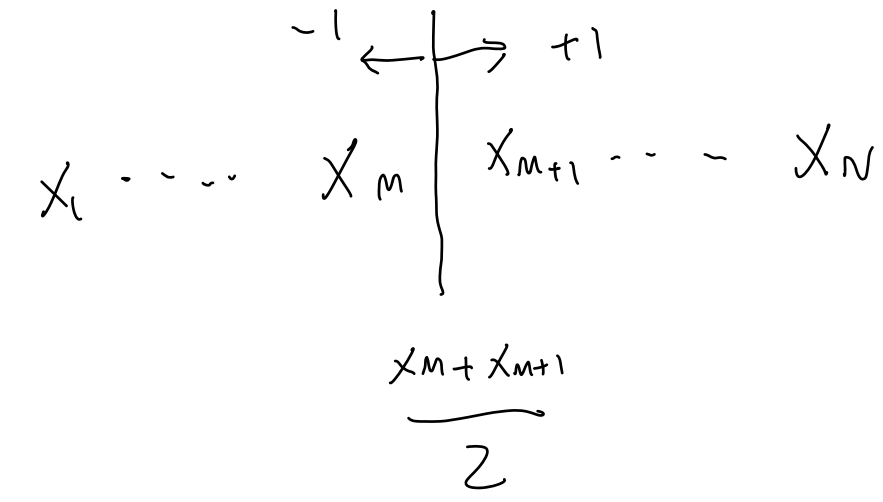
1

(c)




$$\text{sign}\left(x - \frac{x_m + x_{m+1}}{2}\right)$$

2

$$\text{margin}(b, w) = \frac{1}{|w|} = \frac{1}{\left|\sum_{n=1}^{N} a_n y_n z_n\right|}$$

(b)

(1) ✗

(2) ✗

(3) ✓

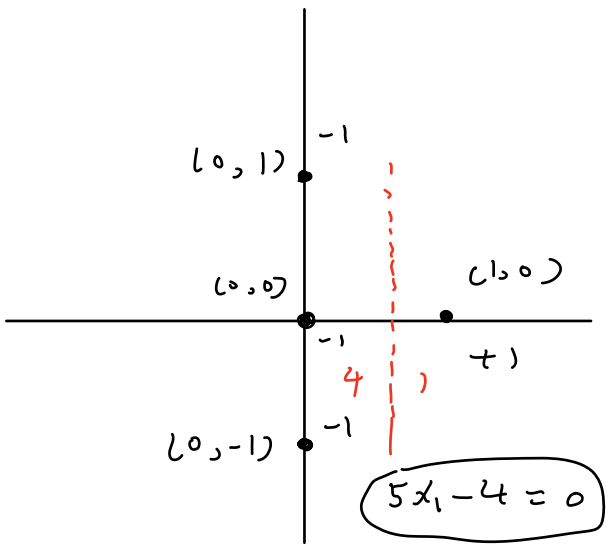
(4) ✗

(5) ✗

(6) ✗

3

(1)




negative margin : positive margin = 4 : 1

4

$$\max_{\text{all } \alpha_n \geq 0} \left( \min_{b,w} \frac{1}{2} w^T w + \sum_{n=1}^{N} \alpha_n (\square - y_n (w^T z_n + b)) \right)$$

$$\square = 1 \quad \text{if } y_n > 0$$

$$\square = p_- \quad \text{if } y_n < 0$$

(c)

..

跟 slide 上的推導一樣

$$\max_{\text{all } a_n \geq 0,\ \sum y_n \alpha_n = 0,\ w = \sum \alpha_n y_n z_n} -\frac{1}{2} \left| \sum_{n=1}^{N} a_n y_n z_n \right|^2 + \sum_{n=1}^{N} \alpha_n \square$$

$$\Rightarrow \min \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} a_n a_m y_n y_m z_n^T z_m + \left( - \sum_{n=1}^{N} a_n \square \right)$$

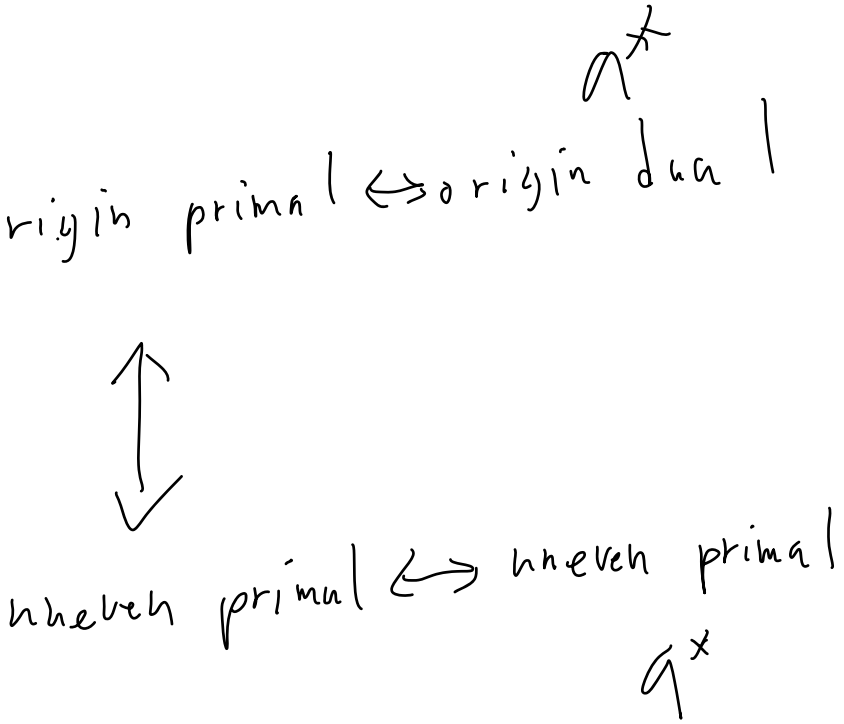
subject to $\sum_{n=1}^{N} y_n a_n = 0$

$a_n \geq 0$ for $n = 1, 2 \cdots N$

$$- \sum_{n=1}^{N} [y_n = +1] a_n - \sum_{n=1}^{N} p_- [y_n = -1] a_n$$

(b)

(a)


$g^*$
origin primal $\leftrightarrow$ origin dual
$\updownarrow$
uneven primal $\leftrightarrow$ uneven primal
$g^*$

6

(9)

$$k(x, x') = (x^T \cdot x')^Q$$

$$= (x_1 x_1' + x_2 x_2' + \cdots + x_d x_d')^Q$$

$$= \varphi(x)^T \varphi(x')$$

$\varphi(x)$ dimesion = $k(x, x')$展開後有幾項

$$\left.\begin{matrix} x_1 x_1' + \cdots + x_d x_d' \\ \vdots \\ x_1 x_1' + \cdots + x_d x_d' \end{matrix}\right\} Q$$

看成同物分箱 Q 同物 d 箱

$$\binom{Q+d-1}{Q}$$

n

(d)

$$|\Phi(x)-\Phi(x')|^2 = \Phi(x)^2 - 2\Phi(x)\Phi(x') + \Phi(x')^2$$

$$= (1+x^Tx)^2 - 2(1+x^Tx')^2 + (1+x'^Tx')^2$$

$$= x^Tx^2 + 2x^Tx - 4x^Tx' - 2x^Tx'^2 + 2x'^Tx' + x'^Tx'^2$$

$$= 1 + 2 + 2 + 1 \underbrace{- 4x^Tx' - 2x^Tx'^2}_{-1 \le x^Tx' \le 1}$$

面2部求極値：2

$$\Downarrow$$

$$-2(x^Tx'+1)^2 + 2$$

$$= 6 + 2$$

$$= 8$$

8

$$W_{t+1} = W_t + y_{n(t)} \phi(x_{n(t)})$$

(1)

$$W_0 = 0 \cdot \phi(x_1) + 0 \cdot \phi(x_2) + \cdots \phi(x_t)$$

,

那個有錯 ,

就加 $y_{n(t)} \phi(x_{n(t)})$;

$$W_t = a_{t(1)} \phi(x_1) + a_{t(2)} \phi(x_2) + \cdots a_{t(n)} \phi(x_t)$$

$$W_{t+1} = a_{t+1(1)} \phi(x_1) + \cdots a_{t+1(n)} \phi(x_t)$$

$$+ y_{n(t)} \phi(x_{n(t)})$$

⇓

$\phi(x_{n(t)})$ 的 $\alpha$ 增加了 $y_{n(t)}$

9

$$\min_{w,b,\xi} \ \frac{1}{2} w^T w + \sum_{n=1}^{N} u_n \cdot \xi_n$$

subject to $y_n(w^T \phi(x_n) + b) \geq 1 - \xi_n$

$\xi_n \geq 0 \quad n = 1, \dots N$

$\Downarrow$ dual

$$\max_{a_n \geq 0 \ \beta_n \geq 0} \left( \min_{b,w,\xi} \ \frac{1}{2} w^T w + \sum_{n=1}^{N} u_n \xi_n + \sum_{n=1}^{N} a_n (1 - \xi_n - y_n(w^T z_n + b)) + \sum_{n=1}^{N} \beta_n \cdot (-\xi_n) \right)$$

$L(b, w, \xi, a, \beta)$

$$\frac{\partial L(b, w, \xi, a, \beta)}{\partial \xi_n} = u_n - a_n - \beta_n = 0$$

$$\Rightarrow \max_{a_n \geq 0 \ \beta_n \geq 0 \ \beta_n = u_n - a_n} \left( \min_{b,w} \ \frac{1}{2} w^T w + \sum_{n=1}^{N} a_n (1 - y_n(w^T z_n + b)) \right)$$

$\Downarrow$ 見 slide 上推導一樣

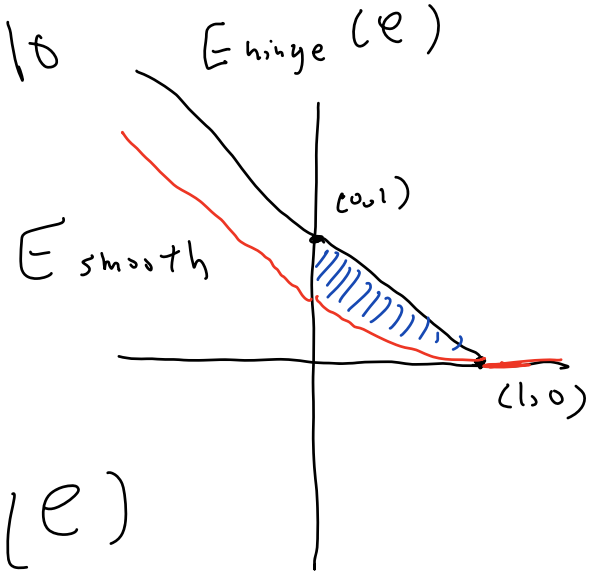
$$\min_{a} \ \frac{1}{2} w^T w - \sum_{n=1}^{N} a_n$$

$$= \min_{a} \ \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} a_n a_m y_n y_m z_n^T z_m - \sum_{n=1}^{N} a_n$$

subject to $y_n(w^T \phi(x_n) + b) \geq 1 - \xi_n$

$\xi_n \geq 0 \quad n = 1 \sim N$

10




$$\int_0^1 \left(1 - e - \frac{1}{2}(1-e)^2\right)^2 de = \frac{8}{60} = \frac{2}{15}$$

11 (a) 4.6462

12 (c) $\frac{37}{4435}$

13 (d) 607

14 (d)

15 (b)

16 (a) 0.1 1 10 100 1000

7424 256 0 0 0

```
import math
import scipy
import random
import numpy as np
from libsvm.svmutil import *

train_y, train_x = svm_read_problem('./train.scale')
test_y, test_x = svm_read_problem('./test.scale')

train_class = []
for i in range(len(train_y)):
    if train_y[i] not in train_class:
        train_class.append(train_y[i])

''' problem 11
for i in range(len(train_y)):
    if train_y[i] != 5.0:
        train_y[i] = 0.0
prob  = svm_problem(train_y, train_x, isKernel=True)
param = svm_parameter('-c 10 -t 0')
m = svm_train(prob, param)
coef = m.get_sv_coef()
sv = m.get_SV()
w = dict()
for i in range(len(coef)):
    for j in sv[i]:
        sv[i][j] *= coef[i][0]
        if j not in w:
            w[j] = sv[i][j]
        else:
            w[j] += sv[i][j]
sum = 0.0
for i in w.values():
    sum += i*i
print(math.sqrt(sum))
'''


'''problem 12 13
for i in range(len(train_y)):
    if train_y[i] != 6.0:
        train_y[i] = 0.0
prob  = svm_problem(train_y, train_x, isKernel=True)
param = svm_parameter('-c 10 -t 1 -d 3 -r 1 -g 1')
m = svm_train(prob, param)
p_label, p_acc, p_val = svm_predict(train_y, train_x, m)
print(p_acc)
'''

'''problem 14 15
for i in range(len(train_y)):
    if train_y[i] != 1.0:
        train_y[i] = 0.0
for i in range(len(test_y)):
    if test_y[i] != 1.0:
        test_y[i] = 0.0
prob  = svm_problem(train_y, train_x, isKernel=True)
param = svm_parameter('-c 0.1 -t 2 -g 1')
m = svm_train(prob, param)
p_label, p_acc, p_val = svm_predict(test_y, test_x, m)
print(p_acc)
'''
par = [0.1, 1, 10, 100, 1000]
score = {0.1:0, 1:0, 10:0, 100:0, 1000:0}
for i in range(len(train_y)):
    if train_y[i] != 1.0:
```

```
        train_y[i] = 0.0
for t in range(1000):
    val_X = []
    val_Y = []
    train_X = []
    train_Y = []
    random_ind = random.sample(list(range(len(train_x))), k=200)
    max = 0
    max_ind = 0
    for i in range(len(train_x)):
        if i in random_ind:
            val_X.append(train_x[i])
            val_Y.append(train_y[i])
        else:
            train_X.append(train_x[i])
            train_Y.append(train_y[i])

    for j in par:
        prob  = svm_problem(train_Y, train_X, isKernel=True)
        param = svm_parameter('-c 0.1 -t 2 -g '+str(j))
        m = svm_train(prob, param)
        p_label, p_acc, p_val = svm_predict(val_Y, val_X, m)
        if p_acc[0] > max:
            max = p_acc[0]
            max_ind = j

    score[max_ind] += 1
print(score)
'''for i in train_class:
    train_Y = np.zeros((len(train_y)))
    for c in range(len(train_y)):
        if train_y[c] == i:
            train_Y[c] = i
    for c in C:
        for j in range(T):
            random.seed(j)
            kernal_parm = '-t 0'
            for k in gamma:
                kernal_parm += ' -g '+str(k)
                prob  = svm_problem(train_Y, train_x, isKernel=True)
                param = svm_parameter('-c '+str(c)+' '+kernal_parm)
                m = svm_train(prob, param)
                print(m.get_SV(), type(m.get_SV()))

'''
```