Learning with Support Vectors

Presentation by: Venugopalakrishna Y. R., SPC Lab, IISc
(Prof. Yaser S. Abu-Mostafa's ML course slides are used to explain SVMs)

1^{st}, Sep 2012

Outline

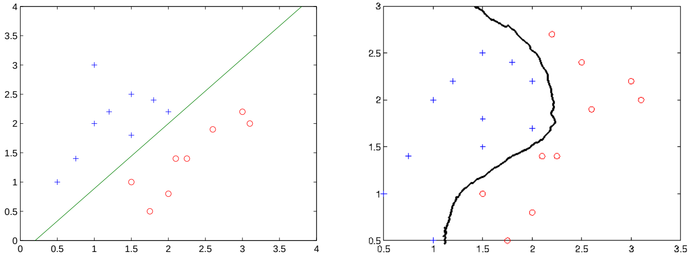
- Introduction to Machine Learning
- Notion of Similarity
- A Simple Pattern Recognition Algorithm
- Learning Theory and Learning algorithms
- Support Vector Machines

Introduction to Machine Learning

- Learning the pattern in the data to find a rule to predict
- Input patterns: $x_1, x_2, \ldots, x_m \in \mathcal{X}$
- Outputs: $y_1, x_2, \ldots, y_m \in \mathcal{Y}$
- Supervised learning and Unsupervised learning

Supervised Learning: Classification

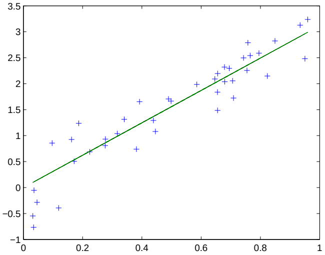
- Training data: $\{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\} \in \mathcal{X} \times \{\pm 1\}$
- Example: Binary Classification



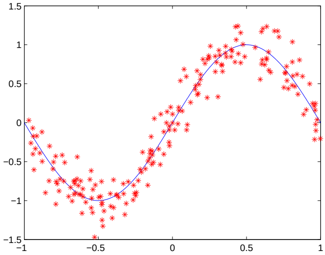
(a) (b)

Supervised Learning: Regression

- Training data: $\{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\} \in \mathcal{X} \times \mathbb{R}$



(c) Linear Regression



(d) Non-linear Regression

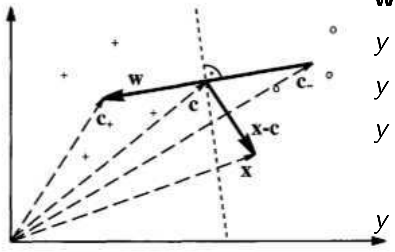
Similarity in data

- Goal: Learn a function that agrees with training data and generalizes for unseen data
- Given a new pattern $x \in \mathcal{X}$, chose a y s.t. (x, y) is similar to training data
- Need to map the input patterns to a space where the similarity in data can be measured
- $k : \mathcal{X} \times \mathcal{X} \to \mathbf{R}$
- k is symmetric, i.e. $k(x, x') = k(x', x)$

Dot Product

- Let $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^N$, simple similarity measure is $\langle \mathbf{x}, \mathbf{x}' \rangle$
- $\|\mathbf{x}\| = \sqrt{\langle \mathbf{x}, \mathbf{x}' \rangle}$, $\cos\theta = \frac{<\mathbf{x},\mathbf{x}'>}{\|\mathbf{x}\| \ \|\mathbf{x}'\|}$
- Distance between two vectors $\mathbf{x}$ and $\mathbf{z}$ is $\|\mathbf{x} - \mathbf{z}\|$
- Map $x \in \mathcal{X}$ to a space $\mathcal{H}$ where dot product is defined
- If $\Phi : \mathcal{X} \to \mathcal{H}$,then $k(x, x') := \langle \mathbf{x}, \mathbf{x}' \rangle =< \Phi(x), \Phi(x') >$

A Binary Pattern Recognition Example



$\mathbf{c}_+$ and $\mathbf{c}_-$ are class means

$\mathbf{w} = \mathbf{c}_+ - \mathbf{c}_-$ and $\mathbf{c} = \frac{\mathbf{c}_+ + \mathbf{c}_-}{2}$

$y = sgn(\langle \mathbf{x} - c, \mathbf{w} \rangle)$

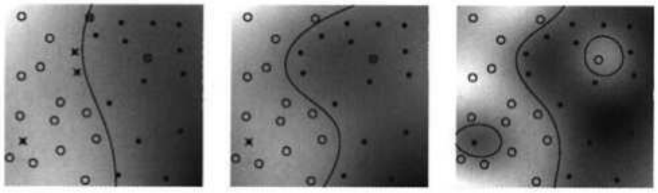
$y = sgn(\langle \mathbf{x}, \mathbf{c}_+ \rangle - \langle \mathbf{x} - \mathbf{c}_- \rangle + b)$

$$y = sgn(\frac{1}{m_+} \sum_{i:y_i=+1} k(x, x_i) - \frac{1}{m_-} \sum_{i:y_i=-1} k(x, x_i))$$

$$y = sgn(\sum_{i=1}^{m} \alpha_i k(x, x_i) + b)$$

- Generally, PR algorithms have this form with kernels centered on training examples
- All input patterns may not be used

Learning Theory



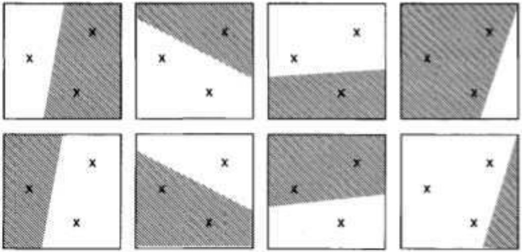
- Learning Theory helps in designing algorithm which choses a function class that leads to small test error

Error in learning

- Let the (x, y) is drawn independently from unknown $\mathbf{P}(x, y)$, and our prediction is $f(x)$
- Loss function: $\frac{|f(x)-y|}{2}$
- Empirical risk: $R_{emp}(f) = \frac{1}{2m}\sum_{i=1}^{m}|f(x) - y|$
- Actual risk: $R(f) = \frac{1}{2}\int |f(x) - y| d\mathbf{P}(x, y)$
- Small empirical risk doesn't imply small actual risk
- So function class of f is restricted to the one which has capacity to suit amount of training data

Capacity concept: VC Dimension

- m input patterns can be labelled in 2^m ways
- A rich function class can realize all 2^m separations, then it is said to shatter all m patterns



- VC Dimension: The largest number of input patterns h, that a fuction class can shatter

VC Bound

- If $h < m$, is the VC dimension of a function class that a learning machine can implement, independent of $\mathbf{P}(x, y)$ generating (x, y), with probabiltiy at least $1 - \delta$
  $R(f) \leq R_{emp}(f) + \phi(h, m, \delta)$ holds
  where $\phi(h, m, \delta) = \sqrt{\frac{1}{m}(h(\ln \frac{2m}{h} + 1) + \ln \frac{4}{\delta})}$
- When $\mathbf{P}(x, y) = \mathbf{P}(x)\mathbf{P}(y)$ with ± 1 equally likely, no good way to predict class of test pattern
- With a function class of large h, we can make training error zero, but $\phi(h, m, \delta)$ so test error is large
- To make non-trivial prediction about test error, function class must be restricted

Support Vector Classification

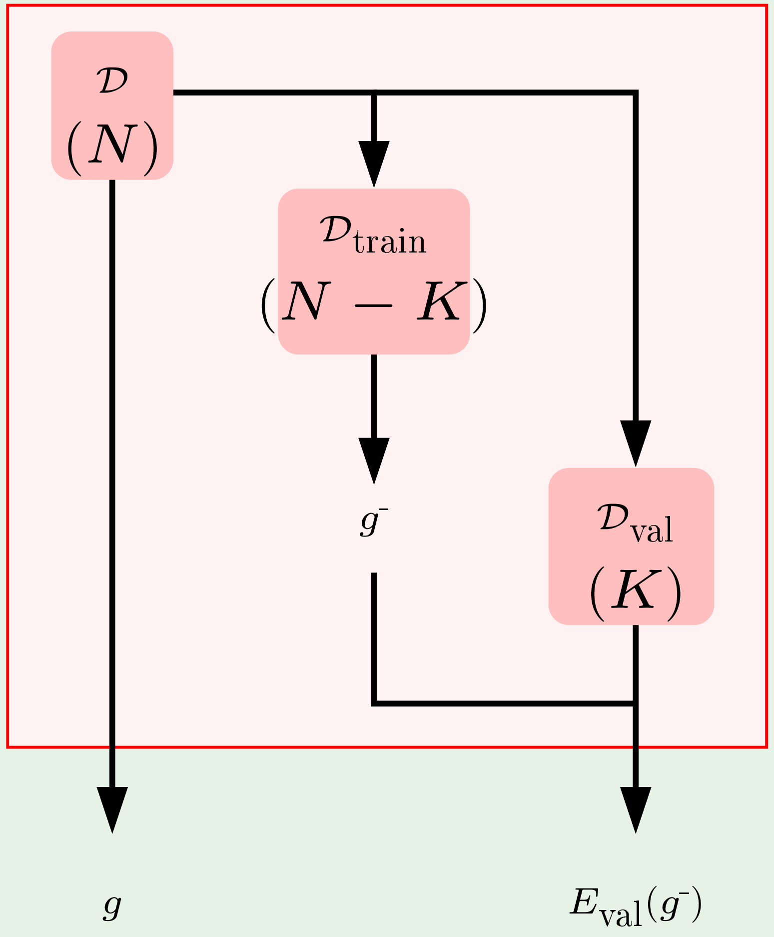
- Vapnik considered the class of (linearly separable) hyperplanes in $\mathcal{H}$

  i.e. $\mathbf{w}^t\mathbf{x} + b = 0$ where $\mathbf{w} \in \mathcal{H}$ and $b \in \mathbb{R}$ correponding to decision functions $f(x) = sgn(\mathbf{w}^t\mathbf{x} + b)$
- Maximizing the separation between any training point and hyperplane
- $\max_{\mathbf{w},b} \min\{\|\mathbf{x} - \mathbf{x}_i\| : \mathbf{x} \in \mathcal{H}, \mathbf{w}^t\mathbf{x} + b = 0, i = 1, \ldots, m\}$

I have used slides from Prof. Yaser S. Abu-Mostafa's course on SVMs.

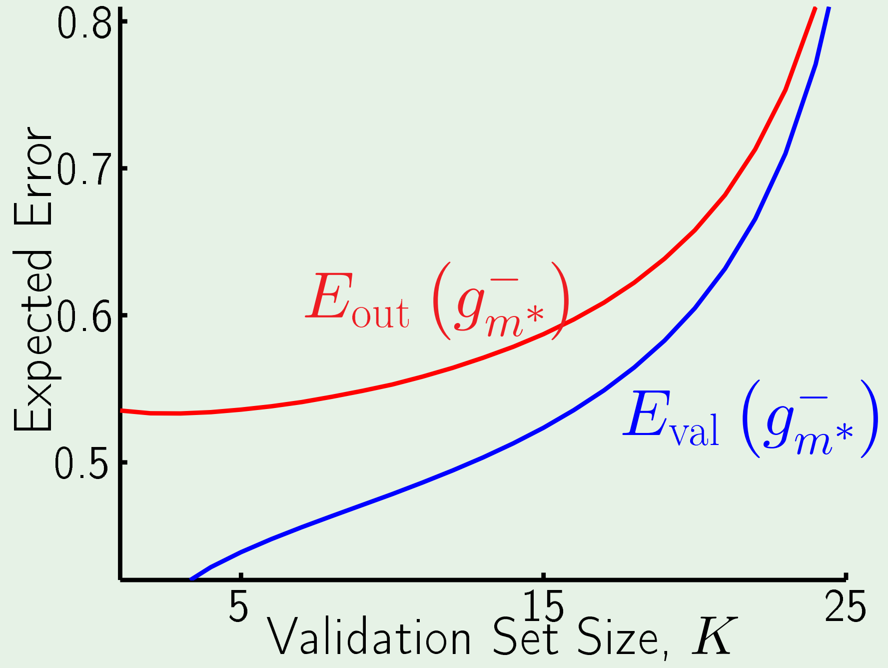
Review of Lecture 13

- Validation



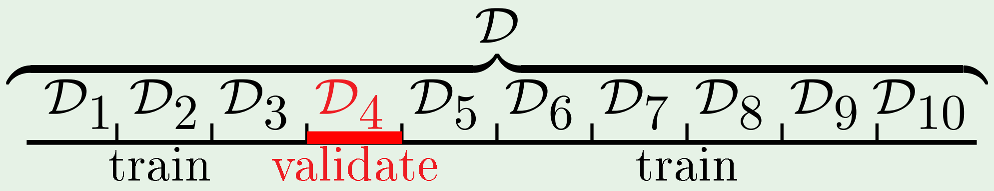
$E_{\text{val}}(g^-)$ estimates $E_{\text{out}}(g)$

- Data contamination



$\mathcal{D}_{\text{val}}$ slightly contaminated

- Cross validation



10-fold cross validation

Learning From Data

Yaser S. Abu-Mostafa
California Institute of Technology

Lecture 14: Support Vector Machines

Sponsored by Caltech's Provost Office, E&AS Division, and IST • Thursday, May 17, 2012

Outline

- Maximizing the margin

- The solution

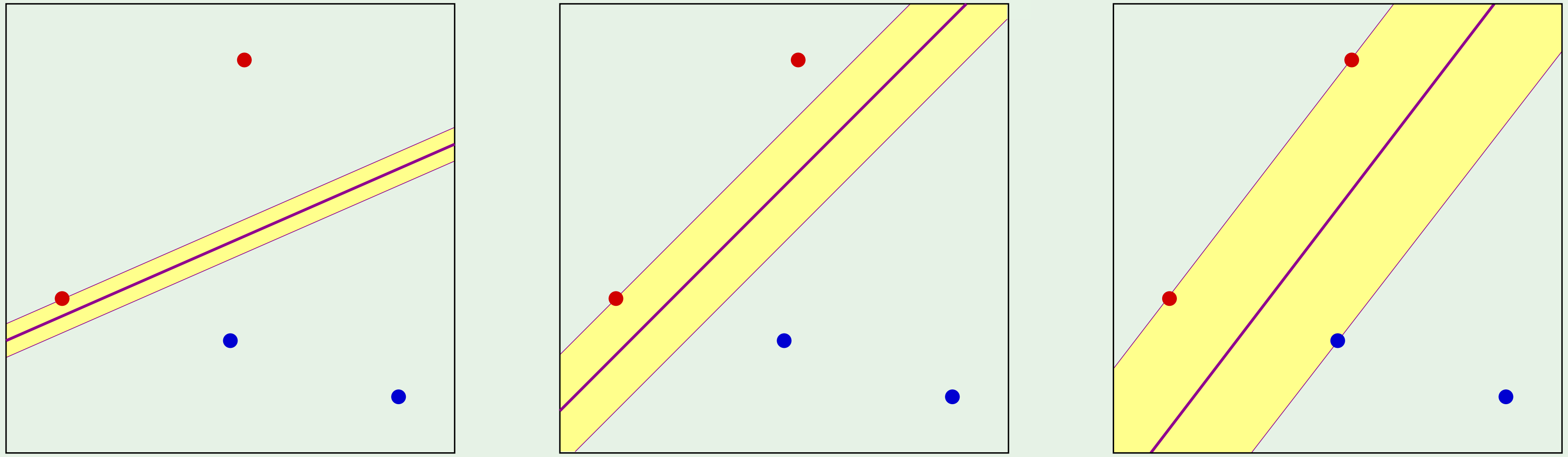
- Nonlinear transforms

Better linear separation

Linearly separable data

Different separating lines

Which is best?



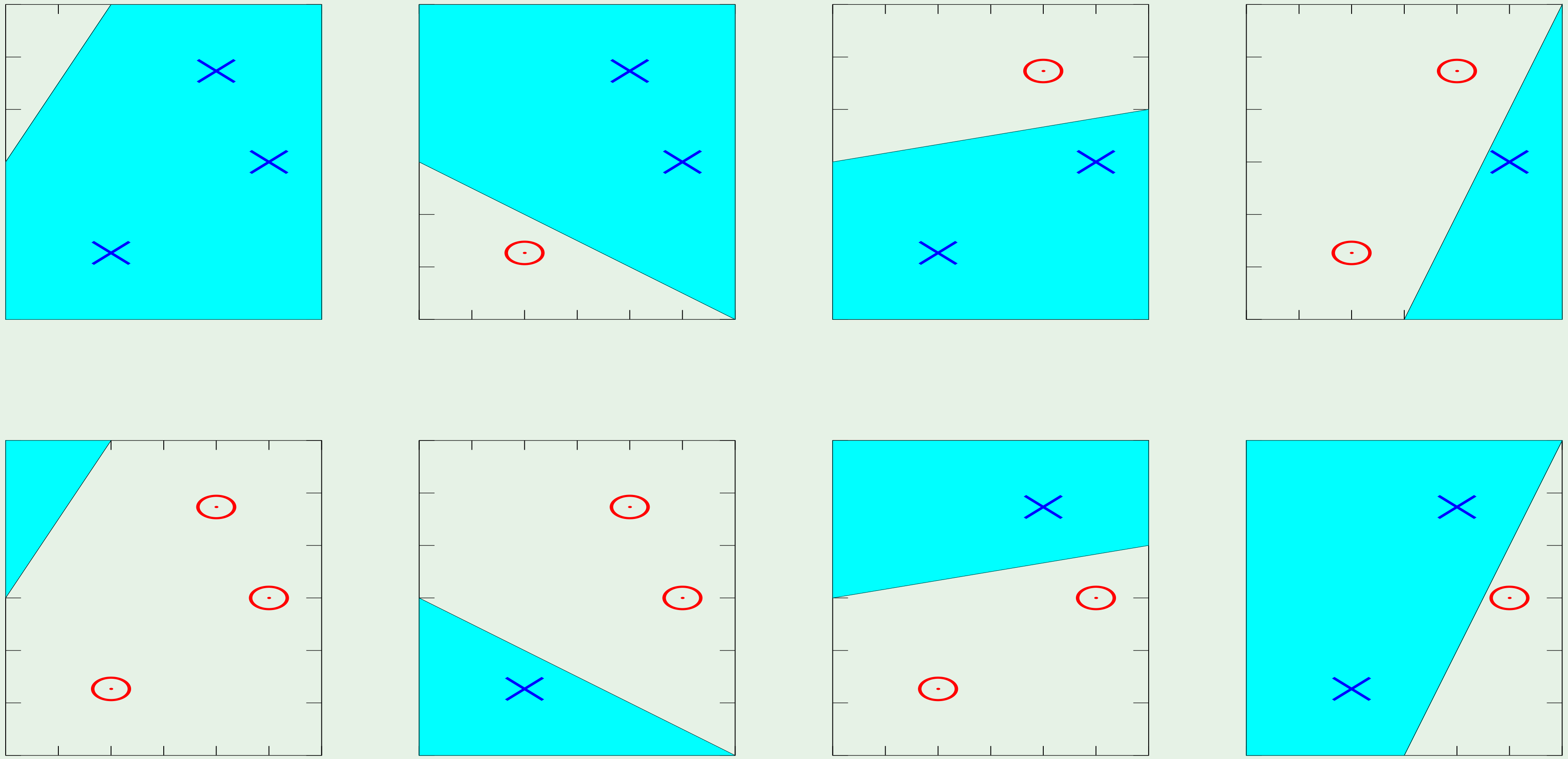
Two questions:

1. Why is bigger margin better?

2. Which $\mathbf{w}$ maximizes the margin?

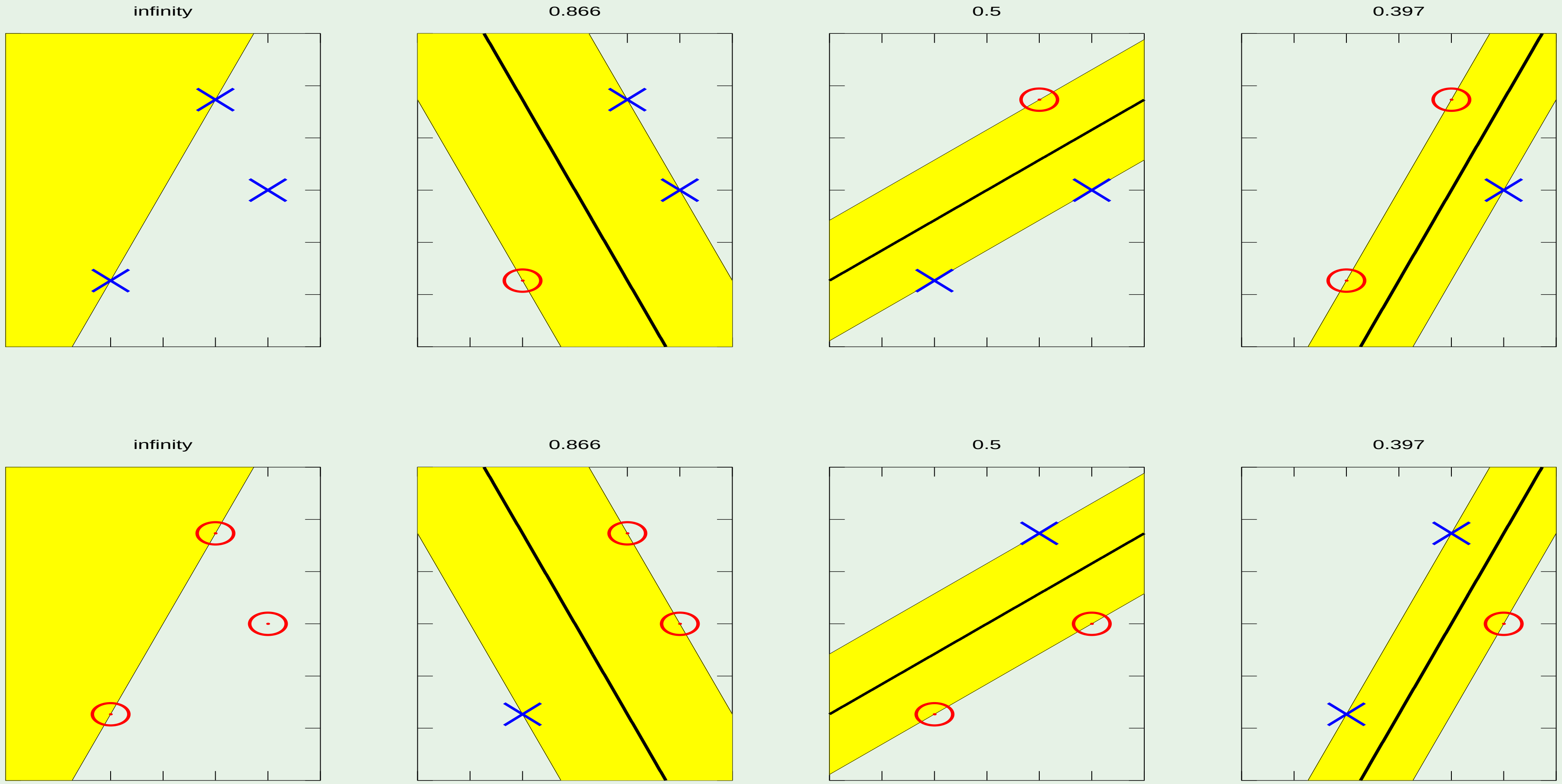
Remember the growth function?

All dichotomies with any line:



Dichotomies with fat margin

Fat margins imply fewer dichotomies



Finding $\mathbf{w}$ with large margin

Let $\mathbf{x}_n$ be the nearest data point to the plane $\mathbf{w}^\intercal\mathbf{x} = 0$. How far is it?

2 preliminary technicalities:

1. **Normalize $\mathbf{w}$:**

   $$|\mathbf{w}^\intercal\mathbf{x}_n| = 1$$

2. **Pull out w_0:**

   $\mathbf{w} = (w_1, \cdots, w_d)$ apart from b

   The plane is now $\boxed{\mathbf{w}^\intercal\mathbf{x} + b = 0}$ (no x_0)

Computing the distance

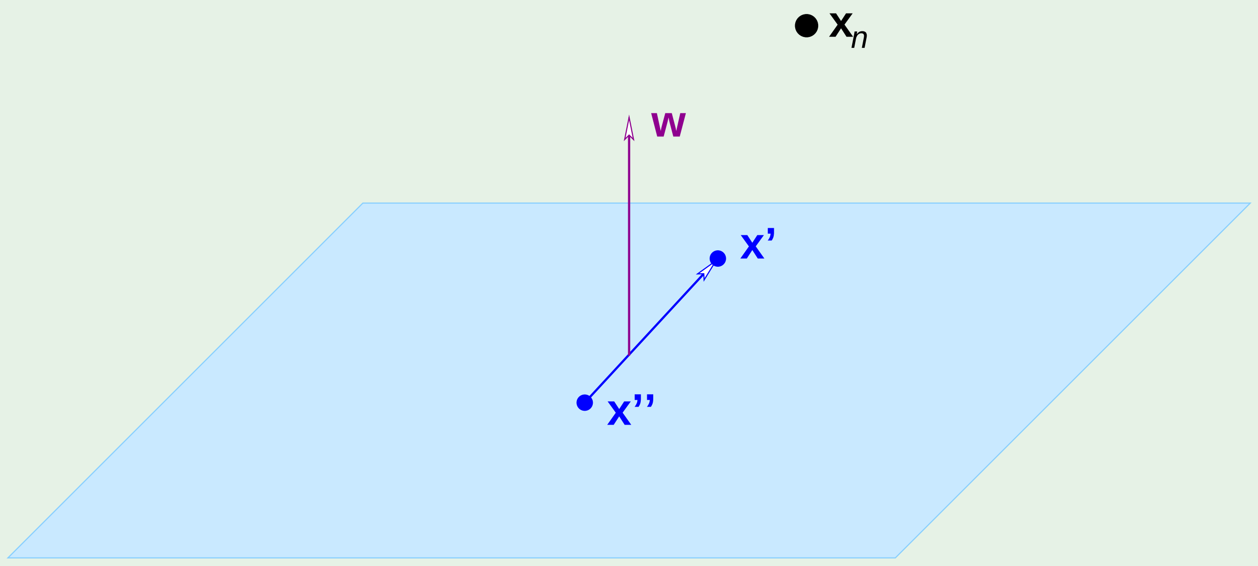
The distance between $\mathbf{x}_n$ and the plane $\mathbf{w}^{\intercal}\mathbf{x} + b = 0$ where $|\mathbf{w}^{\intercal}\mathbf{x}_n + b| = 1$

The vector $\mathbf{w}$ is $\perp$ to the plane in the $\mathcal{X}$ space:

Take $\mathbf{x}'$ and $\mathbf{x}''$ on the plane

$$\mathbf{w}^{\intercal}\mathbf{x}' + b = 0 \quad \text{and} \quad \mathbf{w}^{\intercal}\mathbf{x}'' + b = 0$$

$$\Longrightarrow \quad \mathbf{w}^{\intercal}(\mathbf{x}' - \mathbf{x}'') = 0$$



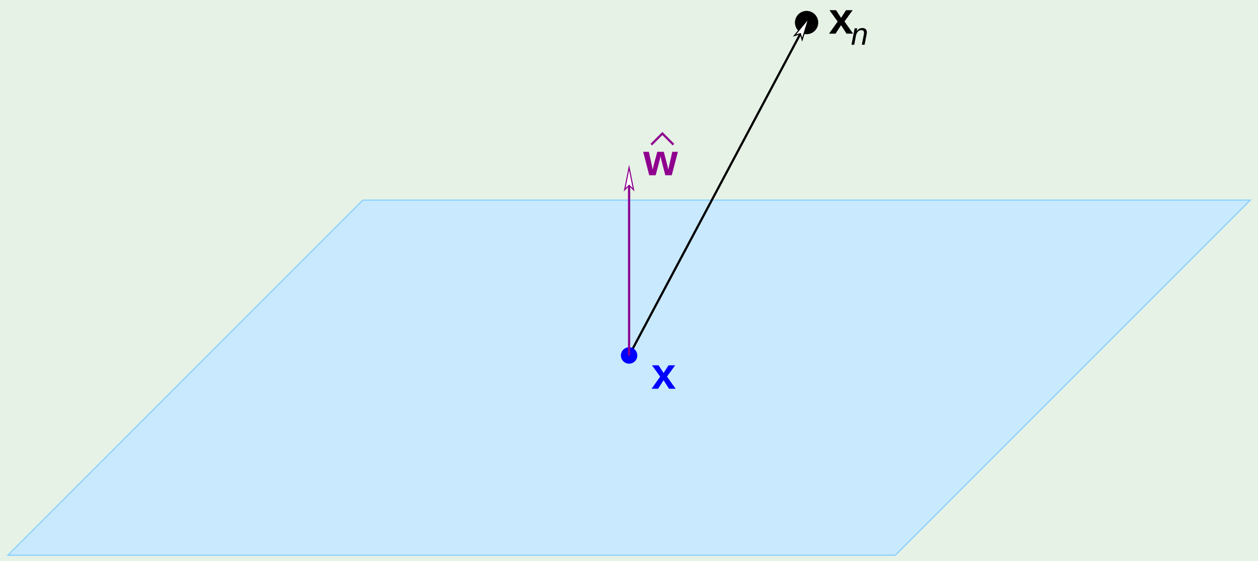
and the distance is ...

Distance between $\mathbf{x}_n$ and the plane:

Take any point $\mathbf{x}$ on the plane

Projection of $\mathbf{x}_n - \mathbf{x}$ on $\mathbf{w}$

$$\hat{\mathbf{w}} = \frac{\mathbf{w}}{\|\mathbf{w}\|} \implies \text{distance} = \left|\hat{\mathbf{w}}^{\mathsf{T}}(\mathbf{x}_n - \mathbf{x})\right|$$



$$\text{distance} = \frac{1}{\|\mathbf{w}\|}\left|\mathbf{w}^{\mathsf{T}}\mathbf{x}_n - \mathbf{w}^{\mathsf{T}}\mathbf{x}\right| = \frac{1}{\|\mathbf{w}\|}\left|\mathbf{w}^{\mathsf{T}}\mathbf{x}_n + b - \mathbf{w}^{\mathsf{T}}\mathbf{x} - b\right| = \frac{1}{\|\mathbf{w}\|}$$

The optimization problem

Maximize $\dfrac{1}{\|\mathbf{w}\|}$

subject to $\min\limits_{n=1,2,\ldots,N} |\mathbf{w}^\intercal \mathbf{x}_n + b| = 1$

Notice: $|\mathbf{w}^\intercal \mathbf{x}_n + b| = y_n \left(\mathbf{w}^\intercal \mathbf{x}_n + b\right)$

Minimize $\dfrac{1}{2}\, \mathbf{w}^\intercal \mathbf{w}$

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

Outline

- Maximizing the margin
- The solution
- Nonlinear transforms

Constrained optimization

Minimize $\quad \frac{1}{2}\, \mathbf{w}^{\intercal}\mathbf{w}$

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

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

Lagrange? inequality constraints $\Longrightarrow$ KKT

We saw this before

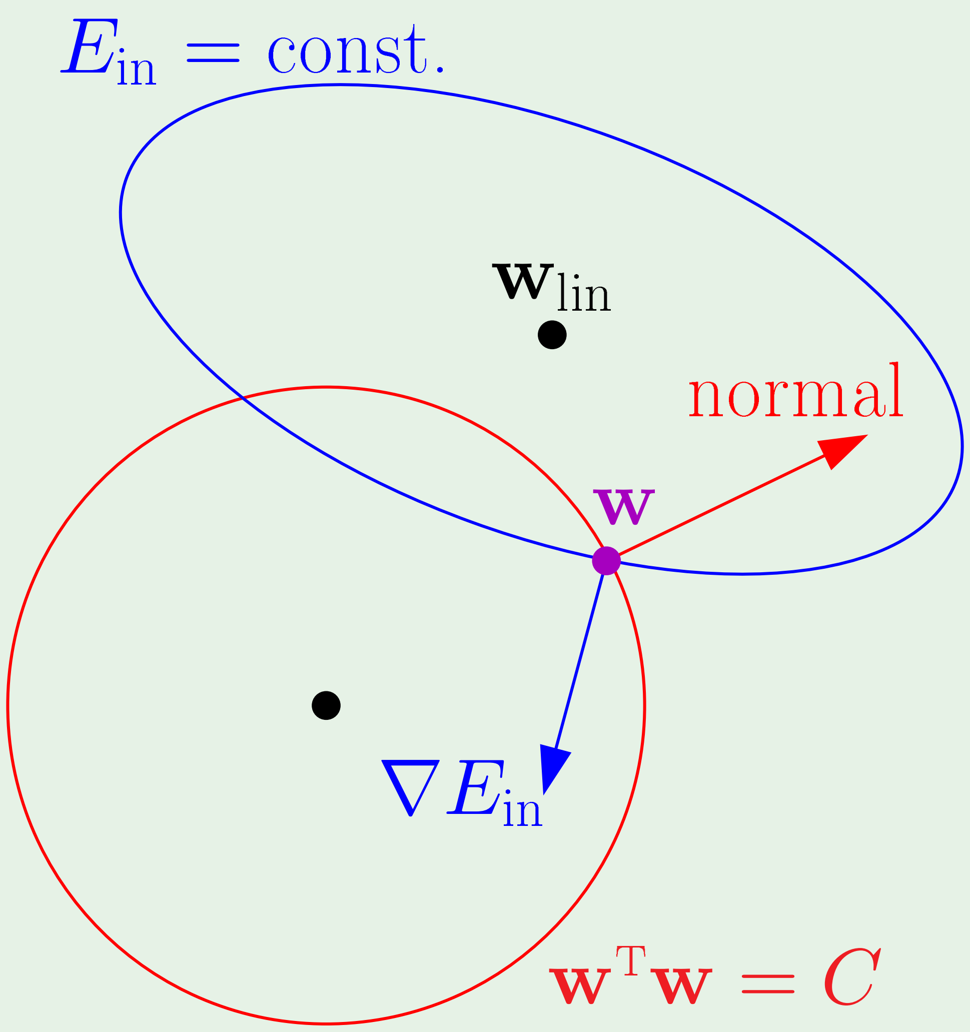
Remember regularization?

Minimize $\quad E_{\text{in}}(\mathbf{w}) = \frac{1}{N}(\mathrm{Z}\mathbf{w} - \mathbf{y})^{\intercal}(\mathrm{Z}\mathbf{w} - \mathbf{y})$

subject to: $\quad \mathbf{w}^{\intercal}\mathbf{w} \leq C$

∇E_{in} normal to constraint

	optimize	**constrain**
Regularization:	E_{in}	$\mathbf{w}^{\intercal}\mathbf{w}$
SVM:	$\mathbf{w}^{\intercal}\mathbf{w}$	E_{in}



Lagrange formulation

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

w.r.t. $\mathbf{w}$ and b and maximize w.r.t. each $\alpha_n \geq 0$

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

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

Substituting ...

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

in the Lagrangian

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

we get

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

Maximize w.r.t. to $\boldsymbol{\alpha}$ subject to $\alpha_n \geq 0$ for $n = 1, \cdots, N$ **and** $\sum_{n=1}^{N} \alpha_n y_n = 0$

The solution - quadratic programming

$$\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\,\boldsymbol{\alpha}^{\intercal} \underbrace{\begin{bmatrix} y_1y_1\ \mathbf{x}_1^{\intercal}\mathbf{x}_1 & y_1y_2\ \mathbf{x}_1^{\intercal}\mathbf{x}_2 & \ldots & y_1y_N\ \mathbf{x}_1^{\intercal}\mathbf{x}_N \\ y_2y_1\ \mathbf{x}_2^{\intercal}\mathbf{x}_1 & y_2y_2\ \mathbf{x}_2^{\intercal}\mathbf{x}_2 & \ldots & y_2y_N\ \mathbf{x}_2^{\intercal}\mathbf{x}_N \\ \ldots & \ldots & \ldots & \ldots \\ y_Ny_1\ \mathbf{x}_N^{\intercal}\mathbf{x}_1 & y_Ny_2\ \mathbf{x}_N^{\intercal}\mathbf{x}_2 & \ldots & y_Ny_N\ \mathbf{x}_N^{\intercal}\mathbf{x}_N \end{bmatrix}}_{\text{quadratic coefficients}} \boldsymbol{\alpha} + \underbrace{(-\mathbf{1}^{\intercal})}_{\text{linear}}\boldsymbol{\alpha}$$

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

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

QP hands us $\boldsymbol{\alpha}$

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

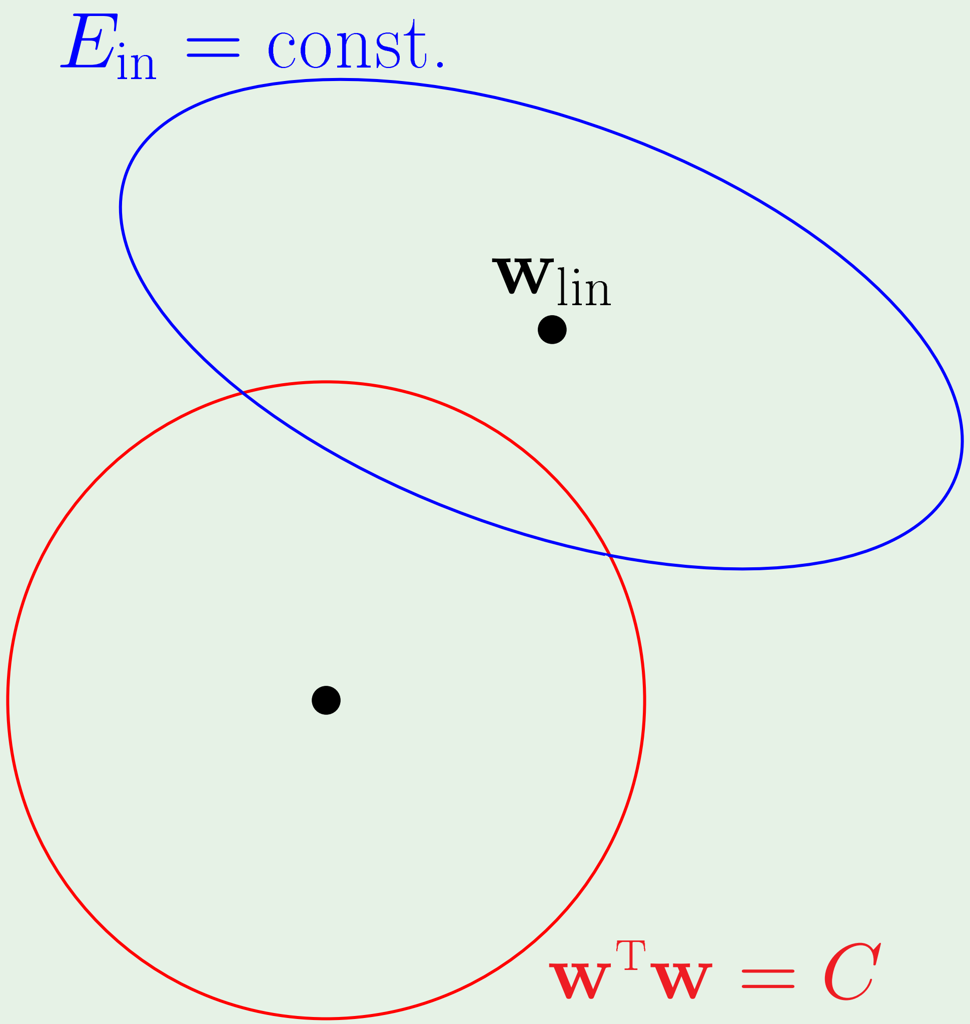
$$\Longrightarrow \quad \mathbf{w} \; = \; \sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n$$

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

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

We saw this before!

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



Support vectors

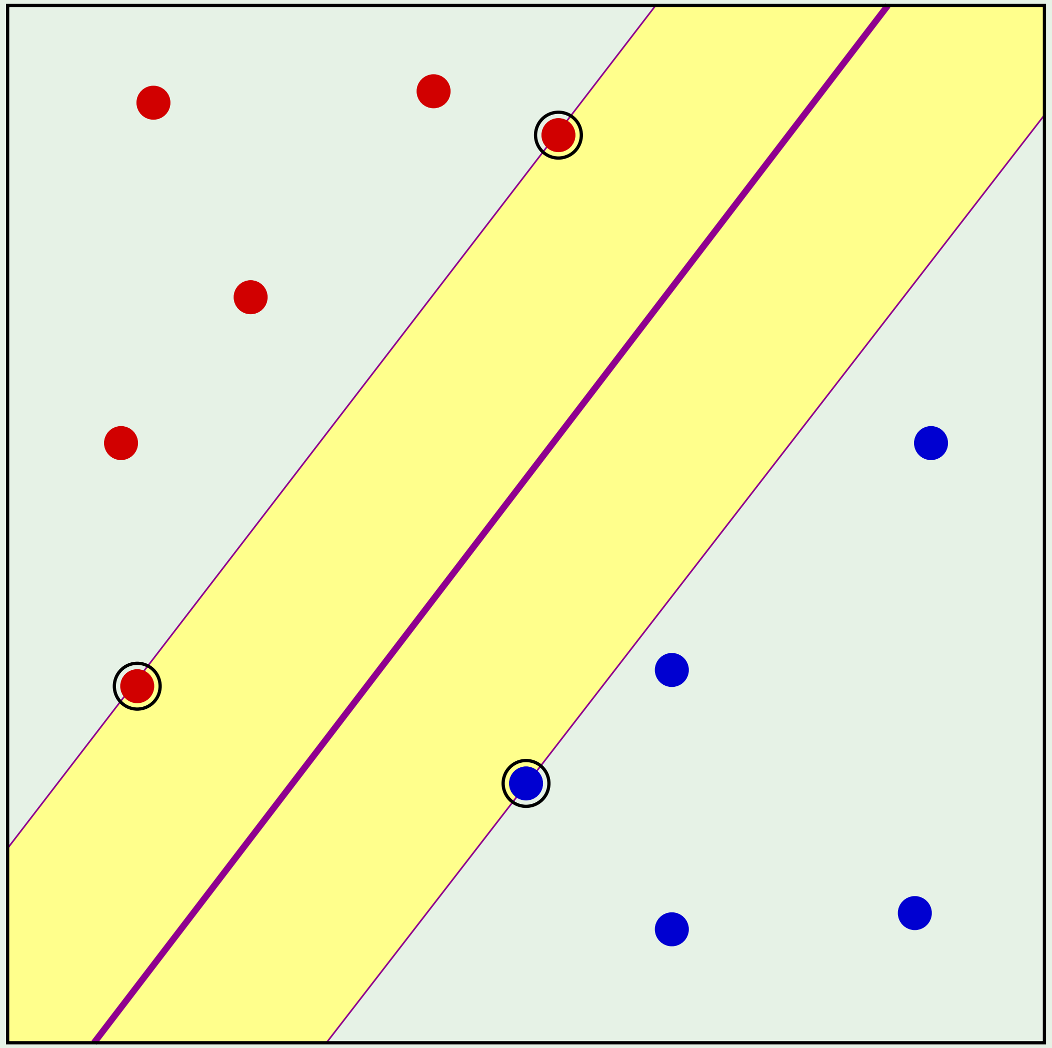
Closest $\mathbf{x}_n$'s to the plane: achieve the margin

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

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

Solve for b using any SV:

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

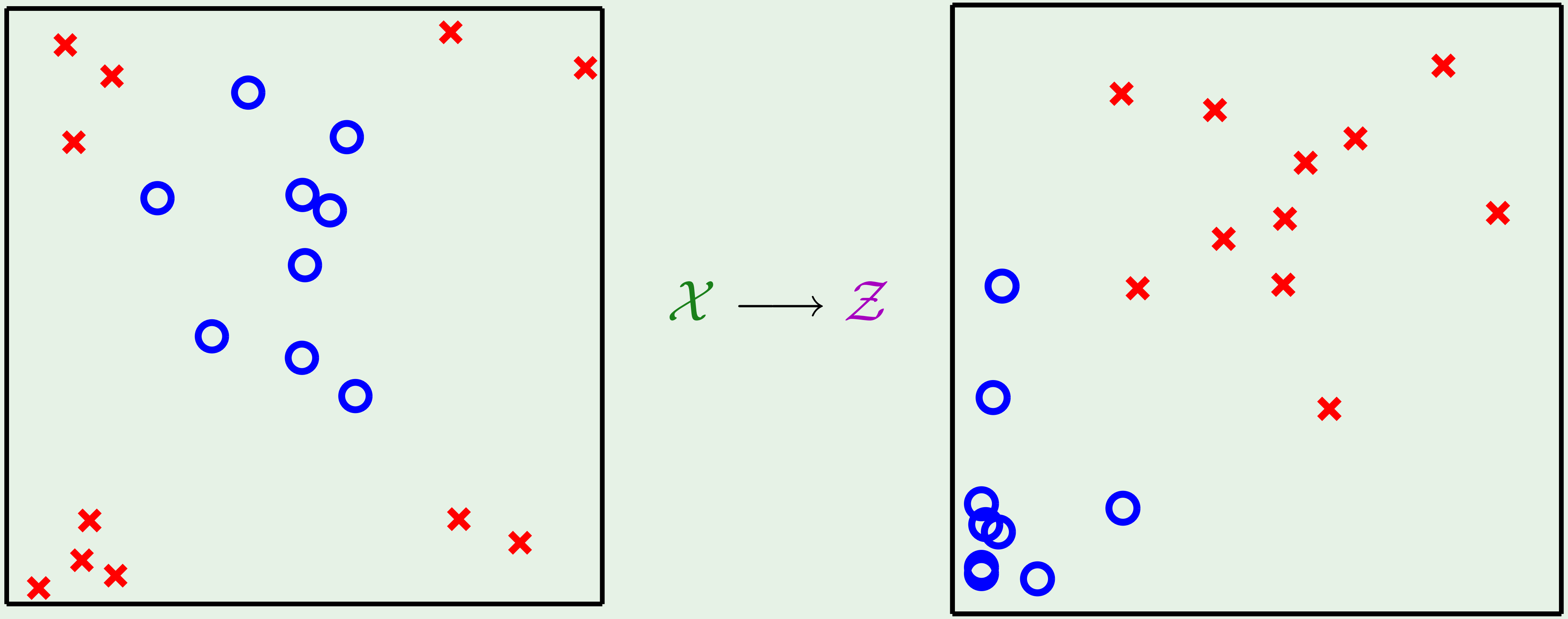


Outline

- Maximizing the margin
- The solution
- Nonlinear transforms

z instead of x

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



"Support vectors" in $\mathcal{X}$ space

Support vectors live in $\mathcal{Z}$ space

In $\mathcal{X}$ space, "pre-images" of support vectors

The margin is maintained in $\mathcal{Z}$ space

Generalization result

$$\mathbb{E}[E_{\text{out}}] \leq \frac{\mathbb{E}[\#\text{ of SV's}]}{N-1}$$