Soft-margin SVM

Q1:

No, there isn't such a guarantee that all training samples in D will be assigned the correct label by the fitted model. As we can see from the influence of the penalty C on the model, when C grows larger, the model tend to have the shape of hard-margin SVM. When it becomes smaller, we get a larger margin region around the decision boundary, and the region may cover some data points. In a properer model, some outliers (data points) may be misclassified in some cases.

Q2:

When we compute the gradient of Lagrangian, the result:

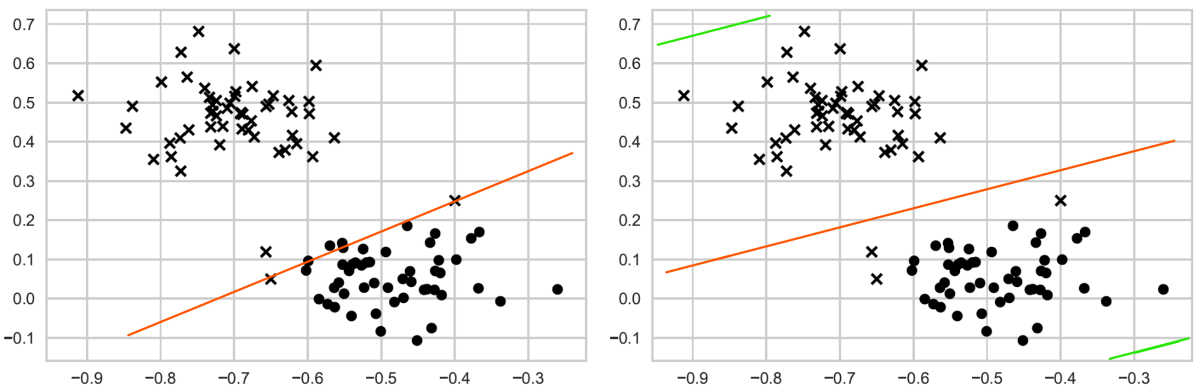
$$C - \alpha_i - \mu_i = 0$$

with α and μ are larger or equal than zero. If $C = 0$, this will lead ξ to have no influence, which means we throw away all the constraints. Finally, the SVM will generate an arbitrary model, which may highly useless. For $C < 0$, we look back at the cost function and the requirement:

$$\text{minimize } f_0(\boldsymbol{w}, b, \boldsymbol{\xi}) = \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_{i=1}^{N}\xi_i$$

the second part of the right side is not negative, minimize the equation will let this part be infinite small, thus $\xi_i \to \infty$. Of course, this is also not a good model.

Q3:



(a) $C = 10^{10}$ (b) $C = 10^{-10}$

Figure 1: Sketch of the decision boundary according to C

C "determines how heavy a violation is punished"; an extremely large C (the left case) will punish a violation really hard, this gives a model no slackness region to allow some "noise" data, and the model to tend to behave like hard-margin SVM. An small C will give slack variable lots of space to grow.

Kernels

Q4:

We do a transformation of the equation:

$$
\begin{aligned}
k(\boldsymbol{x}_1, \boldsymbol{x}_2) &= \sum_{i=1}^{N} a_i(\boldsymbol{x}_1^T \boldsymbol{x}_2)^i + a_0 \\
&= \sum_{i=0}^{N} a_i(\boldsymbol{x}_1^T \boldsymbol{x}_2)^i
\end{aligned} \tag{1}
$$

As $\boldsymbol{x}_1^T \boldsymbol{x}_2$ is a valid kernel, $(\boldsymbol{x}_1^T \boldsymbol{x}_2)^i = (\boldsymbol{x}_1^T \boldsymbol{x}_2)(\boldsymbol{x}_1^T \boldsymbol{x}_2)...(\boldsymbol{x}_1^T \boldsymbol{x}_2)$ is also a valid kernel; for $a_i > 0$, the kernel is still preserved; for $a_i = 0$, the whole expression becomes 0, which is also valid.

Q5:

First check the Taylor series of the expression (for $x \in (0, 1)$):

$$\frac{1}{1-x} = \sum_{n=0}^{\infty} x^n$$

thus we can expand the kernel function:

$$
\begin{aligned}
\frac{1}{1 - x_1 x_2} =& \sum_{n=0}^{\infty} (x_1 x_2)^n \\
=& 1 + (x_1 x_2) + (x_1 x_2)^2 + ... + (x_1 x_2)^\infty
\end{aligned} \tag{2}
$$

We can express this use a basis function

$$\Phi(x) = [1 \; x \; x^2 \; x^3 \; ... \; x^\infty]^T$$

thus, according to the definition, the kernel can be expressed with this function:

$$\frac{1}{1 - x_1 x_2} = \Phi(x_1)^T \Phi(x_2)$$

Q6:

a)

Find the total number of duplicated characters in string y and x (include the duplication times of a single character).

b)

We define a basis function:

$$\Phi(m) = (s_1 \;\; s_2 \;\; s_3 \;\; ... \; s_v)^T$$

for $i \in (1, 2, 3, .., v)$, $s_i \in \mathbb{Z}_0$. s_i denotes the times of the character $a_i \in S$ (preserves the order in S) in string "m" shows. Thus the kernel function can be expressed with:

$$k(x, y) = \Phi(x)^T \Phi(y) = x_1 y_1 + x_2 y_2 + x_3 y_3 + ... + x_v y_v$$

thus this kernel is valid.

Gaussian kernel

Q7:

Consider $y_i$can be chosen from set$(1, -1)$, the classification equation is:

$$y_i(\sum_i \alpha_i y_i k_G(x_1, x_2) + b) > 0$$

Assume $b = 0$

$$\sum_i \alpha_i y_i^2 k_G(x_1, x_2) > 0$$

Let α strictly larger than zero, and since $y_i^2 > 0$, we only need $K_G(x_1, x_2) > 0$ to make sure the data is correctly classified.
As the kernel is in an exponential form, thus the condition is fulfilled. Considering the property of computing in a computer, we cannot let σ go close to zero.