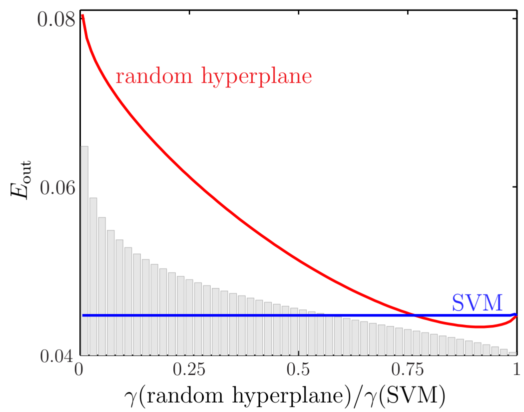
Learning From Data
Lecture 25
The Kernel Trick

Learning with only inner products
The Kernel

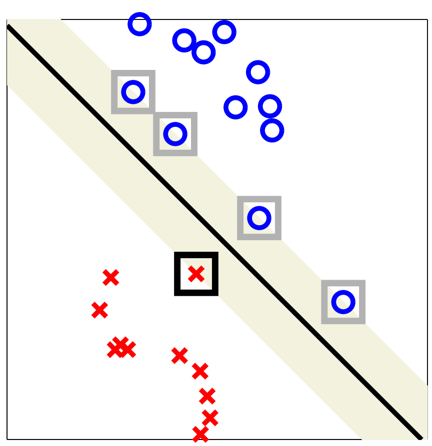
M. Magdon-Ismail
CSCI 4100/6100

RECAP: # Large Margin is Better

Controling Overfitting

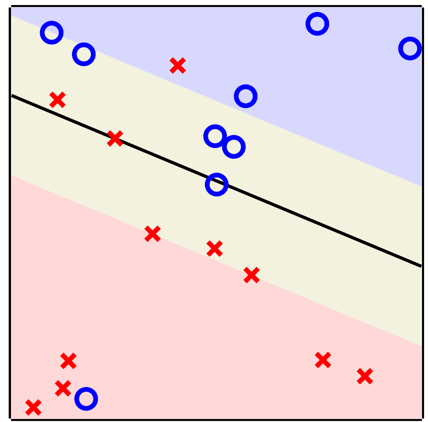


Theorem. $d_{\text{VC}}(\gamma) \le \left\lceil \frac{R^2}{\gamma^2} \right\rceil + 1$

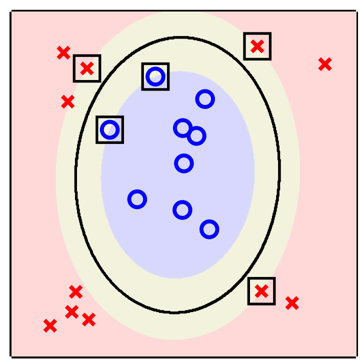


$$E_{\text{cv}} \le \frac{\#\ \text{support vectors}}{N}$$

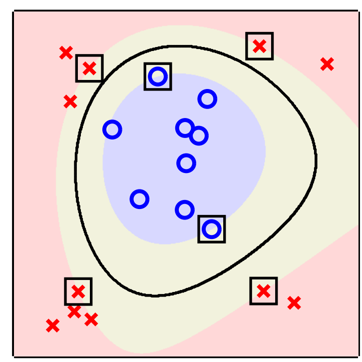
Non-Separable Data



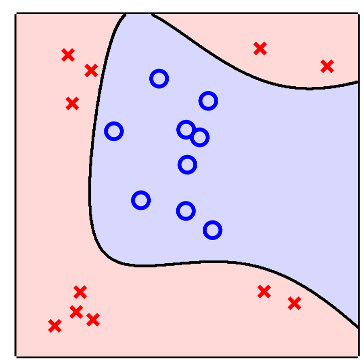
$$\begin{aligned} \underset{b,\mathbf{w},\boldsymbol{\xi}}{\text{minimize}} \quad & \tfrac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + C\textstyle\sum_{n=1}^{N}\xi_n \\ \text{subject to:} \quad & y_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b) \ge 1 - \xi_n \\ & \xi_n \ge 0 \qquad\qquad \text{for } n = 1, \ldots, N \end{aligned}$$



Φ_2 + SVM



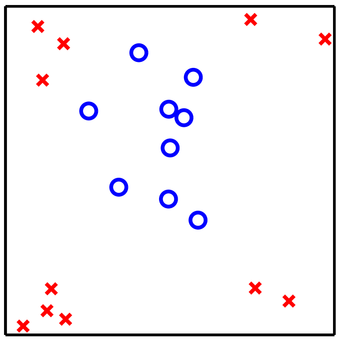
Φ_3 + SVM



Φ_3 + pseudoinverse algorithm

Complex hypothesis that does not overfit because it is 'simple', controlled by only a few support vectors.

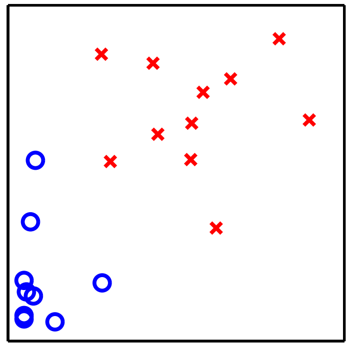
Recall: Mechanics of the Nonlinear Transform



1. Original data

$\mathbf{x}_n \in \mathcal{X}$

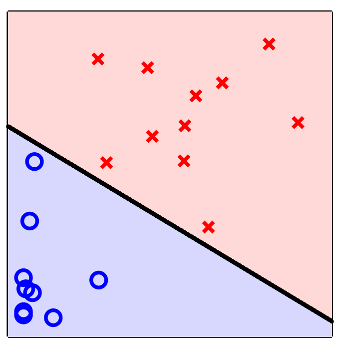
$\xrightarrow{\ \mathbf{\Phi}\ }$



2. Transform the data

$\mathbf{z}_n = \Phi(\mathbf{x}_n) \in \mathcal{Z}$

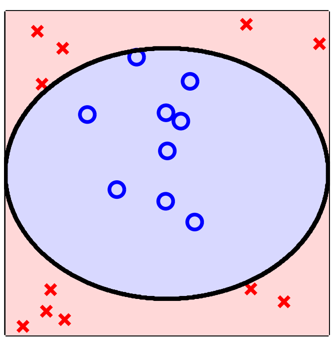
$\downarrow$



3. Separate data in $\mathcal{Z}$-space

$\tilde{g}(\mathbf{z}) = \text{sign}(\tilde{\mathbf{w}}^{\text{T}}\mathbf{z})$

$\xleftarrow{\ `\mathbf{\Phi}^{-1}\text{'}\ }$



4. Classify in $\mathcal{X}$-space

$g(\mathbf{x}) = \tilde{g}(\Phi(\mathbf{x})) = \text{sign}(\tilde{\mathbf{w}}^{\text{T}}\Phi(\mathbf{x}))$

$\underline{\mathcal{X}\text{-space is } \mathbb{R}^d}$	$\underline{\mathcal{Z}\text{-space is } \mathbb{R}^{\tilde{d}}}$
$\mathbf{x} = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix}$	$\mathbf{z} = \mathbf{\Phi}(\mathbf{x}) = \begin{bmatrix} 1 \\ \Phi_1(\mathbf{x}) \\ \vdots \\ \Phi_{\tilde{d}}(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} 1 \\ z_1 \\ \vdots \\ z_{\tilde{d}} \end{bmatrix}$
$\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$	$\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N$
$y_1, y_2, \ldots, y_N$	$y_1, y_2, \ldots, y_N$
no weights	$\tilde{\mathbf{w}} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{\tilde{d}} \end{bmatrix}$
$d_{\text{VC}} = d + 1$	$\boldsymbol{d_{\text{VC}} = d + 1}$
$g(\mathbf{x}) = \text{sign}(\tilde{\mathbf{w}}^{\text{T}}\mathbf{\Phi}(\mathbf{x}))$	

Have to **transform** the data to the $\mathcal{Z}$-space.

This Lecture

How to use nonlinear transforms without **physically transforming** data to $\mathcal{Z}$-space.

Primal Versus Dual

Primal

$$\underset{b,\mathbf{w}}{\text{minimize}} \quad \tfrac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}$$

$$\text{subject to: } y_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b) \geq 1 \quad \text{for } n = 1, \ldots, N$$

$$g(\mathbf{x}) = \text{sign}(\mathbf{w}^{\mathrm{T}}\mathbf{x} + b)$$

$d+1$ optimization variables $\mathbf{w}, b$

Dual

$$\underset{\boldsymbol{\alpha}}{\text{minimize}} \quad \tfrac{1}{2}\sum_{n,m=1}^{N} \alpha_n\alpha_m y_n y_m(\mathbf{x}_n^{\mathrm{T}}\mathbf{x}_m) - \sum_{n=1}^{N}\alpha_n$$

$$\text{subject to: } \sum_{n=1}^{N}\alpha_n y_n = 0$$

$$\alpha_n \geq 0 \quad \text{for } n = 1, \ldots, N$$

$$\begin{aligned} \mathbf{w}^* &= \sum_{n=1}^{N}\alpha_n^* y_n \mathbf{x}_n \\ b^* &= y_s - \mathbf{w}^{\mathrm{T}}\mathbf{x}_s \qquad (\alpha_s^* > 0) \end{aligned}$$

support vectors

$$\begin{aligned} g(\mathbf{x}) &= \text{sign}(\mathbf{w}^{*\mathrm{T}}\mathbf{x} + b^*) \\ &= \text{sign}\left(\sum_{n=1}^{N}\alpha_n^* y_n \mathbf{x}_n^{\mathrm{T}}(\mathbf{x} - \mathbf{x}_s) + y_s\right) \end{aligned}$$

N optimization variables $\boldsymbol{\alpha}$

Primal Versus Dual - Matrix Vector Form

Primal

$$\underset{b,\mathbf{w}}{\text{minimize}} \quad \tfrac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}$$

subject to: $y_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b) \geq 1 \quad \text{for } n = 1, \ldots, N$

$$g(\mathbf{x}) = \text{sign}(\mathbf{w}^{\mathrm{T}}\mathbf{x} + b)$$

$d + 1$ optimization variables $\mathbf{w}, b$

Dual

$$\underset{\boldsymbol{\alpha}}{\text{minimize}} \quad \tfrac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\mathrm{G}\boldsymbol{\alpha} - \mathbf{1}^{\mathrm{T}}\boldsymbol{\alpha} \qquad (\mathrm{G}_{nm} = y_n y_m \mathbf{x}_n^{\mathrm{T}}\mathbf{x}_m)$$

subject to: $\mathbf{y}^{\mathrm{T}}\boldsymbol{\alpha} = 0$

$$\boldsymbol{\alpha} \geq \mathbf{0}$$

$$\mathbf{w}^* = \sum_{n=1}^{N} \alpha_n^* y_n \mathbf{x}_n$$

support vectors

$$b^* = y_s - \mathbf{w}^{\mathrm{T}}\mathbf{x}_s \qquad (\alpha_s^* > 0)$$

$$\begin{aligned} g(\mathbf{x}) &= \text{sign}(\mathbf{w}^{*\mathrm{T}}\mathbf{x} + b^*) \\ &= \text{sign}\left(\sum_{n=1}^{N} \alpha_n^* y_n \mathbf{x}_n^{\mathrm{T}}(\mathbf{x} - \mathbf{x}_s) + y_s\right) \end{aligned}$$

N optimization variables $\boldsymbol{\alpha}$

Deriving the Dual: The Lagrangian

$$\mathcal{L} = \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + \sum_{n=1}^{N} \underbrace{\alpha_n}_{\text{lagrange multipliers}} \cdot \underbrace{(1 - y_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b))}_{\text{the constraints}}$$

minimize w.r.t. $b, \mathbf{w}$ ← unconstrained

maximize w.r.t. $\boldsymbol{\alpha} \geq \mathbf{0}$

Formally: use KKT conditions to transform the primal.

Intuition

- $1 - y_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b) > 0 \implies \alpha_n \to \infty$ gives $\mathcal{L} \to \infty$
- Choose $(b, \mathbf{w})$ to min $\mathcal{L}$, so $1 - y_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b) \leq 0$
- $1 - y_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b) \underbrace{<}_{\text{non support vectors}} 0 \implies \alpha_n = 0$ (max $\mathcal{L}$ w.r.t. α_n)

Conclusion

At the optimum, $\alpha_n(y_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b) - 1) = 0$, so

$$\mathcal{L} = \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}$$

is minimized and the constraints are satisfied

$$1 - y_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b) \leq 0$$

Unconstrained Minimization w.r.t. $(b, \mathbf{w})$

$$\mathcal{L} = \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} - \sum_{n=1}^{N} \alpha_n \cdot (y_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b) - 1)$$

Set $\frac{\partial \mathcal{L}}{\partial b} = 0$:

$$\frac{\partial \mathcal{L}}{\partial b} = \sum_{n=1}^{N} \alpha_n y_n \qquad \Longrightarrow \qquad \sum_{n=1}^{N} \alpha_n y_n = 0$$

Set $\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = 0$:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \mathbf{w} - \sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n \qquad \Longrightarrow \qquad \mathbf{w} = \sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n$$

Substitute into $\mathcal{L}$ to maximize w.r.t. $\boldsymbol{\alpha} \geq \mathbf{0}$

$$\begin{aligned} \mathcal{L} &= \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} - \mathbf{w}^{\mathrm{T}}\sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n - b\sum_{n=1}^{N} \alpha_n y_n + \sum_{n=1}^{N} \alpha_n \\ &= -\frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + \sum_{n=1}^{N} \alpha_n \\ &= -\frac{1}{2}\sum_{m,n=1}^{N} \alpha_n \alpha_m y_n y_m \mathbf{x}_n^{\mathrm{T}}\mathbf{x}_m + \sum_{n=1}^{N} \alpha_n \end{aligned}$$

$$\begin{aligned} \underset{\boldsymbol{\alpha}}{\text{minimize}} \quad & \tfrac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\mathrm{G}\boldsymbol{\alpha} - \mathbf{1}^{\mathrm{T}}\boldsymbol{\alpha} \\ \text{subject to:} \quad & \mathbf{y}^{\mathrm{T}}\boldsymbol{\alpha} = 0 \\ & \boldsymbol{\alpha} \geq \mathbf{0} \end{aligned}$$

$(\mathrm{G}_{nm} = y_n y_m \mathbf{x}_n^{\mathrm{T}}\mathbf{x}_m)$

$$\mathbf{w} = \textstyle\sum_{n=1}^{N} \alpha_n^* y_n \mathbf{x}_n$$

$$\begin{aligned} \alpha_s > 0 \quad &\Longrightarrow \quad y_s(\mathbf{w}^{\mathrm{T}}\mathbf{x}_s + b) - 1 = 0 \\ &\Longrightarrow \quad b = y_s - \mathbf{w}^{\mathrm{T}}\mathbf{x}_s \end{aligned}$$

Example — Our Toy Data Set

signed data matrix
$\downarrow$

$$\mathrm{X} = \begin{bmatrix} 0 & 0 \\ 2 & 2 \\ 2 & 0 \\ 3 & 0 \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} -1 \\ -1 \\ +1 \\ +1 \end{bmatrix} \quad \longrightarrow \quad \mathrm{X}_s = \begin{bmatrix} 0 & 0 \\ -2 & -2 \\ 2 & 0 \\ 3 & 0 \end{bmatrix} \quad \longrightarrow \quad \mathrm{G} = \mathrm{X}_s \mathrm{X}_s^{\mathrm{T}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 8 & -4 & -6 \\ 0 & -4 & 4 & 6 \\ 0 & -6 & 6 & 9 \end{bmatrix}$$

Quadratic Programming

$$\begin{array}{ll} \underset{\mathbf{u}}{\text{minimize}} & \frac{1}{2}\mathbf{u}^{\mathrm{T}}\mathrm{Q}\mathbf{u} + \mathbf{p}^{\mathrm{T}}\mathbf{z} \\ \text{subject to:} & \mathrm{A}\mathbf{u} \geq \mathbf{c} \end{array}$$

Dual SVM

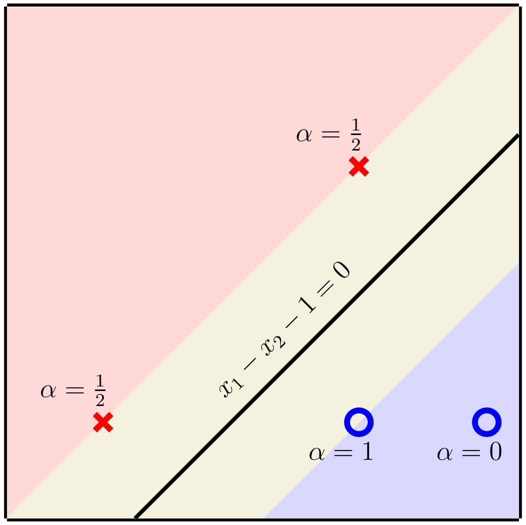
$$\begin{array}{ll} \underset{\boldsymbol{\alpha}}{\text{minimize}} & \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\mathrm{G}\boldsymbol{\alpha} - \mathbf{1}^{\mathrm{T}}\boldsymbol{\alpha} \\ \text{subject to:} & \mathbf{y}^{\mathrm{T}}\boldsymbol{\alpha} = 0 \\ & \boldsymbol{\alpha} \geq \mathbf{0} \end{array}$$

$$\left.\begin{array}{c} \mathbf{u} = \boldsymbol{\alpha} \\ \mathrm{Q} = \mathrm{G} \\ \mathbf{p} = -\mathbf{1}_N \\ \mathrm{A} = \begin{bmatrix} \mathbf{y}^{\mathrm{T}} \\ -\mathbf{y}^{\mathrm{T}} \\ \mathrm{I}_N \end{bmatrix} \\ \mathbf{c} = \begin{bmatrix} 0 \\ 0 \\ \mathbf{0}_N \end{bmatrix} \end{array}\right\} \xrightarrow{\mathrm{QP}(\mathrm{Q},\mathbf{p},\mathrm{A},\mathbf{c})} \quad \boldsymbol{\alpha}^* = \begin{bmatrix} \frac{1}{2} \\ \frac{1}{2} \\ 1 \\ 0 \end{bmatrix}$$

$$\mathbf{w} = \sum_{n=1}^{4} \alpha_n^* y_n \mathbf{x}_n = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

$$b = y_1 - \mathbf{w}^{\mathrm{T}}\mathbf{x}_1 = -1$$

$$\gamma = \frac{1}{\|\mathbf{w}\|} = \frac{1}{\sqrt{2}}$$



non-support vectors $\implies$ $\alpha_n = 0$
only support vectors can have $\alpha_n > 0$

Dual QP Algorithm for Hard Margin linear-SVM

1: **Input:** $\mathrm{X}, \mathbf{y}$.

2: Let $\mathbf{p} = -\mathbf{1}_N$ be the N-vector of ones and $\mathbf{c} = \mathbf{0}_{N+2}$ the N-vector of zeros. Construct matrices Q and A, where

$$\underbrace{\mathrm{X}_\mathrm{s} = \begin{bmatrix} —y_1\mathbf{x}_1^{\mathrm{T}}— \\ \vdots \\ —y_N\mathbf{x}_N^{\mathrm{T}}— \end{bmatrix}}_{\text{signed data matrix}}, \qquad \mathrm{Q} = \mathrm{X}_s\mathrm{X}_s^{\mathrm{T}}, \qquad \mathrm{A} = \begin{bmatrix} \mathbf{y}^{\mathrm{T}} \\ -\mathbf{y}^{\mathrm{T}} \\ \mathrm{I}_{N\times N} \end{bmatrix}$$

3: $\boldsymbol{\alpha}^* \leftarrow \mathsf{QP}(\mathrm{Q}, \mathbf{c}, \mathrm{A}, \mathbf{a})$.

4: Return

$$\mathbf{w}^* = \sum_{\alpha_n^*>0} \alpha_n^* y_n \mathbf{x}_n$$

$$b^* = y_s - \mathbf{w}^{\mathrm{T}}\mathbf{x}_s \qquad (\alpha_s^* > 0)$$

5: The final hypothesis is $g(\mathbf{x}) = \mathrm{sign}(\mathbf{w}^{*\mathrm{T}}\mathbf{x} + b^*)$.

$$\begin{array}{ll} \underset{\boldsymbol{\alpha}}{\text{minimize}} & \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\mathrm{G}\boldsymbol{\alpha} - \mathbf{1}^{\mathrm{T}}\boldsymbol{\alpha} \\ \text{subject to:} & \mathbf{y}^{\mathrm{T}}\boldsymbol{\alpha} = 0 \\ & \boldsymbol{\alpha} \geq \mathbf{0} \end{array}$$

$\uparrow$

Some packages allow equality and bound constraints to directly solve this type of QP

Primal Versus Dual (Non-Separable)

Primal

$$\underset{b,\mathbf{w},\boldsymbol{\xi}}{\text{minimize}} \quad \tfrac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + C\sum_{n=1}^{N}\xi_n$$

$$\text{subject to: } y_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b) \geq 1 - \xi_n$$

$$\xi_n \geq 0 \quad \text{for } n = 1, \ldots, N$$

$$g(\mathbf{x}) = \text{sign}(\mathbf{w}^{\mathrm{T}}\mathbf{x} + b)$$

$N + d + 1$ optimization variables $b, \mathbf{w}, \boldsymbol{\xi}$

Dual

$$\underset{\boldsymbol{\alpha}}{\text{minimize}} \quad \tfrac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\mathrm{G}\boldsymbol{\alpha} - \mathbf{1}^{\mathrm{T}}\boldsymbol{\alpha}$$

$$\text{subject to: } \mathbf{y}^{\mathrm{T}}\boldsymbol{\alpha} = 0$$

$$\mathbf{C} \geq \boldsymbol{\alpha} \geq \mathbf{0}$$

$$\begin{aligned} \mathbf{w}^* &= \sum_{n=1}^{N} \alpha_n^* y_n \mathbf{x}_n \\ b^* &= y_s - \mathbf{w}^{\mathrm{T}}\mathbf{x}_s \qquad (C > \alpha_s^* > 0) \end{aligned}$$

$$\begin{aligned} g(\mathbf{x}) &= \text{sign}(\mathbf{w}^{*\mathrm{T}}\mathbf{x} + b^*) \\ &= \text{sign}\left(\sum_{n=1}^{N} \alpha_n^* y_n \mathbf{x}_n^{\mathrm{T}}(\mathbf{x} - \mathbf{x}_s) + y_s\right) \end{aligned}$$

N optimization variables $\boldsymbol{\alpha}$

Dual SVM is an Inner Product Algorithm

$\underline{\mathcal{X}\text{-Space}}$

$$\underset{\boldsymbol{\alpha}}{\text{minimize}} \quad \tfrac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\mathrm{G}\boldsymbol{\alpha} - \mathbf{1}^{\mathrm{T}}\boldsymbol{\alpha}$$

$$\text{subject to: } \mathbf{y}^{\mathrm{T}}\boldsymbol{\alpha} = 0$$

$$\mathbf{C} \geq \boldsymbol{\alpha} \geq \mathbf{0}$$

$$G_{nm} = y_n y_m (\mathbf{x}_n^{\mathrm{T}}\mathbf{x}_m)$$

$$g(\mathbf{x}) = \text{sign}\left(\sum_{\alpha_n^*>0} \alpha_n^* y_n (\mathbf{x}_n^{\mathrm{T}}\mathbf{x}) + b^*\right)$$

$$C > \alpha_s^* > 0$$

$$b^* = y_s - \sum_{\alpha_n^*>0} \alpha_n^* y_n (\mathbf{x}_n^{\mathrm{T}}\mathbf{x}_s)$$

Dual SVM is an Inner Product Algorithm

$\underline{\mathcal{Z}\text{-Space}}$

$$\underset{\boldsymbol{\alpha}}{\text{minimize}} \quad \tfrac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\mathrm{G}\boldsymbol{\alpha} - \mathbf{1}^{\mathrm{T}}\boldsymbol{\alpha}$$

$$\text{subject to: } \mathbf{y}^{\mathrm{T}}\boldsymbol{\alpha} = 0$$

$$\mathbf{C} \geq \boldsymbol{\alpha} \geq \mathbf{0}$$

$$G_{nm} = y_n y_m(\mathbf{z}_n^{\mathrm{T}}\mathbf{z}_m)$$

$$g(\mathbf{x}) = \text{sign}\left(\sum_{\alpha_n^*>0} \alpha_n^* y_n(\mathbf{z}_n^{\mathrm{T}}\mathbf{z}) + b^*\right)$$

$$C > \alpha_s^* > 0$$

$$b^* = y_s - \sum_{\alpha_n^*>0} \alpha_n^* y_n(\mathbf{z}_n^{\mathrm{T}}\mathbf{z}_s)$$

Dual SVM is an Inner Product Algorithm

$\underline{\mathcal{Z}\text{-Space}}$

$$\underset{\boldsymbol{\alpha}}{\text{minimize}} \quad \tfrac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\mathrm{G}\boldsymbol{\alpha} - \mathbf{1}^{\mathrm{T}}\boldsymbol{\alpha}$$

$$\text{subject to: } \mathbf{y}^{\mathrm{T}}\boldsymbol{\alpha} = 0$$

$$\mathbf{C} \geq \boldsymbol{\alpha} \geq \mathbf{0}$$

$$G_{nm} = y_n y_m(\mathbf{z}_n^{\mathrm{T}}\mathbf{z}_m)$$

$$g(\mathbf{x}) = \text{sign}\left(\sum_{\alpha_n^*>0} \alpha_n^* y_n(\mathbf{z}_n^{\mathrm{T}}\mathbf{z}) + b\right)$$

$$C > \alpha_s^* > 0$$

$$b = y_s - \sum_{\alpha_n^*>0} \alpha_n^* y_n(\mathbf{z}_n^{\mathrm{T}}\mathbf{z}_s)$$

Can we compute $\mathbf{z}^{\mathrm{T}}\mathbf{z}'$ without needing $\mathbf{z} = \Phi(\mathbf{x})$ to visit $\mathcal{Z}$-space?

The Kernel $K(\cdot,\cdot)$ for a Transform $\Phi(\cdot)$

The Kernel tells you how to compute the inner product in $\mathcal{Z}$-space

$$K(\mathbf{x},\mathbf{x}') = \Phi(\mathbf{x})^{\mathrm{T}}\Phi(\mathbf{x}') = \mathbf{z}^{\mathrm{T}}\mathbf{z}'$$

Example: 2nd-order polynomial transform

$$\Phi(\mathbf{x}) = \begin{bmatrix} x_1 \\ x_2 \\ x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{bmatrix}$$

$$\begin{aligned} K(\mathbf{x},\mathbf{x}') &= \Phi(\mathbf{x})^{\mathrm{T}}\Phi(\mathbf{x}') = \begin{bmatrix} x_1 \\ x_2 \\ x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{bmatrix} \bullet \begin{bmatrix} x_1' \\ x_2' \\ {x_1'}^2 \\ \sqrt{2}x_1'x_2' \\ {x_2'}^2 \end{bmatrix} \qquad \leftarrow O(d^2) \\ &= x_1x_1' + x_2x_2' + x_1^2{x_1'}^2 + 2x_1x_2x_1'x_2' + x_2^2{x_2'}^2 \\ &= \left(\frac{1}{2} + \mathbf{x}^{\mathrm{T}}\mathbf{x}'\right)^2 - \frac{1}{4} \end{aligned}$$

$\uparrow$
computed quickly
in $\mathcal{X}$-space, in $O(d)$

The Gaussian Kernel is Infinite-Dimensional

$$K(\mathbf{x}, \mathbf{x}') = e^{-\gamma \|\mathbf{x}-\mathbf{x}'\|^2}$$

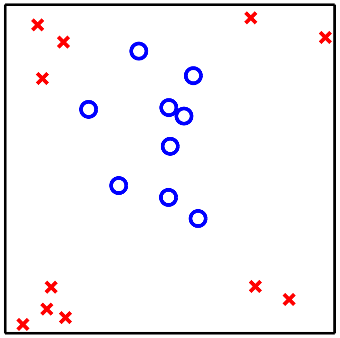
Example: Gaussian Kernel in 1-dimension

$$\Phi(\mathbf{x}) = \begin{bmatrix} e^{-x^2}\sqrt{\frac{2^0}{0!}} \\ e^{-x^2}\sqrt{\frac{2^1}{1!}}x \\ e^{-x^2}\sqrt{\frac{2^2}{2!}}x^2 \\ e^{-x^2}\sqrt{\frac{2^3}{3!}}x^3 \\ e^{-x^2}\sqrt{\frac{2^4}{4!}}x^4 \\ \vdots \end{bmatrix}$$

(infinite dimensional Φ)

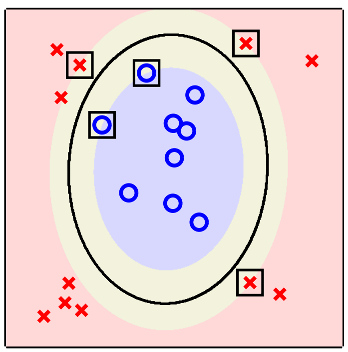
$$\begin{aligned} K(x, x') &= \Phi(x)^{\mathrm{T}}\Phi(x') = \begin{bmatrix} e^{-x^2}\sqrt{\frac{2^0}{0!}} \\ e^{-x^2}\sqrt{\frac{2^1}{1!}}x \\ e^{-x^2}\sqrt{\frac{2^2}{2!}}x^2 \\ e^{-x^2}\sqrt{\frac{2^3}{3!}}x^3 \\ e^{-x^2}\sqrt{\frac{2^4}{4!}}x^4 \\ \vdots \end{bmatrix} \bullet \begin{bmatrix} e^{-x'^2}\sqrt{\frac{2^0}{0!}} \\ e^{-x'^2}\sqrt{\frac{2^1}{1!}}x' \\ e^{-x'^2}\sqrt{\frac{2^2}{2!}}x'^2 \\ e^{-x'^2}\sqrt{\frac{2^3}{3!}}x'^3 \\ e^{-x'^2}\sqrt{\frac{2^4}{4!}}x'^4 \\ \vdots \end{bmatrix} \\ &= e^{-x^2}e^{-x'^2}\sum_{i=0}^{\infty}\frac{(2xx')^i}{i!} \\ &= e^{-(x-x')^2} \end{aligned}$$

The Kernel Allows Us to Bypass $\mathcal{Z}$-space



$\mathbf{x}_n \in \mathcal{X}$

$$\Big\downarrow K(\cdot,\cdot)$$



$$g(\mathbf{x}) = \text{sign}\left(\sum_{\alpha_n^*>0} \alpha_n^* y_n K(\mathbf{x}_n, \mathbf{x}) + b^*\right)$$

$$b^* = y_s - \sum_{\alpha_n^*>0} \alpha_n^* y_n K(\mathbf{x}_n, \mathbf{x}_s)$$

1: **Input:** X, $\mathbf{y}$, regularization parameter C

2: Compute G: $\mathrm{G}_{nm} = y_n y_m K(\mathbf{x}_n, \mathbf{x}_m)$.

3: Solve (QP):

$$\left.\begin{array}{ll} \underset{\boldsymbol{\alpha}}{\text{minimize:}} & \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\mathrm{G}\boldsymbol{\alpha} - \mathbf{1}^{\mathrm{T}}\boldsymbol{\alpha} \\ \text{subject to:} & \mathbf{y}^{\mathrm{T}}\boldsymbol{\alpha} = 0 \\ & \mathbf{C} \geq \boldsymbol{\alpha} \geq \mathbf{0} \end{array}\right\} \longrightarrow \begin{array}{l} \boldsymbol{\alpha}^* \\ \text{index } s : C > \alpha_s^* > 0 \end{array}$$

4: $$b^* = y_s - \sum_{\alpha_n^*>0} \alpha_n^* y_n K(\mathbf{x}_n, \mathbf{x}_s)$$

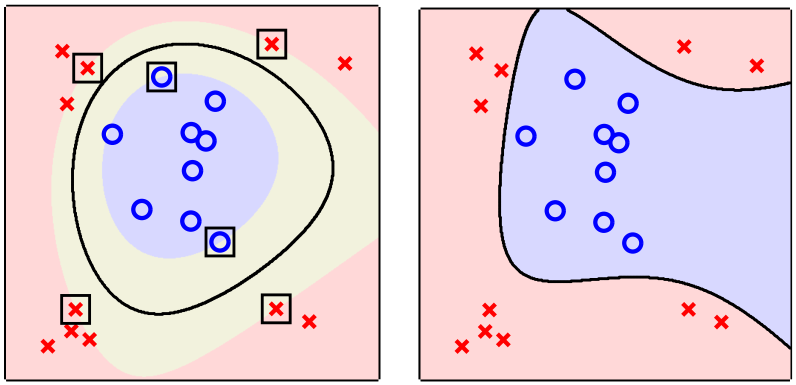
5: The final hypothesis is

$$g(\mathbf{x}) = \text{sign}\left(\sum_{\alpha_n^*>0} \alpha_n^* y_n K(\mathbf{x}_n, \mathbf{x}) + b^*\right)$$

The Kernel-Support Vector Machine

Overfitting

SVM Regression



high $\tilde{d} \rightarrow$ complicated separator

small # support vectors $\rightarrow$ low effective complexity

Can go to high (infinite) $\tilde{d}$

Computation

Inner products with Kernel

$$K(\cdot, \cdot)$$

high $\tilde{d} \rightarrow$ expensive or infeasible computation

kernel $\rightarrow$ computationally feasible to go to high $\tilde{d}$

Can go to high (infinite) $\tilde{d}$