7.1.1 重なりのあるクラス分布

平成 28 年 9 月 11 日

概 要

PRMLの「7.1.1 重なりのあるクラス分布」についての実装と考察

目 次

1 問題設定

重なりのあるクラス分布に対する, サポートベクトルマシンについて考える.

2 アルゴリズム

一般には一部のデータの誤分類を許すように SVM を修正する.
まず, スラック変数 $\xi_n \geq 0$ $(n = 1, \ldots, N)$ を導入する. データが正しく分類されかつマージンの境界の上または外側に存在する場合は $\xi_n = 0$. それ以外は $\xi_n = |t_n - y(\mathbf{x}_n)|$ ととる. つまり,

$$\begin{aligned} \xi = 0 &\quad (\text{正しく分類されマージン境界の上または外側}) \\ 0 < \xi < 1 &\quad (\text{正しく分類され決定面とマージン境界との間}) \\ \xi = 1 &\quad (\text{決定面の上}) \\ \xi > 1 &\quad (\text{誤分類されている}) \end{aligned}$$

となる. これより, 制約条件は

$$t_n y(\mathbf{x}_n) \geq 1 - \xi_n, \ n = 1, \ldots, N \quad (7.20)$$

と修正する. この条件の下で, ソフトにペナルティを与えつつマージンを最大化することが目的で

$$C \sum_{n=1}^{N} \xi_n + \frac{1}{2}\|\mathbf{w}\|^2 \quad (7.21)$$

を最大化する. これにはラグランジュ未定乗数法を用いる.

$$L(\mathbf{w}, b, \boldsymbol{\xi}, \mathbf{a}, \boldsymbol{\mu}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{n=1}^{N}\xi_n - \sum_{n=1}^{N} a_n\{t_n y(\mathbf{x}_n) - 1 + \xi_n\} - \sum_{n=1}^{N}\mu_n\xi_n \quad (7.22)$$

ここで, KKT 条件は

$$a_n \geq 0 \ (7.23), \ t_n y(\mathbf{x}_n) - 1 + \xi_n \geq 0 \ (7.24), \ a_n(t_n y(\mathbf{x}_n) - 1 + \xi_n) = 0 \ (7.25)$$

$$\mu_n \geq 0 \ (7.26), \ \xi_n \geq 0 \ (7.27), \ \mu_n \xi_n = 0 \ (7.28)$$

で与えられる. また, $\mathbf{w}, b, \boldsymbol{\xi}$ で微分したものを 0 とすると,

$$\mathbf{w} = \sum_{n=1}^{N} a_n t_n \boldsymbol{\phi}(\mathbf{x}_n) \ (7.29), \ \sum_{n=1}^{N} a_n t_n = 0 \ (7.30), \ a_n = C - \mu_n \ (7.31)$$

この結果から, 双対系のラグランジュ関数が得られ

$$\tilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m) \quad (7.32)$$

これを最大化する. ただし, 以下の条件の下で

$$0 \leq a_n \leq C \ (7.33), \ \sum_{n=1}^{N} a_n t_n = 0 \ (7.34)$$

これによって, パラメータ $\mathbf{a}$ が求まり, 前と同様に

$$b = \frac{1}{N_M}\sum_{n \in M}\left(t_n - \sum_{m \in S} a_m t_m k(\mathbf{x}_n, \mathbf{x}_m)\right) \quad (7.37)$$

でパラメータ b が求まる. ただし, M は $0 \leq a_n \leq C$ となるデータ点の添え字集合である.

3 コード

重なりのあるクラス分布における SVM のコード (SVM1.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"""
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,:])
#共役勾配法
for C in [0.05,5,500]:
        Q = cvxopt.matrix(K)
        p = cvxopt.matrix(-np.ones(N))
        temp1 = np.diag([-1.0]*N)
        temp2 = np.identity(N)
        G = cvxopt.matrix(np.vstack((temp1, temp2)))
        temp1 = np.zeros(N)
        temp2 = np.ones(N) * C
        h = cvxopt.matrix(np.hstack((temp1, temp2)))
        A = cvxopt.matrix(t, (1,N))
        b = cvxopt.matrix(0.0)
        sol = cvxopt.solvers.qp(Q, p, G, h, A, b)
        a = np.array(sol['x']).reshape(N)
        print("\n",a)
        """bの決定"""
        b=0
        Ns,Nm=0,0
        S,M=[],[]
        for n in range(N):
          if 10**-5<a[n]:
            S.append(n)
            Ns+=1
            if a[n]<C:
              M.append(n)
              Nm+=1
        for n in M:
          sum_b=0
          for m in S:
              sum_b+=a[m]*t[m]*gauss(x[n,:],x[m,:])
          b+=t[n]-sum_b
        b/=Nm
        print(b)
        #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
```

4 結果

$N = 30, 50,\ C = 0.05, 5, 500$ 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.

図 1: $N = 30,\ C = 0.05, 5, 500$

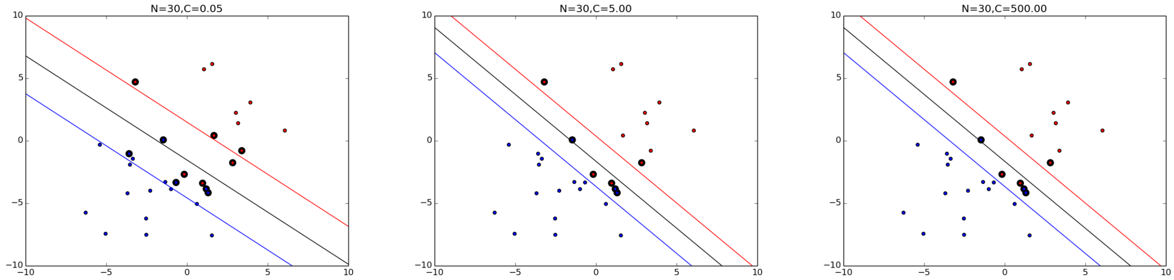
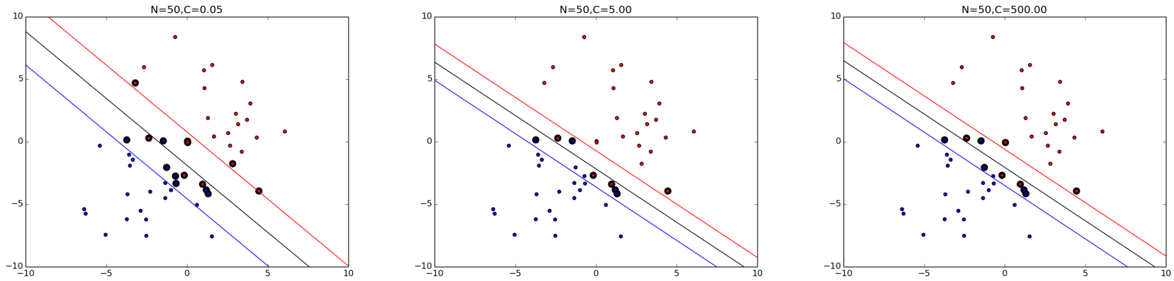


図 2: $N = 50,\ C = 0.05, 5, 500$



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.

$\theta = 1$ のとき

図 3: $N = 30,\ C = 0.05, 5, 500$

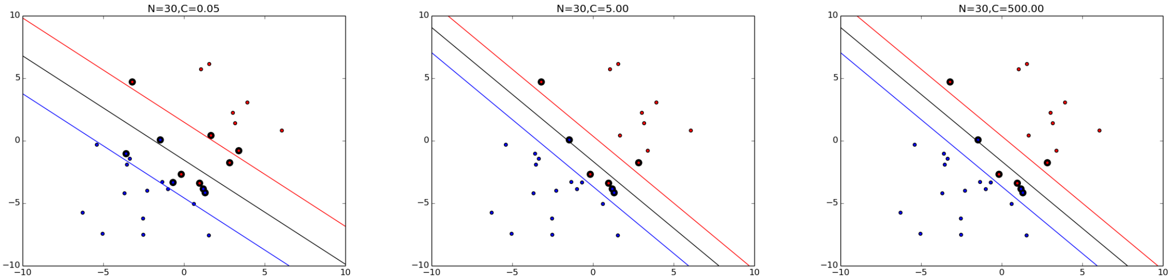
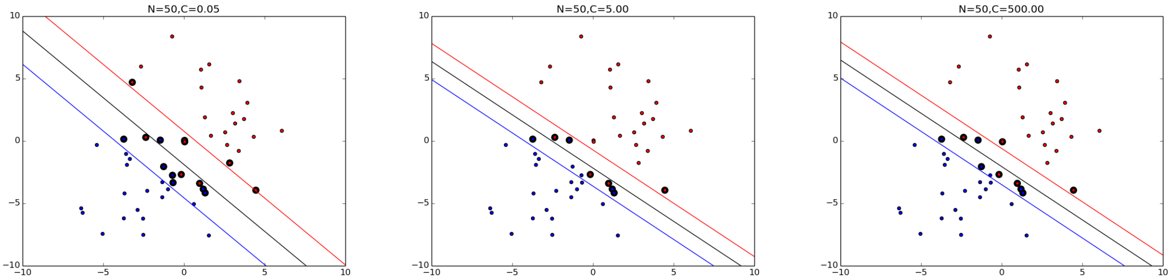


図 4: $N = 50,\ C = 0.05, 5, 500$



$\theta = 2$ のとき

図 5: $N = 30,\ C = 0.05, 5, 500$

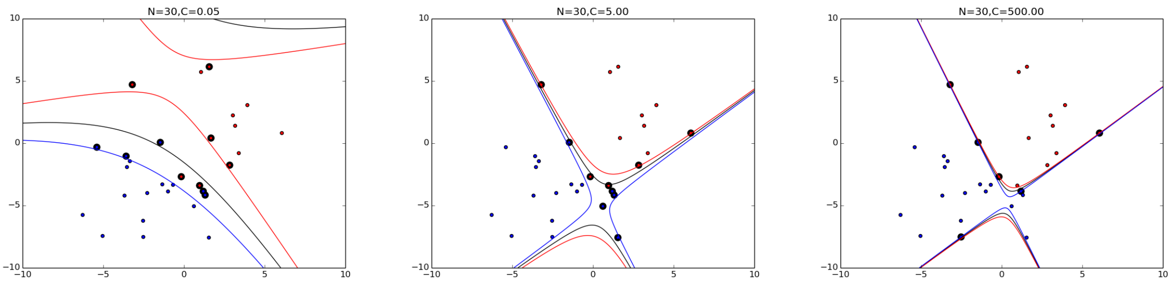
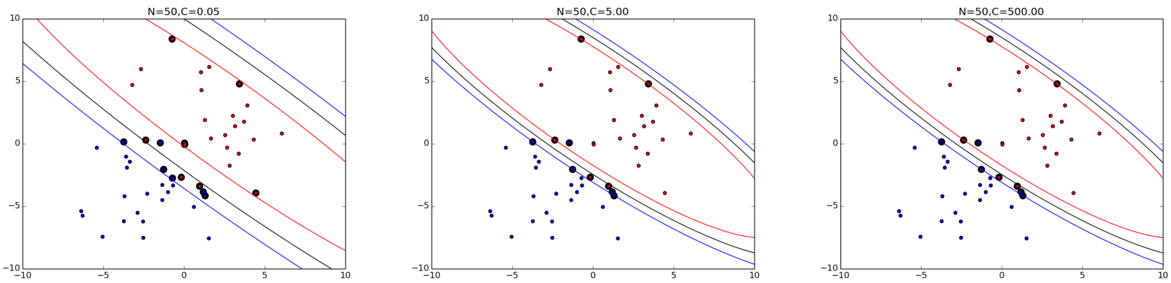


図 6: $N = 50,\ C = 0.05, 5, 500$



$\theta = 3$ のとき

図 7: $N = 30,\ C = 0.05, 5, 500$

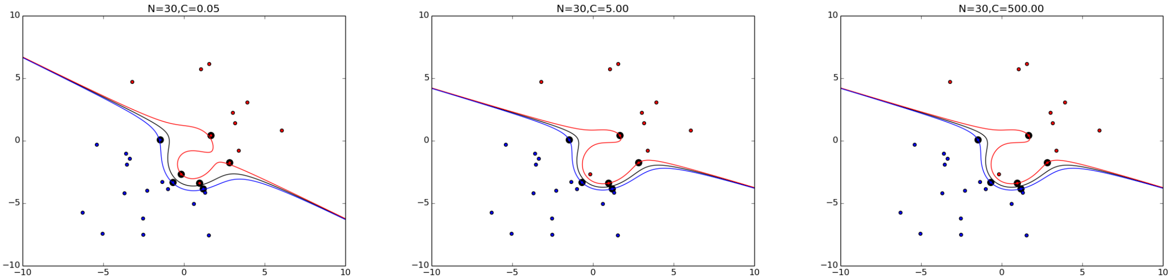
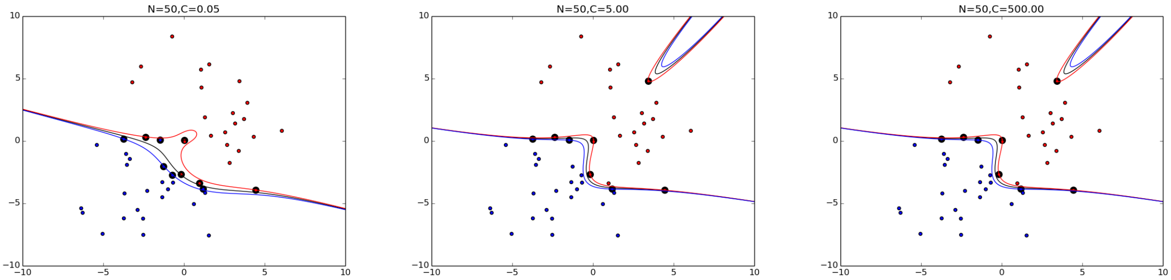


図 8: $N = 50,\ C = 0.05, 5, 500$



4.3 ガウスカーネル

カーネル関数に以下のガウスカーネルを用いた. $\theta = 0.05$ とした.

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

図 9: $N = 30,\ C = 0.05, 5, 500$

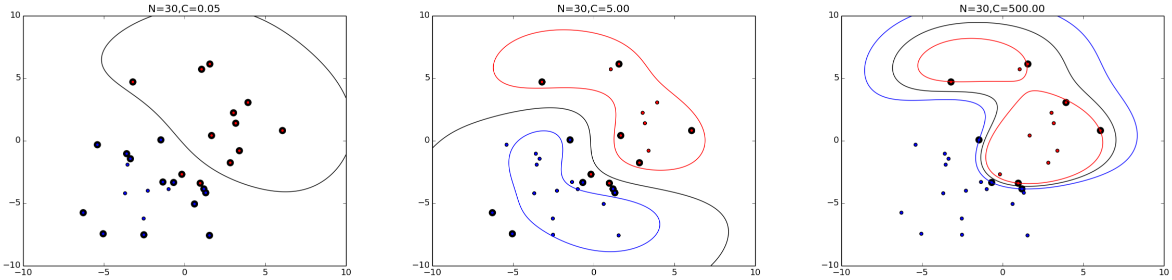
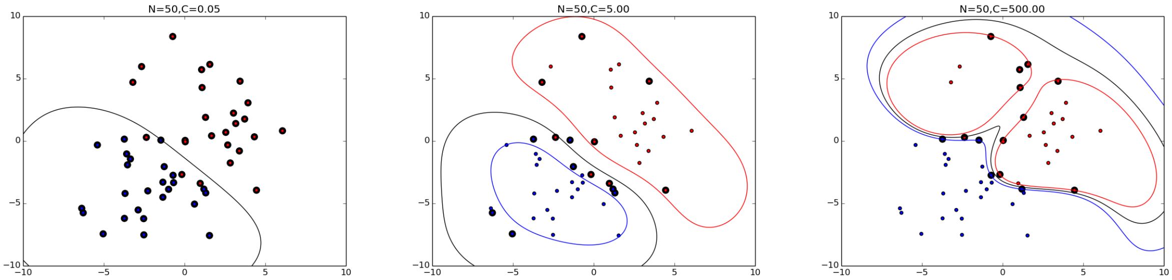


図 10: $N = 50,\ C = 0.05, 5, 500$



ガウスカーネルは面白かったので追加する．重なりを多くし $N = 100, 200$ に対し $\theta = 0.05, 1$ で試した．

$\theta = 0.05$ のとき

図 11: $N = 100,\ C = 0.05, 5, 500$

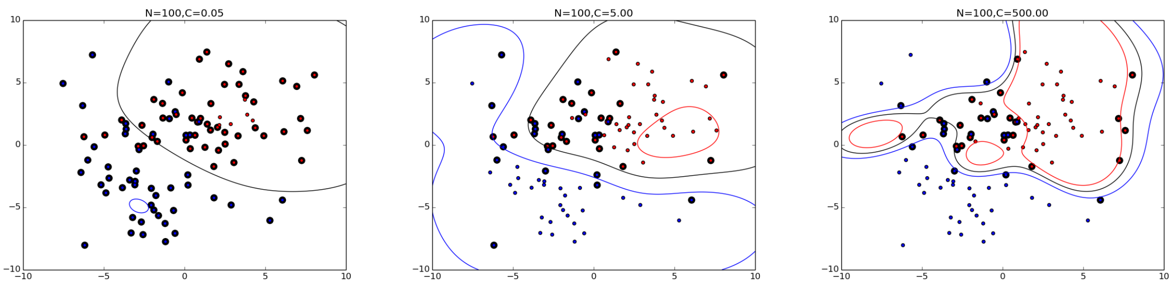
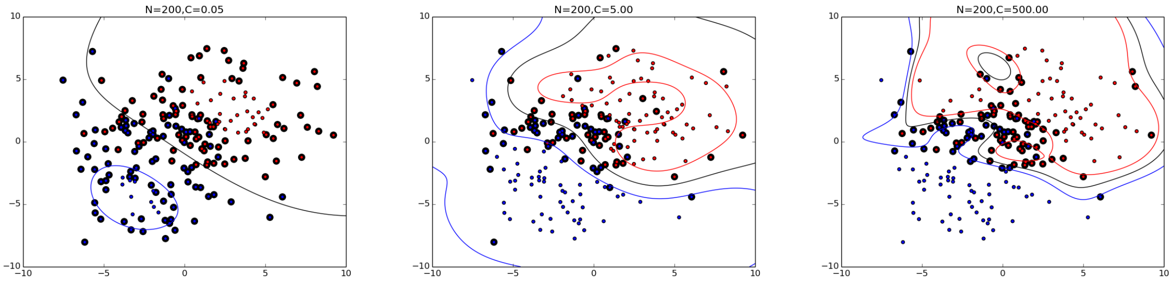


図 12: $N = 200,\ C = 0.05, 5, 500$



$\theta = 1$ のとき

図 13: $N = 100,\ C = 0.05, 5, 500$

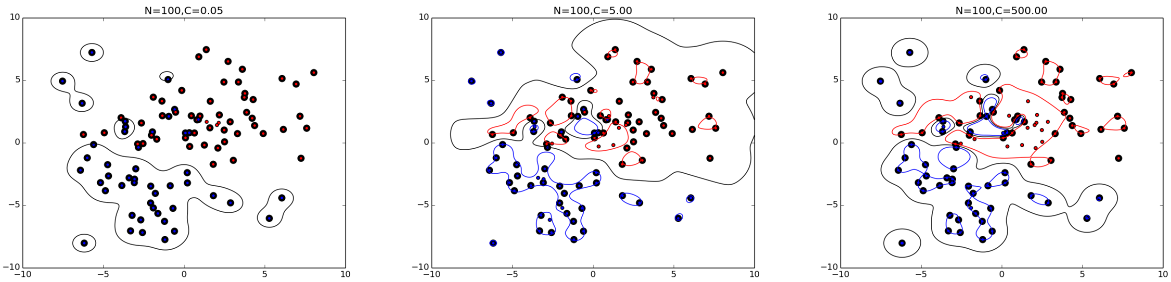
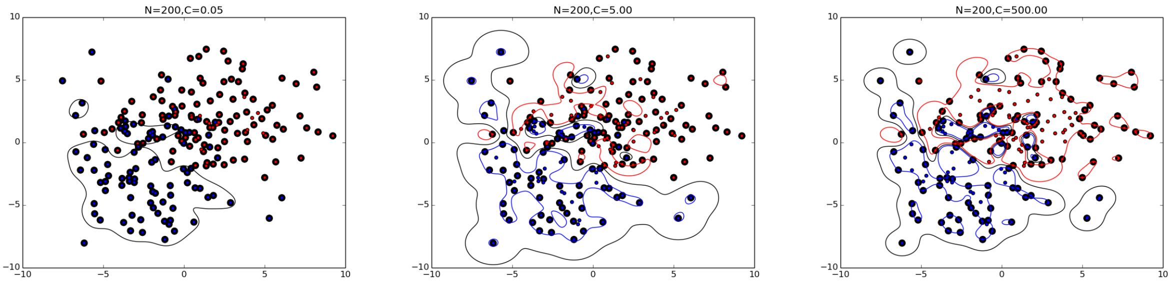


図 14: $N = 200,\ C = 0.05, 5, 500$



4.4 まとめ

サポートベクトルは全体のデータ数よりは少ないが, 線形分離可能なときよりも多い.
C が大きくなるにつれマージンが狭くなる. そして, 誤分類は減るが過学習によってしまう.
図 11〜図 14 より, パラメータの最適化が非常に重要であることが分かる.

5 ν-SVM

結果は重なりのあるクラス分布に対する SVM と同じであるが, 違った定式化を与えることができる.

$$\tilde{L}(\mathbf{a}) = -\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m) \quad (7.38)$$

を次の条件の下で最大化する.

$$0 \leq a_n \leq 1/N \ (7.39),\ \sum_{n=1}^{N} a_n t_n = 0 \ (7.40)\ ,\ \sum_{n=1}^{N} a_n \geq \nu \ (7.41)$$