Machine Learning Homework 5

**1**

由題目推得

$$h(x) = sign(wx + b) = sign(-(x - \frac{x_{M+1} + x_M}{2}))$$
$$w = -1 \ \ , \ \ b = \frac{x_{M+1} + x_M}{2}$$

答案為 [b].

**2**

由距離公式 $distance(x, b, w) = \frac{1}{\|w\|}|w^T x + b|$ 加上條件 $y_n(w^T x_n + b) \geq 1$ 推得最近的點到 decision boundary 的值為 $\|w\|^{-1}$

以$\min_{b,w} \frac{1}{2} w^T w \ \ s.t. \ \ y_n(w^T z_n + b) \geq 1$ 使用 lagrange multiplier 得到 $L(b, w, a) = \frac{1}{2} w^T w + \sum_{n=1}^{N} \alpha_n (1 - y_n(w^T z_n + b))$，經過上課一連串推導轉換為

$$\max_{all\ \alpha_n > 0, \sum y_n \alpha_n = 0, w = \sum \alpha_n y_n z_n} -\frac{1}{2} \|\sum_{n=1}^{N} \alpha_n y_n z_n\|^2 + \sum_{n=1}^{N} \alpha_n$$

在最佳解時其值與$\frac{1}{2}\|w\|^2$ 相同。固

$$\frac{1}{\|w\|} = (2 \sum_{n=1}^{N} \alpha_n - \|\sum_{n=1}^{N} \alpha_n y_n z_n\|^2)^{-1/2}$$

得到 (3)、(6) 符合，答案為 [c].

**3**

將四點以 $x, y$ 座標畫出來後可觀察出 decision boundary 與 y 軸平行，且該直線介於 $x = 0$ 與 $x = 1$ 之間且與兩者的距離比為 $4 : 1$，固得到 decision boundary 為 $x - \frac{4}{5} = 0$，再代入 $x_4$ 得到 $w_0(1 - \frac{4}{5}) \geq 1$ 推得 $w = (5, 0)$，進一步得到 $b = -4$。
答案為 [c].

**4**

Uneven margin SVM 的 lagrange multiplier 如下

$$\max_{all\ \alpha_n \geq 0} \frac{1}{2}w^T w + \sum_{n=1}^{N} [\![y_n = +1]\!] \alpha_n (1 - y_n(w^T x_n + b)) + \\ \sum_{n=1}^{N} [\![y_n = -1]\!] \alpha_n (\rho^- - y_n(w^T x_n + b))$$

將其對 $b$ 做偏微分得到

$$\frac{\partial L(w,b,\alpha)}{\partial b} = \sum_{n=1}^{N} [\![y_n = +1]\!] - \alpha_n y_n + \sum_{n=1}^{N} [\![y_n = -1]\!] - \alpha_n y_n = 0$$

$$\sum_{n=1}^{N} \alpha_n y_n = 0$$

因此上式中的 $b$ 可以消除，之後再對 $w_i$ 偏微分得到

$$\frac{\partial L(w,b,\alpha)}{\partial w_i} = w_i - \sum_{n=1}^{N} [\![y_n = +1]\!] - \alpha_n y_n x_{n,i} \\ + \sum_{n=1}^{N} [\![y_n = -1]\!] - \alpha_n y_n x_{n,i} = 0$$

$$w = \sum_{n=1}^{N} \alpha_n y_n x_n$$

因此可將原式化進一步轉化為

$$\max_{all\ \alpha_n \geq 0, \sum y_n \alpha_n = 0, w = \sum \alpha_n y_n x_n} -\frac{1}{2} \|\alpha_n y_n x_n\|^2 + \sum_{n=1}^{N} [\![y_n = +1]\!] \alpha_n \\ + \sum_{n=1}^{N} [\![y_n = -1]\!] \rho^- \alpha_n$$

最後得到

$$\min_{\alpha} \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m x_n^T x_m - \sum_{n=1}^{N} [\![y_n = +1]\!] \alpha_n - \sum_{n=1}^{N} [\![y_n = -1]\!]\ \rho^- \alpha_n$$

$$subject\ \ to\ \ \sum_{n=1}^{N} y_n \alpha_n = 0;\ \ \alpha_n \geq 0, for\ \ n = 1, 2, ..., N$$

答案為 [c].

**5**

我不會寫，但我猜答案是 [a].

**6**

題目要求的 $\Phi(x)$ 可看成是從 $d$ 個維度中組合出 $Q$ 個可重複的 $x_i$ 的重複組合問題，故其維度為 $H_Q^d = C_{d-1}^{Q+d-1} = C_Q^{Q+d-1}$。
答案為 [a].

**7**

由距離公式推得

$$\begin{aligned}\|\Phi(x) - \Phi(x')\|^2 &= \|\Phi(x)\Phi(x) - 2\Phi(x)\Phi(x') + \Phi(x')\Phi(x')\| \\ &= K_2(x,x) - 2K_2(x,x') + K_2(x',x')\end{aligned}$$

由於 $x, x'$ 為單位向量，故 $K_2(x,x) = K_2(x',x') = (1+1)^2 = 4$，而 $K_2(x,x')$ 的最小值為 $x^T x' = -1$ 時 $K_2(x,x') = (1-1)^2 = 0$，故 $\max\|\Phi(x) - \Phi(x')\|^2 = 4 - 0 + 4 = 8$。
答案為 [d].

**8**

由下列公式

$$w_{t+1} = w_t + y_{n(t)}\Phi(x_{n(t)})$$
$$w_t = \sum_{n=1}^{N} \alpha_t[n]\Phi(x_n)$$

可看出當 $(\Phi(x_{n(t)}), y_{n(t)})$ 分類錯誤時，對應到的 $\alpha_t[n(t)]\Phi(x_{n(t)})$ 會更新為 $\alpha_t[n(t)]\Phi(x_{n(t)}) + y_{n(t)}\Phi(x_{n(t)}) = (\alpha_t[n(t)] + y_{n(t)})\Phi(x_{n(t)})$，故 $\alpha_{t+1}[n(t)] = \alpha_t[n(t)] + y_{n(t)}$。
答案為 [c].

## 9

加上 lagrange multiplier 後得到下式

$$\max_{\alpha_n \geq 0, \beta_n \geq 0} \frac{1}{2} w^T w + \sum_{n=1}^{N} u_n \xi_n + \sum_{n=1}^{N} \alpha_n (1 - \xi_n - y_n (w^T \Phi(x_n) + b)) + \sum_{n=1}^{N} \beta_n (-\xi_n)$$

將其對 $\xi_n$ 做偏微分得到

$$\frac{\partial L(w, b, \xi, \alpha, \beta))}{\partial \xi_n} = \mu_n - \alpha_n - \beta_n = 0$$

在不失其最佳性的情況下得到 $\beta_n = u_n - \alpha_n, 0 \leq \alpha_n \leq u_n$，故可將上式簡化為

$$\begin{aligned}\max_{0 \leq \alpha_n \leq u_n, \beta_n = u_n - \alpha_n} & \frac{1}{2} w^T w + \sum_{n=1}^{N} \alpha_n (1 - y_n (w^T \Phi(x_n) + b)) + \sum_{n=1}^{N} (u_n - \alpha_n - \beta_n) \xi_n \\ &= \frac{1}{2} w^T w + \sum_{n=1}^{N} \alpha_n (1 - y_n (w^T \Phi(x_n) + b))\end{aligned}$$

之後如同一般的SVM一樣，對 $b$ 做偏微後得到 $\sum_{n=1}^{N} \alpha_n y_n = 0$；對 $w_i$ 做偏微後得到 $w = \sum_{n=1}^{N} \alpha_n y_n \Phi(x_n)$，最後推得

$$\begin{aligned}&\min_{\alpha} \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m \Phi(x_n)^T \Phi(x_m) - \sum_{n=1}^{N} \alpha_n \\ subject\ \ to\ \ &\sum_{n=1}^{N} y_n \alpha_n = 0; \ \ 0 \leq \alpha_n \leq u_n, for\ \ n = 1, 2, ..., N\end{aligned}$$

答案為 [a].

## 10

透過積分計算得到

$$\begin{aligned}\int_0^1 (E_{smooth} - E_{hinge})^2 d\rho = \int_0^1 (\frac{1}{2}(1 - \rho^2)^2 d\rho &= \int_0^1 \frac{1}{4} (\rho^4 - 2\rho^2 + 1) d\rho \\ &= \frac{1}{4} (\frac{1}{5}\rho^5 - \frac{2}{3}\rho^3 + \rho) \Big|_0^1 = \frac{2}{15}\end{aligned}$$

答案為 [e].

### 11, 12, 13, 14, 15, 16

```
from libsvm.svmutil import *
import numpy as np
import math
import tqdm

#=========================================basic function=========================================

def E_01(y, predict):
    y = np.reshape(y, -1)
    predict = np.reshape(predict, -1)
    #print('E_01:', np.sum(y != predict), '/', y.shape[0])
    return np.sum(y != predict) / y.shape[0]

#=========================================Transform==============================================

def OneLabelTransform(data, target):
    temp = data.copy()
    for i in range(len(temp)):
        if temp[i] == target:
            temp[i] = 1
        else:
            temp[i] = -1

    return np.array(temp).astype(np.int)

#=========================================liblinear==============================================

def MySVM(x_train, y_train, x_test, y_test, par, ReturnNode=False):
    prob = svm_problem(y_train, x_train)
    param = svm_parameter(par)
    model = svm_train(prob, param)
    p_label, p_acc, p_val = svm_predict(y_test, x_test, model, '-q')

    if ReturnNode:
        return E_01(y_test, p_label), model.nSV[0] + model.nSV[1]
    else:
        return E_01(y_test, p_label)

#================================================================================================

def main():
    y_train, x_train = svm_read_problem('satimage.scale')
    y_test, x_test = svm_read_problem('satimage.scale.t')

    p11, p12, p13, p14, p15, p16 = True, True, True, True, True, True

    if p11:
        y_train_5 = OneLabelTransform(y_train, 5)

        prob = svm_problem(y_train_5, x_train)
        param = svm_parameter('-s 0 -t 0 -c 10 -q')
        model = svm_train(prob, param)
        p_label, p_acc, p_val = svm_predict(y_train_5, x_train, model, '-q')

        w = np.zeros(len(x_train) + 1)

        for i in range(model.l):
            for node in  model.SV[i]:
                if node.index == -1:
                    break
                w[node.index] += model.sv_coef[0][i] * node.value

        print('Problem 11:', '|w| =', math.sqrt(np.dot(w, w)))
```

```python
    if p12 or p13:
        label, Ein, NodeNumber = [], [], []
        for i in range(2, 7):
            y_train_i = OneLabelTransform(y_train, i)
            par = '-s 0 -t 1 -d 3 -c 10 -g 1 -r 1 -q'
            ein, nn = MySVM(x_train, y_train_i, x_train, y_train_i, par, ReturnNode=True)

            label.append(i)
            Ein.append(ein)
            NodeNumber.append(nn)

        if p12:
            ans = np.argmax(Ein)
            print('Problem 12:', 'OVA class =', label[ans], 'with Ein', Ein[ans])

        if p13:
            ans = np.argmax(NodeNumber)
            print('Problem 13:', 'Node number =', NodeNumber[ans])

    if p14:
        y_train_1 = OneLabelTransform(y_train, 1)
        y_test_1 = OneLabelTransform(y_test, 1)
        C, Eout = [0.01, 0.1, 1, 10, 100], []

        for c in C:
            par = '-s 0 -t 2 -g 10 -c ' + str(c) + ' -q'
            eout = MySVM(x_train, y_train_1, x_test, y_test_1, par)
            Eout.append(eout)

        ans = np.argmin(Eout)
        print('Problem 14:', 'C =', C[ans], 'with Eout', Eout[ans])

    if p15:
        y_train_1 = OneLabelTransform(y_train, 1)
        y_test_1 = OneLabelTransform(y_test, 1)
        G, Eout = [0.1, 1, 10, 100, 1000], []

        for g in G:
            par = '-s 0 -t 2 -c 0.1 -g ' + str(g) + ' -q'
            eout = MySVM(x_train, y_train_1, x_test, y_test_1, par)
            Eout.append(eout)

        ans = np.argmin(Eout)
        print('Problem 15:', 'Gamma =', G[ans], 'with Eout', Eout[ans])

    if p16:
        T = 1000
        y_train_1 = OneLabelTransform(y_train, 1)
        G = [0.1, 1, 10, 100, 1000]
        score = np.zeros(len(G))

        for i in tqdm.trange(T):
            rd_val = np.random.choice(len(x_train), 200)
            rd_tra = np.delete(np.array(np.arange(len(x_train))), rd_val)
            x_val, y_val = np.array(x_train)[rd_val], np.array(y_train_1)[rd_val]
            x_tra, y_tra = np.array(x_train)[rd_tra], np.array(y_train_1)[rd_tra]

            Eval = []

            for g in G:
                par = '-s 0 -t 2 -c 0.1 -g ' + str(g) + ' -q'
                ev = MySVM(x_tra, y_tra, x_val, y_val, par)
                Eval.append(ev)

            best = np.argmin(Eval)
            score[best] += 1

            if(i > 0 and i % 100 == 0):
                print(score)

        ans = np.argmax(score)
        print('Problem 16:', 'Gamma =', G[ans], 'for selected', score[ans], 'times')


if __name__ == '__main__':
    main()
```

答案依序為[a], [c], [e], [d], [b], [a].