7.1 最大マージン分類器

平成 28 年 9 月 11 日

概 要

PRML の「7.1 最大マージン分類器」についての実装と考察

目 次

1 問題設定

線形分離可能な例にたいする, サポートベクトルマシンについて考える.
2次計画法の部分はcvxoptを用いた.
プロットはcoutourがいい感じ.

2 アルゴリズム

まずは, 線形分離可能な例について考える.

$$y(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) + b \quad (7.1)$$

訓練データは入力ベクトル $\mathbf{x}_1, \ldots, \mathbf{x}_N$ と目標値 $t_1, \ldots, t_N$ ($t_n \in \{-1, 1\}$ とし, $t_n = +1$ となるデータ点については $y(\mathbf{x}_n) > 0$, $t_n = -1$ となるデータ点については $y(\mathbf{x}_n), < 0$ となるとする. このとき

$$t_n y(\mathbf{x}_n) > 0 \quad n = 1, \ldots, N$$

が成り立つ.
SVMはマージン最大化を目標としている. まず決定面 $y(\mathbf{x}) = 0$ とデータ点 $\mathbf{x}$ との距離は $|y(\mathbf{x})|/\|\mathbf{w}\|$ で与えられるため

$$argmax_{\mathbf{w},b} \left\{ \frac{1}{\|\mathbf{w}\|} min_n [t_n(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n) + b)] \right\} \quad (7.3)$$

と問題を定式化できる.
さらに, $W \to \kappa \mathbf{w}, b \to \kappa b$ という変換に対して決定面は不変であるため, 決定面に最も近い点(サポートベクトル)について

$$t_n(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n) + b) = 1 \quad (7.4)$$

となるように変換すると, 全ての点で

$$t_n(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n) + b) \geq 1 \quad (7.5)$$

が成立する. 結局, この制約のもとで

$$argmin_{\mathbf{w},b} \frac{1}{2} \|\mathbf{w}\|^2 \quad (7.6)$$

を解けばよい.
この制約付き最小化問題には, ラグランジュの未定乗数法が使えて

$$L(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{n-1}^{N} a_n \{t_n(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n) + b) - 1\} \quad (7.7)$$

の $\mathbf{w}, b$ についての最小化, $\mathbf{a}$ についての最大化を考える. ただし, ラグランジュ乗数 a_n は $a_n \geq 0$ である.
$L(\mathbf{w}, b, \mathbf{a})$ を $\mathbf{w}, b$ で微分したものを0とすると

$$\mathbf{w} = \sum_{n=1}^{N} a_n t_n \boldsymbol{\phi}(\mathbf{x}_n) \quad (7.8) \quad 0 = \sum_{n=1}^{N} a_n t_n \quad (7.9)$$

が得られる. これを $L(\mathbf{w}, b, \mathbf{a})$ に代入することで双対表現が得られ

$$\tilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m) \quad (7.10)$$

の $\mathbf{a}$ についての最大化を行う. ただし, 以下の条件の下で

$$a_n \geq 0 \ (7.11), \ \sum_{n=1}^{N} a_n t_n = 0 \ (7.12)$$

これには, チャンキング (射影共役勾配法) や分解法, 逐次最小問題最適化法などが用いられる.
これで, パラメータ $\mathbf{a}$ は確定したので b を考えるが, 下の (7.13) より

$$t_n \left(\sum_{m \in S} a_m t_m k(\mathbf{x}_n, \mathbf{x}_m) + b \right) = 1 \quad (7.17)$$

ただし, S はサポートベクトルの添え字集合で $n \in S$ である. これだけでも b は求まるが, 計算安定性の観点から

$$b = \frac{1}{N_S} \sum_{n \in S} \left(t_n - \sum_{m \in S} a_m t_m k(\mathbf{x}_n, \mathbf{x}_m) \right) \quad (7.18)$$

ただし, N_S はサポートベクトルの総数.
以上で必要なパラメータが確定し, 新たなデータ点に対しては (7.1) より

$$y(\mathbf{x}) = \sum_{n=1}^{N} a_n t_n k(\mathbf{x}, \mathbf{x}_n) + b \quad (7.13)$$

の正負によって分類を行う.

3 コード

SVM のコード (SVM.py).

```
"""カーネル関数の定義"""
theta=0.05
def gauss(x,z):
  return np.exp(-theta*norm(x-z)**2)

"""aの決定(制約付き2次計画法)"""
a=np.zeros(N)
I=np.ones(N)
K=np.zeros((N,N))
for n in range(N):
  for m in range(N):
    K[n,m]+=t[n]*t[m]*gauss(x[n,:],x[m,:])
Q = cvxopt.matrix(K)
p = cvxopt.matrix(-np.ones(N))          # -1がN個の列ベクトル
G = cvxopt.matrix(np.diag([-1.0]*N))    # 対角成分が-1のNxN行列
h = cvxopt.matrix(np.zeros(N))          # 0がN個の列ベクトル
A = cvxopt.matrix(t, (1,N))             # N個の教師信号が要素の行ベクトル(1xN)
b = cvxopt.matrix(0.0)                  # 定数0.0
sol = cvxopt.solvers.qp(Q, p, G, h, A, b)# 二次計画法でラグランジュ乗数aを求める
a = np.array(sol['x']).reshape(N)       # 'x'がaに対応する

"""bの決定"""
b=0
Ns=0
S=[] #サポートベクトル
for n in range(N):
  if abs(a[n])>10**-5:
    S.append(n)
    Ns+=1
for n in S:
```

```
  sum_b=0
  for m in S:
      sum_b+=a[m]*t[m]*gauss(x[n,:],x[m,:])
  b+=t[n]-sum_b
b/=Ns

#求まったパラメータからモデル関数を作り
def model(z):
  res=0
  for n in range(N):
    res+=a[n]*t[n]*gauss(z,x[n,:])
  return res+b
```

今現在 cvxopt が ubuntu の python2 でしか使えない.

4 結果

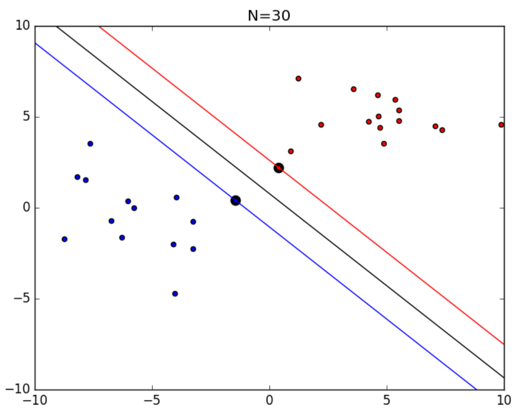
$N = 30, 50$ に対して, $y(\boldsymbol{x}) = -1, 0, 1$ となる面とサポートベクトルをプロットした.

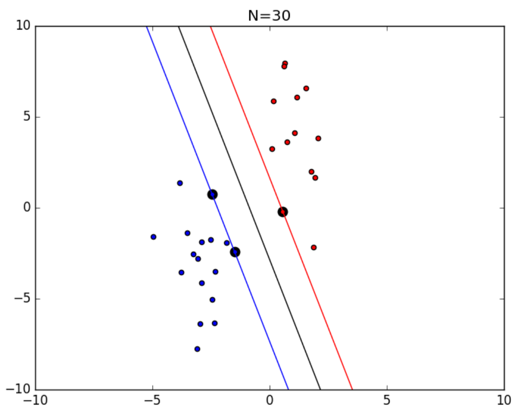
4.1 内積カーネル

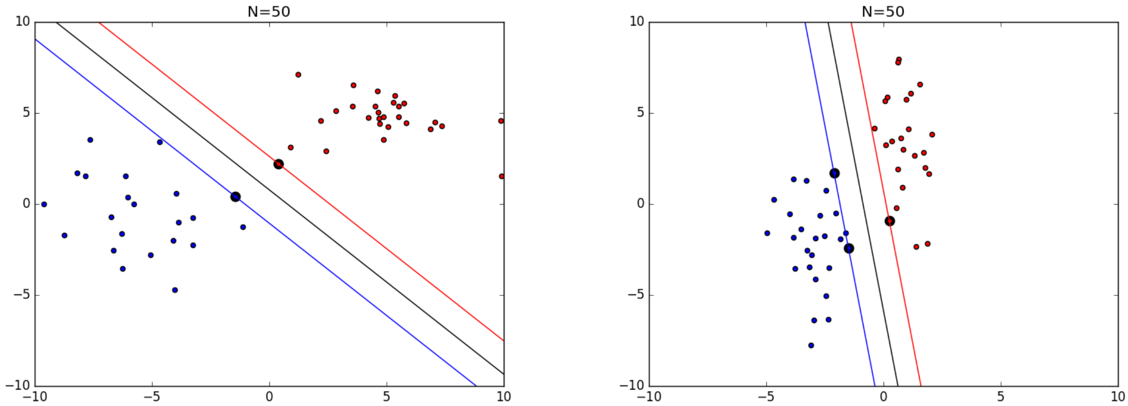
カーネル関数に

$$k(\boldsymbol{x}_n, \boldsymbol{x}_m) = \boldsymbol{x}_n^T \boldsymbol{x}_m$$

を用いた.







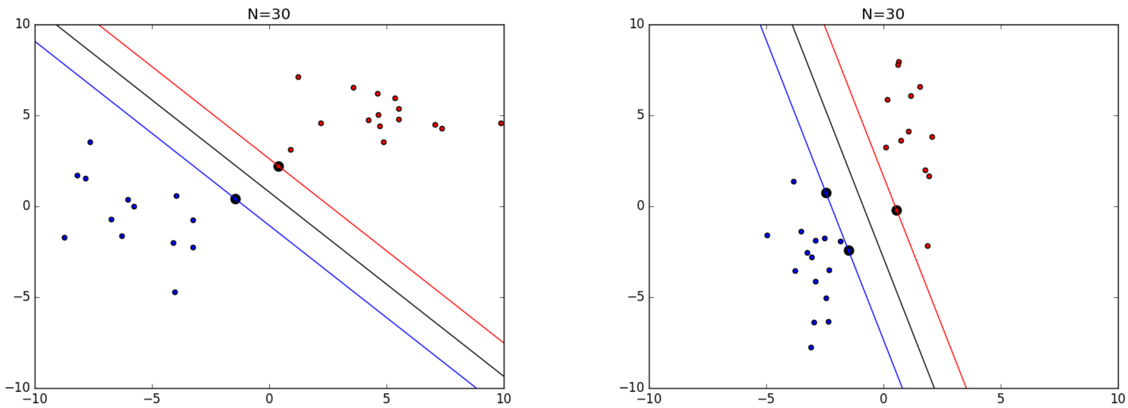
単純な線形分離となる．サポートベクトルも決定境界も正しいと思われる．

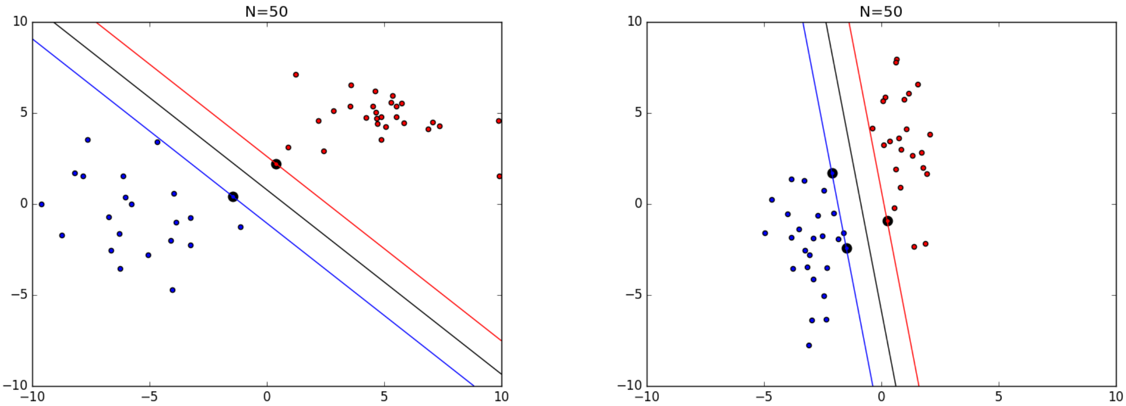
4.2　多項式カーネル

カーネル関数に

$$k(\boldsymbol{x}_n, \boldsymbol{x}_m) = (\boldsymbol{x}_n^T \boldsymbol{x}_m + 1)^\theta$$

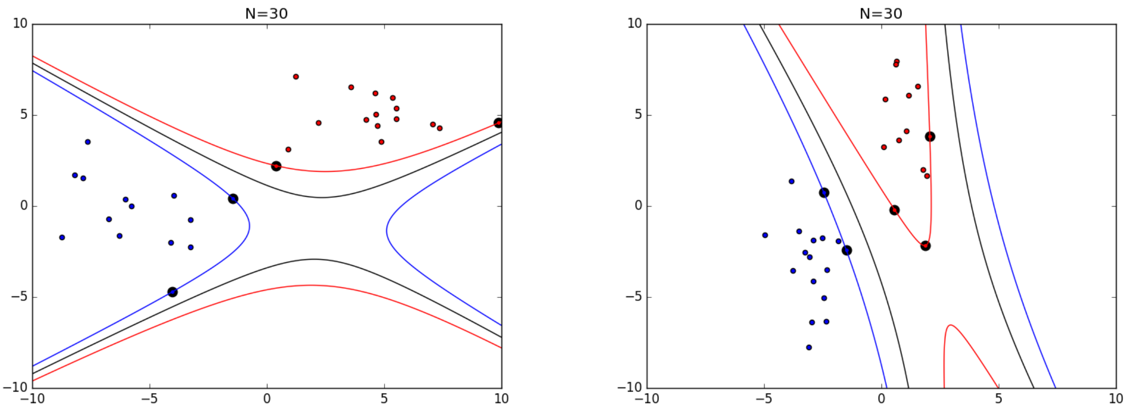
を用いた．$\theta = 1, 2, 3$ で試した．
まず $\theta = 1$ では

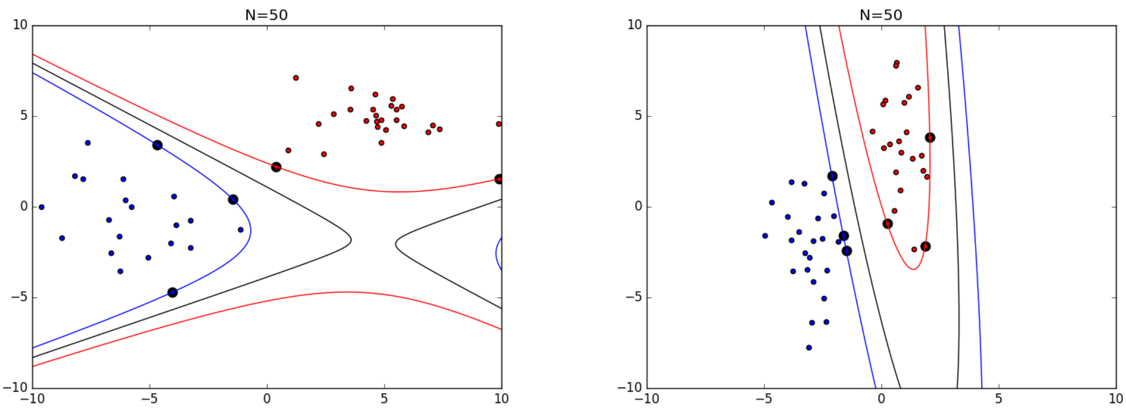




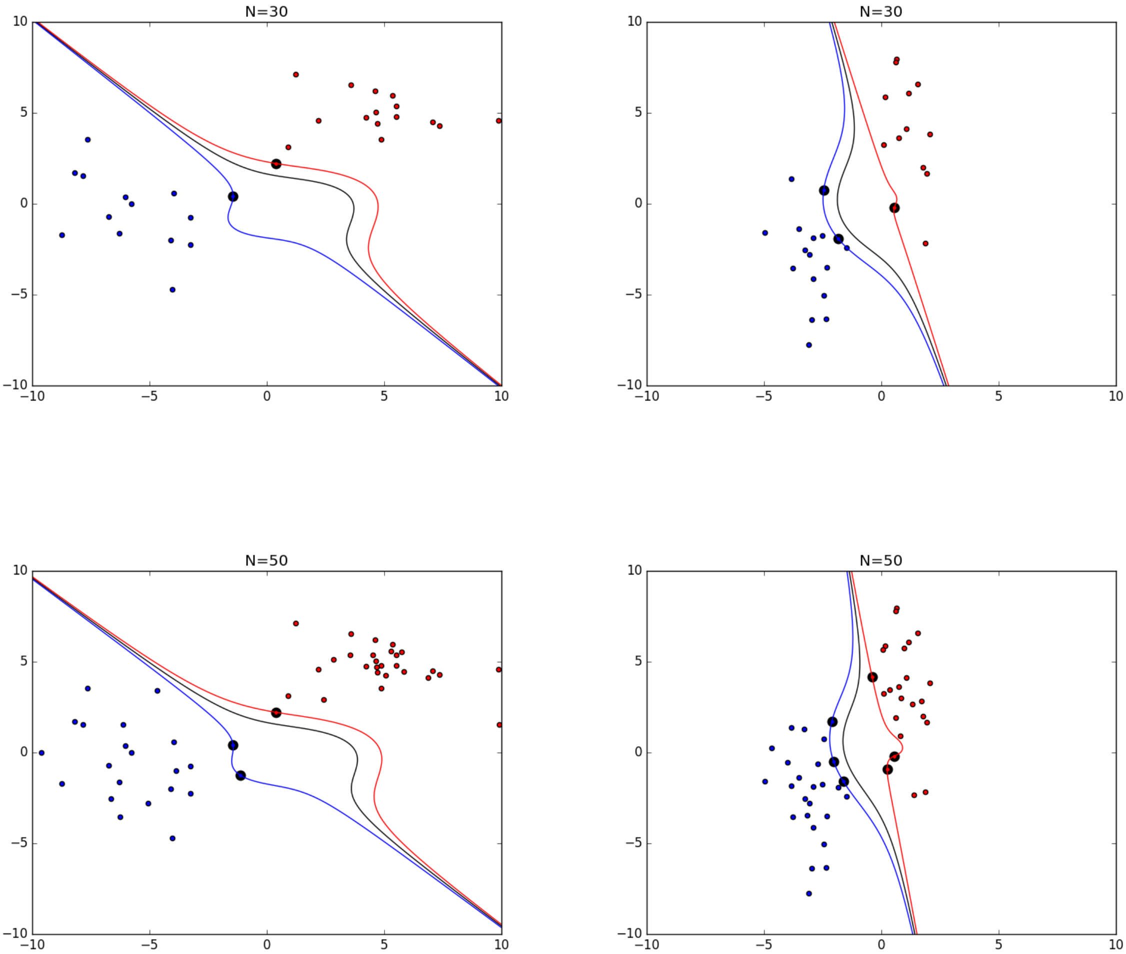
内積カーネルと等しくなった.

$\theta = 2$ では





$\theta = 3$ では



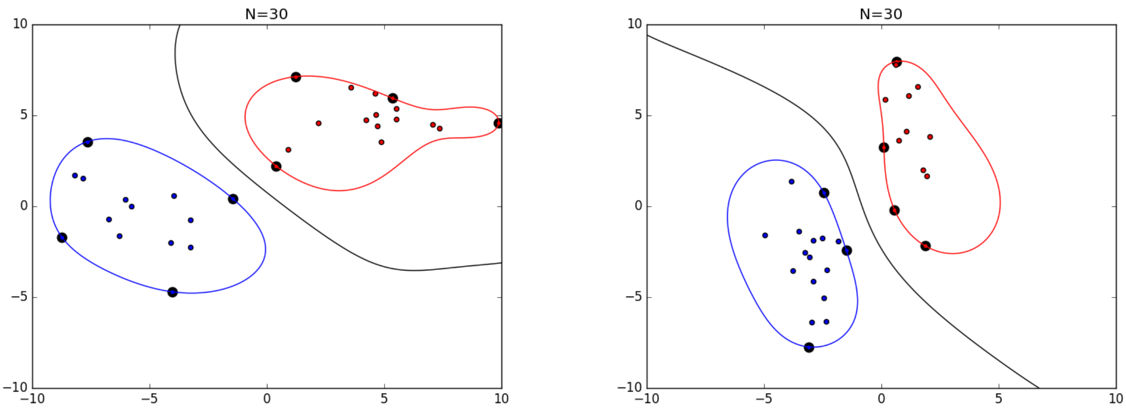
単純な線形分離となる．サポートベクトルも決定境界も正しいと思われる．

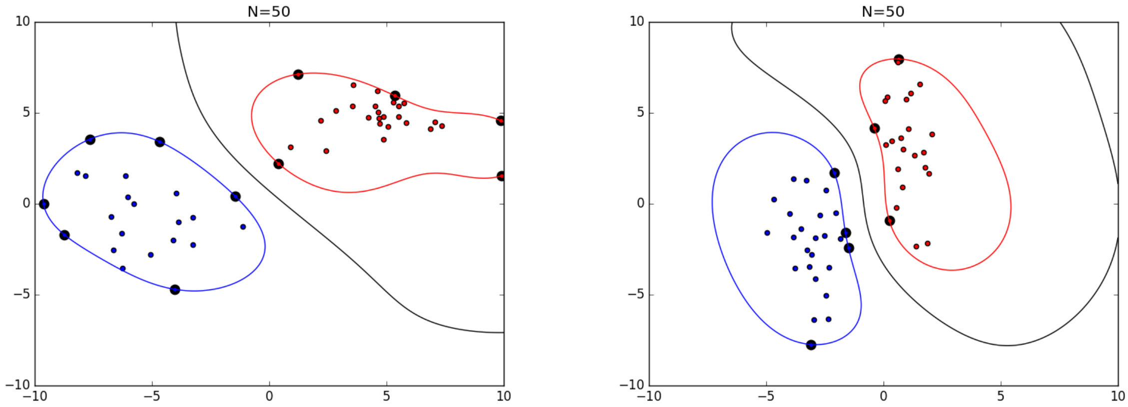
4.3　ガウスカーネル

カーネル関数に

$$k(\boldsymbol{x}_n, \boldsymbol{x}_m) = exp(-\theta\|\boldsymbol{x}_n - \boldsymbol{x}_m\|^2)$$

を用いた．





サポートベクトルも決定境界も正しいと思われる.

5　まとめ

ラグランジュの未定乗数法を解くために cvxopt が必要になる点が少々面倒くさい.
ガウスカーネルの結果はかなりいい感じに決定境界が定まった. ただ, 確率的でない点は少々問題である.