LEUPHANA
UNIVERSITÄT LÜNEBURG

INTRODUCTION TO MACHINE LEARNING

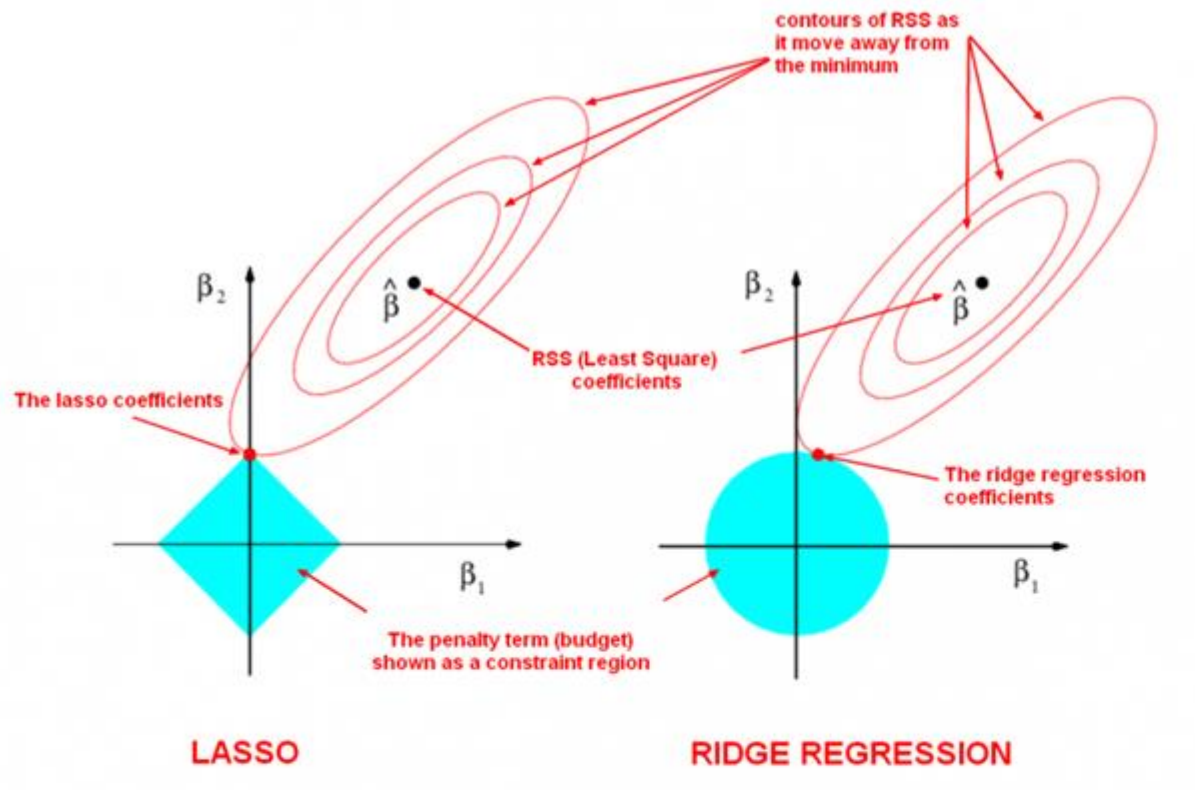
01. Support Vector Machines
Winter 2020/2021

Recap: regularized regression

— In LASSO regression we use

$$E_{\text{in}}(\mathbf{w}) + \frac{\lambda}{N} |\text{w}|$$

— LASSO regression can be used for variable selection



Source: https://www.quora.com/How-would-you-describe-the-difference-between-linear-regression-lasso-regression-and-ridge-regression

Agenda

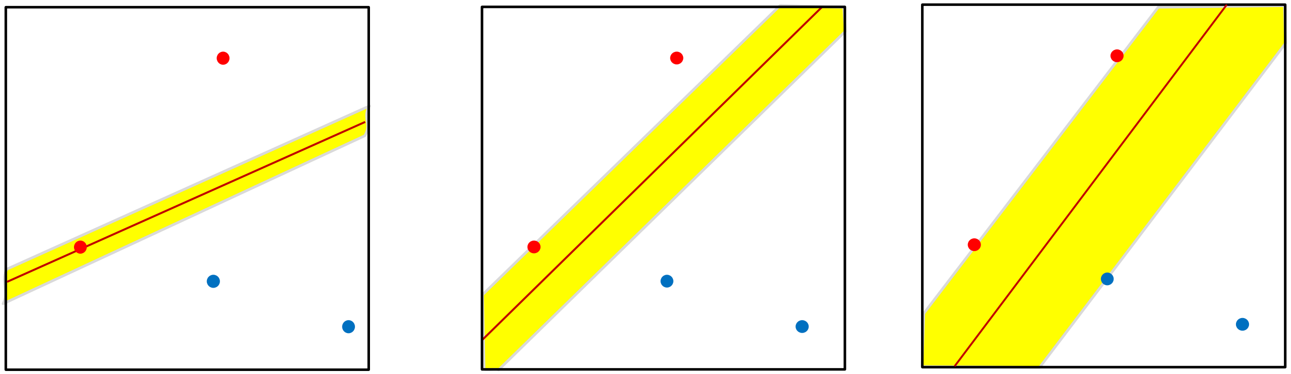
- Introduction
- Learning problem & linear classification
- Linear models: regression & logistic regression
- Non-linear transformation, overfitting & regularization
- **Support Vector Machines and kernel learning**
- Neural Networks: shallow [and deep]
- Theoretical foundation of supervised learning
- Unsupervised learning

Better linear separation

Linearly separable data

Different separating lines

What means best?



Two questions:

- Why is bigger margin better?
- Which **w** maximizes the margin?

Finding w with large margin

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

2 preliminary technicalities:

- Normalize w:

$$|\mathrm{w}^\mathsf{T} x_n| = 1$$

- Pull out w_0:
  $\mathbf{w} = (w_1, \ldots, w_d)$ as b
- The plane is now defined by $\boxed{\mathbf{w}^\mathsf{T}\mathbf{x} + b = 0}$
  (no x_0)

Computing the distance

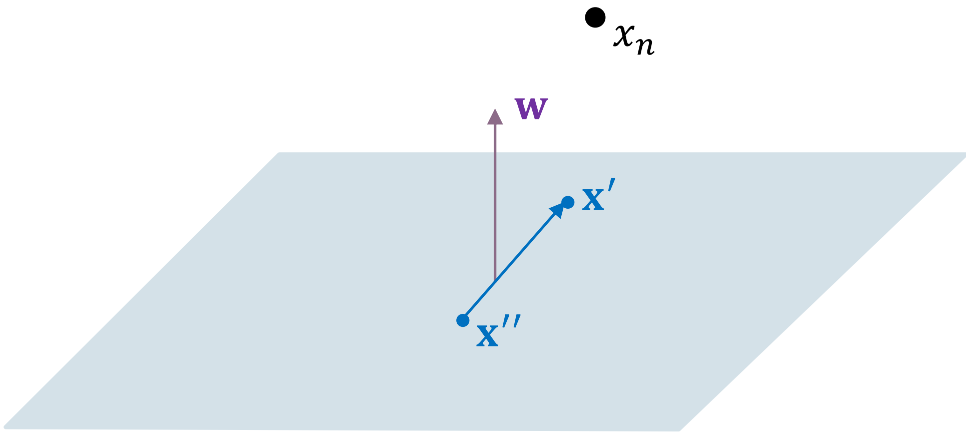
The distance between x_n and the plane $\mathrm{w}^{\mathsf{T}}\mathbf{x} + b = 0$ where $|\mathrm{w}^{\mathsf{T}}\mathrm{x}_n + b| = 1$

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

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

$\mathrm{w}^{\mathsf{T}}\mathbf{x}' + b = 0$ and $\mathrm{w}^{\mathsf{T}}\mathbf{x}'' + b = 0$

$\Longrightarrow \mathrm{w}^{\mathsf{T}}(\mathbf{x}' - \mathbf{x}'') = 0$

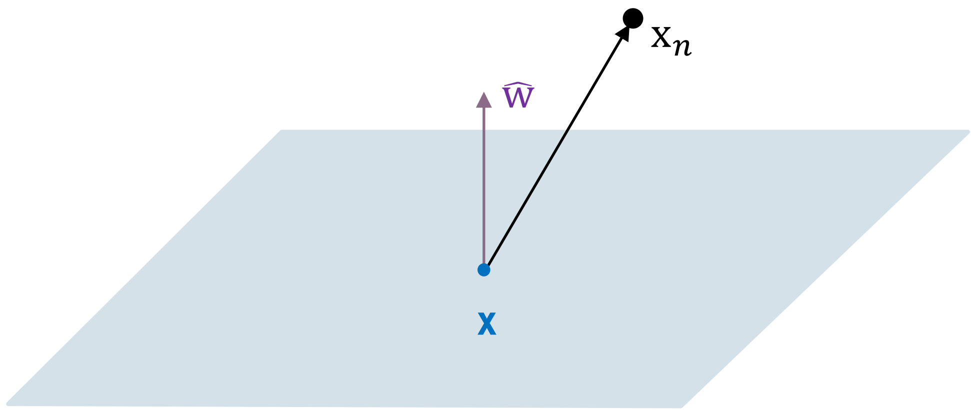


and the distance is …

Distance between x_n and the plane:

Take any point x on the plane

Projection $\mathrm{x}_n - \mathrm{x}$ on w



$$\hat{\mathrm{w}} = \frac{\mathrm{w}}{\|\mathrm{w}\|} \implies \text{distance} = |\hat{\mathrm{w}}^\intercal(\mathrm{x}_n - \mathrm{x})|$$

$$\text{distance} = \frac{1}{\|\mathrm{w}\|}|\mathrm{w}^\intercal \mathrm{x}_n - \mathrm{w}^\intercal \mathrm{x}| = \frac{1}{\|\mathrm{w}\|}|\mathrm{w}^\intercal \mathrm{x}_n + b - \mathrm{w}^\intercal \mathrm{x} - b| = \frac{1}{\|\mathrm{w}\|}$$

The optimization problem

Maximize $\frac{1}{\|\mathrm{w}\|}$

subject to $\min_n |\mathrm{w}^\intercal \mathrm{x}_n + b| = 1$

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

Minimize $\frac{1}{2}\mathrm{w}^\intercal \mathrm{w}$

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

How can we solve this optimization problem?

Lagrange formulation (using Karush-Kuhn-Tucker)

Minimize $\mathcal{L}(\mathbf{w}, b, \alpha) = \frac{1}{2}\mathbf{w}^\intercal\mathbf{w} - \sum_{\boldsymbol{n}=\mathbf{1}}^{\boldsymbol{N}} \boldsymbol{\alpha}_{\boldsymbol{n}}(\boldsymbol{y}_{\boldsymbol{n}}(\mathrm{w}^\intercal\mathrm{x}_n + b) - \mathbf{1})$

w.r.t. $\mathbf{w}$ and b and maximize w.r.t. each $\boldsymbol{\alpha}_{\boldsymbol{n}} \geq \mathbf{0}$

$$\nabla_{\mathrm{w}}\mathcal{L} = \mathbf{w} - \sum_{\boldsymbol{n}=\mathbf{1}}^{\boldsymbol{N}} \boldsymbol{\alpha}_{\boldsymbol{n}}\boldsymbol{y}_{\boldsymbol{n}}\mathbf{x}_{\boldsymbol{n}} = \mathbf{0}$$

$$\frac{\partial\mathcal{L}}{\partial b} = -\sum_{\boldsymbol{n}=\mathbf{1}}^{\boldsymbol{N}} \boldsymbol{\alpha}_{\boldsymbol{n}}\boldsymbol{y}_{\boldsymbol{n}} = \mathbf{0}$$

Substituting …

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

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

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

Maximize w.r.t. to α <u>subject to</u> $\alpha_n \geq 0$ for $n = 1, \ldots, N$ and $\sum_{n=1}^{N} \alpha_n y_n = \mathbf{0}$

The solution – quadratic programming

$$\min_{\alpha} \frac{1}{2}\alpha^{\intercal} \underbrace{\begin{bmatrix} y_1 y_1 \mathrm{x}_1^{\intercal}\mathrm{x}_1 & y_1 y_2 \mathrm{x}_1^{\intercal}\mathrm{x}_2 & \cdots & y_1 y_{\mathrm{N}} \mathrm{x}_1^{\intercal}\mathrm{x}_N \\ y_2 y_1 \mathrm{x}_2^{\intercal}\mathrm{x}_1 & y_2 y_2 \mathrm{x}_2^{\intercal}\mathrm{x}_2 & \cdots & y_2 y_N \mathrm{x}_2^{\intercal}\mathrm{x}_N \\ \vdots & \vdots & \ddots & \vdots \\ y_N y_1 \mathrm{x}_N^{\intercal}\mathrm{x}_1 & y_N y_2 \mathrm{x}_N^{\intercal}\mathrm{x}_2 & \cdots & y_N y_N \mathrm{x}_N^{\intercal}\mathrm{x}_N \end{bmatrix} \alpha}_{\text{quadratic coefficients}} + \underbrace{(-1^{\intercal})}_{\text{linear}}\alpha$$

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

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

Quadratic Programming finds the α's

Solution: $\alpha = \alpha_1, \dots, \alpha_N$

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

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

$$\alpha_n (y_n (\mathrm{w}^\intercal \mathrm{x}_n + b) - 1) = 0$$

That leads to the conclusion

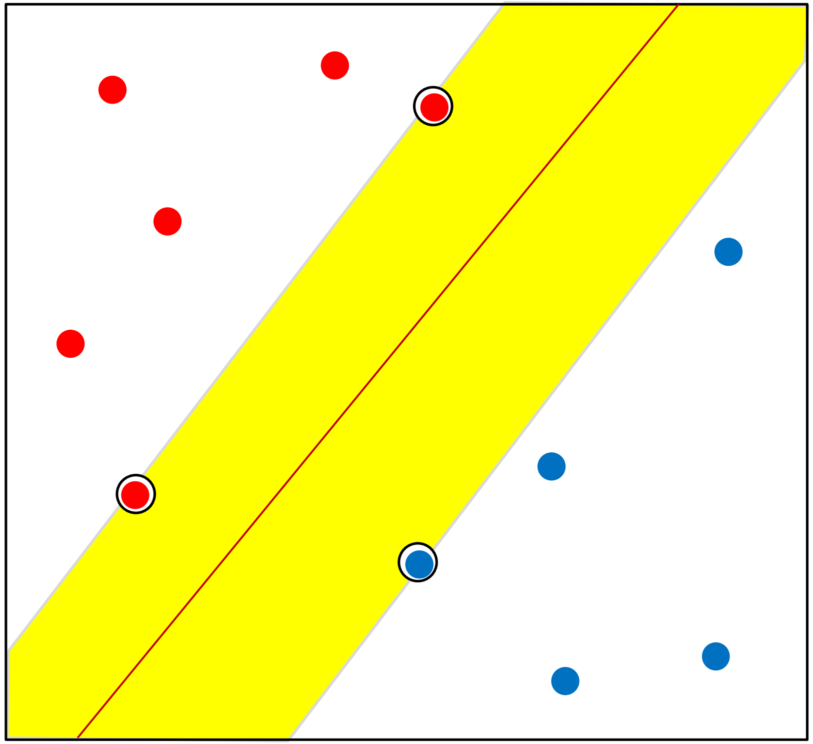
$\alpha_n > 0 \Rightarrow \mathrm{x}_n$ **is a support vector**

Support vectors

Closest x_n's to the plane: achieve the margin

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

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



Solve for b using any SV:

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

An interesting insight

— If we can express $\mathbf{w}$ in terms of a linear combinations of $\mathbf{x_n}$:

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

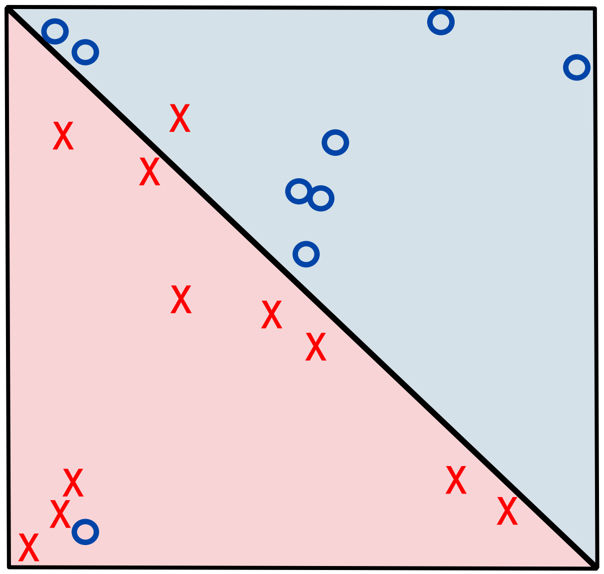
— … then we can express the decision function as

$$\mathrm{h}(\mathbf{x}) = \operatorname{sign}\left(\sum_{\mathrm{x}_n \text{ is SV}} \alpha_n y_n \mathbf{x_n}^T \mathbf{x} + b\right)$$

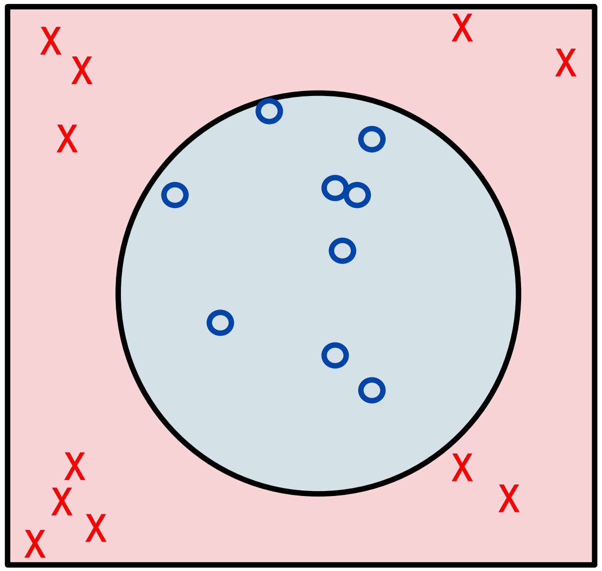
… so it also depends on the linear combination

What, if data is not linearly seperable?

slightly:



seriously:

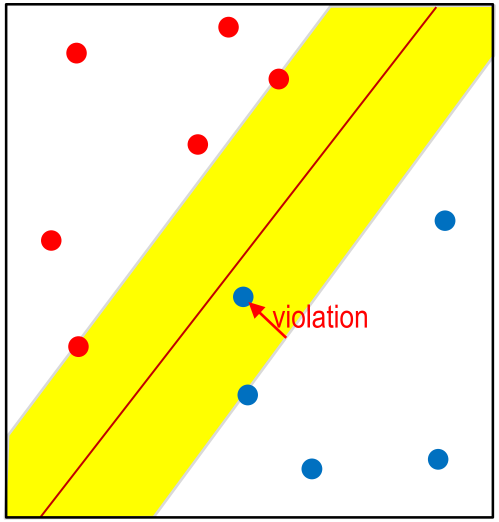


Accepting margin violations

Margin violation: $y_n(w^\top \mathrm{x}_n + b) \geq 1$ fails

Require: $y_n(w^\top \mathrm{x}_n + b) \geq 1 - \xi_n$ with $\xi_n \geq 0$

Total violation $= \sum_{n=1}^{N} \xi_n$



The new optimization

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

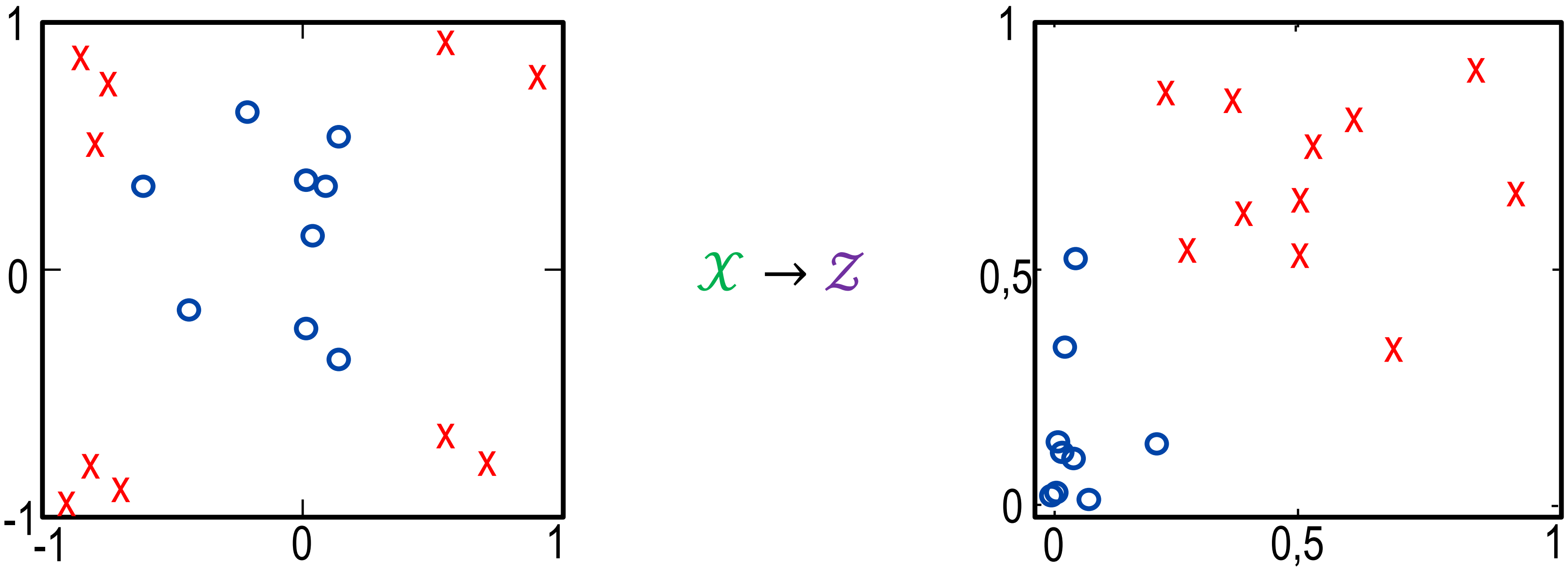
Subject to $y_n(\mathrm{w}^{\mathsf{T}}\mathrm{x}_n + b) \geq 1 - \xi_n$ for $n = 1, \dots, N$

and $\xi_n \geq 0$ for $n = 1, \dots, N$

$\mathrm{w} \in \mathbb{R}^d, \quad b \in \mathbb{R}, \quad \xi \in \mathbb{R}^N$

Using non-linear transformations

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



What do we need from the $\mathcal{Z}$ space

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

Constraints: $\boldsymbol{\alpha_n} \geq \mathbf{0}$ for $n = 1, \ldots, N$ and $\sum_{n=1}^{N} \boldsymbol{\alpha_n y_n} = 0$

$\boxed{g(\mathbf{x}) = sign(\mathrm{w}^\top \mathbf{z} + b)}$ need $\mathbf{z}_n{}^\top \mathbf{z}$

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

and b : $\boldsymbol{y_m}(\mathrm{w}^\top \mathbf{z}_m + b) = \mathbf{1}$ need $\mathbf{z}_n{}^\top \mathbf{z}_m$

Generalized inner product

Given two points $\mathbf{x}$ and $\mathbf{x}' \in \mathcal{X}$, we need $\mathbf{z}^\mathsf{T}\mathbf{z}'$

Let $\mathbf{z}^\mathsf{T}\mathbf{z}' = K(\mathbf{x}, \mathbf{x}')$ (the kernel) "inner product" of $\mathbf{x}$ and $\mathbf{x}'$

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

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

$$K(\mathbf{x}, \mathbf{x}') = \mathbf{z}^\mathsf{T}\mathbf{z}' = 1 + x_1 x_1' + x_2 x_2' + x_1^2 x_1'^2 + x_2^2 x_2'^2 + x_1 x_1' x_2 x_2'$$

The final hypothesis

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

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

where $b = y_m - \sum_{\boldsymbol{\alpha}_n > 0} \alpha_n y_n K(\boldsymbol{x_n}, \boldsymbol{x_m})$

for any support vector $(\boldsymbol{\alpha}_m > 0)$

Design your own kernel

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

1. It is symmetric and 2. The matrix $\begin{bmatrix} K(\mathrm{x}_1, \mathrm{x}_1) & K(\mathrm{x}_1, \mathrm{x}_2) & \cdots & K(\mathrm{x}_1, \mathrm{x}_N) \\ K(\mathrm{x}_2, \mathrm{x}_1) & K(\mathrm{x}_2, \mathrm{x}_2) & \dots & K(\mathrm{x}_2, \mathrm{x}_N) \\ \vdots & \vdots & \ddots & \vdots \\ K(\mathrm{x}_N, \mathrm{x}_1) & K(\mathrm{x}_N, \mathrm{x}_2) & \cdots & K(\mathrm{x}_N, \mathrm{x}_N) \end{bmatrix}$

is positive semi-definite

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

Backup

Lagrange formulation

$$\mathcal{L}(\mathbf{w}, b, \xi, \alpha, \beta) =$$

$$\frac{1}{2}\mathbf{w}^{\intercal}\mathbf{w} + C\sum_{n=1}^{N}\xi_n - \sum_{\boldsymbol{n}=\mathbf{1}}^{\boldsymbol{N}}\boldsymbol{\alpha}_{\boldsymbol{n}}(\boldsymbol{y}_{\boldsymbol{n}}(\mathrm{w}^{\intercal}\mathrm{x}_n + b) - \mathbf{1} + \xi_n) - \sum_{\boldsymbol{n}=\mathbf{1}}^{\boldsymbol{N}}\beta_n\xi_n$$

Minimize w.r.t. $\mathbf{w}$, b and ξ and maximize w.r.t. each $\boldsymbol{\alpha}_{\boldsymbol{n}} \geq \mathbf{0}$ and $\beta_n \geq 0$

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

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

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

and the solution is …

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

subject to $0 \le \alpha_n \le \boldsymbol{C}$ for $n = 1, \ldots, N$ and $\sum_{n=1}^{N} \alpha_n y_n = 0$

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

minimizes $\frac{1}{2}\mathrm{w}^{\intercal}\mathrm{w} + C\sum_{n=1}^{N} \xi_n$

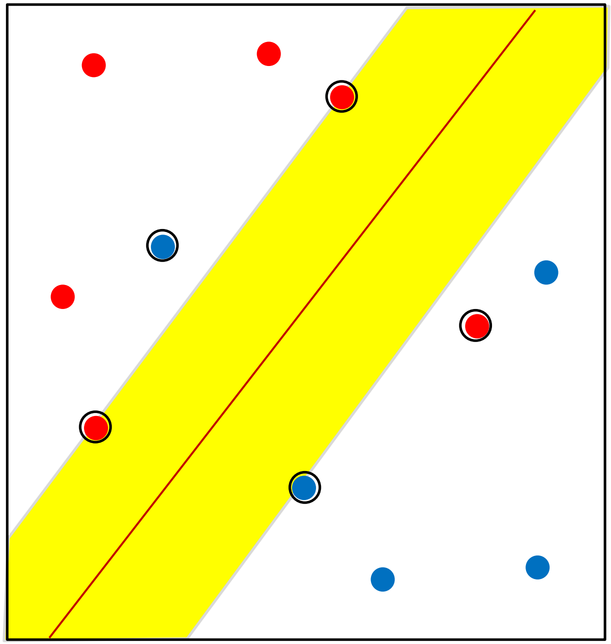
Types of support vectors

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

$$y_n(\mathrm{w}^\intercal \mathrm{x}_n + b) = 1 \qquad (\xi_n = 0)$$

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

$$y_n(\mathrm{w}^\intercal \mathrm{x}_n + b) < 1 \qquad (\xi_n > 0)$$



The Kernel trick

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

Example: Consider $K(\mathbf{x}, \mathbf{x}') = (1 + \mathbf{x}^{\mathsf{T}}\mathbf{x}')^2 = (1 + \boldsymbol{x_1 x_1'} + \boldsymbol{x_2 x_2'})^2$

$$= 1 + \boldsymbol{x_1^2 x_1'^2} + \boldsymbol{x_2^2 x_2'^2} + \mathbf{2}\boldsymbol{x_1 x_1'} + \mathbf{2}\boldsymbol{x_2 x_2'} + \mathbf{2}\boldsymbol{x_1 x_1' x_2 x_2'}$$

This is an inner product!

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

$$(\mathbf{1}, \boldsymbol{x_1'^2}, \boldsymbol{x_2'^2}, \sqrt{\mathbf{2}}\boldsymbol{x'_1}, \sqrt{\mathbf{2}}\boldsymbol{x'_2}, \sqrt{\mathbf{2}}\boldsymbol{x'_1 x'_2})$$

We only need $\mathcal{Z}$ to exist!

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

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

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

$$K(x, x') = \exp(-(\boldsymbol{x} - \boldsymbol{x}')^{\mathbf{2}})$$

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

Kernel formulation of SVM

Remember quadratic programming? The only difference is now:

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

Everything else is the same.