, \boldsymbol{x}_m) + b \right) &= \sum_{n \in S} t_n \\
\sum_{n \in S} \sum_{m \in S} a_m t_m k(\boldsymbol{x}_n, \boldsymbol{x}_m) + N_s b &= \sum_{n \in S} t_n \ (\because t_n^2 = 1) \\
N_s b &= \sum_{n \in S} t_n - \sum_{n \in S} \sum_{m \in S} a_m t_m k(\boldsymbol{x}_n, \boldsymbol{x}_m) \\
b &= \frac{1}{N_s} \sum_{n \in S} \left(t_n - \sum_{n \in S} \sum_{m \in S} a_m t_m k(\boldsymbol{x}_n, \boldsymbol{x}_m) \right)
\end{aligned} \tag{15}$$

ここで N_s はサポートベクトルの総数である。

【得られた解に関する解釈】

残りは読み合わせ。

・(7.19) を導入した理由がよくわからない。

・図 7.2 から SVM の解がサポートベクトルとなるデータ点にのみ依存することが確かめられる。この図において、緑色の丸印がついていない点を動かしても、分類境界（太線）を変えることはない。

以上、訓練データ点が特徴空間において線形分離な可能な場合について SVM によるクラス分類を考えた。次回はもう少し複雑な設定で SVM を適用する方法を学ぶ。具体的に言うとクラスの条件付き分布に重なりが存在し訓練データを完全に分離することが望ましくないケースについて考える。このとき、分類境界に対して誤分類を許すことで、より汎化性能を高める手法を適用することになる（ソフトマージンへの緩和)。