**ML Techniques HW5**

**1. D**

我們做完feature transform之後，可以發現：  
X=⎡⎢⎣1−24100124⎤⎥⎦,y=⎡⎢⎣−1+1−1⎤⎥⎦X=[1−24100124],y=[−1+1−1]  
列出可知：  
⎧⎪⎨⎪⎩−w1+2w2−4w3−b≥1(1)w1+b≥1(2)−w1−2w2−4w3−b≥1(3){−w1+2w2−4w3−b≥1(1)w1+b≥1(2)−w1−2w2−4w3−b≥1(3)  
從(1)+(2)(1)+(2)可知−4w3≥2−2w2≥2−4w3≥2−2w2≥2，可知w3≤−1/2w3≤−1/2。  
我們就令w2=0,w3=−1/2w2=0,w3=−1/2，注意這符合(1)−(3)(1)−(3)的式子。  
從(1)(1)可知−w1−b≥−1−w1−b≥−1，w1+b≤1w1+b≤1，不過(2)(2)是w1+b≥1w1+b≥1，所以w1+b=1w1+b=1。為了使wTwwTw最小，令w1=0,b=1w1=0,b=1。  
所以：  
w=⎡⎢⎣00−1/2⎤⎥⎦,b=1w=[00−1/2],b=1  
注意這會讓上面三條等式成立，所以他們都是support vector。

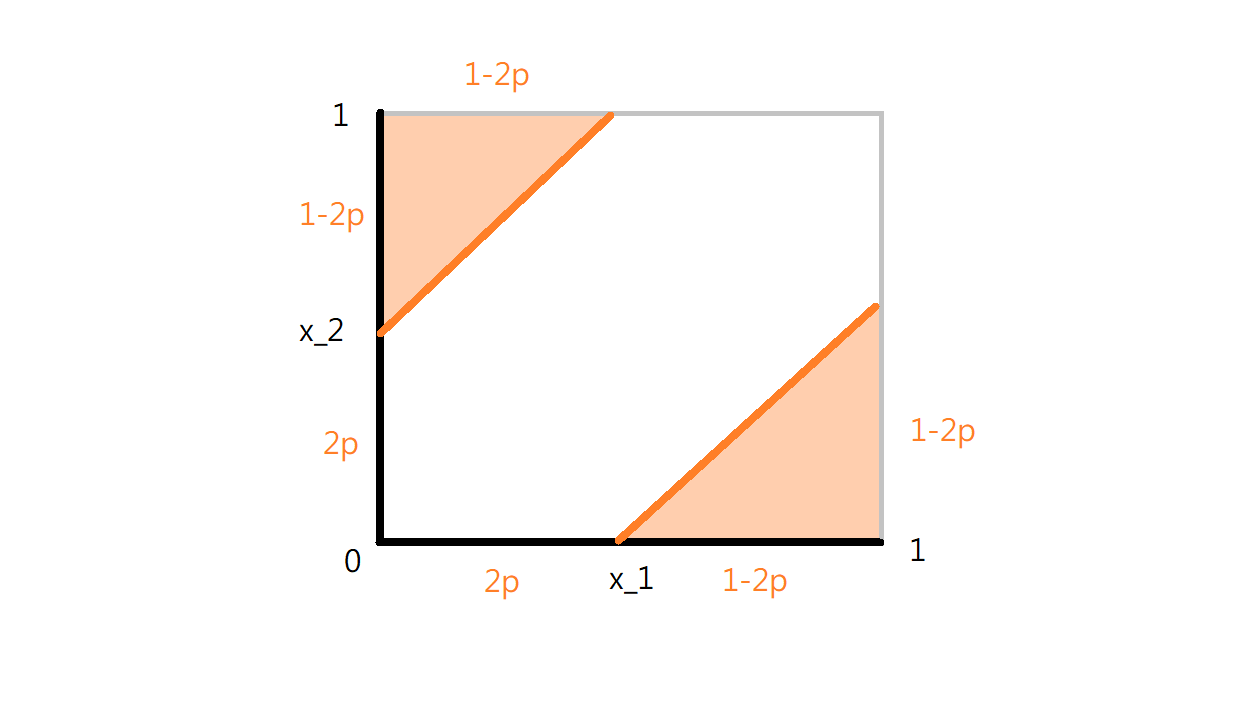
**2. B**

直接拿上面的ww和bb，又我們知道margin(w)=min(dist(ϕ(x),w))margin(w)=min(dist(ϕ(x),w))，又三者的|wTϕ(x)+b|=1|wTϕ(x)+b|=1，所以margin(w)=1/∥w∥=2margin(w)=1/‖w‖=2。

**3. E**

反正生出一組能seperate又讓margin最大的(w,b)(w,b)即可。可知一個可行的hypothesis應該是sign(x−θ)sign(x−θ)，且θ=(xM+xM+1)/2θ=(xM+xM+1)/2。因為只求margin所以也不scale了。  
可知margin應該為xMxM與xM+1xM+1的距離除2，即(xM+1−xM)/2(xM+1−xM)/2。

**4. A**

首先，當y1=y2y1=y2的時候，不論是等於+1還是-1，都可以找到一個hypothesis，硬要找一個的話就說sign(x+100)sign(x+100)還有sign(x−100)sign(x−100)好了。所以這兩個狀況找到的dichotomies數量期望值總共是2。  
至於兩個y1≠y2y1≠y2的狀況，我們畫分布圖看。注意為了使margin大於ρρ，他們的距離必須大於2ρ2ρ：  
  
橫軸是x1x1可能擺放的位置，從0到1；而縱軸的橘色區域是對應的x2x2可以擺放的位置。例如x1=0x1=0時x2x2只能放在[2p,1][2p,1]的位置內。橘色數字代表長度。  
因為總面積為11，可知可行區域的比例即為(1−2p)2(1−2p)2。因為y1≠y2y1≠y2各有(+1,−1)(+1,−1)與(−1,+1)(−1,+1)兩種可能，所以這個狀況的期望值即為2⋅(1−2p)22⋅(1−2p)2。  
因為y1,y2y1,y2是多少的機率都是平均分布的，所以dichotomies期望值為2+2⋅(1−2p)22+2⋅(1−2p)2。

**5. C**

基本上就是重導：  
minb,w12wTwsubject to ∀n,yn(wTxn+b)≥ρ+⋅[[yn=1]]+ρ−⋅[[yn=−1]]minb,w12wTwsubject to ∀n,yn(wTxn+b)≥ρ+⋅[[yn=1]]+ρ−⋅[[yn=−1]]  
然後把Lagrange function寫出來：  
L(b,w,α)=12wTw+N∑n=1(αn((ρ+⋅[[yn=1]]+ρ−⋅[[yn=−1]])−yn(wTxn+b)))L(b,w,α)=12wTw+∑n=1N(αn((ρ+⋅[[yn=1]]+ρ−⋅[[yn=−1]])−yn(wTxn+b)))  
接下來就是導出maxα(minb,wL(b,w,α))maxα(minb,wL(b,w,α))，事實上這個lagrange function幾乎跟原本的沒什麼兩樣，只是把常數11改成(ρ+⋅[[yn=1]]+ρ−⋅[[yn=−1]])(ρ+⋅[[yn=1]]+ρ−⋅[[yn=−1]])而已，注意他對ww和bb微分都為0，所以那些constraint基本上都沒變，我們導一次：  
∂L∂b=0=0+N∑n=1αnyn⇒N∑n=1αnyn=0∂L∂b=0=0+∑n=1Nαnyn⇒∑n=1Nαnyn=0  
跟原本的constraint一樣，lagrange function變成：  
L(b,w,α)=12wTw+N∑n=1(αn((ρ+⋅[[yn=1]]+ρ−⋅[[yn=−1]])−ynwTxn))L(b,w,α)=12wTw+∑n=1N(αn((ρ+⋅[[yn=1]]+ρ−⋅[[yn=−1]])−ynwTxn))  
接下來如法炮製：  
∂L∂wi=0=wi+N∑n=1(−αnynxn,i)⇒w=N∑n=1αnynxn∂L∂wi=0=wi+∑n=1N(−αnynxn,i)⇒w=∑n=1Nαnynxn  
Lagrange function變成：  
L(b,w,α)=12wTw+N∑n=1αn(ρ+⋅[[yn=1]]+ρ−⋅[[yn=−1]])−wTw=−12wTw+N∑n=1αn(ρ+⋅[[yn=1]]+ρ−⋅[[yn=−1]])=−12∥N∑n=1αnynxn∥2+N∑n=1αn(ρ+⋅[[yn=1]]+ρ−⋅[[yn=−1]])L(b,w,α)=12wTw+∑n=1Nαn(ρ+⋅[[yn=1]]+ρ−⋅[[yn=−1]])−wTw=−12wTw+∑n=1Nαn(ρ+⋅[[yn=1]]+ρ−⋅[[yn=−1]])=−12‖∑n=1Nαnynxn‖2+∑n=1Nαn(ρ+⋅[[yn=1]]+ρ−⋅[[yn=−1]])  
我們對lagrange function取負並改成求max，得□◻為：  
−N∑n=1αn(ρ+⋅[[yn=1]]+ρ−⋅[[yn=−1]])−∑n=1Nαn(ρ+⋅[[yn=1]]+ρ−⋅[[yn=−1]])

**6. A**

我們就隨便拿一張SVM的圖片來，就說是201\_handout的第17頁吧，其實沒有圖片也沒差。  
事實上，那條線應該只有平移而已，因為我們已經有讓margin最小化的αα了(也意味我們有ww，因為ww可以用αα做出來)。至於ρρ們會影響的東西是這條線到底比較靠近+1部分的SV還是-1部分的SV，使得−∑∀SVρ(ySV)⋅αSV−∑∀SVρ(ySV)⋅αSV最小化。−∑∀SVρ(ySV)⋅αSV−∑∀SVρ(ySV)⋅αSV即為第五題求得的式子。

**7. D**

內積的定義有一條∀ϕ(x)≠0,⟨ϕ(x),ϕ(x)⟩>0∀ϕ(x)≠0,⟨ϕ(x),ϕ(x)⟩>0。  
請看( D )，如果∃x:K(x,x)<1∃x:K(x,x)<1，那log2K(x,x)<0log2⁡K(x,x)<0，違反內積定義。  
因為題目只要求not alwys a valid kernel，這就是一個例子：舉例如ϕ(x)=[1/2,x]ϕ(x)=[1/2,x]時當x=0x=0時K(x,x)=14<1K(x,x)=14<1時( D )的kernel不可行，因為⟨ϕ(x),ϕ(x)⟩=−2<0,ϕ(x)≠0⟨ϕ(x),ϕ(x)⟩=−2<0,ϕ(x)≠0。

**8. C**

∥ϕ(x)−ϕ(x′)∥2=⟨ϕ(x)−ϕ(x′),ϕ(x)−ϕ(x′)⟩=⟨ϕ(x),ϕ(x)⟩−2⟨ϕ(x),ϕ(x′)⟩+⟨ϕ(x′),ϕ(x′)⟩=exp(−γ∥x−x∥2)−2exp(−γ∥x−x′∥2)+exp(−γ∥x′−x′∥2)=2−2exp(−γ∥x−x′∥2)‖ϕ(x)−ϕ(x′)‖2=⟨ϕ(x)−ϕ(x′),ϕ(x)−ϕ(x′)⟩=⟨ϕ(x),ϕ(x)⟩−2⟨ϕ(x),ϕ(x′)⟩+⟨ϕ(x′),ϕ(x′)⟩=exp⁡(−γ‖x−x‖2)−2exp⁡(−γ‖x−x′‖2)+exp⁡(−γ‖x′−x′‖2)=2−2exp⁡(−γ‖x−x′‖2)  
我們知道exp(−γ∥x−x′∥2)∈(0,1]exp⁡(−γ‖x−x′‖2)∈(0,1]，所以tight upper bound為2。

**9. D**

首先，我們研究K(x,x′)K(x,x′)。對於D={(xn,yn)}Nn=1D={(xn,yn)}n=1N，因為∀a,b∈D,a≠b:∥xa−xb∥≥ϵ∀a,b∈D,a≠b:‖xa−xb‖≥ϵ，所以把他代入到KK：  
K(xa,xb)=exp(−γ∥xa−xb∥2)≤exp(−γϵ2):=ξK(xa,xb)=exp⁡(−γ‖xa−xb‖2)≤exp⁡(−γϵ2):=ξ  
所以我們得知，如果想要對於所有DD中的相異xx的KK都不大於某個數字ξξ的話，只要γ=−ln(ξ)ϵ2γ=−ln⁡(ξ)ϵ2即可。這在最後會使用到。  
接下來，我們需要讓^hh^對於每個(xa,ya)∈D(xa,ya)∈D都預測對，又K(xa,xa)=1K(xa,xa)=1。我們代入：  
^h(xa)=sign(∑∀nynαnK(xn,xa))=sign(∑∀nynK(xn,xa))=sign(ya+∑∀n≠aynK(xn,xa))=yah^(xa)=sign(∑∀nynαnK(xn,xa))=sign(∑∀nynK(xn,xa))=sign(ya+∑∀n≠aynK(xn,xa))=ya  
注意所有K>0K>0，並且為了使yaya的符號成為最終的符號，剩下點的ynK(xn,xa)ynK(xn,xa)都不可凌駕於yaya之上。考慮最扯狀況∀n≠a:yn=−ya∀n≠a:yn=−ya，那可知∑∀n≠aK(xn,xa)∑∀n≠aK(xn,xa)必須小於1。  
為了這個保證，必須使所有K(xn,xa)<1N−1K(xn,xa)<1N−1，因為我們去除了點(xa,ya)(xa,ya)所以我們只加N−1N−1個點。加上最前面的推導，我們知道γγ的tightest lower bound為−ln(1N−1)ϵ2=ln(N−1)ϵ2−ln⁡(1N−1)ϵ2=ln⁡(N−1)ϵ2。

**10. C**

在wt+1wt+1我們要加一份yn(t)ϕ(xn(t))yn(t)ϕ(xn(t))，對應到αα就是讓αn(t)αn(t)加yn(t)yn(t)。

**11. A**

wTtϕ(x)=(N∑n=1αt,nϕ(xn))Tϕ(x)=N∑n=1αt,nK(xn,x)wtTϕ(x)=(∑n=1Nαt,nϕ(xn))Tϕ(x)=∑n=1Nαt,nK(xn,x)

**12. B**

因為∀n:αn=C∀n:αn=C，但是根據complementary slackness，可知∀n,αn(1−ξn−yn(wTxn+b))=0∀n,αn(1−ξn−yn(wTxn+b))=0，可知∀n:1−ξn−yn(wTxn+b)=0∀n:1−ξn−yn(wTxn+b)=0。  
所以ynb=1−ξn−yn∑Ni=1αiyixTixnynb=1−ξn−yn∑i=1NαiyixiTxn。令yn=−1yn=−1會得到b的下界，我們令yn=1yn=1：  
ynb=1⋅b=1−ξn≥0−N∑i=1yiαiK(xi,xn)≤1−N∑i=1yiαiK(xi,xn)ynb=1⋅b=1−ξn⏟≥0−∑i=1NyiαiK(xi,xn)≤1−∑i=1NyiαiK(xi,xn)  
所以最大可能bb取min∀n:yn=1(1−∑Ni=1yiαiK(xi,xn))min∀n:yn=1(1−∑i=1NyiαiK(xi,xn))。取min的原因是因為上式對

**13. E**

我們就不多說直接來吧：  
minb,w,ξ12wTw+CN∑n=1ξ2nsubject to yn(wTϕ(xn)+b)≥1−ξn,∀n⇒L(b,w,ξ,α)=12wTw+CN∑n=1ξ2n+N∑n=1αn(1−ξn−yn(wTϕ(xn)+b))⎧⎪

⎪

⎪

⎪

⎪

⎪

⎪

⎪

⎪

⎪

⎪

⎪⎨⎪

⎪

⎪

⎪

⎪

⎪

⎪

⎪

⎪

⎪

⎪

⎪⎩∂L∂ξn=0=2Cξn−αn⇒ξn=αn2C∂L∂wi=0=wi−N∑n=1αnynϕ(xn)i⇒w=N∑n=1αnynϕ(xn)∂L∂b=0=N∑n=1αnyn⇒N∑n=1αnyn=0⇒L(b,w,ξ,α)=12wTw+CN∑n=1ξ2n+N∑n=1αn(1−ξn−yn(wTϕ(xn)+b))=12wTw+N∑n=1αn(1−yn(wTϕ(xn)))+N∑n=1(Cξ2n−αnξn)=−12wTw+N∑n=1αn+N∑n=1(α2n4C−α2n2C)=−12N∑n=1N∑m=1αnαmynymK(xn,xm)+N∑n=1αn−N∑n=1(α2n4C)=−12N∑n=1N∑m=1αnαmynym(K(xn,xm)+[[n=m]]2C)+N∑n=1αnminb,w,ξ12wTw+C∑n=1Nξn2subject to yn(wTϕ(xn)+b)≥1−ξn,∀n⇒L(b,w,ξ,α)=12wTw+C∑n=1Nξn2+∑n=1Nαn(1−ξn−yn(wTϕ(xn)+b)){∂L∂ξn=0=2Cξn−αn⇒ξn=αn2C∂L∂wi=0=wi−∑n=1Nαnynϕ(xn)i⇒w=∑n=1Nαnynϕ(xn)∂L∂b=0=∑n=1Nαnyn⇒∑n=1Nαnyn=0⇒L(b,w,ξ,α)=12wTw+C∑n=1Nξn2+∑n=1Nαn(1−ξn−yn(wTϕ(xn)+b))=12wTw+∑n=1Nαn(1−yn(wTϕ(xn)))+∑n=1N(Cξn2−αnξn)=−12wTw+∑n=1Nαn+∑n=1N(αn24C−αn22C)=−12∑n=1N∑m=1NαnαmynymK(xn,xm)+∑n=1Nαn−∑n=1N(αn24C)=−12∑n=1N∑m=1Nαnαmynym(K(xn,xm)+[[n=m]]2C)+∑n=1Nαn  
所以LL取負後就是題目的式子。注意最後一行變得有點唐突，他做了三件事：

* 把一個∑∑變成兩個。這勢必要把加的數目用某種方法從N2N2減到NN。
* 進去前面的雙重∑∑裡面，他使用了[[n=m]][[n=m]]來達到讓加的數目變成NN次加上同時有好好的把α2nαn2用[[n=m]][[n=m]]重現。注意y2n=1yn2=1。
* 因為外面有1212，所以提出一個1212，從1414變成1212。

繁雜程度讓人思考當時到底是怎麼想到能直接這樣塞入而不是另外寫一個矩陣加到原本的QQ裡面的…其實這樣好像也是個方法，畢竟α2nαn2會變成identity matrix的二次式形式，所以裡面用[[n=m]][[n=m]]表示identity matrix。這樣解釋應該比較恰當。

**14. E**

如果你真的有看完上面的證明的話，已經寫了：  
∂L∂ξn=0=2Cξn−αn⇒ξn=αn2C∂L∂ξn=0=2Cξn−αn⇒ξn=αn2C  
所以選E。

**15. D**

注意，根據http://www.csie.ntu.edu.tw/~cjlin/libsvm/faq.html#f4151，我們需要改動svm.cpp來算出∥w∥‖w‖。我們把svm.cpp的第747行開始的部分改寫成：

// calculate objective value

{

double v = 0;

int i;

for(i=0;i<l;i++)

v += alpha[i] \* (G[i] + p[i]);

si->obj = v/2;

/\* find ||w|| \*/

double sum\_a = 0;

for(i=0;i<l;i++)

sum\_a += alpha[i];

printf("[\*] |w|^2 = %lf\n", 2\*(si->obj + sum\_a));

}

然後給定python code：

from svmutil import \*

import numpy as np

VALUE = 3

C = 10

def TransformY(y):

return list(map(lambda x: 2\*x-1, np.array(y) == VALUE))

# Read data in LIBSVM format

y, x = svm\_read\_problem('../satimage.scale')

m = svm\_train(TransformY(y), x, '-s 0 -t 0 -c 10')

有輸出：

$ python ML\_HW5.py

........\*..\*.............\*[\*] |w|^2 = 71.518938

optimization finished, #iter = 22874

nu = 0.108695

obj = -4784.881503, rho = 3.623003

nSV = 500, nBSV = 466

Total nSV = 500

Accuracy = 95.8061% (4249/4435) (classification)

對71.518938開根號，得8.45694。

**16. B**

from svmutil import \*

import numpy as np

VALUE = 0

def TransformY(y):

return list(map(lambda x: 2\*x-1, np.array(y) == VALUE))

# Read data in LIBSVM format

y, x = svm\_read\_problem('../satimage.scale')

y\_p, x\_p = svm\_read\_problem('../satimage.scale.t')

ls = []

for i in range(1, 6):

VALUE = i

m = svm\_train(TransformY(y), x, '-s 0 -t 1 -c 10 -d 2 -g 1 -r 1')

p\_label, p\_acc, p\_val = svm\_predict(TransformY(y), x, m)

ls.append([i, np.mean(np.array(p\_label) != np.array(TransformY(y)))])

print(ls)

最後跑出來是[[1, 0.0006764374295377678], [2, 0.0], [3, 0.022322435174746337], [4, 0.040135287485907556], [5, 0.006764374295377677]]，所以最少的是2。

**17. C**

用完全跟16題一模一樣隻字不改的code：

from svmutil import \*

import numpy as np

VALUE = 0

def TransformY(y):

return list(map(lambda x: 2\*x-1, np.array(y) == VALUE))

# Read data in LIBSVM format

y, x = svm\_read\_problem('../satimage.scale')

y\_p, x\_p = svm\_read\_problem('../satimage.scale.t')

ls = []

for i in range(1, 6):

VALUE = i

m = svm\_train(TransformY(y), x, '-s 0 -t 1 -c 10 -d 2 -g 1 -r 1')

p\_label, p\_acc, p\_val = svm\_predict(TransformY(y), x, m)

ls.append([i, np.mean(np.array(p\_label) != np.array(TransformY(y)))])

print(ls)

但是我們看完整的輸出，可以看出來：

$ python ML\_HW5.py

............\*...\*[\*] |w|^2 = 303.202590

optimization finished, #iter = 15704

nu = 0.008385

obj = -220.264016, rho = 0.006089

nSV = 145, nBSV = 10

Total nSV = 145

Accuracy = 99.9324% (4432/4435) (classification)

.\*.\*[\*] |w|^2 = 64.891626

optimization finished, #iter = 2908

nu = 0.001463

obj = -32.447042, rho = 2.547625

nSV = 87, nBSV = 0

Total nSV = 87

Accuracy = 100% (4435/4435) (classification)

.................................\*.........................\*............\*[\*] |w|^2 = 685.915723

optimization finished, #iter = 69995

nu = 0.072509

obj = -2872.825882, rho = 6.225147

nSV = 433, nBSV = 244

Total nSV = 433

Accuracy = 97.7678% (4336/4435) (classification)

.........................................\*......................\*..........\*[\*] |w|^2 = 1314.283655

optimization finished, #iter = 73743

nu = 0.132555

obj = -5221.665416, rho = 2.338587

nSV = 712, nBSV = 499

Total nSV = 712

Accuracy = 95.9865% (4257/4435) (classification)

...........................\*..............\*[\*] |w|^2 = 736.472889

optimization finished, #iter = 41434

nu = 0.031060

obj = -1009.282166, rho = -2.036284

nSV = 259, nBSV = 68

Total nSV = 259

Accuracy = 99.3236% (4405/4435) (classification)

[[1, 0.0006764374295377678], [2, 0.0], [3, 0.022322435174746337], [4, 0.040135287485907556], [5, 0.006764374295377677]]

可以發現，看他的nSV的話，最大712。

**18. D/E**→→**E**

from svmutil import \*

import numpy as np

VALUE = 6

def TransformY(y):

return list(map(lambda x: 2\*x-1, np.array(y) == VALUE))

gamma\_choices = [0.1, 1, 10, 100, 1000]

gamma\_ls = [0] \* 5

# Read data in LIBSVM format

y, x = svm\_read\_problem('../satimage.scale')

y = TransformY(y)

y, x = np.array(y), np.array(x)

y\_t, x\_t = svm\_read\_problem('../satimage.scale.t')

y\_t = TransformY(y\_t)

y\_t, x\_t = np.array(y\_t), np.array(x\_t)

E\_out\_ls = []

for C in [0.01, 0.1, 1, 10, 100]:

# find best gamma

m = svm\_train(y, x, '-s 0 -t 2 -c {} -g {} -q'.format(C, 10))

p\_label, p\_acc, p\_val = svm\_predict(y\_t, x\_t, m, '-q')

E\_out = 1 - p\_acc[0] \* 0.01

E\_out\_ls.append(E\_out)

print(E\_out\_ls)

最後結果是[0.235, 0.16349999999999998, 0.10650000000000004, 0.09699999999999998, 0.09699999999999998]，最後兩個都一樣是最小值，所以DE都行。在這裡我們選E。

**19. B**

from svmutil import \*

import numpy as np

VALUE = 6

def TransformY(y):

return list(map(lambda x: 2\*x-1, np.array(y) == VALUE))

# Read data in LIBSVM format

y, x = svm\_read\_problem('../satimage.scale')

y = TransformY(y)

y, x = np.array(y), np.array(x)

y\_t, x\_t = svm\_read\_problem('../satimage.scale.t')

y\_t = TransformY(y\_t)

y\_t, x\_t = np.array(y\_t), np.array(x\_t)

ls = []

for gamma in [0.1, 1, 10, 100, 1000]:

m = svm\_train(y, x, '-s 0 -t 2 -c {} -g {}'.format(0.1, gamma))

p\_label, p\_acc, p\_val = svm\_predict(y\_t, x\_t, m)

print(p\_acc)

ls.append([gamma, np.mean(np.array(p\_label) != np.array(y\_t))])

print(ls)

做出來是[[0.1, 0.0985], [1, 0.07], [10, 0.1635], [100, 0.235], [1000, 0.235]]，選1。

**20. B**

from svmutil import \*

import numpy as np

VALUE = 6

def TransformY(y):

return list(map(lambda x: 2\*x-1, np.array(y) == VALUE))

gamma\_choices = [0.1, 1, 10, 100, 1000]

gamma\_ls = [0] \* 5

# Read data in LIBSVM format

y, x = svm\_read\_problem('../satimage.scale')

y = TransformY(y)

y, x = np.array(y), np.array(x)

for \_ in range(1000):

# make data

print("[\*] cur exp\_n =", \_, ", gamma\_ls =", gamma\_ls)

choice\_ls = np.random.choice(len(x), 200)

non\_choice\_ls = np.delete(np.array(np.arange(len(x))), choice\_ls)

test\_x = x[choice\_ls]

test\_y = y[choice\_ls]

train\_x = x[non\_choice\_ls]

train\_y = y[non\_choice\_ls]

# find best gamma

E\_val\_ls = []

for i, gamma in enumerate(gamma\_choices):

m = svm\_train(train\_y, train\_x, '-s 0 -t 2 -c {} -g {} -q'.format(0.1, gamma))

p\_label, p\_acc, p\_val = svm\_predict(test\_y, test\_x, m, '-q')

E\_val = 1 - p\_acc[0] \* 0.01

E\_val\_ls.append(E\_val)

print(E\_val\_ls)

gamma\_ls[np.argmin(E\_val\_ls)] += 1

print(gamma\_ls)

跑了實在有夠久之後最後一行跑出來是[265, 735, 0, 0, 0]，選B。

發表於 [**HackMD**](https://hackmd.io/)

 41

[讚賞](https://hackmd.io/@Kaiserouo/HkmAoasjP) [收藏](https://hackmd.io/@Kaiserouo/HkmAoasjP) [訂閱](https://hackmd.io/@Kaiserouo/HkmAoasjP)