Machine Learning

Lecture 12: Support Vector Machine

王浩

信息科学与技术学院

Email: wanghao1@shanghaitech.edu.cn

本节内容

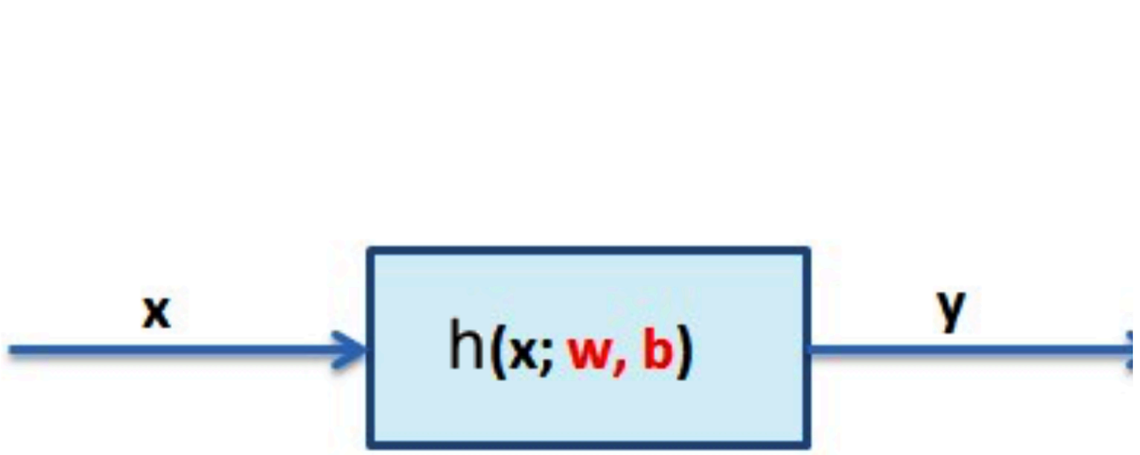
- SVM
- soft-margin
- dual
- Kernel

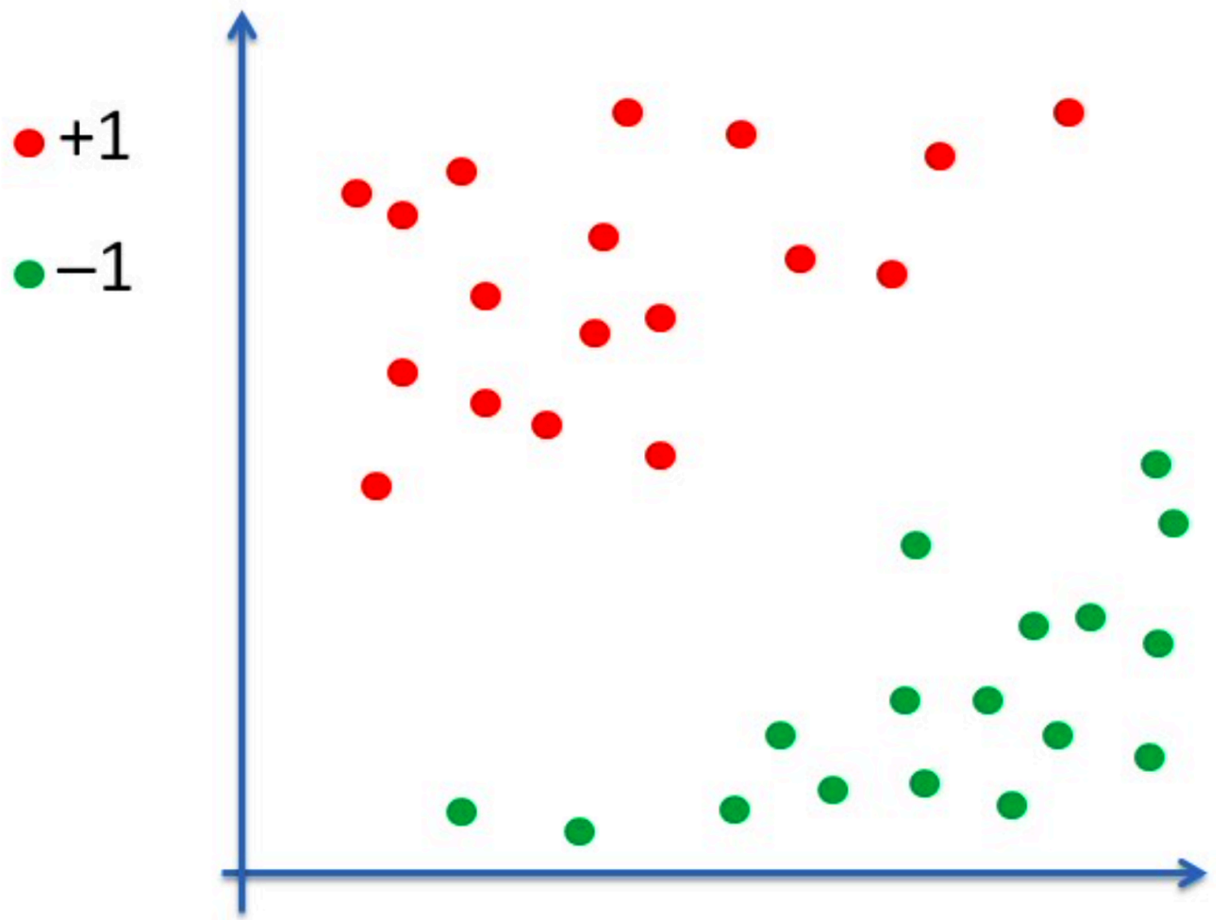
Support Vector Machine (SVM)

Very robust linear model for classification

Labeled Data

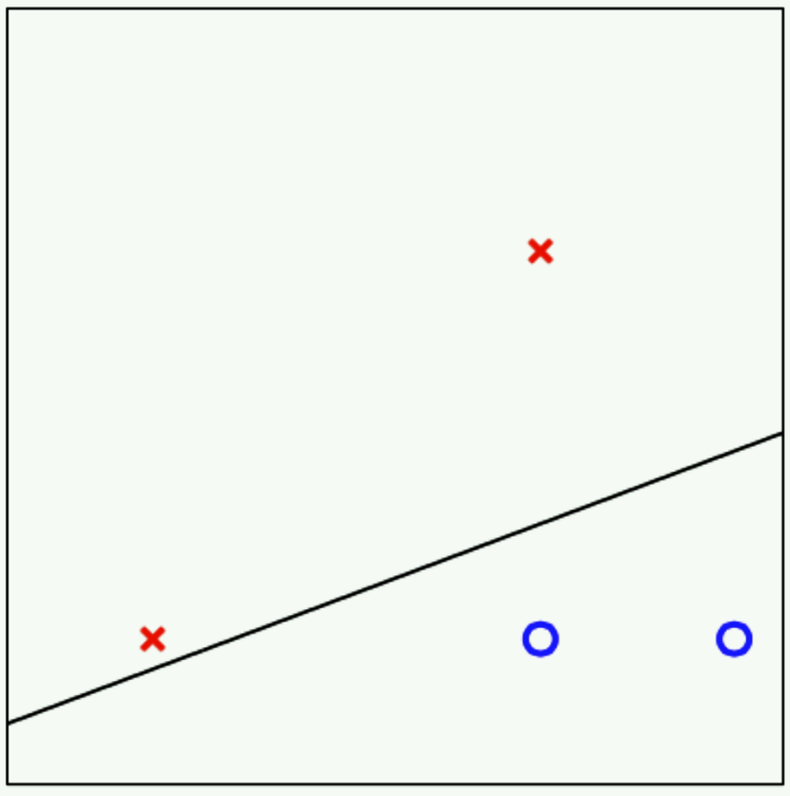
$$\mathcal{D} = \{(\mathbf{x}^i, y^i) | \mathbf{x}^i \in \mathbb{R}^n, y^i \in \{-1, 1\}\}_{i=1}^m$$

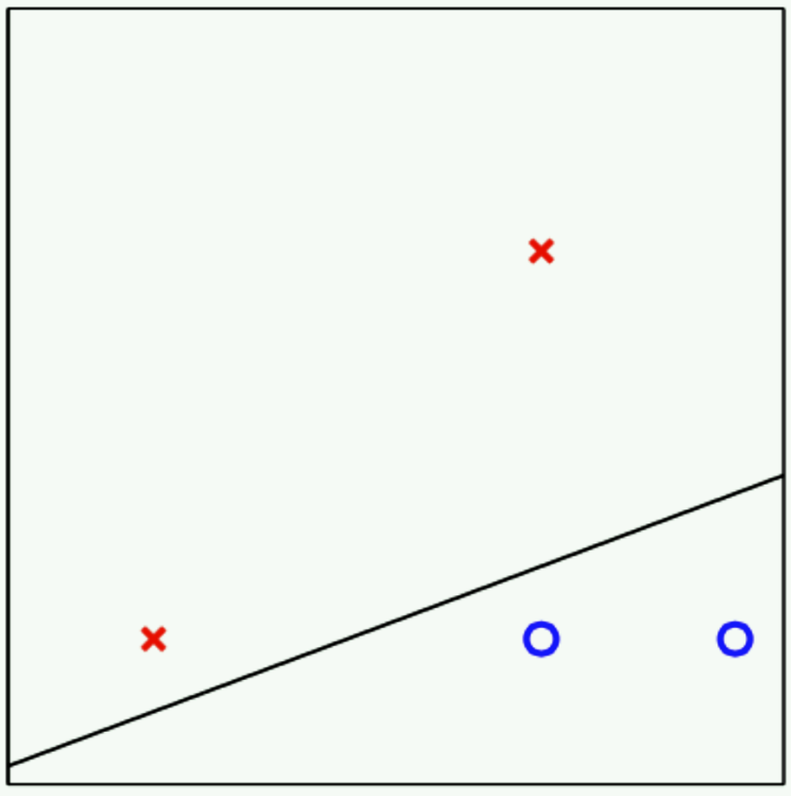


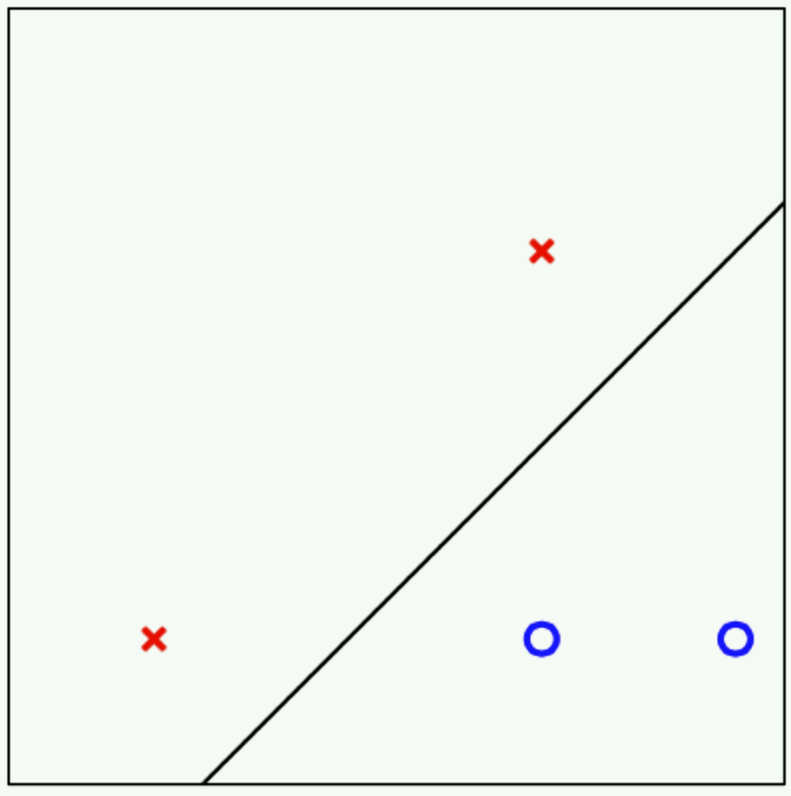


Support Vector Machine (SVM)

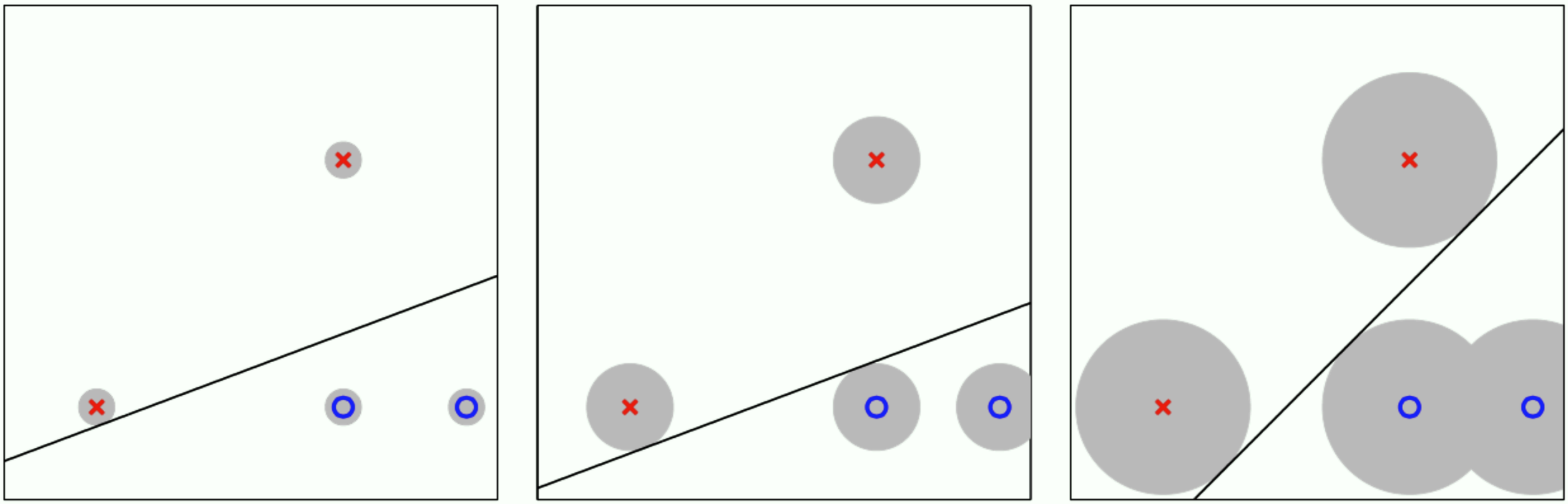
Which separator do you pick?





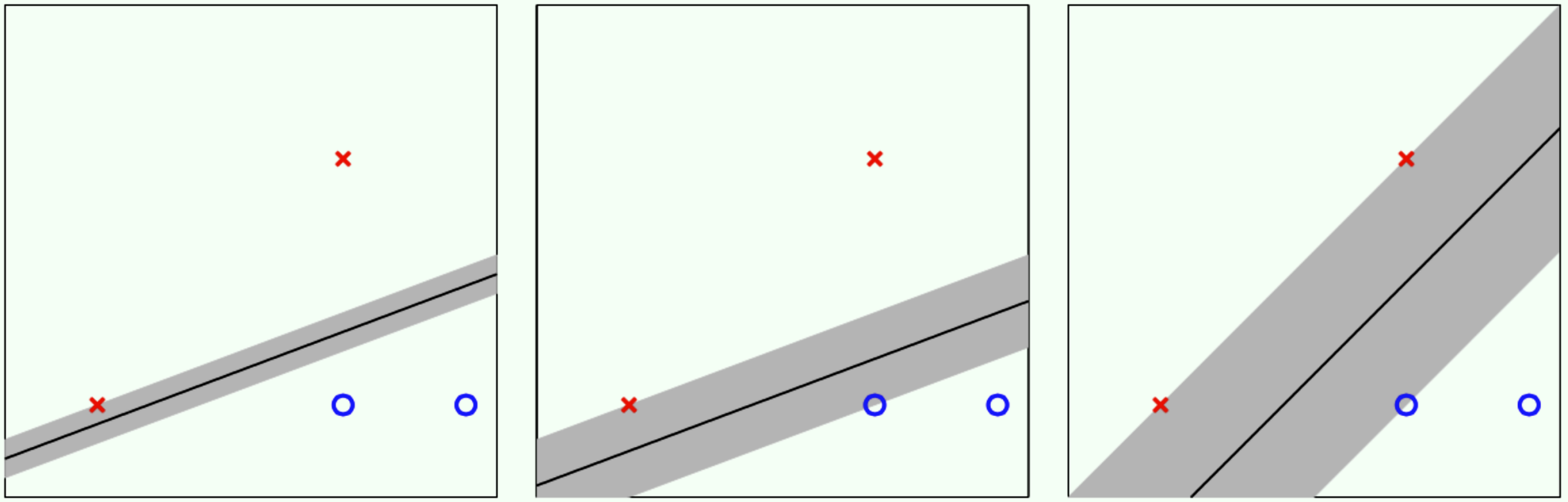


Robustness to noise



Being robust to noise (measurement noise) is good (remember regularization)

Thicker cushion means more robustness



We call such hyperplanes *fat*

1. Is a fatter hyperplane better than a thin one?
2. Can we efficiently find the fattest separating hyperplane?

Pulling out the bias

Before

$$\mathbf{x}, \mathbf{w} \in \mathbb{R}^{n+1}$$

$$\mathbf{x} = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}, \mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{bmatrix}$$

$$\text{signal} = \mathbf{w}^T\mathbf{x}$$

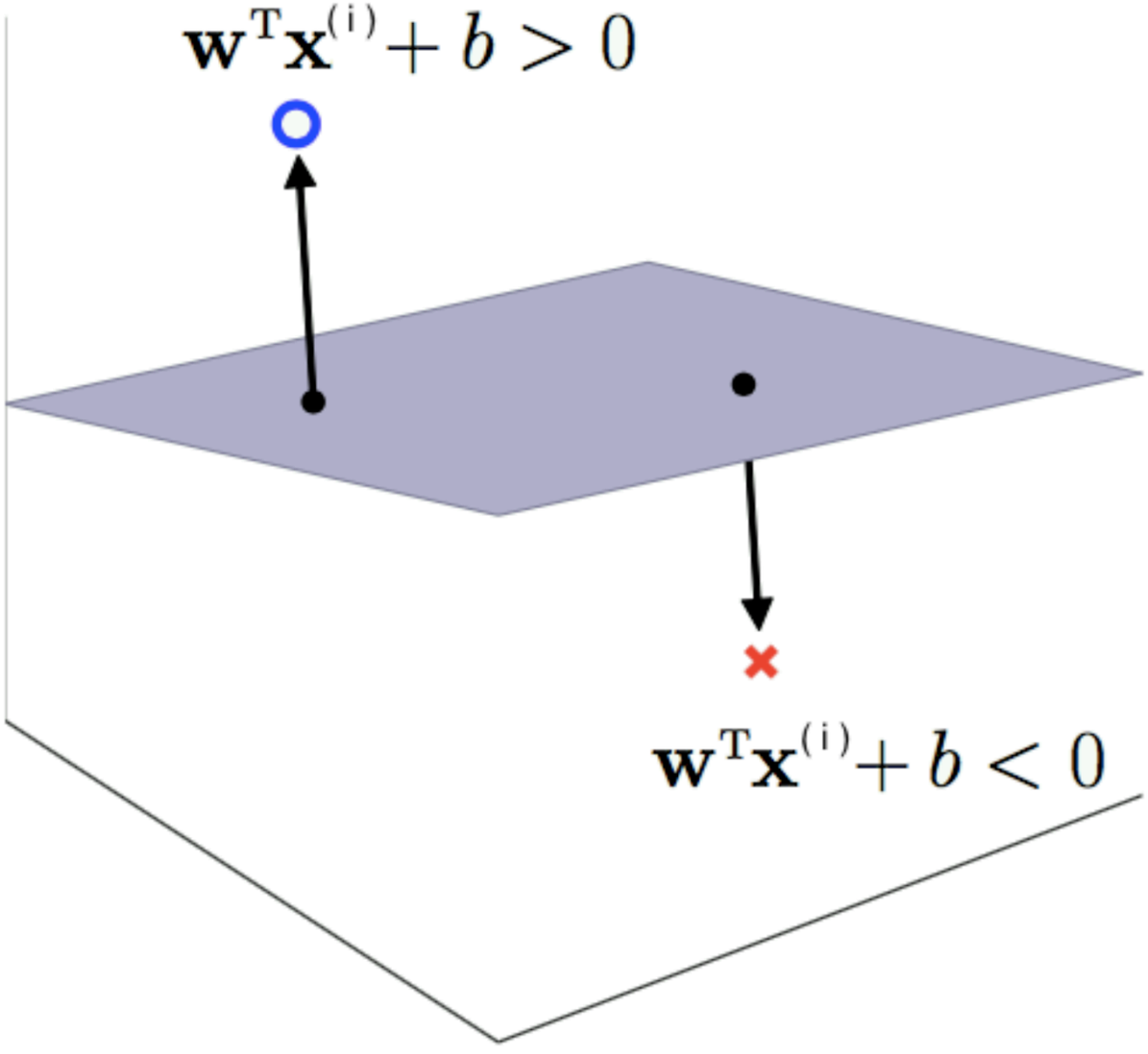
Now

$$\mathbf{x}, \mathbf{w} \in \mathbb{R}^{n}, b \in \mathbb{R}$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \mathbf{w} = \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix}$$

$$\text{signal} = \mathbf{w}^T\mathbf{x} + b$$

Separating the data



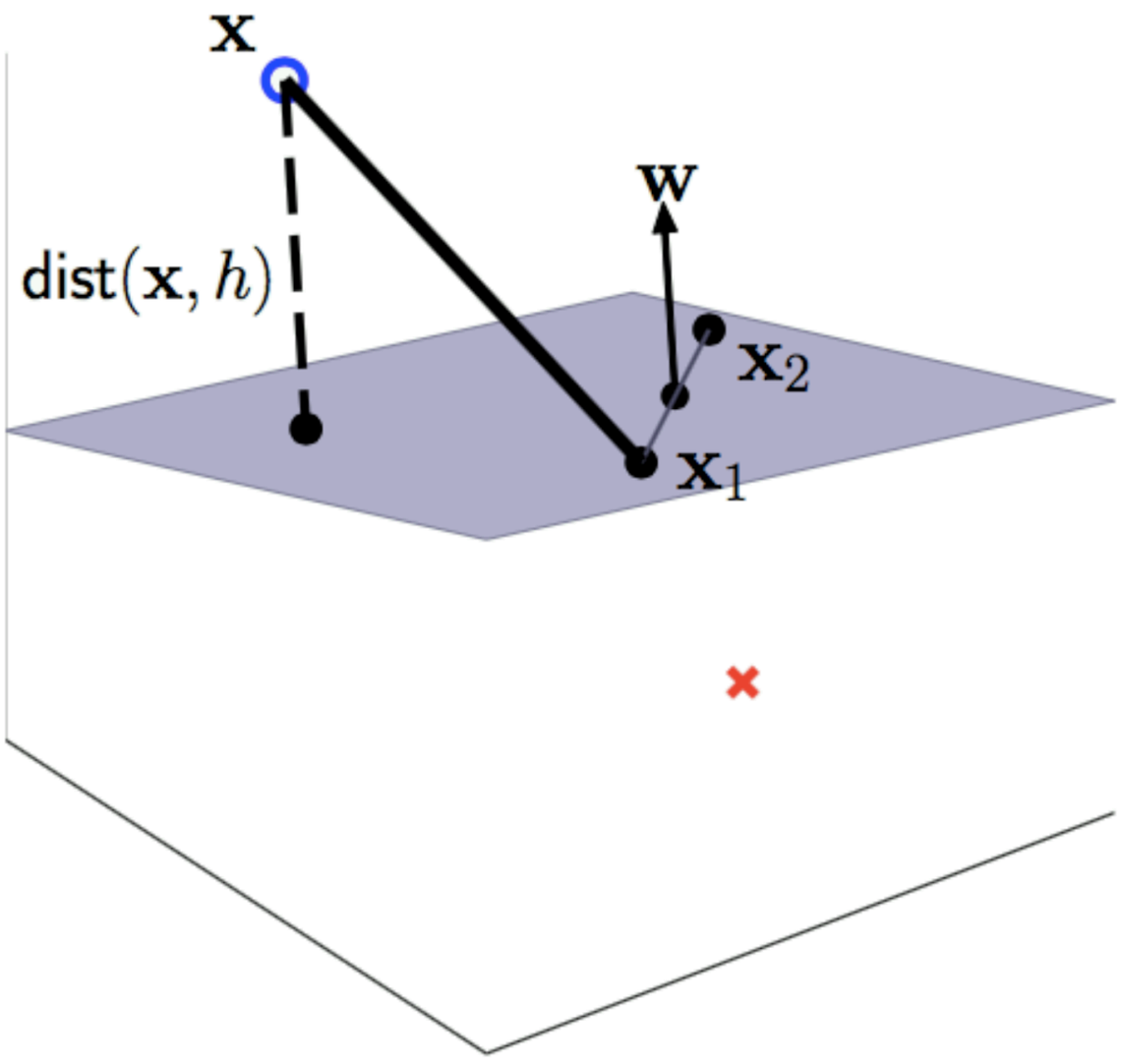
Hyperplane $h = (b, \mathbf{w})$
h separates the data means:

$$y^i(\mathbf{w}^T\mathbf{x}^i + b) > 0$$

By rescaling the weights and bias

$$\min_{i=1,\ldots,m} \; y^i(\mathbf{w}^T\mathbf{x}^i + b) = 1$$

Distance to the hyperplane



$\mathbf{w}$ is normal to the hyperplane

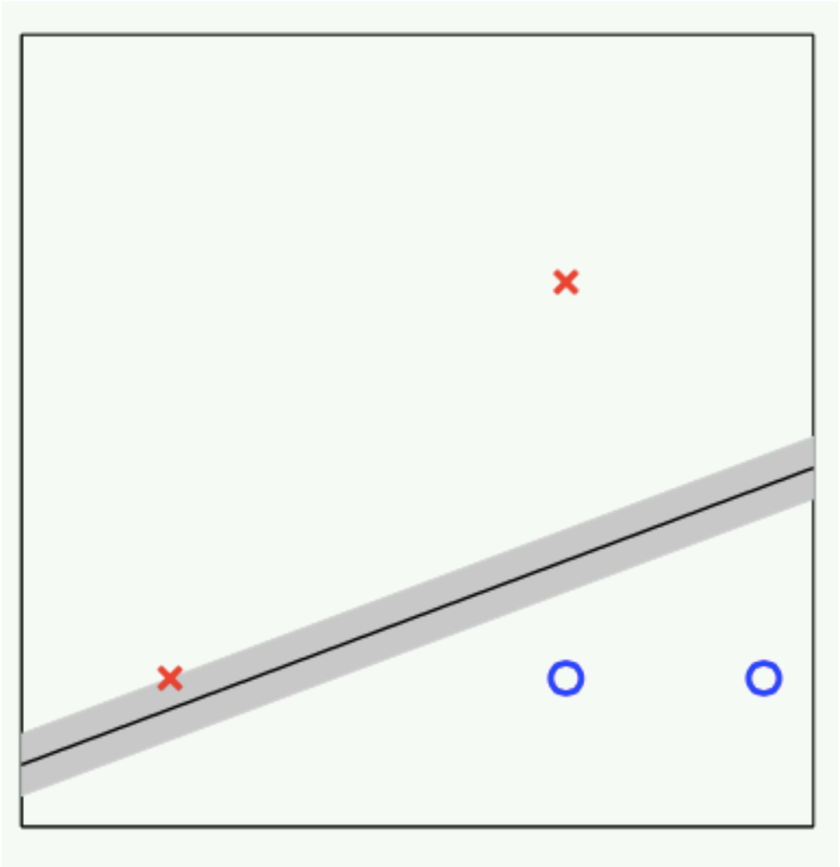
$$\mathbf{w}^T(\mathbf{x}_2 - \mathbf{x}_1) = \mathbf{w}^T\mathbf{x}_2 - \mathbf{w}^T\mathbf{x}_1 = 0.$$

Unit norm $\mathbf{u} = \mathbf{w}/\|\mathbf{w}\|$

$$\begin{aligned}\text{dist}(\mathbf{x}, h) &= |\mathbf{u}^T(\mathbf{x} - \mathbf{x}_1)| \\ &= \frac{1}{\|\mathbf{w}\|}|\mathbf{w}^T\mathbf{x} - \mathbf{w}^T\mathbf{x}_1| \\ &= \frac{1}{\|\mathbf{w}\|}|\mathbf{w}^T\mathbf{x} + b|\end{aligned}$$

Fatness of a separating hyperplane

$$\text{dist}(\mathbf{x}, h) = \frac{1}{\|\mathbf{w}\|}|\mathbf{w}^T\mathbf{x} + b|$$



since

$$|\mathbf{w}^T\mathbf{x}^i + b| = |y^i \cdot (\mathbf{w}^T\mathbf{x}^i + b)| = y^i \cdot (\mathbf{w}^T\mathbf{x}^i + b)$$

$$\text{dist}(\mathbf{x}^i, h) = \frac{1}{\|\mathbf{w}\|} \cdot y^i(\mathbf{w}^T\mathbf{x}^i + b)$$

$$\begin{aligned}\text{fatness} &= \min_i \text{dist}(\mathbf{x}^i, h) \\ &= \frac{1}{\|\mathbf{w}\|} \cdot \min_i y^i(\mathbf{w}^T\mathbf{x} + b) \\ &= \frac{1}{\|\mathbf{w}\|}\end{aligned}$$

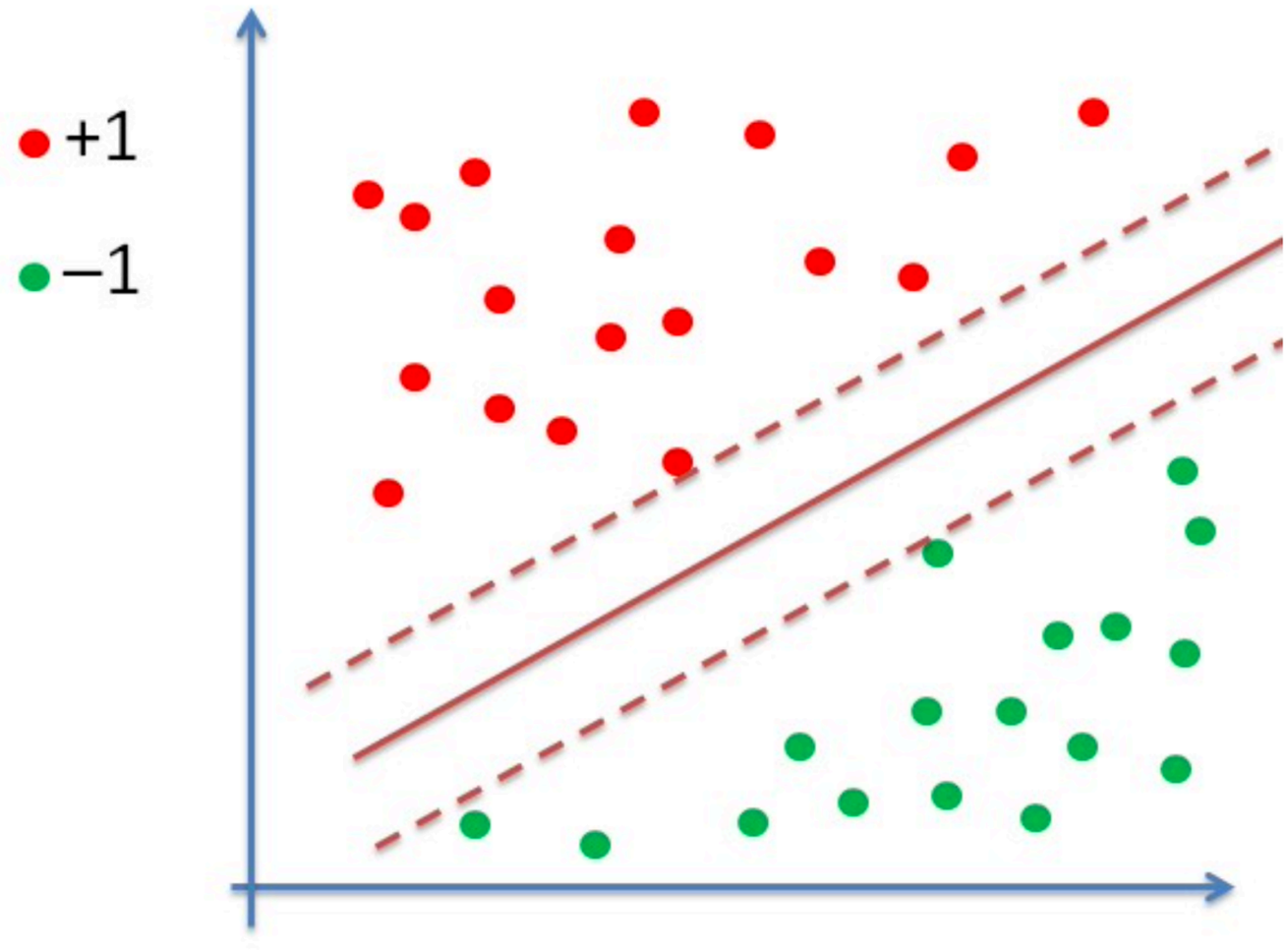
Maximizing the margin

$$\text{margin} = \frac{1}{\|\mathbf{w}\|}$$

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\|\mathbf{w}\|_2^2 \\ \text{subject to} \quad & \min_{i=1,\ldots,m} y^i(\mathbf{w}^T\mathbf{x}^i + b) = 1 \end{aligned}$$

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\|\mathbf{w}\|_2^2 \\ \text{subject to} \quad & y^i(\mathbf{w}^T\mathbf{x}^i + b) \geq 1, \quad i = 1, ..., m \end{aligned}$$

Alternative explanation



Hyperplane:

$$\mathbf{w}^T\mathbf{x} + b = 0$$

Margins:

$$\mathbf{w}^T\mathbf{x} + b = 1, \quad \mathbf{w}^T\mathbf{x} + b = -1$$

$$\begin{cases} \mathbf{w}^T\mathbf{x}^i + b \geq 1, & \text{if } y^i = 1 \\ \mathbf{w}^T\mathbf{x}^i + b \leq -1, & \text{if } y^i = -1 \end{cases}$$

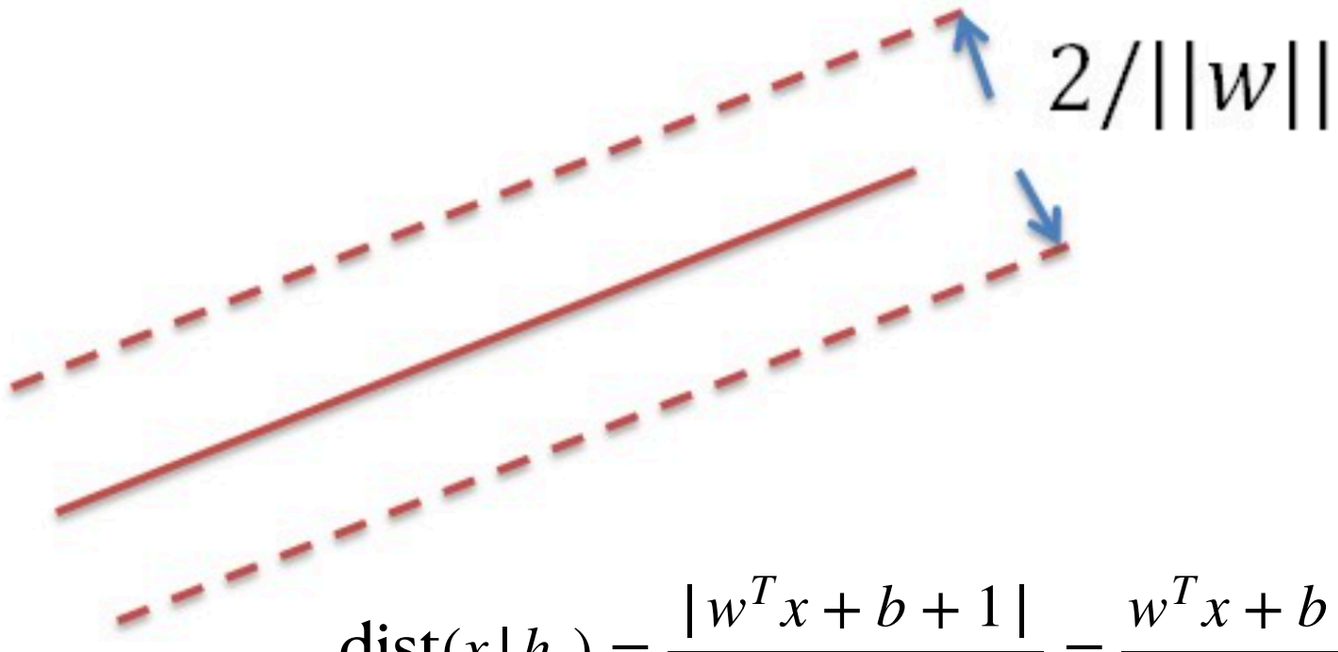
Alternative explanation

$$\text{minimize} \quad \frac{1}{2}\|\mathbf{w}\|_2^2$$

$$\text{subject to} \quad y^i(\mathbf{w}^T\mathbf{x}^i + b) \geq 1, \quad i = 1, ..., m$$

$$\begin{cases} \mathbf{w}^T\mathbf{x}^i + b \geq 1, \\ \mathbf{w}^T\mathbf{x}^i + b \leq -1, \end{cases}$$

$$\text{dist}(x \mid h_1) = \frac{|w^T x + b - 1|}{\|w\|_2^2} = \frac{-w^T x - b + 1}{\|w\|_2^2}$$



$$\text{dist}(x \mid h_2) = \frac{|w^T x + b + 1|}{\|w\|_2^2} = \frac{w^T x + b + 1}{\|w\|_2^2}$$

Example

$$\mathbf{X} = \begin{bmatrix} 0 & 0 \\ 2 & 2 \\ 2 & 0 \\ 3 & 0 \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} -1 \\ -1 \\ +1 \\ +1 \end{bmatrix}$$

$$\begin{aligned} -b &\geq 1 \quad (i) \\ -(2w_1 + 2w_2 + b) &\geq 1 \quad (ii) \\ 2w_1 + b &\geq 1 \quad (iii) \\ 3w_1 + b &\geq 1 \quad (iv) \end{aligned}$$

(i) and (iii) gives $w_1 \geq 1$

(ii) and (iii) gives $w_2 \leq -1$

So, $\frac{1}{2}(w_1^2 + w_2^2) \geq 1$ $(b = -1, w_1 = 1, w_2 = -1)$

Optimal Hyperplane

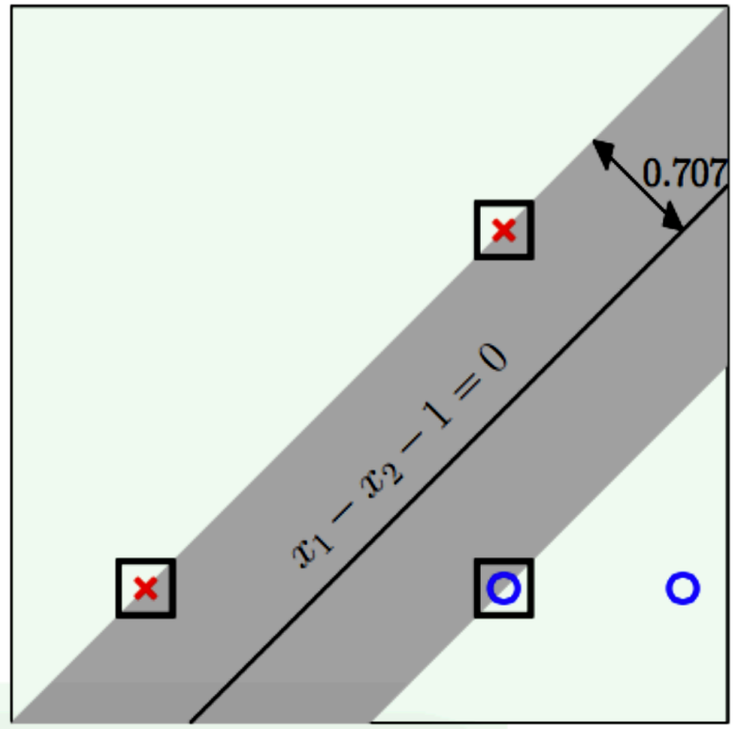
$$g(\mathbf{x}) = \text{sign}(x_1 - x_2 - 1)$$

margin: $$\frac{1}{\| \mathbf{w}^* \|} = \frac{1}{\sqrt{2}} \approx 0.707.$$

For data points (i), (ii) and (iii) $y\ (\mathbf{w}^{*\mathrm{T}}\mathbf{x} + b^*) = 1$

↑

Support Vectors



Quadratic Programming

minimize $\frac{1}{2}\mathbf{u}^T\mathbf{Q}\mathbf{u} + \mathbf{p}^T\mathbf{u}$

subject to $\mathbf{A}\mathbf{u} \geq \mathbf{c}$

minimize $\frac{1}{2}\|\mathbf{w}\|_2^2$

subject to $y^i(\mathbf{w}^T\mathbf{x}^i - b) \geq 1,$
$i = 1, ..., m$

minimize $\frac{1}{2}\mathbf{u}^T\mathbf{Q}\mathbf{u} + \mathbf{p}^T\mathbf{u}$

subject to $\mathbf{A}\mathbf{u} \geq \mathbf{c}$

$$\mathbf{u} = [b, \mathbf{w}]^T$$

$$\mathbf{Q} = \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix}, \mathbf{p} = 0$$

$$\mathbf{A} = \begin{bmatrix} y^1 & y^1(\mathbf{x}^1)^T \\ \vdots & \vdots \\ y^m & y^m(\mathbf{x}^m)^T \end{bmatrix}, \mathbf{c} = \mathbf{1}$$

Exercise:

$$\mathrm{X} = \begin{bmatrix} 0 & 0 \\ 2 & 2 \\ 2 & 0 \\ 3 & 0 \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} -1 \\ -1 \\ +1 \\ +1 \end{bmatrix}$$

$$\begin{aligned} -b &\geq 1 \quad (i) \\ -(2w_1 + 2w_2 + b) &\geq 1 \quad (ii) \\ 2w_1 + b &\geq 1 \quad (iii) \\ 3w_1 + b &\geq 1 \quad (iv) \end{aligned}$$

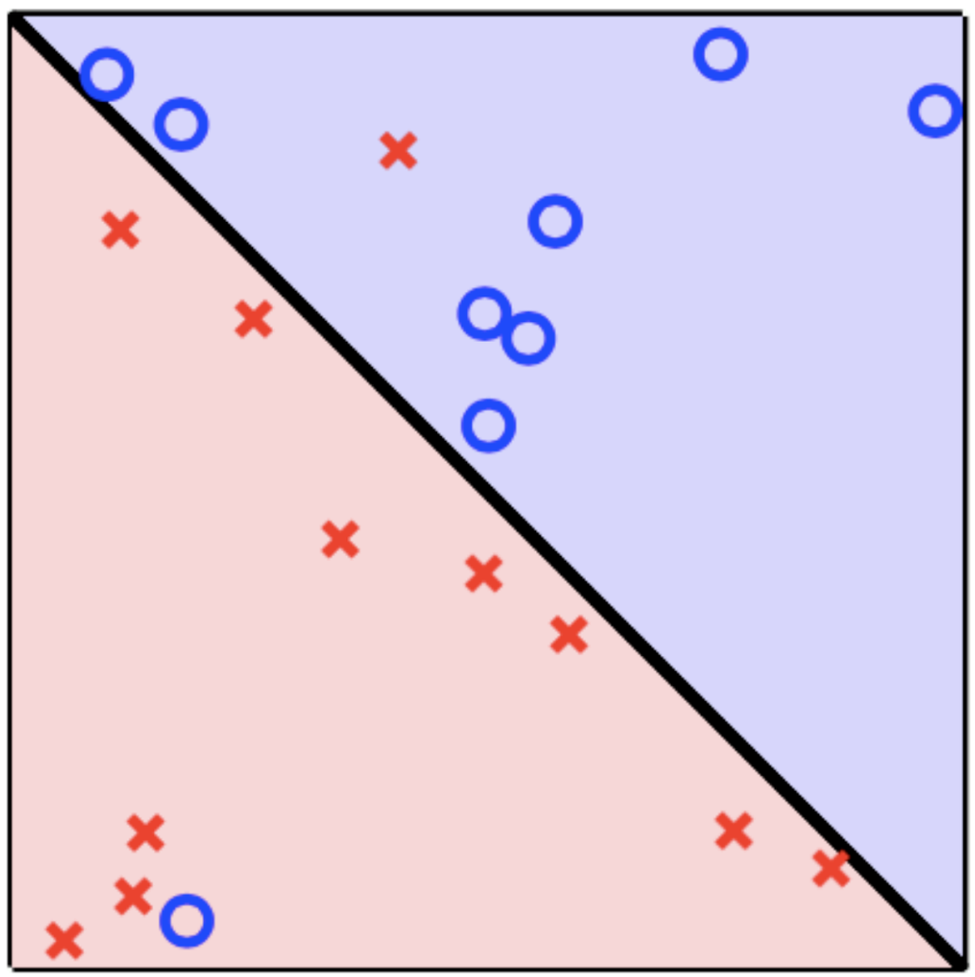
Show that

$$\mathrm{Q} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \mathbf{p} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad \mathrm{A} = \begin{bmatrix} -1 & 0 & 0 \\ -1 & -2 & -2 \\ 1 & 2 & 0 \\ 1 & 3 & 0 \end{bmatrix} \quad \mathbf{c} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

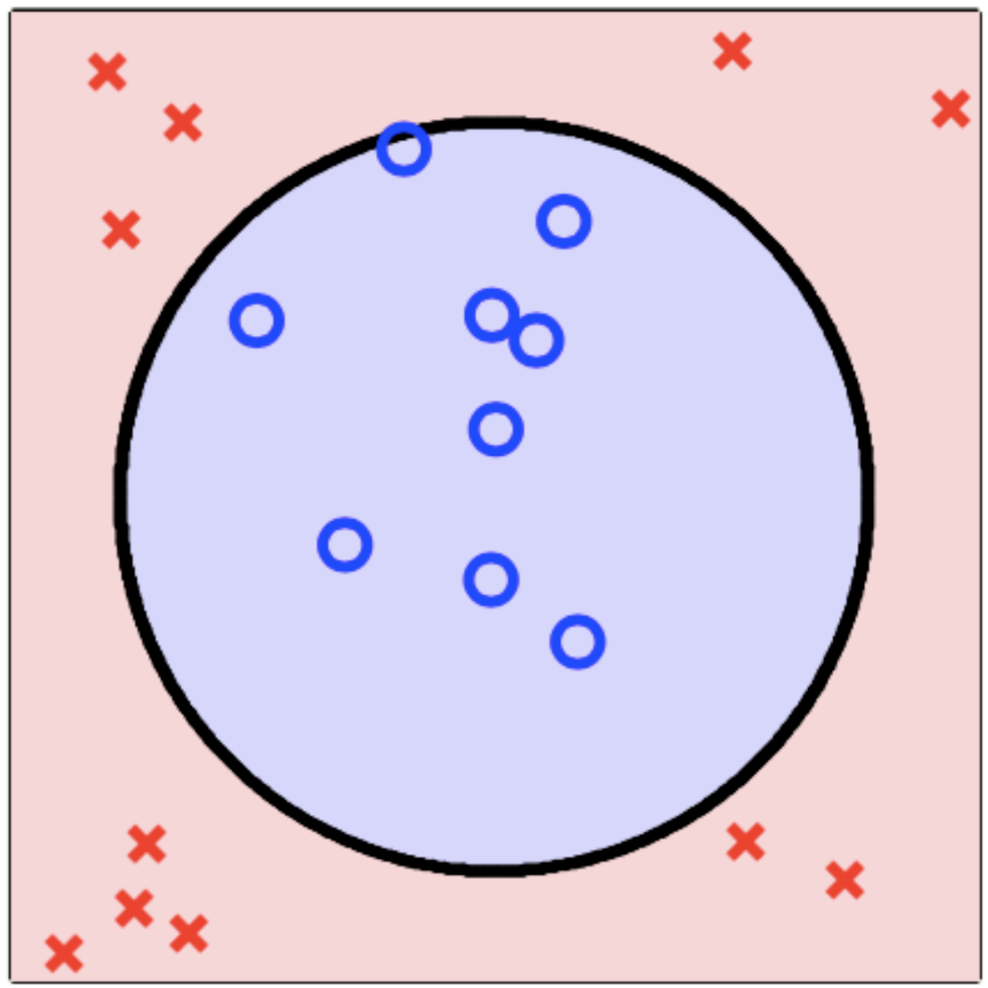
Use your QP-solver to give

$$(b^*, w_1^*, w_2^*) = (-1, 1, -1)$$

Non-separable data

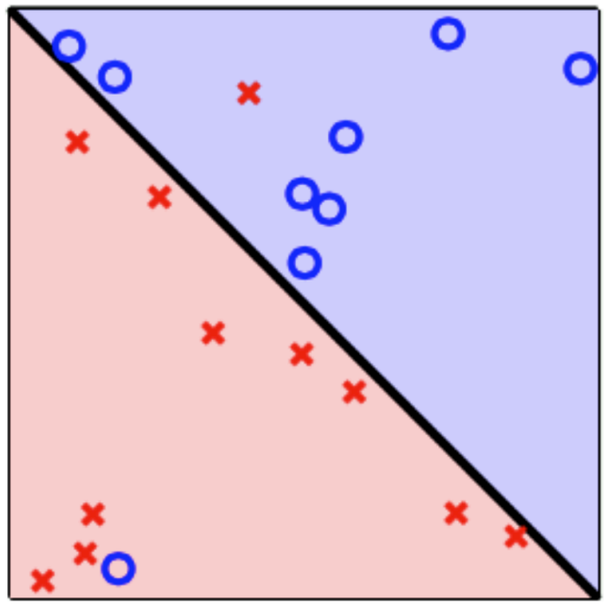


tolerate error



nonlinear transform

Soft-margin SVM

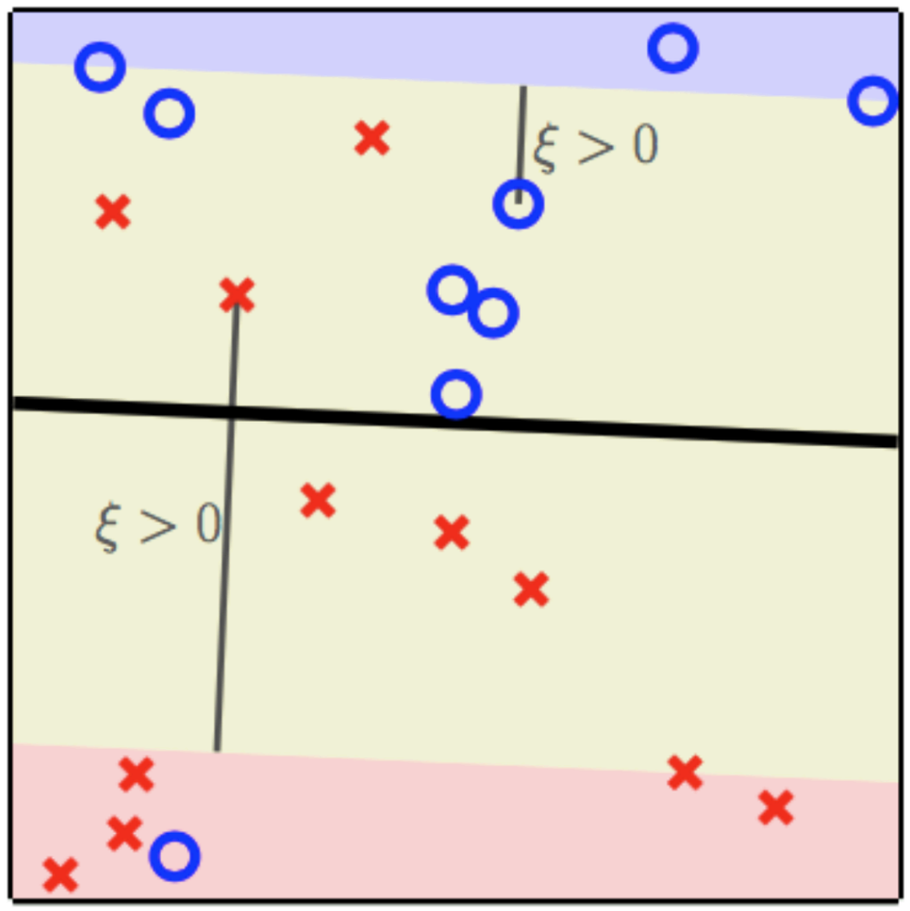


$$\begin{aligned} \text{minimize} \quad & \tfrac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^m \xi^i \\ \text{subject to} \quad & y^i(\mathbf{w}^T\mathbf{x}^i + b) \geq 1 - \xi^i, \\ & \xi^i \geq 0, \quad i = 1, ..., m \end{aligned}$$

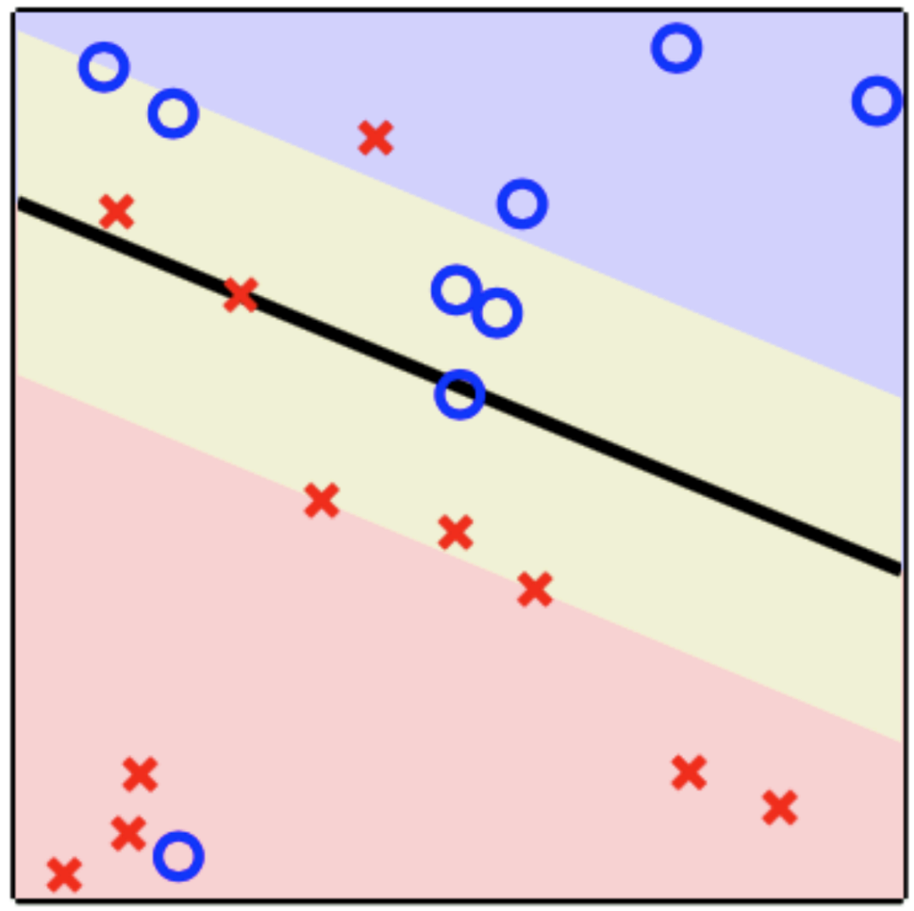
tolerate error

- Trades off "soft in-sample error' $\sum_{i=1}^{m} \xi^i$ with weight norm $\frac{1}{2}\mathbf{w}^T\mathbf{w}$, like regularization.
- C plays the role of a regularization parameter
- Choice of C is important

Non-separable data



$C = 1$

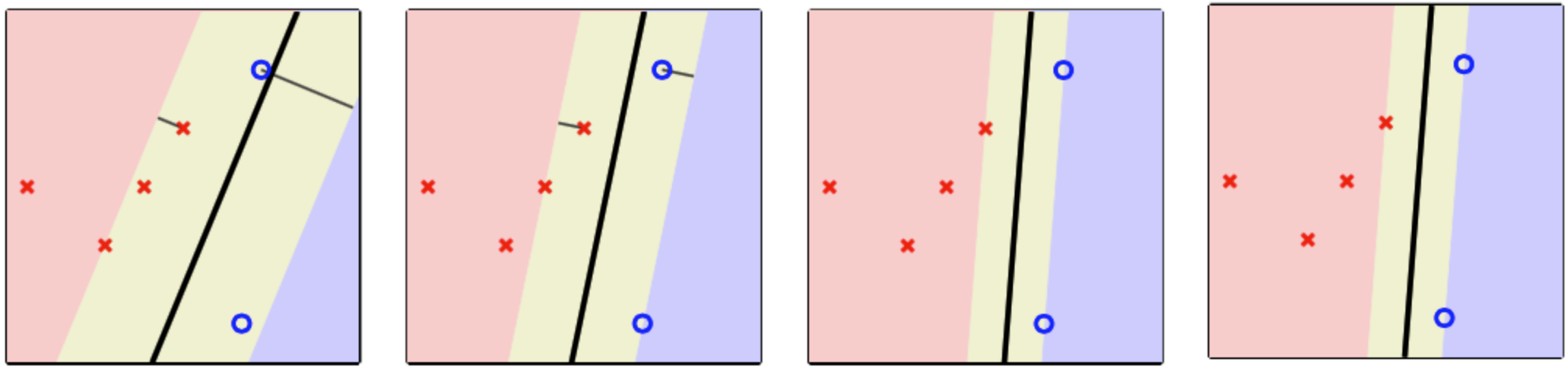


$C = 500$

minimize $\quad \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{m} \xi^i$

subject to $\quad y^i(\mathbf{w}^T\mathbf{x}^i - b) \geq 1 - \xi^i,$
$\xi^i \geq 0, \quad i = 1, ..., m$

Soft margin SVM with separable data



$$\begin{aligned} \text{minimize} \quad & \tfrac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{m}\xi^i \\ \text{subject to} \quad & y^i(\mathbf{w}^T\mathbf{x}^i - b) \geq 1 - \xi^i, \\ & \xi^i \geq 0, \quad i = 1, ..., m \end{aligned}$$

Choice of C is important!

SVM and regularization

$$\text{minimize} \quad \|\mathbf{w}\|_2^2 + C\sum_{i=1}^{n} \max\{1 - y^i(\mathbf{w}^T\mathbf{x}^i - b), 0\}$$

ℓ_1-regularized SVM problem (often results in sparse solution)

$$\text{minimize} \quad \|\mathbf{w}\|_1 + C\sum_{i=1}^{n} \max\{1 - y^i(\mathbf{w}^T\mathbf{x}^i - b), 0\}$$

Link to regularization (why SVM is robust...)

Optimal hyperplane

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w}$$
$$\text{s.t.} \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \quad i = 1, ..., N.$$

Regularization

$$\min_{\mathbf{w},b} \quad E_{in}(\mathbf{w})$$
$$\text{s.t.} \quad \mathbf{w}^T\mathbf{w} \leq C, \quad i = 1, ..., N.$$

	optimal hyperplane	regularization
minimize	$\mathbf{w}^T\mathbf{w}$	E_{in}
subject to	$E_{in} = 0$	$\mathbf{w}^T\mathbf{w} \leq C$

The optimal hyperplane performs "automatic" regularization

Evidence that larger margin is better

- Experimental: larger margin gives lower E_{out}; `bias` drops a little and `var` a lot.
- Bound for d_{vc} can be less than $d+1$ - fat hyperplanes generalized better
- E_{vc} bound does not explicitly depend on d

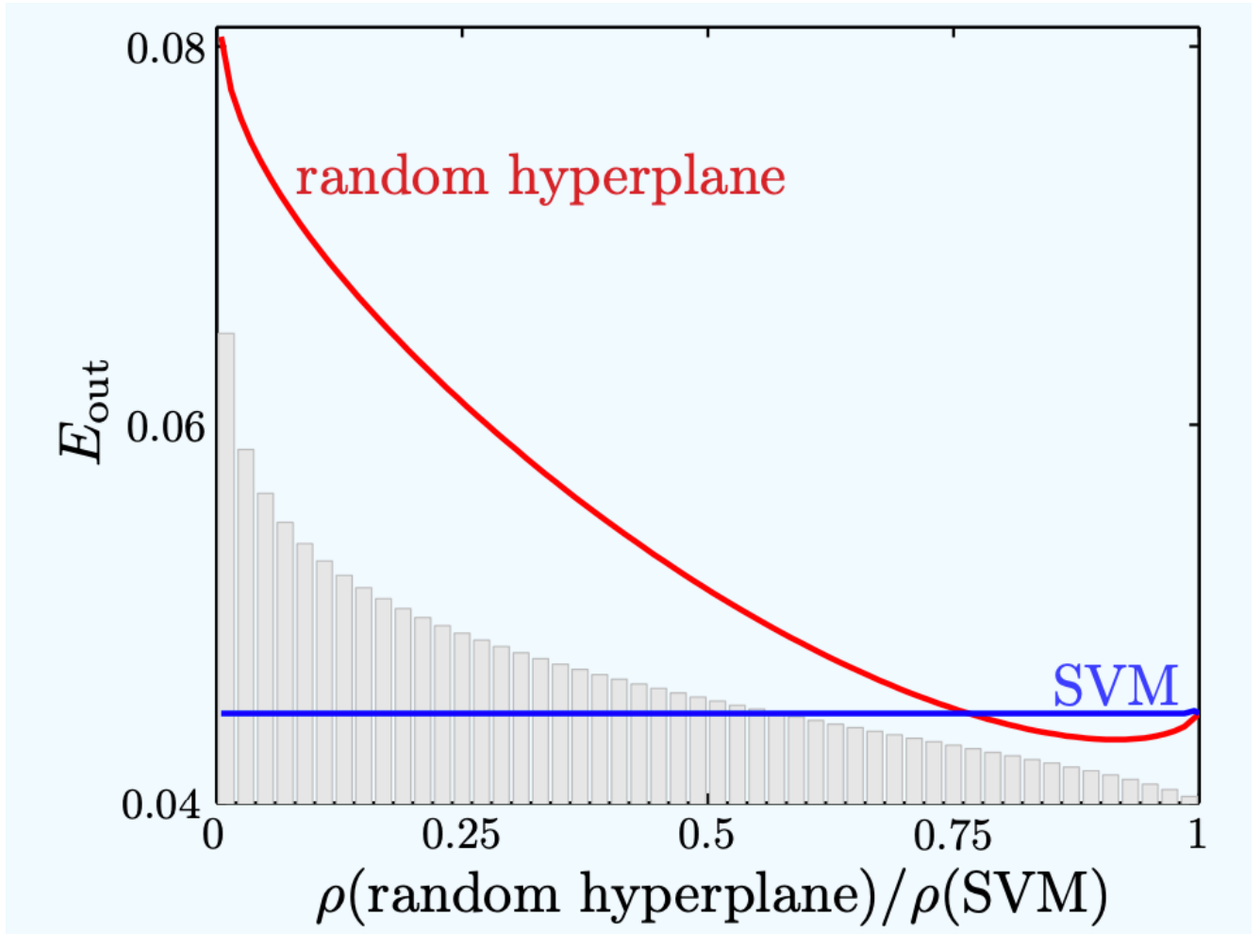
$$\rho = \min_{n=1,\ldots,N} y_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b)$$

Generate a random separable data set $(N = 20)$

Select 50,000 random separating hyperplanes h

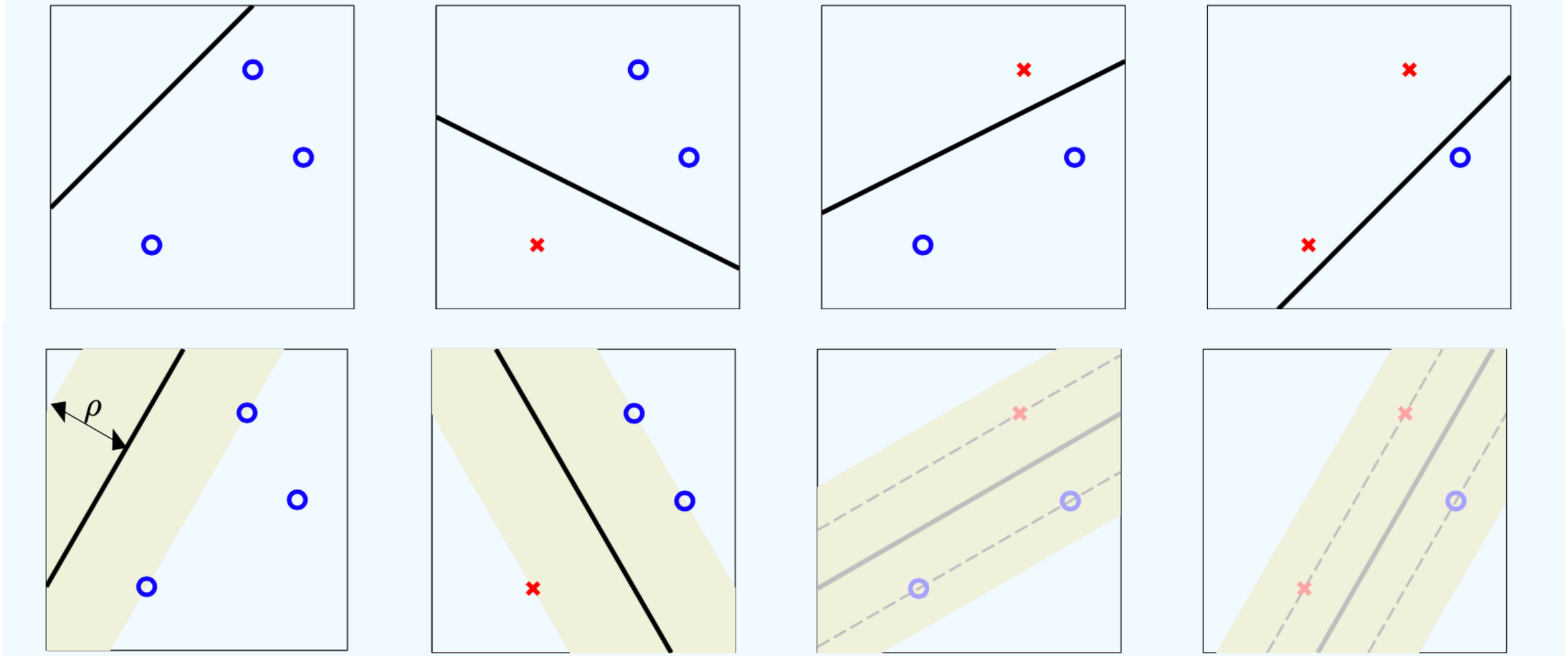
Compute E_{out} and $\rho(h)/\rho(SVM)$

Average over several thousands of random data sets



(Histogram shows relative frequency of different margins)

Fat hyperplanes shatter fewer points

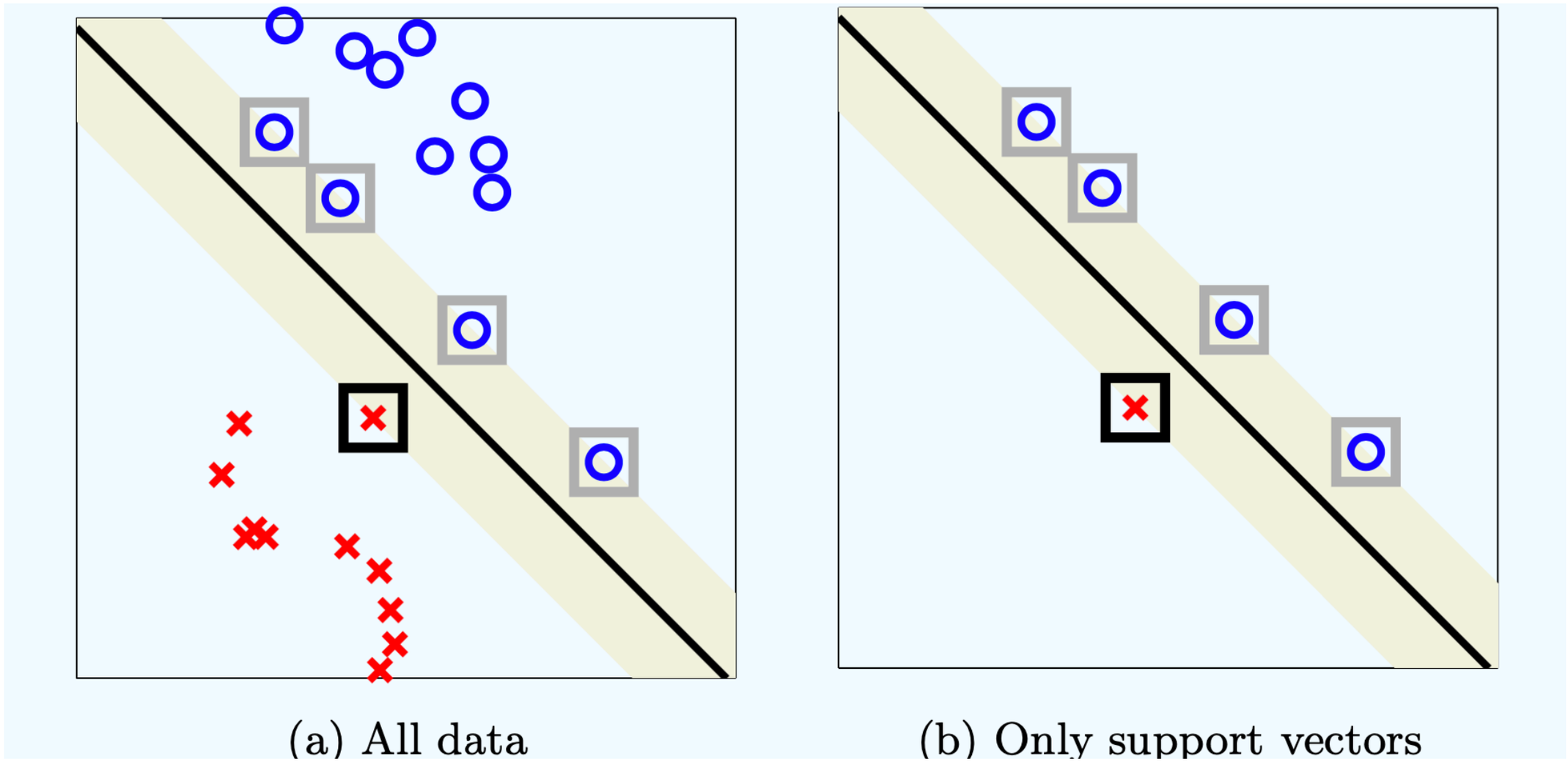


Theorem 8.5 (VC dimension of Fat Hyperplanes)**.** Suppose the input space is the ball of radius R in $\mathbb{R}^d$, so $\|\mathbf{x}\| \leq R$. Then,

$$d_{\text{vc}}(\rho) \leq \lceil R^2/\rho^2 \rceil + 1,$$

where $\lceil R^2/\rho^2 \rceil$ is the smallest integer greater than or equal to R^2/ρ^2.

A bound on E_{cv}



(a) All data (b) Only support vectors

$$E_{\text{cv}} = \frac{1}{N}\sum_{n=1}^{N} \mathsf{e}_n$$

E_{cv} is that it is an unbiased estimate of the expected out-of-sample error for a data set of size $N-1$.

$$E_{\text{cv}}(\text{SVM}) = \frac{1}{N}\sum_{n=1}^{N} \mathsf{e}_n \leq \frac{\#\text{ support vectors}}{N}$$

(no explicit dependence on d)

A bound on E_{cv}

Algorithm For Selecting Separating Hyperplane		
General	PLA	SVM (Optimal Hyperplane)
$d_{\text{VC}} = d + 1$	$E_{\text{cv}} \leq \dfrac{R^2}{N\rho^2}$	$d_{\text{VC}}(\rho) \leq \min\left(\left\lceil \frac{R^2}{\rho^2} \right\rceil, d\right) + 1$ $E_{\text{cv}} \leq \dfrac{\#\text{ support vectors}}{N}$

bias↓ var↓↓

Primal & Lagrange

Minimize $\quad \dfrac{1}{2}\,\mathbf{w}^{\mathsf{T}}\mathbf{w}$

subject to $\quad y_n\left(\mathbf{w}^{\mathsf{T}}\mathbf{x}_n + b\right) \geq 1 \quad$ for $\quad n = 1, 2, \ldots, N$

$$\mathbf{w} \in \mathbb{R}^d,\ b \in \mathbb{R}$$

Minimize $\quad \mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha}) = \dfrac{1}{2}\,\mathbf{w}^{\mathsf{T}}\mathbf{w} - \sum_{n=1}^{N} \alpha_n (y_n\left(\mathbf{w}^{\mathsf{T}}\mathbf{x}_n + b\right) - 1)$

$$\begin{cases} \nabla_{\mathbf{w}}\mathcal{L} = \mathbf{w} - \sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n = \mathbf{0} \\ \dfrac{\partial \mathcal{L}}{\partial b} = -\sum_{n=1}^{N} \alpha_n y_n = 0 \end{cases}$$

Lagrange and Dual

$$\mathbf{w} = \sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n \qquad \text{and} \qquad \sum_{n=1}^{N} \alpha_n y_n = 0$$

$$\mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2} \mathbf{w}^{\mathsf{T}}\mathbf{w} - \sum_{n=1}^{N} \alpha_n \left(y_n \left(\mathbf{w}^{\mathsf{T}}\mathbf{x}_n + b\right) - 1\right)$$

Maximize $\mathcal{L}(\boldsymbol{\alpha}) = \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} y_n y_m \, \alpha_n \alpha_m \, \mathbf{x}_n^{\mathsf{T}} \mathbf{x}_m$

subject to $\alpha_n \geq 0$ for $n = 1, \cdots, N$ and $\sum_{n=1}^{N} \alpha_n y_n = 0$

The dual – QP

$$\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\,\boldsymbol{\alpha}^{\mathsf{T}} \underbrace{\begin{bmatrix} y_1 y_1\, \mathbf{x}_1^{\mathsf{T}}\mathbf{x}_1 & y_1 y_2\, \mathbf{x}_1^{\mathsf{T}}\mathbf{x}_2 & \dots & y_1 y_N\, \mathbf{x}_1^{\mathsf{T}}\mathbf{x}_N \\ y_2 y_1\, \mathbf{x}_2^{\mathsf{T}}\mathbf{x}_1 & y_2 y_2\, \mathbf{x}_2^{\mathsf{T}}\mathbf{x}_2 & \dots & y_2 y_N\, \mathbf{x}_2^{\mathsf{T}}\mathbf{x}_N \\ \dots & \dots & \dots & \dots \\ y_N y_1\, \mathbf{x}_N^{\mathsf{T}}\mathbf{x}_1 & y_N y_2\, \mathbf{x}_N^{\mathsf{T}}\mathbf{x}_2 & \dots & y_N y_N\, \mathbf{x}_N^{\mathsf{T}}\mathbf{x}_N \end{bmatrix}}_{\text{quadratic coefficients}} \boldsymbol{\alpha} + \underbrace{(-\mathbf{1}^{\mathsf{T}})}_{\text{linear}} \boldsymbol{\alpha}$$

subject to $\underbrace{\mathbf{y}^{\mathsf{T}}\boldsymbol{\alpha} = 0}_{\text{linear constraint}}$

$$\underbrace{\mathbf{0}}_{\text{lower bounds}} \leq \boldsymbol{\alpha} \leq \underbrace{\infty}_{\text{upper bounds}}$$

The dual – QP

Solution: $\boldsymbol{\alpha} = \alpha_1, \cdots, \alpha_N$

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

KKT condition: For $n = 1, \cdots, N$

$$\alpha_n \left(y_n \left(\mathbf{w}^\mathsf{T} \mathbf{x}_n + b \right) - 1 \right) = 0$$

We saw this before!

$\alpha_n > 0 \implies \mathbf{x}_n$ is a **support vector**

Closest $\mathbf{x}_n$'s to the plane: achieve the margin

$$\Longrightarrow \quad y_n \left(\mathbf{w}^\mathsf{T} \mathbf{x}_n + b \right) = 1$$

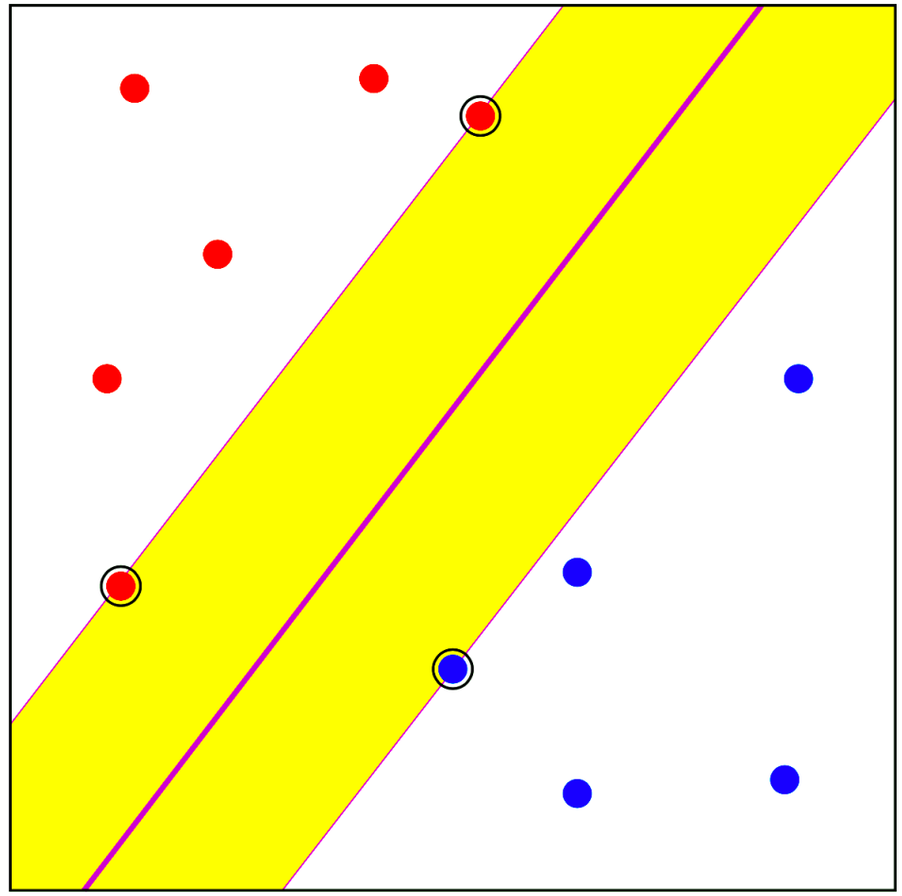
$$\mathbf{w} = \sum_{\mathbf{x}_n \text{ is SV}} \alpha_n y_n \mathbf{x}_n \qquad \mathbf{w}^* = \sum_{n=1}^{N} y_n \alpha_n^* \mathbf{x}_n$$

Solve for b using any SV:

$$y_n \left(\mathbf{w}^\mathsf{T} \mathbf{x}_n + b \right) = 1$$

$$y_s \left(\mathbf{w}^{*\mathrm{T}} \mathbf{x}_s + b^* \right) = 1 \qquad b^* = y_s - \mathbf{w}^{*\mathrm{T}} \mathbf{x}_s$$

$$= y_s - \sum_{n=1}^{N} y_n \alpha_n^* \mathbf{x}_n^\mathrm{T} \mathbf{x}_s$$



The dual for soft-margin

$$\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}\mathbf{w}^\intercal\mathbf{w} + C\sum_{n=1}^{N}\xi_n - \sum_{n=1}^{N}\alpha_n(y_n(\mathbf{w}^\intercal\mathbf{x}_n + b) - 1 + \xi_n) - \sum_{n=1}^{N}\beta_n\,\xi_n$$

Minimize w.r.t. $\mathbf{w}$, b, and $\boldsymbol{\xi}$ and maximize w.r.t. each $\alpha_n \geq 0$ and $\beta_n \geq 0$

$$\nabla_{\mathbf{w}}\mathcal{L} = \mathbf{w} - \sum_{n=1}^{N}\alpha_n y_n \mathbf{x}_n = \mathbf{0}$$

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

$$\frac{\partial\mathcal{L}}{\partial \xi_n} = C - \alpha_n - \beta_n = 0$$

The dual for soft-margin

Maximize $$\mathcal{L}(\boldsymbol{\alpha}) = \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} y_n y_m \, \alpha_n \alpha_m \, \mathbf{x}_n^{\mathsf{T}} \mathbf{x}_m$$ w.r.t. to $\boldsymbol{\alpha}$

subject to $0 \leq \alpha_n \leq C$ for $n = 1, \cdots, N$ and $$\sum_{n=1}^{N} \alpha_n y_n = 0$$

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

minimizes $$\frac{1}{2} \mathbf{w}^{\mathsf{T}} \mathbf{w} + C \sum_{n=1}^{N} \xi_n$$

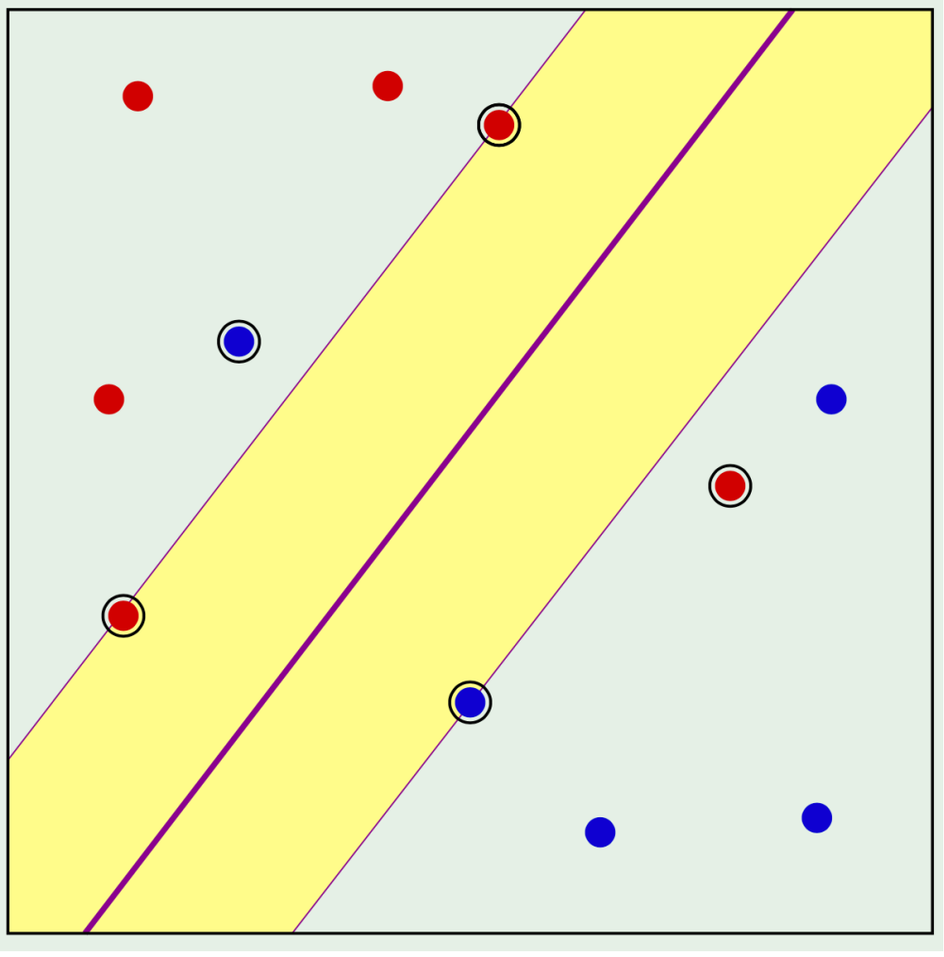
Types of SVs

margin support vectors $\quad (0 < \alpha_n < C)$

$$y_n \left(\mathbf{w}^\mathsf{T} \mathbf{x}_n + b \right) = 1 \qquad (\xi_n = 0)$$

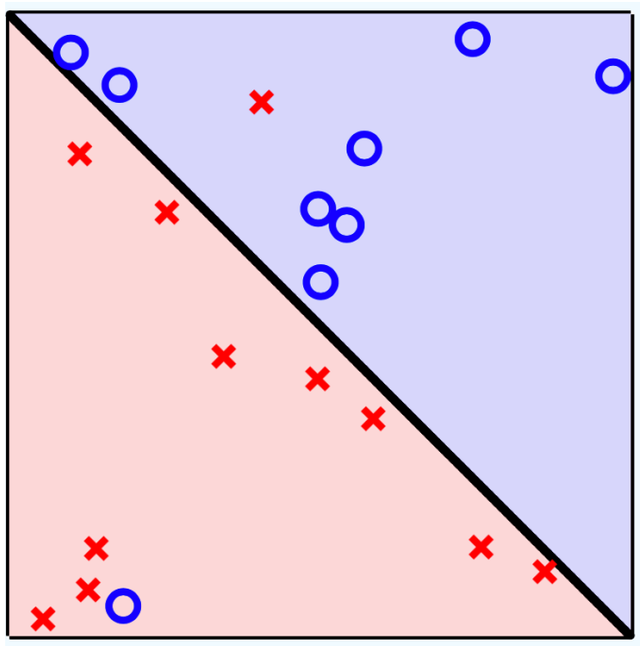
non-margin support vectors $\quad (\alpha_n = C)$

$$y_n \left(\mathbf{w}^\mathsf{T} \mathbf{x}_n + b \right) < 1 \qquad (\xi_n > 0)$$

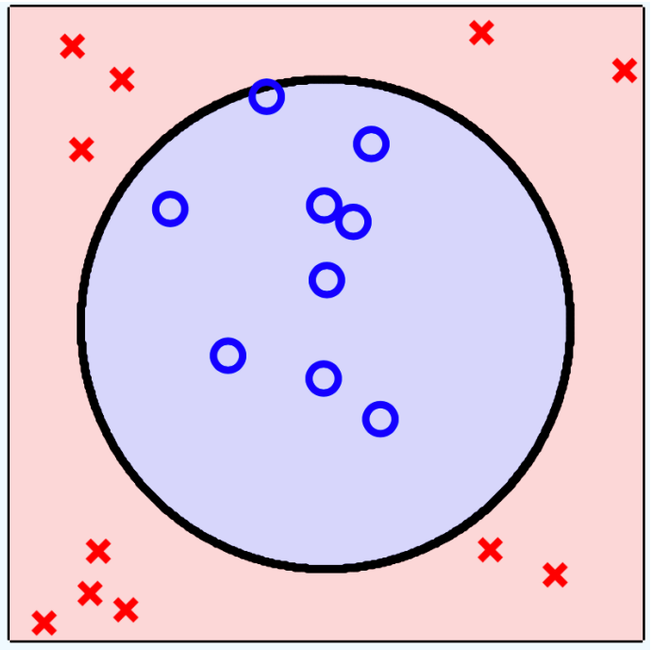


Nonlinear transformation and SVM

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



(a) Few noisy data.



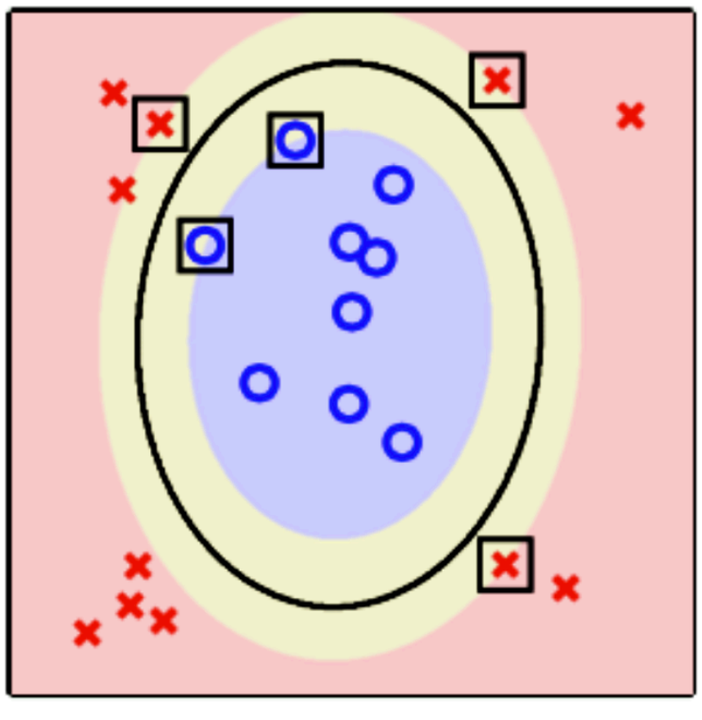
(b) Nonlinearly separable.

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

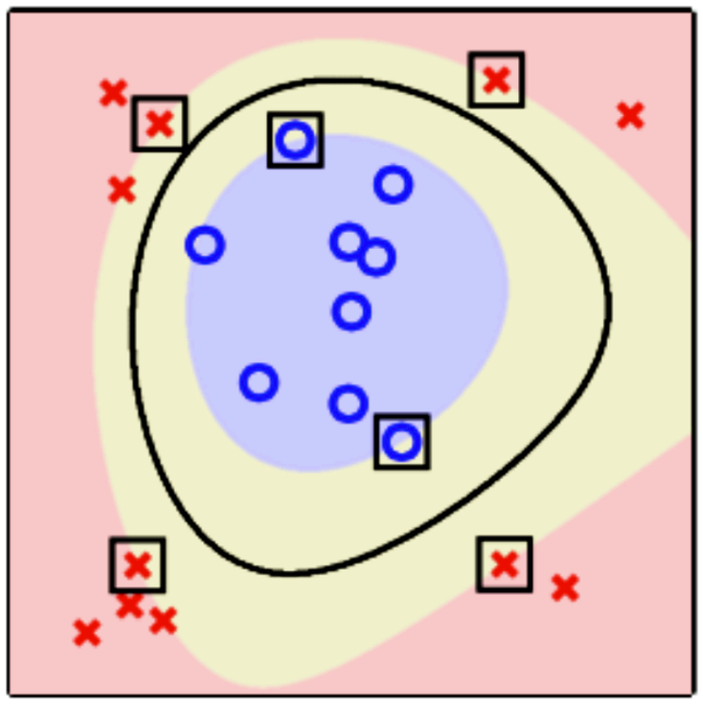
$$\text{subject to:} \quad y_n\left(\tilde{\mathbf{w}}^{\mathrm{T}}\mathbf{z}_n + \tilde{b}\right) \geq 1$$

$$(n = 1, \cdots, N)$$

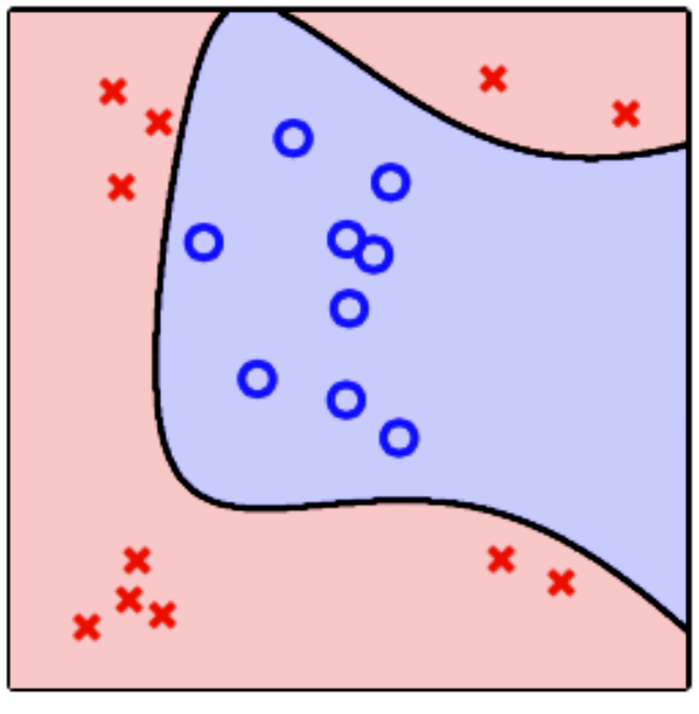
$$\Phi_2(\mathbf{x}) = (x_1, x_2, x_1^2, x_1x_2, x_2^2)$$



Φ_2+ SVM



Φ_3+ SVM



Φ_3+ pseudoinverse algorithm

Nonlinear transformation and SVM

Observations:

1. Φ_3 has almost $2\times$ the parameters of Φ_2
2. Φ_3-SVM does not display significant overfitting compared to Φ_3-regression
3. #support vectors did not double
4. Can go to higher dimensions if #support vectors stays small or margin stays large

	pseudoinverse regression		SVM	
	linear	nonlinear (ϕ)	linear	nonlinear (ϕ)
overfitting	little	lots	tiny	ok
boundary	linear	complex	linear	complex

Going to even higher dimension

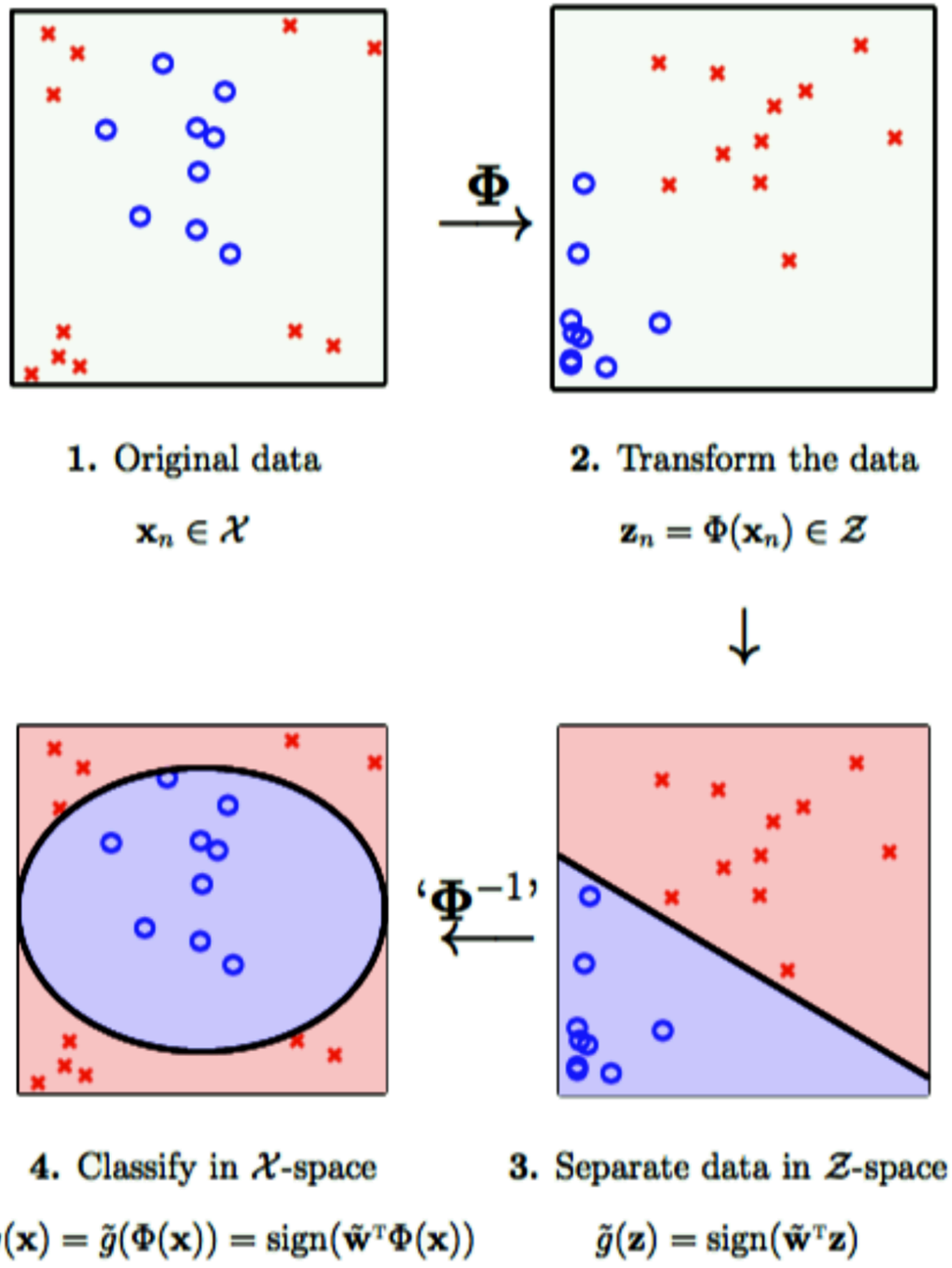
In higher dimension, can control overfitting with # SV or margin ρ

What about:

Efficiency?

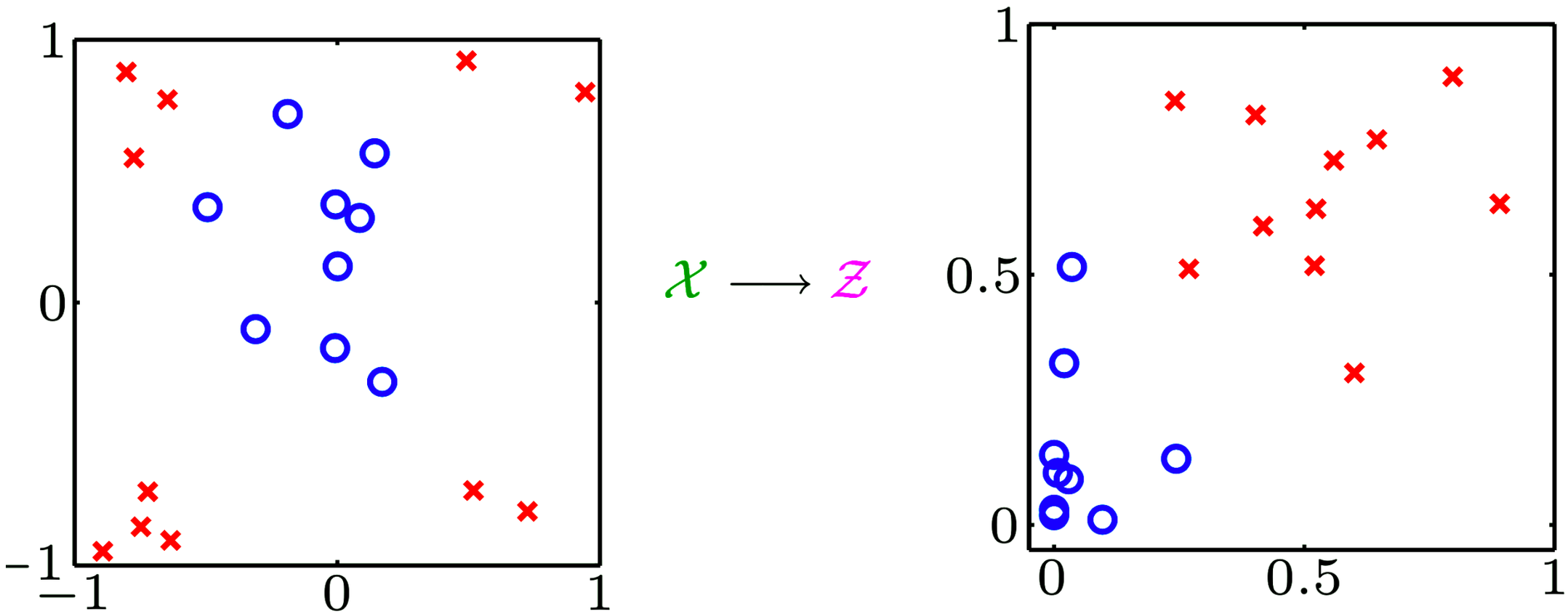
Infinitely many dimensions?

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



Going to even higher dimension

$$\mathcal{L}(\boldsymbol{\alpha}) \;=\; \sum_{n=1}^{N} \alpha_n \;-\; \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} y_n y_m \; \alpha_n \alpha_m \; \mathbf{z}_n^{\mathsf{T}} \mathbf{z}_m$$



$$\boxed{g(\mathbf{x}) = \operatorname{sign}\left(\mathbf{w}^{\mathsf{T}} \mathbf{z} + b\right)} \qquad \text{need} \quad \mathbf{z}_n^{\mathsf{T}} \mathbf{z}$$

where $\mathbf{w} \;=\; \sum_{\mathbf{z}_n \text{ is SV}} \alpha_n y_n \mathbf{z}_n$

and b: $y_m \left(\mathbf{w}^{\mathsf{T}} \mathbf{z}_m + b\right) = 1$ need $\mathbf{z}_n^{\mathsf{T}} \mathbf{z}_m$

Going to even higher dimension

Given two points $\mathbf{x}$ and $\mathbf{x}' \in \mathcal{X}$, we need $\mathbf{z}^\top \mathbf{z}'$

Let $\mathbf{z}^\top \mathbf{z}' = K(\mathbf{x}, \mathbf{x}')$ (the kernel) "inner product" of $\mathbf{x}$ and $\mathbf{x}'$

Example: $\mathbf{x} = (x_1, x_2) \longrightarrow$ 2nd-order Φ

$$\mathbf{z} = \Phi(\mathbf{x}) = (1, x_1, x_2, x_1^2, x_2^2, x_1 x_2)$$

$$K(\mathbf{x}, \mathbf{x}') = \mathbf{z}^\top \mathbf{z}' = 1 + x_1 x'_1 + x_2 x'_2 +$$

$$x_1^2 x'^2_1 + x_2^2 x'^2_2 + x_1 x'_1 x_2 x'_2$$

Kernel

Can we compute $K(\mathbf{x}, \mathbf{x}')$ **without** transforming $\mathbf{x}$ and $\mathbf{x}'$?

Example: Consider $K(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x}^\intercal \mathbf{x}')^2 = (1 + x_1 x'_1 + x_2 x'_2)^2$

$$= 1 + x_1^2 x'^2_1 + x_2^2 x'^2_2 + 2x_1 x'_1 + 2x_2 x'_2 + 2x_1 x'_1 x_2 x'_2$$

This is an inner product!

$$(1 , x_1^2 , x_2^2 , \sqrt{2}\, x_1 , \sqrt{2}\, x_2 , \sqrt{2}\, x_1 x_2)$$

$$(1 , x'^2_1 , x'^2_2 , \sqrt{2} x'_1 , \sqrt{2} x'_2 , \sqrt{2} x'_1 x'_2)$$

Design the kernel

2nd-Order Polynomial Kernel

$$\Phi(\mathbf{x}) = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \\ x_1^2 \\ x_2^2 \\ \vdots \\ x_d^2 \\ \sqrt{2}x_1x_2 \\ \sqrt{2}x_1x_3 \\ \vdots \\ \sqrt{2}x_1x_d \\ \sqrt{2}x_2x_3 \\ \vdots \\ \sqrt{2}x_{d-1}x_d \end{bmatrix}$$

$$\begin{aligned} K(\mathbf{x},\mathbf{x}') &= \Phi(\mathbf{x})^{\mathrm{T}}\Phi(\mathbf{x}') \\ &= \sum_{i=1}^{d} x_i x_i' + \sum_{i=1}^{d} x_i^2 x_i'^2 + 2\sum_{i<j} x_i x_j x_i' x_j' \qquad \leftarrow O(d^2) \\ &= \left(\frac{1}{2} + \mathbf{x}^{\mathrm{T}}\mathbf{x}'\right)^2 - \frac{1}{4} \end{aligned}$$

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

Q-th order polynomial kernel

$$K(\mathbf{x},\mathbf{x}') = (r + \mathbf{x}^{\mathrm{T}}\mathbf{x}')^Q \qquad \leftarrow \text{inhomogeneous kernel}$$

$$K(\mathbf{x},\mathbf{x}') = (\mathbf{x}^{\mathrm{T}}\mathbf{x}')^Q \qquad \leftarrow \text{homogeneous kernel}$$

The polynomial kernel

$\mathcal{X} = \mathbb{R}^d$ and $\Phi : \mathcal{X} \to \mathcal{Z}$ is polynomial of order Q

The "equivalent" kernel $K(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x}^\mathsf{T}\mathbf{x}')^Q$

$$= (1 + x_1x'_1 + x_2x'_2 + \cdots + x_dx'_d)^Q$$

Compare for $d = 10$ and $Q = 100$

Can adjust scale: $K(\mathbf{x}, \mathbf{x}') = (a\mathbf{x}^\mathsf{T}\mathbf{x}' + b)^Q$

Kernel formulation of SVM: QP

$$\underbrace{\begin{bmatrix} y_1 y_1 K(\mathbf{x}_1, \mathbf{x}_1) & y_1 y_2 K(\mathbf{x}_1, \mathbf{x}_2) & \dots & y_1 y_N K(\mathbf{x}_1, \mathbf{x}_N) \\ y_2 y_1 K(\mathbf{x}_2, \mathbf{x}_1) & y_2 y_2 K(\mathbf{x}_2, \mathbf{x}_2) & \dots & y_2 y_N K(\mathbf{x}_2, \mathbf{x}_N) \\ \dots & \dots & \dots & \dots \\ y_N y_1 K(\mathbf{x}_N, \mathbf{x}_1) & y_N y_2 K(\mathbf{x}_N, \mathbf{x}_2) & \dots & y_N y_N K(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix}}_{\text{quadratic coefficients}}$$

Everything else is the same.

Express $g(\mathbf{x}) = \text{sign}\left(\mathbf{w}^\mathsf{T}\mathbf{z} + b\right)$ in terms of $K(-,-)$

$$\mathbf{w} = \sum_{\mathbf{z}_n \text{ is SV}} \alpha_n y_n \mathbf{z}_n \quad \Longrightarrow \quad g(\mathbf{x}) = \text{sign}\left(\sum_{\alpha_n > 0} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}) + b\right)$$

where $b = y_m - \sum_{\alpha_n > 0} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}_m)$

for any support vector $(\alpha_m > 0)$

RBF-Kernel

One dimensional RBF-Kernel

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

$$\begin{aligned} K(\mathbf{x}, \mathbf{x}') &= \Phi(\mathbf{x})^{\mathrm{T}}\Phi(\mathbf{x}') \\ &= e^{-x^2}e^{-x'^2}\sum_{i=0}^{\infty}\frac{(2xx')^i}{i!} \\ &= e^{-x^2}e^{-x'^2}e^{2xx'} \\ &= e^{-(x-x')^2} \end{aligned}$$

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

d-dimensional RBF-Kernel

$$K(\mathbf{x}, \mathbf{x}') = e^{-\gamma\|\mathbf{x}-\mathbf{x}'\|^2} \qquad (\gamma > 0)$$

We only need $\mathcal{Z}$-space to exist

If $K(\mathbf{x}, \mathbf{x}')$ is an inner product in some space $\mathcal{Z}$, we are good.

Example: $$K(\mathbf{x}, \mathbf{x}') \;=\; \exp\left(-\gamma \, \|\mathbf{x} - \mathbf{x}'\|^2\right)$$

Infinite-dimensional $\mathcal{Z}$: take simple case

$$K(x, x') \;=\; \exp\left(-(x - x')^2\right)$$

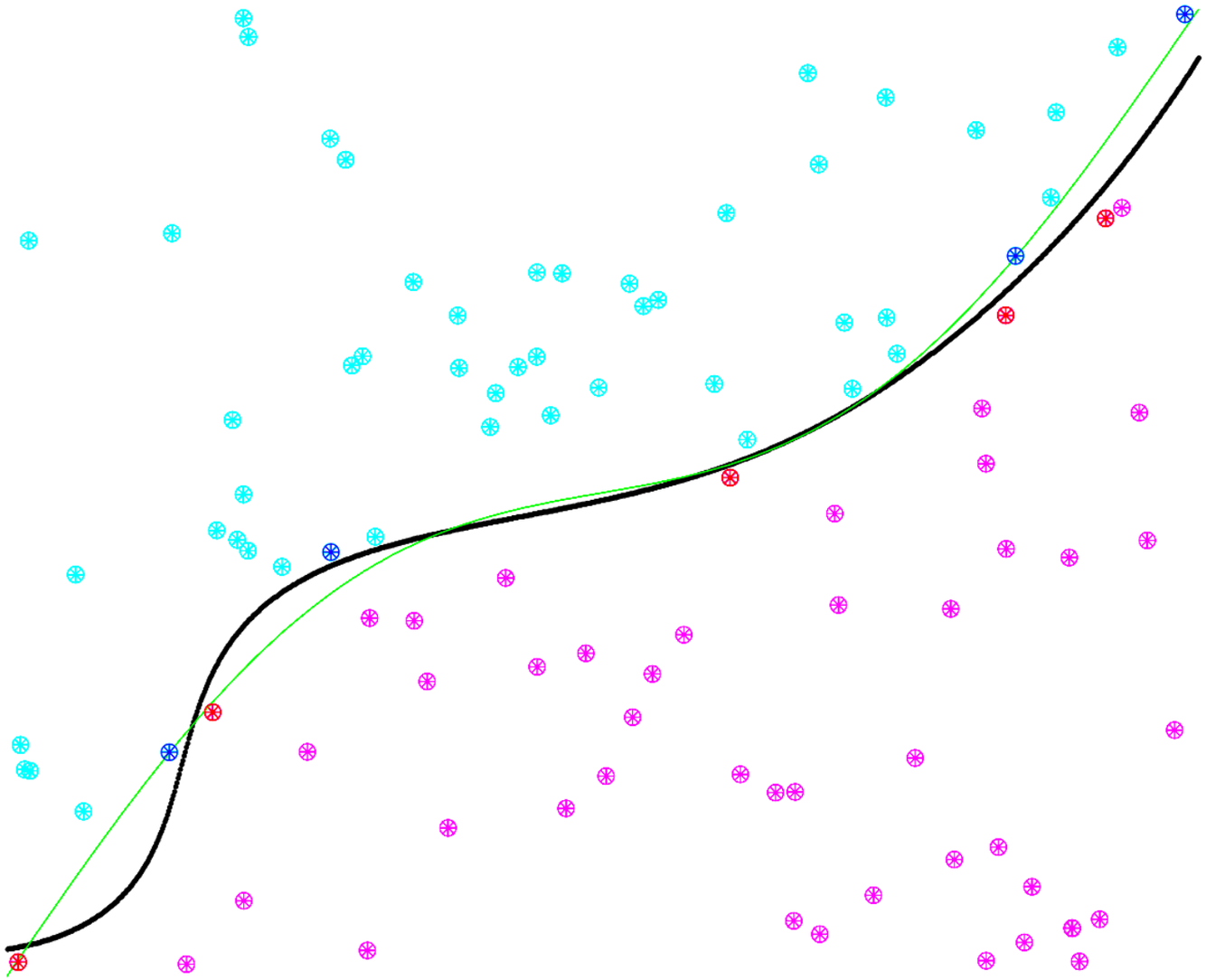
$$= \exp\left(-x^2\right) \exp\left(-x'^2\right) \underbrace{\sum_{k=0}^{\infty} \frac{2^k (x)^k (x')^k}{k!}}_{\exp(2xx')}$$

Example

Slightly non-separable case:

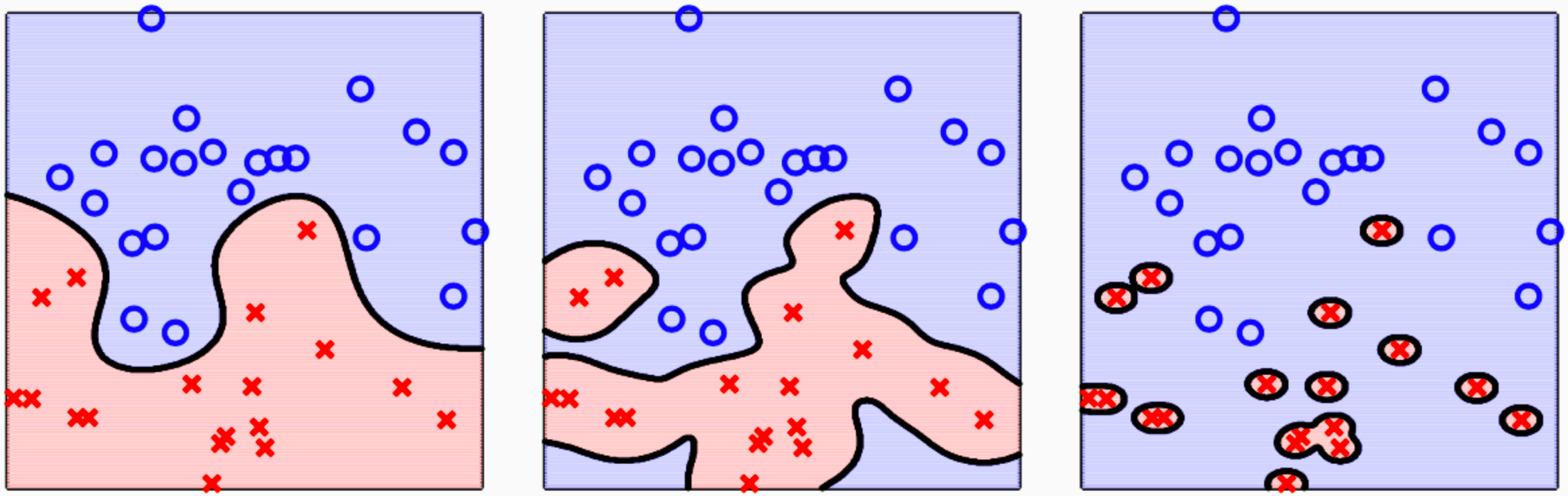
Transforming $\mathcal{X}$ into ∞-dimensional $\mathcal{Z}$

Overkill? Count the support vectors



Choosing RBF-Kernel Width

$$e^{-\gamma \| \mathbf{x} - \mathbf{x}' \|^2}$$



Small γ

Medium γ

Large γ!

Design the kernel

$K(\mathbf{x}, \mathbf{x}')$ is a valid kernel iff

1. It is symmetric and 2.

2. The matrix:

$$\begin{bmatrix} K(\mathbf{x}_1, \mathbf{x}_1) & K(\mathbf{x}_1, \mathbf{x}_2) & \dots & K(\mathbf{x}_1, \mathbf{x}_N) \\ K(\mathbf{x}_2, \mathbf{x}_1) & K(\mathbf{x}_2, \mathbf{x}_2) & \dots & K(\mathbf{x}_2, \mathbf{x}_N) \\ \dots & \dots & \dots & \dots \\ K(\mathbf{x}_N, \mathbf{x}_1) & K(\mathbf{x}_N, \mathbf{x}_2) & \dots & K(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix}$$

is **positive semi-definite**

for any $\mathbf{x}_1, \cdots, \mathbf{x}_N$ (Mercer's condition)

Design the kernel

a given $K(\mathbf{x}, \mathbf{x}')$ is a valid kernel?

Three approaches:

1. By construction
2. Math properties (*Mercer's condition*)
3. Who cares? ☺

Designing Kernels

$$
\begin{aligned}
& K(\mathbf{x}, \mathbf{x}') \\
=& \mathbf{z}^T \mathbf{z}' \\
=& \|\mathbf{z}\| \cdot \|\mathbf{z}'\| \cdot \cos(\theta_{\mathbf{z},\mathbf{z}'}) \\
=& \|\mathbf{z}\| \cdot \|\mathbf{z}'\| \cdot \mathsf{CosSim}(\mathbf{z}, \mathbf{z}')
\end{aligned}
$$

- Construct a similarity measure for the data
- A linear model should be plausible in that transformed space

Comparison of SVM & Logistic Regression

- SVM works well with unstructured and semi-structured data like text and images (Image classification, Recognizing handwriting, Caner detection) while logistic regression works with already identified independent variables.
- SVM is based on geometrical properties of the data while logistic regression is based on statistical approaches.
- The risk of overfitting is less in SVM, while Logistic regression is vulnerable to overfitting.
- Multi-class case
- One-class case
- Regression
- Posterior probability

Comparison of SVM & Logistic Regression

- n = number of features ($x \in \mathbb{R}^{n+1}$), m = number of training examples
- If n is large (relative to m): use logistic regression, or SVM without a kernel

  (e.g. $n \geq m, n = 10^4, m = 10 \sim 10^3$)
- If n is small, m is intermediate: use SVM with Gaussian kernel

  (e.g. $n = 1 \sim 10^3, m = 10 \sim 10^4$)
- If n is small, m is large: create/add more features, then use logistic regression or SVM without a kernel

  (e.g. $n = 1 \sim 10^3, m \geq 5 \times 10^4$)
- Neural network likely to work well for most of these settings, but slower to train