SVM Kernel trick and soft-margin SVM

JrPhy

Introduction

- So far SVM can just solve linear separable data, then how can we deal with the nonlinear separable data?
- One is accepting some mistakes, its so called soft-margin SVM.
- The other is using nonlinear function. It’s so-called kernel SVM.
- Here we introduce the polynomial kernel and Gaussian kernel.

Polynomial kernel

- In the original SVM, we calculate $\min_{\alpha \ge 0} \left(\frac{1}{2} \left\| \sum_j \sum_i \alpha_i \alpha_j y_i y_j \vec{x}^T \vec{x}' \right\|^2 - \sum_i \alpha_i \right)$

- The $\boldsymbol{x^T x}$ is inner product, it can be seen as linear, here we take 2nd polynomial as example. Suppose *f* is a function such that *f* : $R^2 \rightarrow R^3$ and dim($\boldsymbol{x}$) = 2, so there are some terms: (1, x_1, x_1, x_1^2, $x_1 x_2$, x_2^2), so the inner product is

$$\vec{x}^T \vec{x}' \to f(x)^T f(x) = 1 + \sum_{i=1}^{2} x_i x_i' + \sum_{i=1}^{2} \sum_{j=1}^{2} x_i x_i' x_j x_j'$$

$$= 1 + \sum_{i=1}^{2} x_i x_i' + \sum_{i=1}^{2} x_i x_i' \sum_{j=1}^{2} x_j x_j' = 1 + \vec{x}^T \vec{x}' + (\vec{x}^T \vec{x}')(\vec{x}^T \vec{x}')$$

Polynomial kernel

- After transforming to 2nd order polynomial, it still calculates the inner product $\boldsymbol{x}^{\boldsymbol{T}}\boldsymbol{x}$.
- So the kernel is **transform + inner product**, here we use the symbol Φ to represent the kernel.
- So all $\boldsymbol{x} \rightarrow \boldsymbol{\Phi}(\boldsymbol{x})$ are what we want to calculate.
- $\min_{\alpha \geq 0} \left(\frac{1}{2} \left\| \sum_j \sum_i \alpha_i \alpha_j y_i y_j \Phi(\vec{x})^T \Phi(\vec{x}') \right\|^2 - \sum_i \alpha_i \right)$, $g_{\mathrm{SVM}}(\boldsymbol{x}) = \mathrm{sign}(\boldsymbol{w}^{\boldsymbol{T}}\, \boldsymbol{\Phi}(\boldsymbol{x}) + \boldsymbol{b})$
- In general the polynomial kernel has the form:
  - $K_Q(\boldsymbol{x}, \boldsymbol{x}') = (\zeta + \gamma \mathbf{x}^{\boldsymbol{T}}\mathbf{x})^Q$, $\gamma > 0$, $\zeta \geq 0$
- If you set $\zeta = \gamma = Q = 1$, it's called linear kernel.

Gaussian kernel

- The other kernel is Gaussian kernel, it uses Gaussian distribution as the basis.
- $\boldsymbol{x^T x'} =$ $\exp(-(x-x')^2) = \exp(-x^2)\exp(-x'^2)\exp(-2xx')$

$$= \exp(-x^2)\exp(-x'^2)\sum_{i=0}^{\infty}\frac{(2xx')^i}{i!}$$

$$= \sum_{i=0}^{\infty}\exp(-x^2)\sqrt{\frac{2^i}{i!}}x^i \exp(-x'^2)\sqrt{\frac{2^i}{i!}}x'^i$$

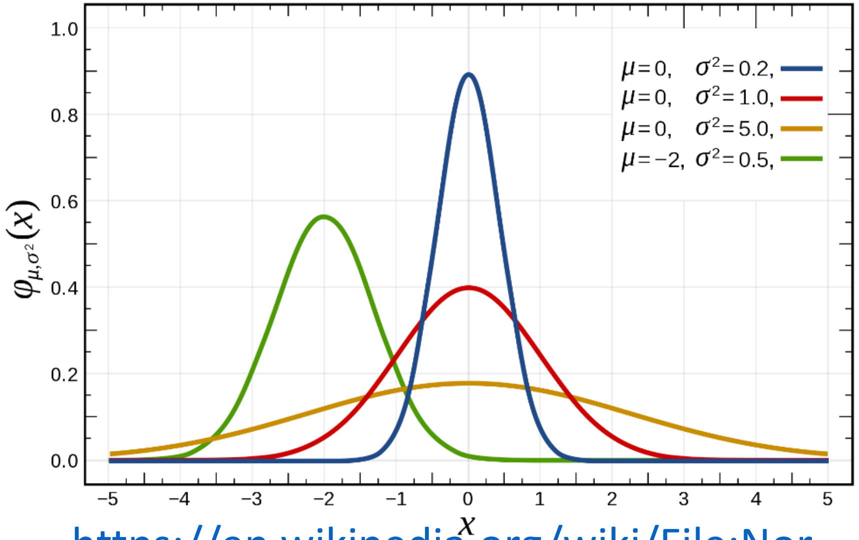
- So Gaussian kernel is $\exp(-x^2)\sqrt{\frac{2^i}{i!}}x^i$

Gaussian kernel

- In general, the Gaussian kernel is written as
  - $K(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\gamma\|\boldsymbol{x} - \boldsymbol{x}'\|^2), \gamma > 0$
- Gaussian distribution:

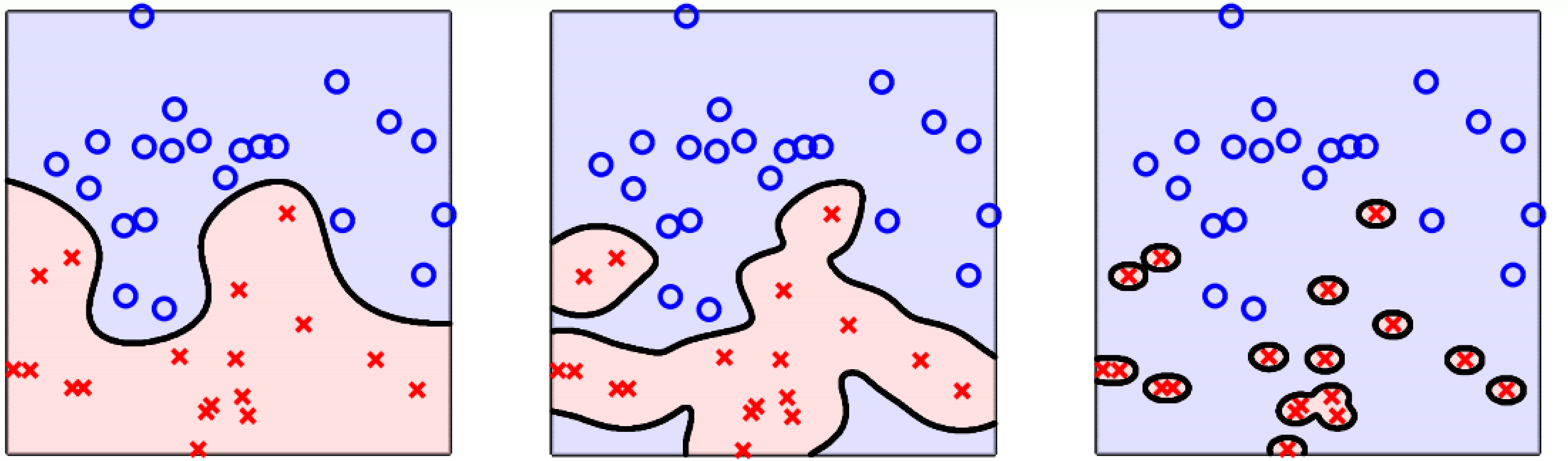
$$N(\mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma}} \exp(-\frac{\| x - \mu \|^2}{2\sigma^2})$$

- The central of the distribution is at $x = \mu$, and the width is determined by σ.
- The central is on the support vector, and the distance is determined by γ,



https://en.wikipedia.org/wiki/File:Normal_Distribution_PDF.svg

Gaussian kernel

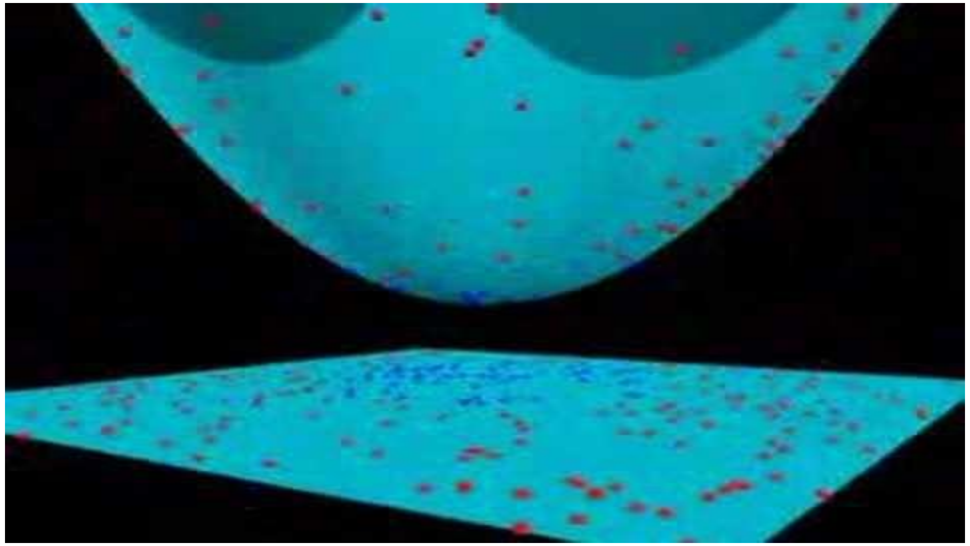


$\exp(-1\|\mathbf{x}-\mathbf{x}'\|^2)$ $\exp(-10\|\mathbf{x}-\mathbf{x}'\|^2)$ $\exp(-100\|\mathbf{x}-\mathbf{x}'\|^2)$

Hsuan Tien Lin, mltech/203_handout

Visualization of kernel transform

- We use kernel to transform our data to higher dimension space, then find a plane to separate the data. Just watch the video below.



- https://www.youtube.com/watch?v=3liCbRZPrZA

Compare

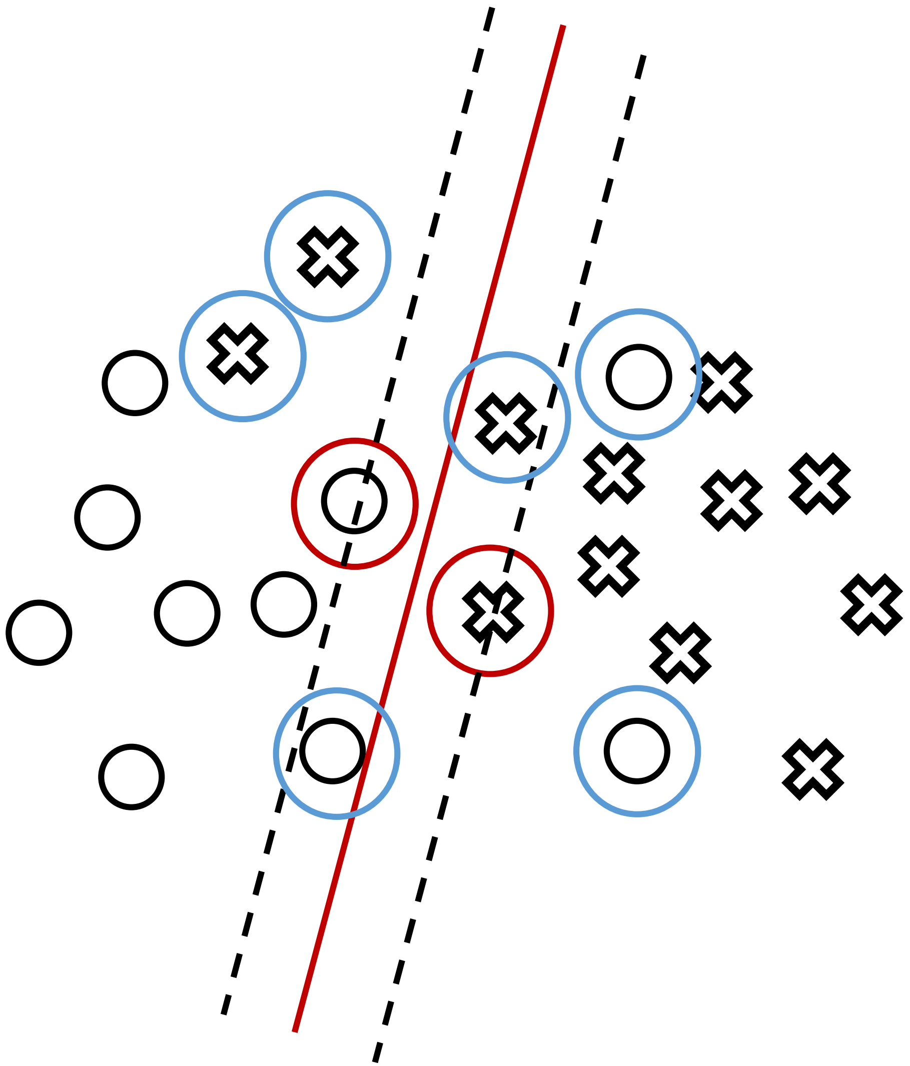
	Linear $K(x, x') = x^T x'$	**Polynomial** $K_Q(x, x') = (\zeta + \gamma x^T x)^Q$, $\gamma > 0, \zeta \geq 0$	**Guass** $K(x, x') = \exp(-\gamma \|x - x'\|^2)$, $\gamma > 0$
Pros	1. Easy to interprets 2. Fast: QP solver 3. Hard to over-fitted	Can use nonlinear separable data.	Most powerfu;
Cons	Can use only linear separable data	1. Slower than linear 2. Not numerical stable if $\|\zeta + \gamma \mathbf{x}^T \mathbf{x}\|$ too big or small	1. Slowest 2. Easy to over-fitted 3. Hard to interprets

Use your own kernel

- Polynomial kernel and Gaussian kernel are used widely, you can use other kernels.
- Kernel is from the inner product, so it satisfies:
    - $K(x, y) = K(y, x)$ and K is semi-definite matrix

Soft-margin SVM

- Another way is to accept some mistakes, or noisy, this is called soft-margin SVM. Then the original SVM is called hard-margin SVM.
- Here we record each mistake as ξ, and we call it penalty.
- I define the area enclosed by SVM is SVM area for convenient.

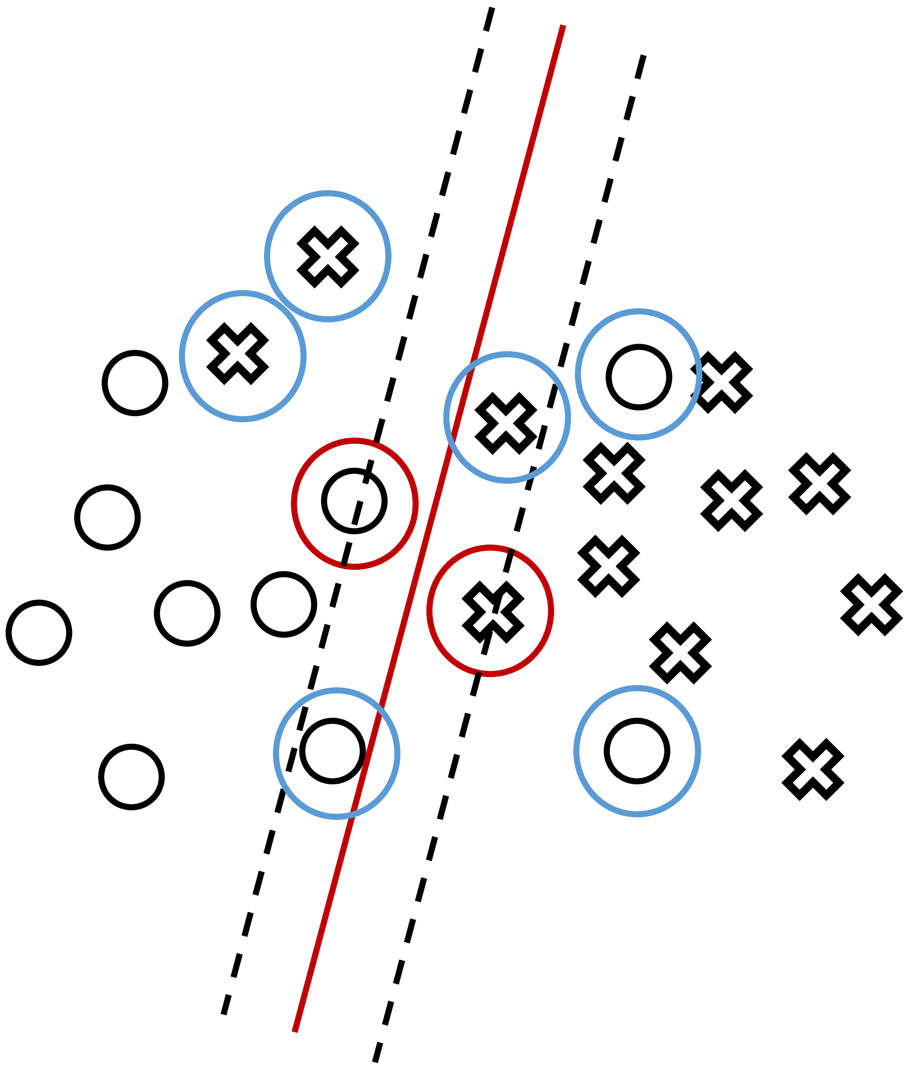


Optimize

$$\min_{b,w} \frac{1}{2}\|w\|^2 \rightarrow \min_{b,w,\xi}\left(\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i\right)$$

- Here C is a parameter to trade-off the width of margin and noisy tolerance.
  - Large C: less noise → narrow width
  - Small C: much noise → wide width
- Condition: $y(\boldsymbol{w}^T\boldsymbol{x} + \boldsymbol{b}) \geq 1$

→ $y(\boldsymbol{w}^T\boldsymbol{x} + \boldsymbol{b}) \geq 1 - \xi_i,\ \xi_i \geq 0$



Optimize

$$L(\alpha, w, b) \rightarrow L(\alpha, w, b, \xi, \beta) = \frac{1}{2}\|w\|^2 + C\sum_i \xi_i + \sum_i \alpha_i[1 - \xi_i - y_i(w^T x + b)] + \sum_i \beta_i \times (-\xi_i)$$

$$\max_{\forall \alpha_i \geq 0} \min_{b,w} L(\alpha, w, b) = \max_{\alpha_n \geq 0, \beta_n \geq 0} \min_{b,w,\xi} L(\alpha, w, b, \xi, \beta)$$

$$\max_{\alpha_n \geq 0, \beta_n \geq 0} \left(\min_{b,w,\xi} \frac{1}{2}\|w\|^2 + C\sum_i \xi_i + \sum_i \alpha_i[1 - \xi_i - y_i(w^T x + b)] + \sum_i \beta_i \times (-\xi_i) \right)$$

$$\frac{\partial}{\partial \xi} L(\alpha, w, b, \xi, \beta) = 0 = C - \alpha_i - \beta_i \rightarrow \beta_i = C - \alpha_i \geq 0 \rightarrow \boxed{C \geq \alpha_i \geq 0, \forall i = 1, 2, 3, \ldots}$$

$$L(\alpha, w, b, \xi, \beta) = \frac{1}{2}\|w\|^2 + C\sum_i \xi_i + \sum_i \alpha_i[1 - \xi_i - y_i(w^T x + b)] + \sum_i (C - \alpha_i) \times (-\xi_i)$$

$$= \frac{1}{2}\|w\|^2 - \sum_i \alpha_i[\xi_i + y_i(w^T x + b)] + \sum_i (C - \alpha_i) \times (-\xi_i) - \sum_i (C - \alpha_i) \times (-\xi_i)$$

$$\boxed{= \frac{1}{2}\|w\|^2 + \sum_i \alpha_i[1 - y_i(w^T x + b)]}$$

Optimize

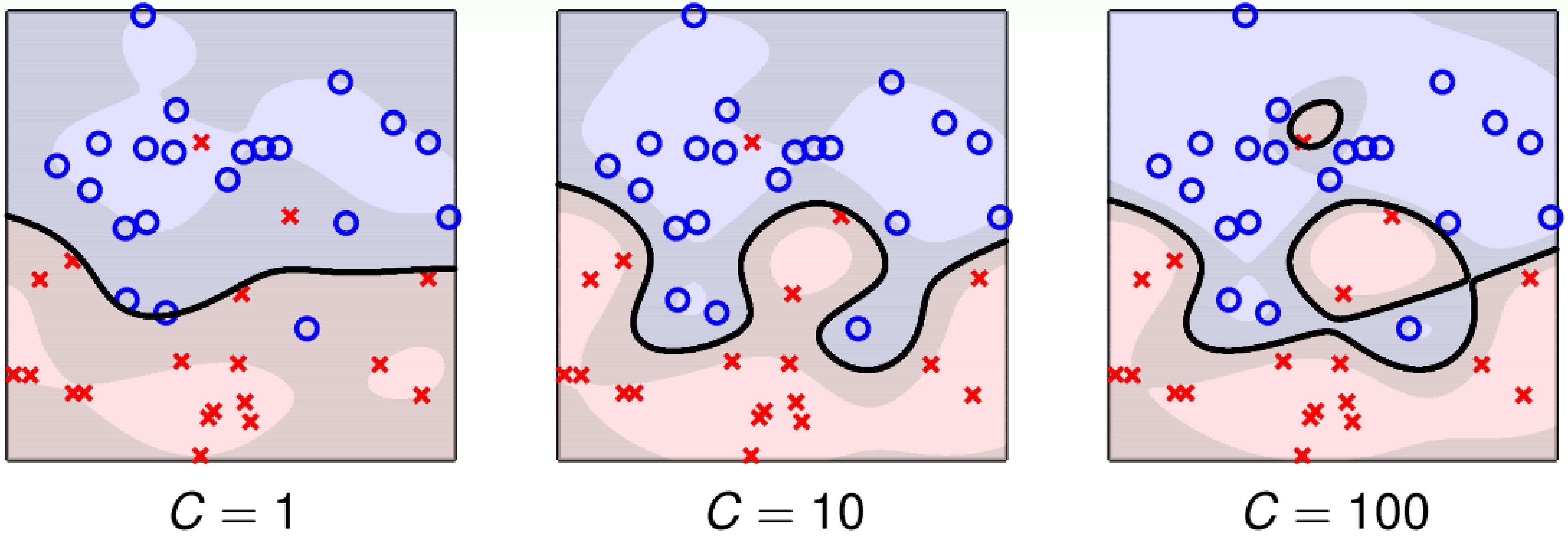
- So the Lagrange is totally the same as the hard-margin SVM, but the constrains are more.
- In hard-margin SVM, $\alpha_i > 0$, but in soft-margin SVM, $C \geq \alpha_i \geq 0$, so that **soft-margin SVM = hard-margin SVM + more constrains**
- Then combine the kernel trick, here we still solve
- $\min_{\alpha \geq 0} \left(\frac{1}{2} \left\| \sum_j \sum_i \alpha_i \alpha_j y_i y_j \Phi(\vec{x})^T \Phi(\vec{x}') \right\|^2 - \sum_i \alpha_i \right)$, $g_{SVM}(\boldsymbol{x}) = \text{sign}(\boldsymbol{w}^{\boldsymbol{T}}\, \boldsymbol{\Phi}(\boldsymbol{x}) + \boldsymbol{b})$
- with $\mathbf{C} \geq \boldsymbol{\alpha_i} \geq \mathbf{0}$

More about the constrain

- In the hard-margin SVM, we just care about those points which α_i do not equal to 0, and no point in the area. But soft-margin SVM allow some points in the area.
- Suppose C is determined, then there are 4 types

$\alpha_i < C, \xi = 0$	This equals to hard-margin SVM, so the point in on the boundary.
$\alpha_i = C, 1 > \xi > 0$	The point is at the correct part, but in the SVM area.
$\alpha_i = C, \xi = 1$	The point is as the hyper plane.
$\alpha_i = C, \xi > 1$	The point is in the incorrect part.

Soft-Margin Gaussian SVM in Action



Hsuan Tien Lin, mltech/203_handout