Machine Learning HW5

b06502152, 許書銓

Hard-Margin SVM and Large Margin

1. The answer should be (d).

   After a polynominal transform we can rewrite the three examples as,

$$(z_1, y_1) = (1, -2, 4, -1)$$

$$(z_2, y_2) = (1, 0, 0, 1)$$

$$(z_3, y_3) = (1, 2, 4, -1)$$

   From the lecture note, we know that standard SVM must fulfill that $\forall\, n \in [1, n],\ y_n(w^T x_n + b) = 1$. Hence, we can write the equations,

$$\begin{cases} -w_1 + 2w_2 - 4w_3 - b = -1 & (1.0) \\ w_1 + b = 1 & (1.1) \\ -w_1 - 2w_2 - 4w_3 - b = -1 & (1.2) \end{cases}$$

   By doing elementrary operations, we can get that

$$\begin{cases} w_1 + b = 1 & (1.3) \\ w_2 = 0 & (1.4) \\ w_3 = -\frac{1}{2} & (1.5) \end{cases}$$

   From equation (1.3), we can see that we cannot compute the exact w_1 from equations (1.0) ~ (1,2). However, we can constrain w_1 as equation (1.3). In order to compute the optimal w^* , which our goal is to minimize $\frac{1}{2}||w^2||$, we can pick $w_1 = 0$ to have the optimal $w^* = (0, 0, -1/2)$.

2. The answer should be (b).

   From the previous problem, we have selected our optimal w^*. Now we would like to know the optimal margin when $w = w^* \ \wedge \ b = b^*$. From the equation (2.0) in the lecture note,

$$\text{margin} = \frac{1}{||w||} = \frac{1}{\frac{1}{2}} = 2 \tag{2.0}$$

3. The answer should be (e).

   From the concept of hard margin SVM, we try to find the optimal hyperplane, which can seperate all points and its label. Now, we have a 1D data from $x_1, x_2, \ldots, x_N$, which fulfill that $x_1 \leq x_2 \leq \ \ldots \ x_M \leq x_{M+1} \ \leq \ \ldots \ \leq x_N$. Moreover, from the problem, we know that all labels from x_1 to x_M are -1 , and all labels from x_{M+1} to x_N are 1 . Hence, the best point, which can seperate all labels is between x_M and x_{M+1}.

   The margin in the sceneroi is in the middle of the points x_M and x_{M+1}, which means $\frac{1}{2}(x_{M+1} - x_M)$.

4. The answer should be (a).

   The expected value of dichotomy is the possiblity times the dichtomies. Here, we can divided the problem into two possible conditions.

   - (-, -), (+, +)

     In this condition, we can always generate these two dichtomies by setting the $h(x) = \lambda$, which $\lambda < 0$ or $\lambda > 1$. If $\lambda < 0$, we will always generate (+, +) condition. On the other hand , if $\lambda > 1$, we will always generate (-, -) condition.

     Hence, in this case the expected value will yield $\mathbb{E} = 2 * 1 = 2$.

   - (-, +), (+, -)

     In this condition, we would like to find two points which fulfill that

$$|x_1 - h(x)| > \rho \ \wedge \ |x_2 - h(x)| > \rho \tag{4.0}$$

     Hence, the possibility of the condition will be $(1 - 2\rho)^2$. Since, if we random pick a hypothesis $h(x) = \lambda, \ \lambda \in [0, 1]$, then each points must be selected outside the interval $[\lambda - \rho, \lambda + \rho]$. THe possibility of the condition then will be $(1 - 2\rho)^2$. Hence, the expected value $\mathbb{E} = 2 * (1 - 2\rho)^2$.

   To sum up, the expected value of the dichotomies will be $\mathbb{E} = 2 + 2 * (1 - 2\rho)^2$.

Dual Problem of Quadratic Programming

5. The answer should be (c).

From the lecture note, we can rewrite the problem

$$\begin{array}{lll} \min\limits_{b,\mathbf{w}} & \frac{1}{2}\mathbf{w^T w} & \\ \text{subject to} & y_n(\mathbf{w^T x_n} + b) \geq \rho_+ & \text{for n such that } \mathrm{y_n} = +1 \\ & y_n(\mathbf{w^T x_n} + b) \geq \rho_- & \text{for n such that } \mathrm{y_n} = -1 \end{array}$$

into

$$\begin{aligned} \mathcal{L}(b, \mathbf{w}, \alpha) \; = & \\ \max_{\text{all } \alpha \geq 0, \Sigma y_n \alpha_n = 0} & (\min_{b,\mathbf{w}} \; (\frac{1}{2}\mathbf{w^T w} + \\ & \Sigma_{n=1}^{N} \alpha_n [|y_n = +1|](\rho_+ - y_n(\mathbf{w^T x_n} + b)) + \\ & \Sigma_{n=1}^{N} \alpha_n [|y_n = -1|](\rho_- - y_n(\mathbf{w^T x_n} + b)))) \end{aligned} \tag{5.0}$$

At optimal $\frac{\partial \mathcal{L}(b,\mathbf{w},\alpha)}{\partial b}$

$$\begin{aligned} \frac{\partial \mathcal{L}(b, \mathbf{w}, \alpha)}{\partial b} &= 0 \\ &= \Sigma_{n=1}^{N} \alpha_n [|y_n = +1|] y_n + \Sigma_{n=1}^{N} \alpha_n [|y_n = -1|] y_n \\ &= \Sigma_{n=1}^{N} \alpha_n y_n \end{aligned} \tag{5.1}$$

Hence, the equation (5.0) may rewrite as

$$\begin{aligned} \mathcal{L}(b, \mathbf{w}, \alpha) \; = & \\ \max_{\text{all } \alpha \geq 0, \Sigma y_n \alpha_n = 0} & (\min_{b,\mathbf{w}} \; (\frac{1}{2}\mathbf{w^T w} + \Sigma_{n=1}^{N} \alpha_n [|y_n = +1|](\rho_+ - y_n(\mathbf{w^T x_n})) + \Sigma_{n=1}^{N} \alpha_n [|y_n = -1|](\rho_- - y_n(\mathbf{w^T x_n})))) \end{aligned} \tag{5.2}$$

At optimal $\frac{\partial \mathcal{L}(b,\mathbf{w},\alpha)}{\partial w_i}$

$$\frac{\partial \mathcal{L}(b,\mathbf{w},\alpha)}{\partial w_i} = 0$$

$$= \mathbf{w} - \Sigma_{n=1}^{N} \alpha_n [|y_n = +1|] y_n x_n + \Sigma_{n=1}^{N} \alpha_n [|y_n = -1|] y_n x_n$$

$$= \mathbf{w} - \Sigma_{n=1}^{N} \alpha_n y_n x_{n,i} \tag{5.3}$$

Hence, the equation (5.1) can rewrite as

$$\mathcal{L}(b,\mathbf{w},\alpha)$$

$$= \max_{\text{all } \alpha \geq 0, \Sigma y_n \alpha_n = 0, \mathbf{w} = \Sigma_{n=1}^{N} \alpha_n y_n x_{n,i}} \left(\left(-\frac{1}{2} \mathbf{w}^{\mathbf{T}} \mathbf{w} + \Sigma_{n=1}^{N} \alpha_n [|y_n = +1|] \rho_+ + \Sigma_{n=1}^{N} \alpha_n [|y_n = -1|] \rho_- \right) \right) \quad (5.4)$$

$$= \min_{\text{all } \alpha \geq 0, \Sigma y_n \alpha_n = 0, \mathbf{w} = \Sigma_{n=1}^{N} \alpha_n y_n x_{n,i}} \left(\left(\frac{1}{2} \mathbf{w}^{\mathbf{T}} \mathbf{w} - \Sigma_{n=1}^{N} \alpha_n [|y_n = +1|] \rho_+ - \Sigma_{n=1}^{N} \alpha_n [|y_n = -1|] \rho_- \right) \right) \quad (5.4)$$

6. The answer should be (e).

From complementary slackness in lecture 202, we know the support vectors fulfill the condition,

$$\text{if } y_n = 1, \quad 1 - \mathbf{w}^T \mathbf{z}_n - b = 0 \tag{6.0}$$
$$\text{if } y_n = -1, \quad 1 + \mathbf{w}^T \mathbf{z}_n + b = 0 \tag{6.1}$$

for tht all support vectors's $\alpha > 0$.

To consider that ρ_+ and $\rho-$ may not be the same, we have to write the complementary slackness for this condition.

$$\text{if } y_n = 1, \quad \alpha_n(\rho_+ - \mathbf{w}^T \mathbf{z}_n - b) = 0 \tag{6.2}$$
$$\text{if } y_n = -1, \quad \alpha_n(\rho_- + \mathbf{w}^T \mathbf{z}_n + b) = 0 \tag{6.3}$$

Consider support vectors, whose $\alpha > 0$; hence,

$$\text{if } y_n = 1, \quad \rho_+ - \mathbf{w}^T \mathbf{z}_n - b = 0 \tag{6.4}$$
$$\text{if } y_n = -1, \quad \rho_- + \mathbf{w}^T \mathbf{z}_n + b = 0 \tag{6.5}$$

If we try to add equation (6.0) and (6.1), we will have

$$2 = \mathbf{w}_{\alpha^*}^T (\mathbf{z}_n - \mathbf{z}_m), \ \ \mathbf{z}_n \text{ for } \mathrm{y_n} = 1 \text{ and } \mathbf{z}_m \text{ for } \mathrm{y_n} = -1 \tag{6.6}$$

$$(\mathbf{z}_n - \mathbf{z}_m) = \frac{2}{\mathbf{w}_{\alpha^*}^T} \tag{6.7}$$

Now, we do the same steps for the consition that ρ_+ and $\rho-$ may not be the same. Summing up equation (6.4) and (6.5), we will get

$$\rho_+ + \rho_- = \mathbf{w}_{\alpha}^T (\mathbf{z}_n - \mathbf{z}_m), \ \ \mathbf{z}_n \text{ for } \mathrm{y_n} = 1 \text{ and } \mathbf{z}_m \text{ for } \mathrm{y_n} = -1 \tag{6.8}$$

From equation (6.7), equation (6.8) can be rewrited as

$$(\rho_+ + \rho_-)\mathbf{w}_{\alpha^*}^T = 2\mathbf{w}_\alpha^T \tag{6.9}$$

Since $\mathbf{w} = \Sigma y_n \alpha_n z_n$, and y_n and z_n is from the data itself, which will not change with different hypothesis. Thus, the only term that will affect $\mathbf{w}$ is α term. The equation (6.9) can further be rewrited as

$$(\rho_+ + \rho_-)\alpha^* = 2\alpha \tag{6.10}$$

$$\alpha = \frac{\rho_+ + \rho_-}{2}\alpha^* \tag{6.11}$$

Properties of Kernels

7. The answer should be (d).

From the lecture note, the necessary condition for a valid kernel function is that the function $\mathrm{K}(\mathbf{x}, \mathbf{x}')$ must always that the matrix K be positive semi-finite. Hence, we should check each option's validity by checking whether it is a positive semi-finite K.

Moreover, from the problem itself, we all know $\mathrm{K}(\mathbf{x}, \mathbf{x}')$ will lead matrix K to a positive semi-finite matrix, which implies that

$$y^T K y \geq 0 \tag{7.0}$$

Furthermore, a positive semi-finite matrix must satisfy that the kth leading principal minor of a matrix K is the determinant of its upper-left k×k sub-matrix. It turns out that a matrix is positive semi-definite if and only if all these determinants are non-negative.

(d) Since the element of $\mathrm{K}(\mathbf{x}, \mathbf{x}')$ is $\in [0, 2)$, if the element in the diagonal in $\mathrm{K}(\mathbf{x}, \mathbf{x}')$ is $\frac{1}{2}$. Then, it will lead the element in the kernel of $\log_2 K(\mathbf{x}, \mathbf{x}')$ to be $\log_2(1/2) = -1$, which is a negative element, vioalting the definition of positive semi-finite kernel.

construct the matrix K from the kernel $\mathrm{K}(\mathbf{x}, \mathbf{x}')$

$$K = \begin{bmatrix} 1 & 0.25 \\ 0.25 & 1 \end{bmatrix} \tag{7.1}$$

Hence, the matrix K' from the kernel $\log_2 \mathrm{K}(\mathbf{x}, \mathbf{x}')$ will be

$$K' = \begin{bmatrix} 0 & -2 \\ -2 & 0 \end{bmatrix} \tag{7.2}$$

It is obvious see that K' is not positive semi-finite, since its determinant is less than 0.

8. The answer should be (c).

From the lecture note, we realize that the distance between two examples in $\mathcal{Z}$ domain will be $||\phi(x) - \phi(x')||^2$,

$$||\phi(x) - \phi(x')||^2 = \phi(x)^T \phi(x) - 2\phi(x)^T \phi(x') + \phi(x')^T \phi(x') \tag{8.0}$$

Furthermore, knrenl function gives us

$$\phi(x)^T \phi(x) = \exp(-\gamma ||x - x||^2) = \exp(0) = 1 \tag{8.1}$$

which implies that the first and thir term in equation (8.0) will be 1. Hence, we can write the equation (8.0) into

$$\begin{aligned} ||\phi(x) - \phi(x')||^2 &= 2 - 2\phi(x)^T \phi(x') \\ &\leq 2 - 0 = 2, \end{aligned} \tag{8.2}$$

9. The answer should be (d).

From the probelm itself, our hypothesis will be

$$h_{1,0}(\mathbf{x}) = \hat{h}(\mathbf{x}) = sign\ (\Sigma_{n=1}^{N} y_n \exp(-\gamma ||\mathbf{x_n} - \mathbf{x}||^2)) \tag{9.0}$$

Now, to make $E_{in}(\hat{h}) = 0$,

$$E_{in}(\hat{h}) = 0 \leftrightarrows \forall y_k,\ k \in [1, N],\ s.t.\ sign(y_k) = sign\ (\Sigma_{n=1}^{N} y_n \exp(-\gamma ||\mathbf{x_n} - \mathbf{x_k}||^2)) \tag{9.1}$$

From the equation (9.1), the condition may rewrite as,

$$y_k(\Sigma_{n=1}^{N} y_n \exp(-\gamma||\mathbf{x_n} - \mathbf{x_k}||^2)) > 0 \tag{9.2}$$

$$y_k(\Sigma_{n=1}^{N}([|y_k \neq y_n|])\, y_n \exp(-\gamma||\mathbf{x_n} - \mathbf{x_k}||^2)\ +\ y_k) > 0 \tag{9.3}$$

$$y_k\ \Sigma_{n=1}^{N}([|y_k \neq y_n|])\, y_n \exp(-\gamma||\mathbf{x_n} - \mathbf{x_k}||^2)\ > -1 \tag{9.4}$$

Since $y_k\ \Sigma_{n=1}^{N}([|y_k \neq y_n|])\ y_n$ is a negative term, the equation (9.4) can rewrite as,

$$\exp(-\gamma||\mathbf{x_n} - \mathbf{x_k}||^2)\ <\ -\frac{1}{y_k\ \Sigma_{n=1}^{N}([|y_k \neq y_n|])\ y_n} \tag{9.5}$$

$$\begin{aligned} -\gamma||\mathbf{x_n} - \mathbf{x_k}||^2\ &<\ \ln(-\frac{1}{y_k\ \Sigma_{n=1}^{N}([|y_k \neq y_n|])\ y_n}) \\ &=\ \ln(y_k\ \Sigma_{n=1}^{N}([|y_k \neq y_n|])\ y_n) \end{aligned} \tag{9.6}$$

$$\gamma\ >\ -\frac{\ln(y_k\ \Sigma_{n=1}^{N}([|y_k \neq y_n|])\ y_n)}{||\mathbf{x_n} - \mathbf{x_k}||^2} \tag{9.7}$$

Since $|\mathbf{x_n} - \mathbf{x_k}| \geq \epsilon$, the equation (9.7) can rewrite as,

$$\begin{aligned} \gamma\ &>\ -\frac{\in (y_k\ \Sigma_{n=1}^{N}([|y_k \neq y_n|])\ y_n)}{||\mathbf{x_n} - \mathbf{x_k}||^2} \\ &>\ -\frac{\ln(y_k\ \Sigma_{n=1}^{N}([|y_k \neq y_n|])\ y_n)}{\epsilon^2} \end{aligned} \tag{9.8}$$

To obtain the tightest lowerbond of γ, if all $sign(y_n) \neq sign(y_k), \forall y_k \neq y_n$ will lead the equation (9.8) to

$$\gamma > \frac{\ln(N-1)}{\epsilon^2} \tag{9.9}$$

Kernel Perceptron Learning Algorithm

10. The answer should be (c).

From the lecture note in Perceptron Learning Algorithm, we updates $\mathbf{w_t}$ to $\mathbf{w_{t+1}}$ by the equation (10.0)

$$\mathbf{w_{t+1}} \leftarrow \mathbf{w_t} + y_{n(t)}\phi(\mathbf{x_n}) \tag{10.0}$$

Since every update is based on the previous example, if we take $\mathbf{w} = 0$, we can represent every $\mathbf{w_t}$ as a linear combination of $\{\phi(\mathbf{x_n})\}_{n=1}^{N}$.

$$\mathbf{w}_t = \Sigma_{n=1}^{N}\alpha_{t,n}\phi(\mathbf{x}_n) \tag{10.1}$$

As we focus on $\mathbf{w}_{t+1}$, the equation (10..1) can be rewrited as,

$$\mathbf{w}_{t+1} = \Sigma_{n=1}^{N}\alpha_{t+1,n}\phi(\mathbf{x}_n) \tag{10.2}$$

By using relationship in the equation (10.1) in the equation(10.0), the euqation (10.0) can be rewrite as,

$$\mathbf{w_{t+1}} \leftarrow \Sigma_{n=1}^{N}\alpha_{t,n}\phi(\mathbf{x}_n) + y_{n(t)}\phi(\mathbf{x_n}) \tag{10.3}$$

Hence, by comparing equation (10.2) and equation (10.3), it is obvious that α will be updated as,

$$\begin{aligned} \alpha_{t+1} &\leftarrow \alpha_t \\ \text{except} \;\; \alpha_{t+1,n(t)} &\leftarrow \alpha_{t,n(t)} + y_n(t) \end{aligned} \tag{10.4}$$

11. The answer should be (a).

In the previous problem (10), we realize the equation (10.1),

$$\mathbf{w}_t = \Sigma_{n=1}^{N}\alpha_{t,n}\phi(\mathbf{x}_n) \tag{10.1}$$

Now we would like to compute

$$\begin{aligned} \mathbf{w}_t^T\phi(\mathbf{x}) &= \Sigma_{n=1}^{T}\alpha_{t,n}\phi(x_n)^T\phi(\mathbf{x}) \\ &= \Sigma_{n=1}^{T}\alpha_{t,n}K(\mathbf{x}_n, \mathbf{x}) \end{aligned} \tag{10.2}$$

Soft-Margin SVM

12. The answer should be (b).

From complementary slackness properties,

$$\alpha_n(1 - \xi_n - y_n(\mathbf{w}^T\mathbf{z}_n + b)) = 0 \tag{12.0}$$
$$(C - \alpha_n)\xi_n = 0 \tag{12.1}$$

From problem itself, all $\alpha_n^* = C$. Hence, from equation (12.0),

$$1 - \xi_n - y_n(\mathbf{w}^T\mathbf{z}_n + b) = 0 \tag{12.2}$$
$$\xi \geq 0 \tag{12.3}$$

Using the result in equation (12.3), and rewrite the equation (12.2).

$$1 - y_n(\mathbf{w}^T\mathbf{z}_n + b) \geq 0 \tag{12.4}$$

Here, i would like to devided the case into $y_n > 0$, or $y_n < 0$.

- Case of $y_n > 0$

  We times y_n, which $y_n > 0$ both side of equation (12.4)

$$y_n - (\mathbf{w}^T\mathbf{z}_n + b) \geq 0 \tag{12.5}$$

  Moreover, all bounded SVs must be satisfied the above equation (12.5)

$$b \leq y_n - \mathbf{w}^T\mathbf{z}_n \tag{12.6}$$

$$= 1 - y_n\mathbf{w}^T\mathbf{z}_n \tag{12.7}$$

$$= 1 - \Sigma_{m=1}^{N} y_m\alpha_m K(\mathbf{x}_n, \mathbf{x}_m) \tag{12.8}$$

  Now, compareing which $\min$ or averagewould lead to the largest, from the equation (12.7), the second term would be smaller when we choose the smaller y_n. Hence, $\min$ one would lead the second term smaller and make the overall b^* larger. Thus, the upper bound of b would be

$$b = b^* = 1 - \Sigma_{m=1}^{N} y_m \alpha_m K(\mathbf{x}_n, \mathbf{x}_m) \tag{12.8}$$

- Case of $y_n < 0$

We times y_n, which $y_n < 0$ both side of equation (12.4)

$$y_n - (\mathbf{w}^T \mathbf{z}_n + b) \leq 0 \tag{12.9}$$

Moreover, all bounded SVs must be satisfied the above equation (12.9)

$$b \geq y_n - \mathbf{w}^T \mathbf{z}_n \tag{12.10}$$

$$= 1 - y_n \mathbf{w}^T \mathbf{z}_n \tag{12.11}$$

$$= 1 - \Sigma_{m=1}^{N} y_m \alpha_m K(\mathbf{x}_n, \mathbf{x}_m) \tag{12.12}$$

Hence, it shows that the lower bound of b would be equation (12.12).

13. The answer sohuld be (e).

From the problem, (P_2) is rewritten as,

$$\begin{aligned} (P_2) \quad \min \quad & \frac{1}{2}\mathbf{w}^T \mathbf{w} + C\, \Sigma_{n=1}^{N}\, \xi_n^2 \\ \text{subjct to} \quad & y_n(\mathbf{w}\phi(\mathbf{x}_n) + b) \geq 1 - \xi_n, \quad \text{for } n = 1, 2, \ldots\ldots N \end{aligned} \tag{13.0}$$

Our goal is to make P_2 look like primal SVM form as,

$$\begin{aligned} (P_0) \quad \min \quad & \frac{1}{2}\mathbf{w}^T \mathbf{w} \\ \text{subjct to} \quad & y_n(\mathbf{w}\phi(\mathbf{x}_n) + b) \geq 1, \quad \text{for } n = 1, 2, \ldots\ldots N \end{aligned} \tag{13.1}$$

Hence, we try to write ξ in w, which means to let $\frac{1}{2}\mathbf{w}^T \mathbf{w} + C\, \Sigma_{n=1}^{N}\, \xi_n^2 = \frac{1}{2}\tilde{\mathbf{w}}^T \tilde{\mathbf{w}}$.

$$\frac{1}{2}\mathbf{w}^T \mathbf{w} + C\, \Sigma_{n=1}^{N}\, \xi_n^2 = \frac{1}{2}\tilde{\mathbf{w}}^T \tilde{\mathbf{w}} \tag{13.2}$$

$$\frac{1}{2}\,(\mathbf{w}^T \mathbf{w} + 2C\, \Sigma_{n=1}^{N}\, \xi_n^2) = \frac{1}{2}\tilde{\mathbf{w}}^T \tilde{\mathbf{w}} \tag{13.3}$$

From the equation (13.3), we know that $\tilde{\mathbf{w}}$ must be,

$$\tilde{\mathbf{w}} = \begin{bmatrix} w \\ \sqrt{2C}\xi_1 \\ \sqrt{2C}\xi_2 \\ \cdots \\ \sqrt{2C}\xi_n \end{bmatrix} \tag{13.4}$$

Furthermore, the constrain can be written in

$$\begin{aligned} & y_n(\mathbf{w}\phi(\mathbf{x}_n) + b) \geq 1 - \xi_n \\ \leftrightarrows\ & y_n(\mathbf{w}\phi(\mathbf{x}_n) + b + \frac{1}{y_n}\xi_n) \geq 1 \\ \leftrightarrows\ & y_n(\mathbf{w}\phi(\mathbf{x}_n) + b + y_n\xi_n) \geq 1 \\ \leftrightarrows\ & y_n(\mathbf{w}\phi(\mathbf{x}_n) + b + (\sqrt{2C}\xi_n)(\frac{y_n}{\sqrt{2}C})) \geq 1 \end{aligned} \tag{13.5}$$

From the equation (13.3), we know that $\tilde{\phi}(\mathbf{x})$ must be,

$$\tilde{\phi}(\mathbf{x}) = \begin{bmatrix} x \\ 0 \\ 0 \\ \cdots \\ \frac{1}{\sqrt{2C}}y_n \\ \cdots \\ 0 \end{bmatrix} \tag{13.6}$$

which means, only when n == m term will have a value $\frac{1}{\sqrt{2C}y_n}$.

Hence, from the lecture note in (201 & 202 & 204), in primal SVM, we can rewrite the equation (P_2) to

$$\begin{aligned} (P_0)\quad \min\quad & \frac{1}{2}\tilde{\mathbf{w}}^T\tilde{\mathbf{w}} \\ \text{subjct to}\quad & y_n(\tilde{\mathbf{w}}\tilde{\phi}(\mathbf{x}_n) + b) \geq 1,\ \text{ for } n = 1, 2, \ldots\ldots N \end{aligned} \tag{13.1}$$

$$\leftrightarrows$$

$$\begin{aligned} (P_0)\quad \min\quad & \frac{1}{2}\Sigma_{n=1}^N\Sigma_{m=1}^N\alpha_n\alpha_m y_n y_m\tilde{\phi}(\mathbf{x}_n)^T\tilde{\phi}(\mathbf{x}_n) - \Sigma_{n=1}^N\alpha_n \\ \text{subjct to}\quad & \Sigma_{n=1}^N y_n\alpha_n = 0 \\ & \alpha_n \geq 0, \text{for } n = 1, 2, \ldots N \end{aligned} \tag{13.7}$$

which

$$\tilde{\phi}(\mathbf{x})^T\tilde{\phi}(\mathbf{x}) = \phi(\mathbf{x})^T\phi(\mathbf{x}) + (\frac{1}{\sqrt{2C}})^2[|n=m|]$$

$$= K(\mathbf{x}_n, \mathbf{x}_m) + \frac{1}{2C}[|n=m|] \tag{13.8}$$

14. The answer should be (e).

From the form of P_2,

$$\begin{aligned} (P_2) \quad \min \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\,\Sigma_{n=1}^{N}\,\xi_n^2 \\ \text{subjct to} \quad & y_n(\mathbf{w}\phi(\mathbf{x}_n)+b) \geq 1-\xi_n, \quad \text{for } n=1,2,\ldots\ldots N \end{aligned} \tag{13.0}$$

we can hide the constrain into

$$(P_2)\ \ \mathcal{L} = \max_{\alpha_n\geq 0}\ (\min\ \ \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\,\Sigma_{n=1}^{N}\,\xi_n^2 + \Sigma_{n=1}^{N}\alpha_n(1-\xi_n - y_n(\mathbf{w}^{\mathbf{T}}\phi(\mathbf{x})+b))) \tag{14.0}$$

To find the optimal solution,

$$\frac{\partial\mathcal{L}}{\partial\xi_n} = 0 = 2C\xi_n^* - \alpha_n^* \tag{14.1}$$

$$\alpha^* = \frac{1}{2C}\xi^* \tag{14.2}$$

15. The answer should be (d).

```
## prob 15
import numpy as np
import pandas as pd
import math
import scipy
import sys

# add the path of liblinear package
LIB_LINSVM_PATH = "./libsvm-master/python"
sys.path.append(LIB_LINSVM_PATH)
from svm import *
from svmutil import *

#main
train_y, train_x = svm_read_problem('./satimage.shape', return_scipy = True)
train_y = train_y == 3
```

```
m = svm_train(train_y, train_x, '-s 0 -t 0 -c 10')

support_vector_coefficients = m.get_sv_coef()
support_vector_coefficients = np.squeeze(support_vector_coefficients)
support_vectors = m.get_SV()
w = np.zeros(train_x.shape[1])
sv = pd.DataFrame(support_vectors,
index=np.arange(len(support_vectors))).sort_index(axis=1).fillna(0).to_numpy()

for i in range(len(support_vectors)):
    w += support_vector_coefficients[i] * sv[i]

print(np.linalg.norm(w))
```

16. The answer should be (b).

```
## problem 16
import numpy as np
import pandas as pd
import math
import scipy
import sys

# add the path of liblinear package
LIB_LINSVM_PATH = "./libsvm-master/python"
sys.path.append(LIB_LINSVM_PATH)
from svm import *
from svmutil import *

#main
train_y, train_x = svm_read_problem('./satimage.shape', return_scipy = True)

for i in range(5):
    print(str(i+1) +' versus not ' + str(i+1))
    train_y_target = train_y == i+1
    m = svm_train(train_y_target, train_x, '-s 0 -t 1 -g 1 -r 1 -d 2 -c 10')
    p_label, p_acc, p_val = svm_predict(train_y_target, train_x, m)
```

17. The answer should be (c).

```
##problem 17
import numpy as np
import pandas as pd
import math
```

```
import scipy
import sys

# add the path of liblinear package
LIB_LINSVM_PATH = "./libsvm-master/python"
sys.path.append(LIB_LINSVM_PATH)
from svm import *
from svmutil import *

#main
train_y, train_x = svm_read_problem('./satimage.shape', return_scipy = True)

for i in range(5):
    print(str(i+1) +' versus not ' + str(i+1))
    train_y_target = train_y == i+1
    m = svm_train(train_y_target, train_x, '-s 0 -t 1 -g 1 -r 1 -d 2 -c 10')
    print(len(m.get_SV()))
    #p_label, p_acc, p_val = svm_predict(train_y_target, train_x, m)
```

18. The answer should be (d) or (e).

```
## problem 18
import numpy as np
import pandas as pd
import math
import scipy
import sys

# add the path of liblinear package
LIB_LINSVM_PATH = "./libsvm-master/python"
sys.path.append(LIB_LINSVM_PATH)
from svm import *
from svmutil import *

#main
train_y, train_x = svm_read_problem('./satimage.shape', return_scipy = True)
test_y, test_x = svm_read_problem('./satimage.scale.t', return_scipy = True)

C = [0.01, 0.1, 1, 10, 100]
for i in range(5):
    print('C = ' + str(C[i]))
    train_y_target = train_y == 6
    test_y_target = test_y == 6
    m = svm_train(train_y_target, train_x, '-s 0 -t 2 -g 10 -c ' + str(C[i]))
    p_label, p_acc, p_val = svm_predict(test_y_target, test_x, m)
```

19. The answer should be (b).

```
## problem 19
import numpy as np
import pandas as pd
import math
import scipy
import sys

# add the path of liblinear package
LIB_LINSVM_PATH = "./libsvm-master/python"
sys.path.append(LIB_LINSVM_PATH)
from svm import *
from svmutil import *

#main
train_y, train_x = svm_read_problem('./satimage.shape', return_scipy = True)
test_y, test_x = svm_read_problem('./satimage.scale.t', return_scipy = True)

r = [0.1, 1, 10, 100, 1000]
for i in range(5):
    print('\gamma = ' + str(r[i]))
    train_y_target = train_y == 6
    test_y_target = test_y == 6
    m = svm_train(train_y_target, train_x, '-s 0 -t 2 -g ' + str(r[i]) + ' -c 0.1')
    p_label, p_acc, p_val = svm_predict(test_y_target, test_x, m)
```

20. The answer should be (b).

```
## problem20
import numpy as np
import pandas as pd
import math
import matplotlib.pyplot as plt
import random
import scipy
import sys

# add the path of liblinear package
LIB_LINSVM_PATH = "./libsvm-master/python"
sys.path.append(LIB_LINSVM_PATH)
from svm import *
from svmutil import *

#main
train_y, train_x = svm_read_problem('./satimage.shape', return_scipy = True)
test_y, test_x = svm_read_problem('./satimage.scale.t', return_scipy = True)

x = np.zeros((train_y.shape[0], 36))
for i in range(train_y.shape[0]):
```

```python
    for j in range(36):
        if train_x[i, j] == None:
            x[i][j] = 0
        else:
            x[i][j] = train_x[i, j]

train_target_y = train_y == 6
train_target_y = train_target_y.reshape(-1,1)
data = np.append(train_target_y, x, axis = 1)

r = [0.1, 1, 10, 100, 1000]

result = []
for i in range(10):
    print(i)
    np.random.shuffle(data)

    train_y = data[:, 0]
    train_x = data[:, 1: ]

    val_y = train_y[:200]
    val_x = train_x[:200, :]
    train_minus_y = train_y[200: ]
    train_minus_x = train_x[200:, :]

    best_acc = 0
    best_idx = 0
    for j in range(5):
        m = svm_train(train_minus_y, train_minus_x, '-s 0 -t 2 -g ' + str(r[j]) + ' -
c 0.1')
        p_label, p_acc, p_val = svm_predict(val_y, val_x, m)

        if(p_acc[0] > best_acc):
            best_acc = p_acc[0]
            best_idx = j

    result.append(best_idx)

plt.hist(result)
print("most number of idx: ", np.bincount(result).argmax())
```