ENCS5341
Machine Learning and Data Science

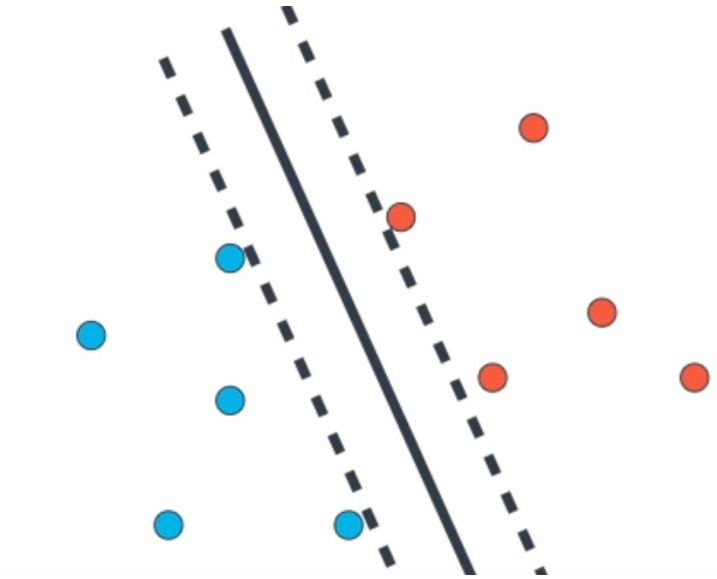
Kernels and SVM

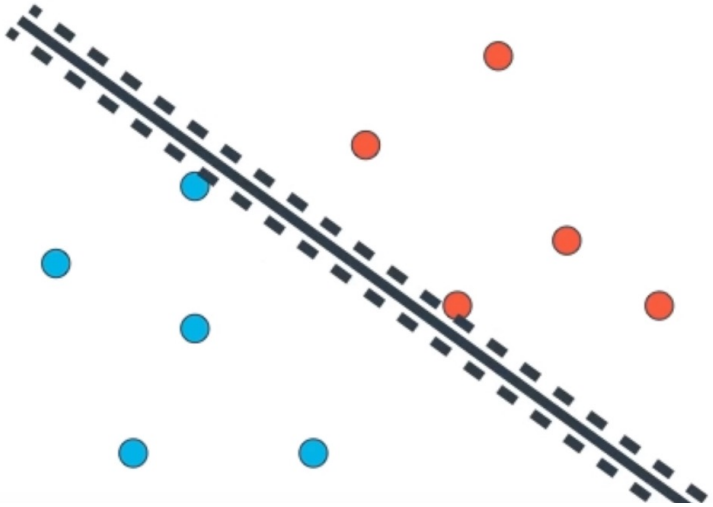
Based on slides prepared by Tamás Horváth

Support Vector Machine

Linear separation

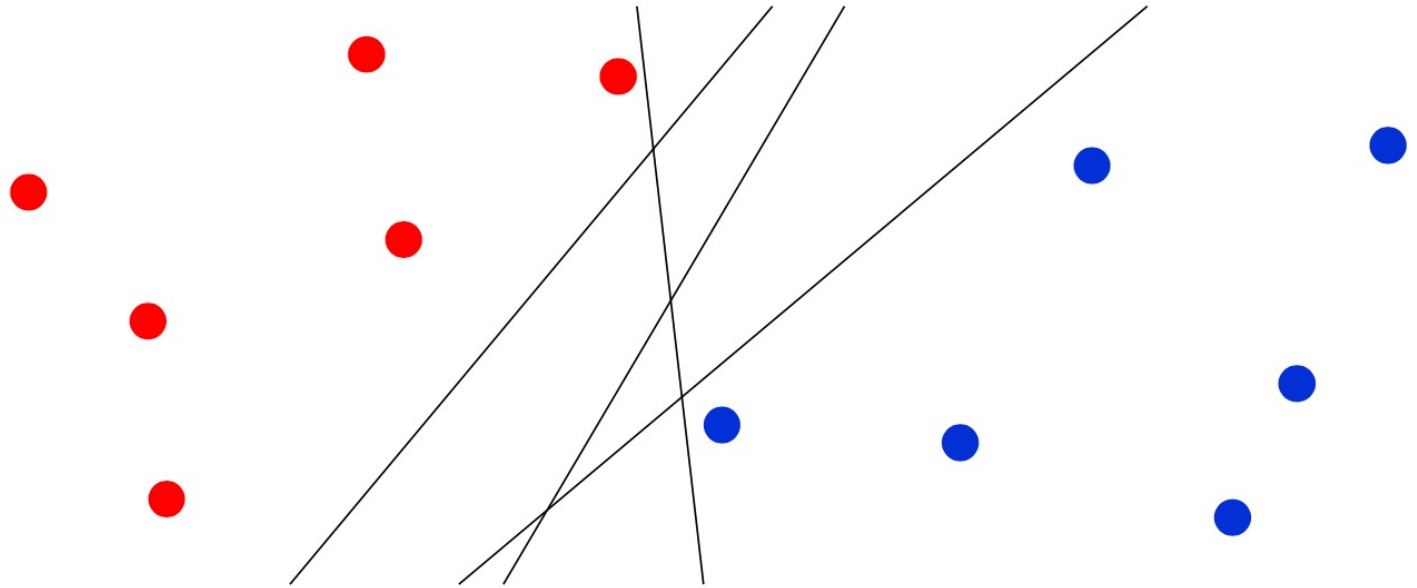
- Consider the following linearly separable binary classification problem. Which line is a better separator?





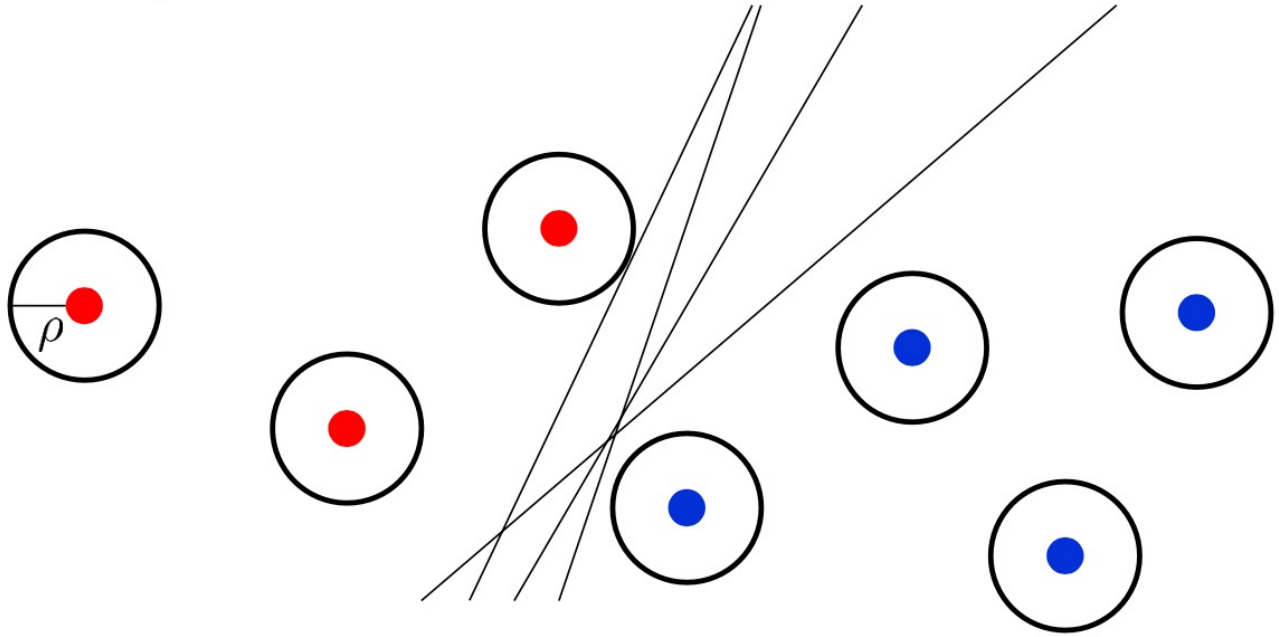
Linear separation

- For a linearly separable data there are infinitely many separating hyperplanes.
- Which one to chose



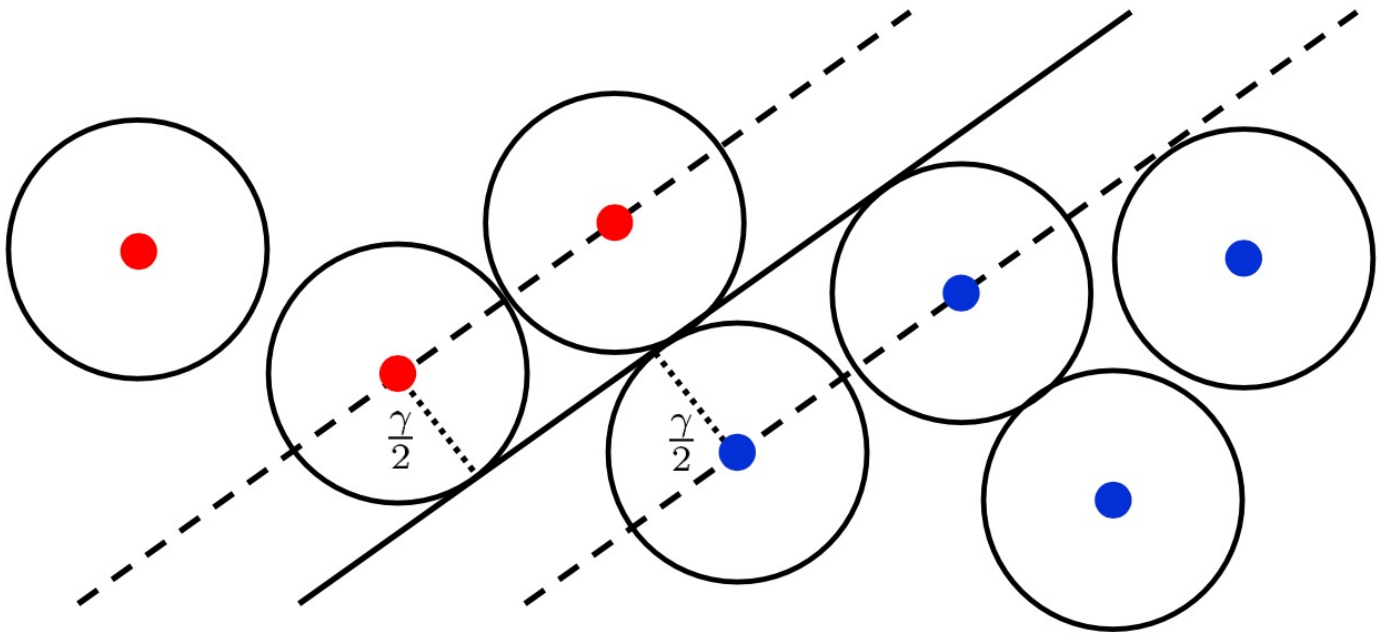
Noise tolerating linear separation

noisy examples: suppose some amount of noise (ρ) has been added to each example



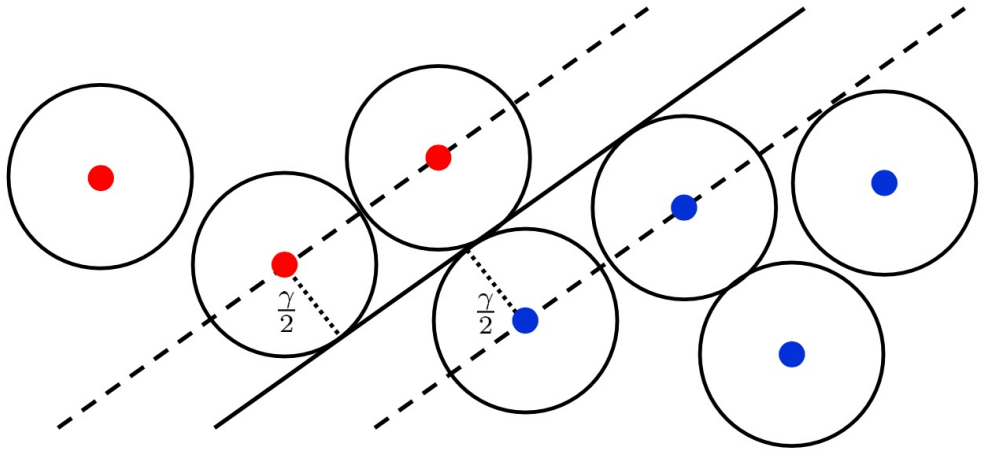
Choose the hyperplane with the largest margin

hyperplane with **maximum** margin (γ):



Arguments for maximum margin hyperplane

- Robust against noise.
- Excellent predictive performance in practice.
- Separating hyperplane becomes unique.



1. Hard Margin Support Vector Machines (Boser, Guyon, and Vapnik, 1992)

Point Hyperplane Distance

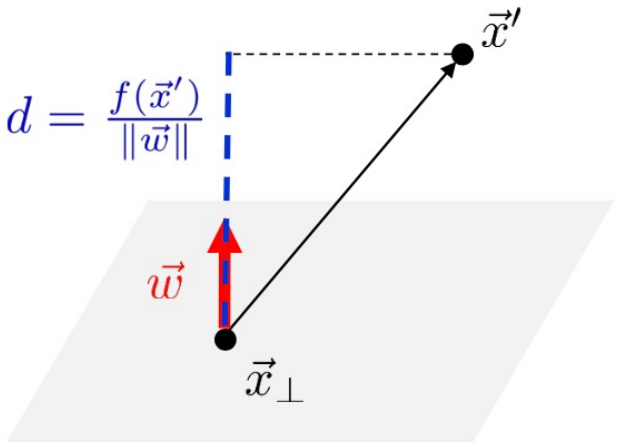
hyperplane for $f(\vec{x}) = \langle \vec{w}, \vec{x} \rangle + b = 0$

Fact 1. $\vec{w}$ is **orthogonal** to the hyperplane, as for any $\vec{x}_1, \vec{x}_2$ on the hyperplane:

$$0 = f(\vec{x}_2) - f(\vec{x}_1) = \langle \vec{w}, \vec{x}_2 \rangle + b - (\langle \vec{w}, \vec{x}_1 \rangle + b) = \langle \vec{w}, \vec{x}_2 - \vec{x}_1 \rangle$$

Fact 2. signed distance of a point $\vec{x}'$ from the hyperplane:

$$\begin{aligned} d &= \left\langle \frac{\vec{w}}{\|\vec{w}\|}, \vec{x}' - \vec{x}_\perp \right\rangle \\ &= \frac{1}{\|\vec{w}\|}(\langle \vec{w}, \vec{x}' \rangle - \langle \vec{w}, \vec{x}_\perp \rangle) \\ &= \frac{1}{\|\vec{w}\|}(\langle \vec{w}, \vec{x}' \rangle + b \underbrace{- \langle \vec{w}, \vec{x}_\perp \rangle - b}_{=0}) \\ &= \frac{f(\vec{x}')}{\|\vec{w}\|} \end{aligned}$$

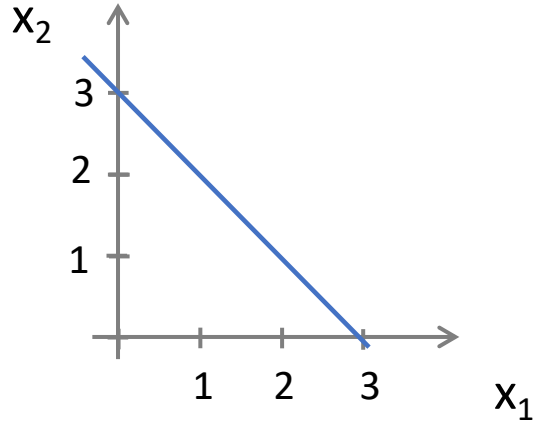


Example: distance from a hyperplane

- $f(x) = x_1 + x_2 - 3$

- Signed distance of the point (0,0) from f is

$$d = \frac{f((0,0))}{\|(1,1)\|} = \frac{-3}{\sqrt{2}}$$

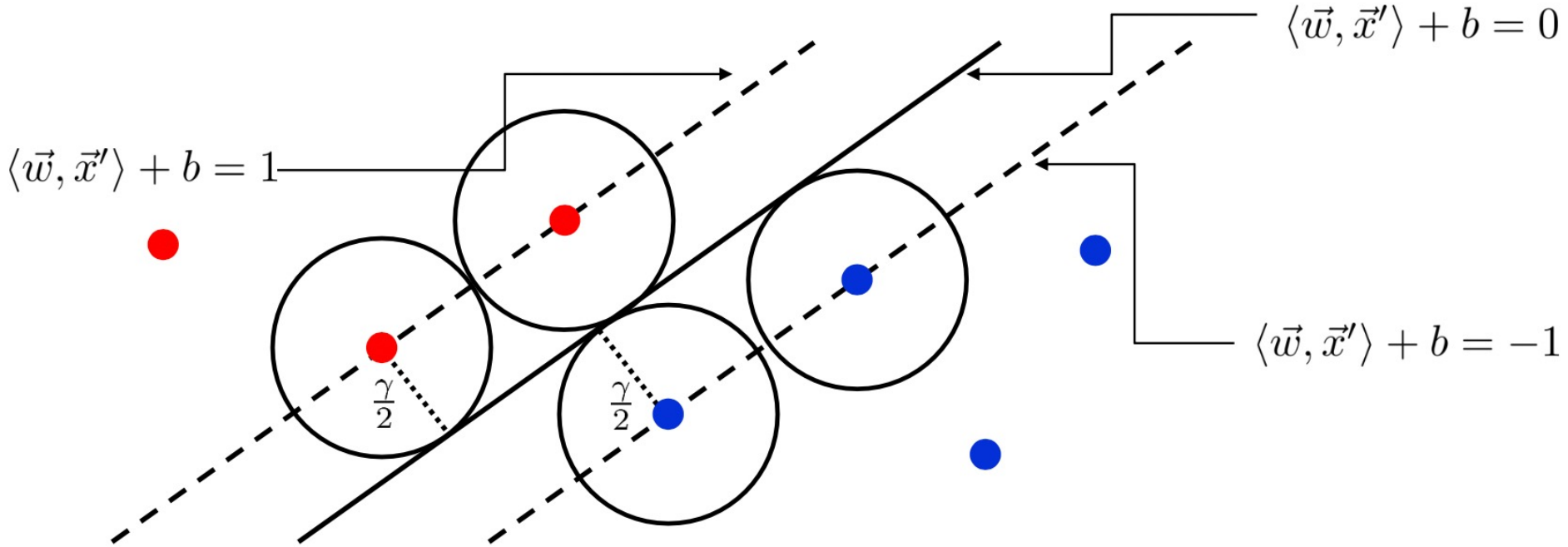


- Signed distance of the point (3,3) from f is

$$d = \frac{f((3,3))}{\|(1,1)\|} = \frac{3}{\sqrt{2}}$$

The Maximum Margin Hyperplane

hyperplane $\langle \vec{w}, \vec{x} \rangle + b = 0$**:** scale $\vec{w}$ and b such that $|\langle \vec{w}, \vec{x}' \rangle + b| = 1$ of all points on the dashed hyperplanes



$\Rightarrow$ **margin:** $\boxed{\gamma = \frac{2}{\|\vec{w}\|}}$

prediction of the class of an unseen instance $\vec{x}$: $\text{sign}(\langle \vec{w}, \vec{x} \rangle + b)$

Support Vector Machines: Hard Margin Constraint

optimization problem for $S = \{(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n)\} \subset \mathbb{R}^d \times \{+1, -1\}$ such that S is linearly separable:

$$\begin{aligned} \max_{\vec{w}, b} \quad & \frac{2}{\|\vec{w}\|} \\ \text{subject to} \quad & |\langle \vec{w}, \vec{x}_i \rangle + b| \geq 1 \quad \text{for } i = 1, \ldots, n \end{aligned}$$

remark: hard margin constraints: **all** data points are classified correctly

problem: objective function is non-convex

Support Vector Machines: Hard Margin Constraint

non-convex optimization problem:

$$\begin{aligned} \max_{\vec{w},b} \quad & \frac{2}{\|\vec{w}\|} \\ \text{subject to} \quad & |\langle \vec{w}, \vec{x}_i \rangle + b| \geq 1 \quad \text{for } i = 1, \ldots, n \end{aligned}$$

equivalent formulation (maximizing $\frac{2}{\|\vec{w}\|}$ is the same as minimizing $\frac{1}{2}\|\vec{w}\|^2$):

$$\begin{aligned} \min_{\vec{w},b} \quad & \frac{1}{2}\|\vec{w}\|^2 \\ \text{s.t.} \quad & -(y_i(\langle \vec{w}, \vec{x}_i \rangle + b) - 1) \leq 0 \quad \text{for } i = 1, \ldots, n \end{aligned}$$

$\Rightarrow$ *quadratic programming problem* (i.e., quadratic objective function with linear inequality constraints)

- could be solved by off-the-shelf programs, still we take its *dual* in order to arrive at a kernel method

Remark

optimization problem:

$$\begin{aligned} \min_{\vec{w},b} \quad & \frac{1}{2}\vec{w}^\top \vec{w} \\ \text{s.t.} \quad & y_i(\langle \vec{w}, \vec{x}_i \rangle + b) - 1 \geq 0, \\ & i = 1, \ldots, n \end{aligned}$$

corresponds to the **regularized empirical risk minimization**:

$$\min_{\vec{w},b} \frac{1}{n} \sum_{i=1}^{n} V(f(\vec{x}_i), y_i) + \lambda \|\vec{w}\|^2$$

- **loss function:** $V(f(\vec{x}), y) = \begin{cases} 0 & \text{if } y \cdot f(\vec{x}) \geq +1 \\ \infty & \text{o/w} \end{cases}$
- **regularization parameter:** any $0 < \lambda < \infty$ results in the same solution

Dual Optimization Problem

solve:

$$\max_{\vec{\alpha}} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \left\langle \vec{x}_i, \vec{x}_j \right\rangle$$

$$\text{s.t.} \quad \sum_{i=1}^{n} \alpha_i y_i = 0 \text{ and}$$

$$\alpha_i \geq 0 \quad i = 1, \ldots, n$$

maximum margin hyperplane: $\left(\vec{w} = \sum_{i=1}^{n} y_i \alpha_i \vec{x}_i \right)$:

$$f(\vec{x}) = \sum_{i=1}^{n} y_i \alpha_i \left\langle \vec{x}_i, \vec{x} \right\rangle + b$$

$$b = -\frac{1}{2} \left(\max_{y_j = -1} \left(\sum_{i=1}^{n} y_i \alpha_i \left\langle \vec{x}_i, \vec{x}_j \right\rangle \right) + \min_{y_i = 1} \left(\sum_{i=1}^{n} y_i \alpha_i \left\langle \vec{x}_i, \vec{x}_j \right\rangle \right) \right)$$

Dual Form: Remark

maximum margin hyperplane: $f(\vec{x}) = \sum_{i=1}^{n} y_i \alpha_i \langle \vec{x}_i, \vec{x} \rangle + b$

optimization theory: the *dual complementary conditions* guarantee that

$$\alpha_i (y_i (\langle \vec{w}, \vec{x}_i \rangle + b) - 1) = 0$$

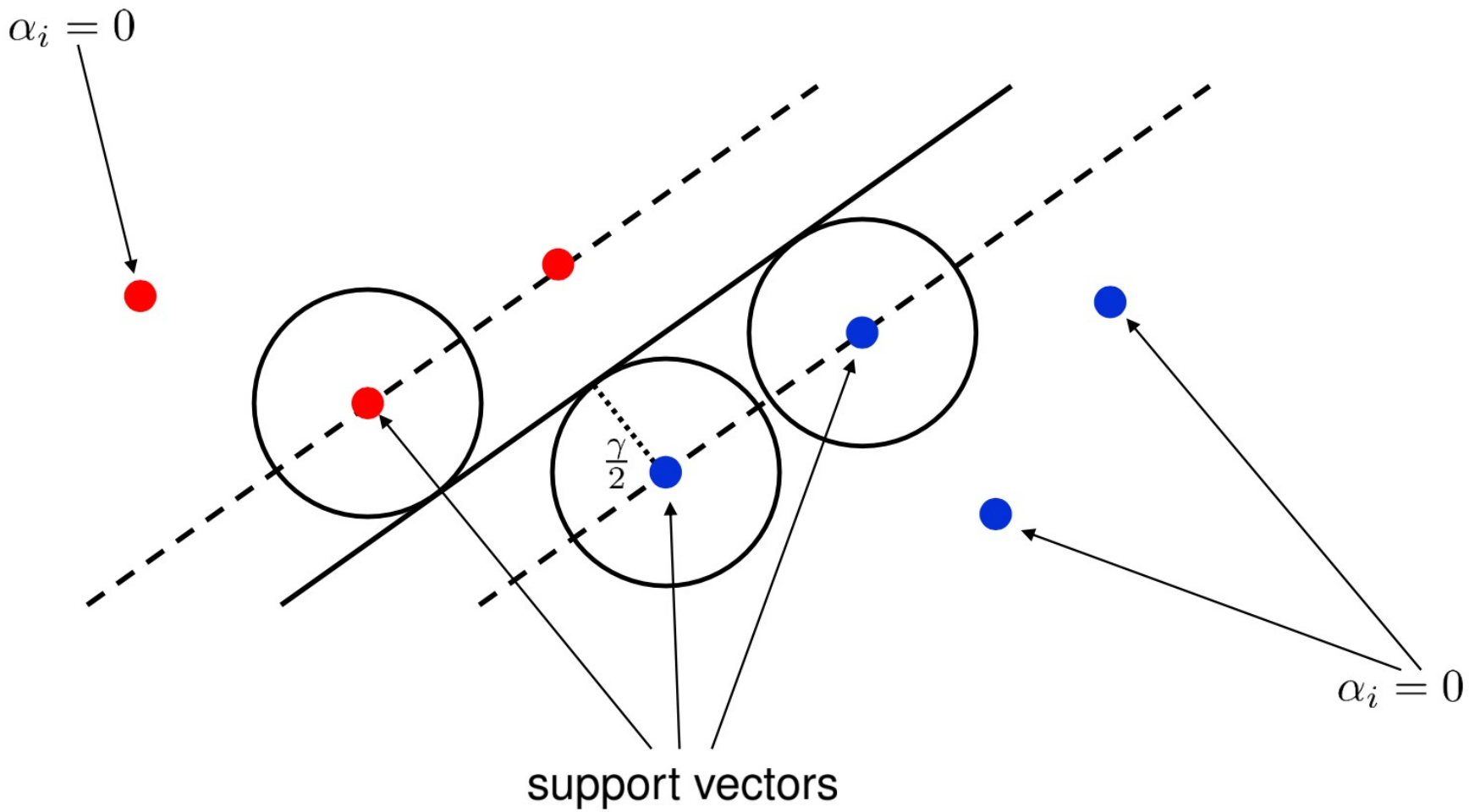
$\Rightarrow$ $\alpha_i = 0$ or $y_i (\langle \vec{w}, \vec{x}_i \rangle + b) = 1$

$\Rightarrow$ only the points on the margin hyperplanes are **active** in the prediction (i.e., $\alpha > 0$); all other points are **inactive** ($\alpha_i = 0$)

- points on the margin hyperplanes: **support vectors**

$\Rightarrow$ **sparse kernel method** because after training, a significant proportion of the data can be discarded; only the support vectors must be kept

Dual Form: Remark



2. Soft Margin Support Vector Machines (Cortes and Vapnik, 1995)

Soft Margin SVM

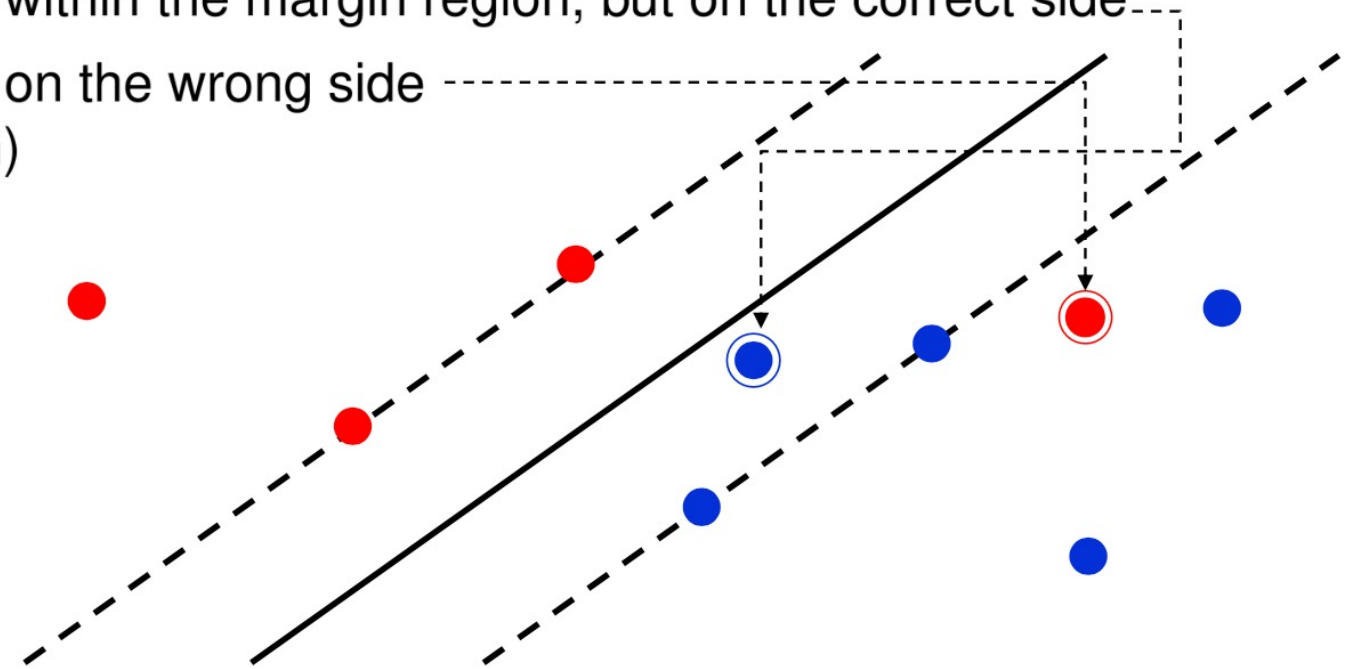
What to do if the data is **not** linearly separable?

idea: allow violations, but penalize them

violation types:

(i) training examples within the margin region, but on the correct side

(ii) training examples on the wrong side (missclassification)



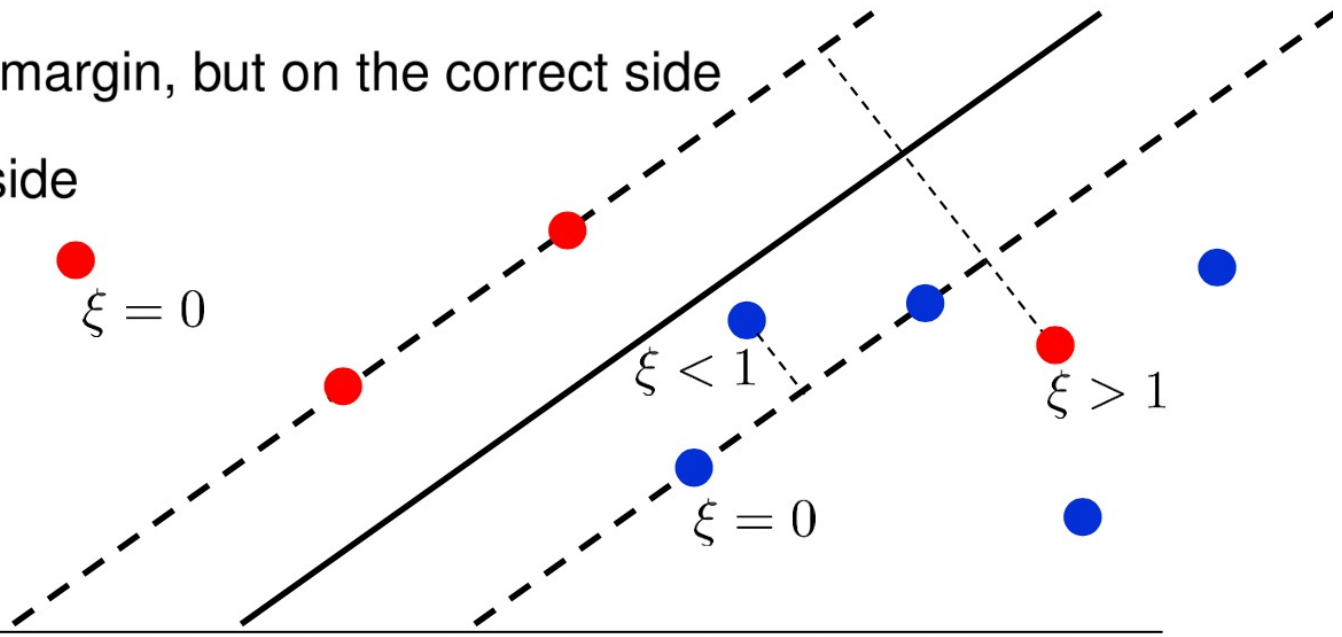
Soft Margin SVM

hard margin constraints are relaxed to **soft margin** constraints:

$$y_i(\langle \vec{w}, \vec{x}_i \rangle + b) \geq 1 - \xi_i \quad \text{for} \quad \xi_i \geq 0 \quad (i = 1, \ldots, n)$$

ξ_i: **slack** variables:

- $\xi = 0$: correct classification
- $0 < \xi \leq 1$: lies inside the margin, but on the correct side
- $\xi > 1$: lies on the wrong side

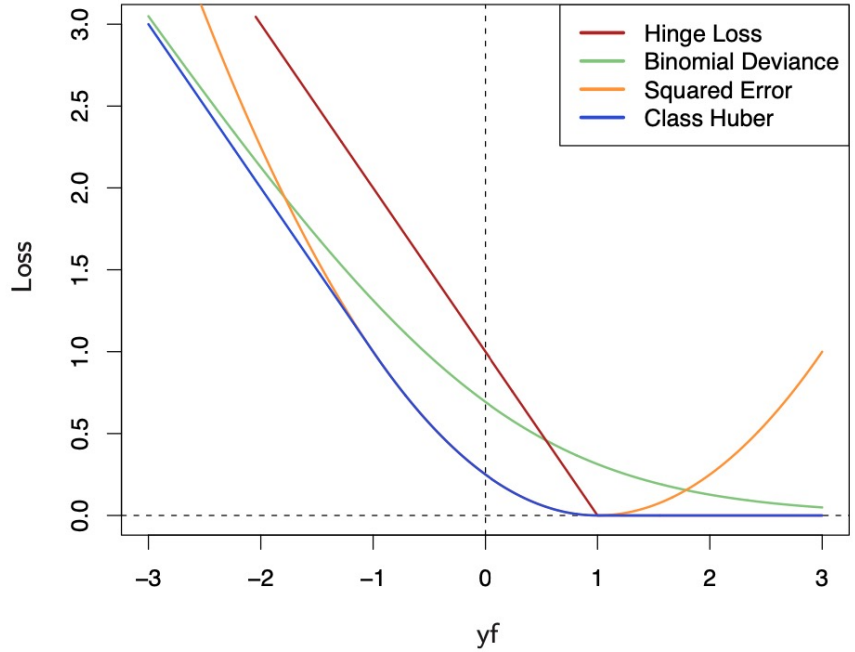


Soft Margin SVM: Optimization Problem

optimization problem with $C > 0$:

$$\min_{\vec{w},b,\vec{\xi}} \quad C\sum_{i=1}^{n}\xi_i + \frac{1}{2}\|\vec{w}\|^2$$

$$\text{s.t.} \quad y_i(\langle \vec{w}, \vec{x}_i\rangle + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \quad i = 1, \ldots, n$$



remarks:

- regularized empirical risk minimization with the hinge loss function

$$V(f(x), y) = \max(0, 1 - y_i f(x))$$

  - f in our case: $f(\vec{x}) = \langle \vec{w}, \vec{x}\rangle + b$
  - $C > 0$ plays the role of the regularization parameter λ $(C = 1/\lambda)$

- $\sum_{i=1}^{n}\xi_i$ is an upper bound on the number of misclassified points

Soft Margin SVM: Dual Form

dual can be obtained in a way similar to the case of hard margins:

$$\max_{\vec{\alpha}} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle \vec{x}_i, \vec{x}_j \rangle$$

$$\text{s.t.} \quad 0 \leq \alpha_i \leq C \quad \text{and} \quad \sum_{i=1}^{n} \alpha_i y_i = 0, \quad i = 1, \ldots, n$$

remark 1: quadratic programming problem

remark 2: almost the same as for hard margin SVM

difference: instead of $0 \leq \alpha_i$ (hard margin) we have the *box constraints*:

$$0 \leq \alpha_i \leq C$$

Soft Margin SVM: Interpretation of the Solution

$\alpha_i = 0$: $\vec{x}_i$ is inactive, i.e., does not contribute to the decision function

- typically this is the case for a large proportion of the training data (sparsity)

$\alpha_i > 0$: for soft margin SVM three types of support vectors:

- data points on the margin
- data points within the margin
- data points on the wrong side of the boundary

The Kernel Trick
(Aronszajn, 1964)

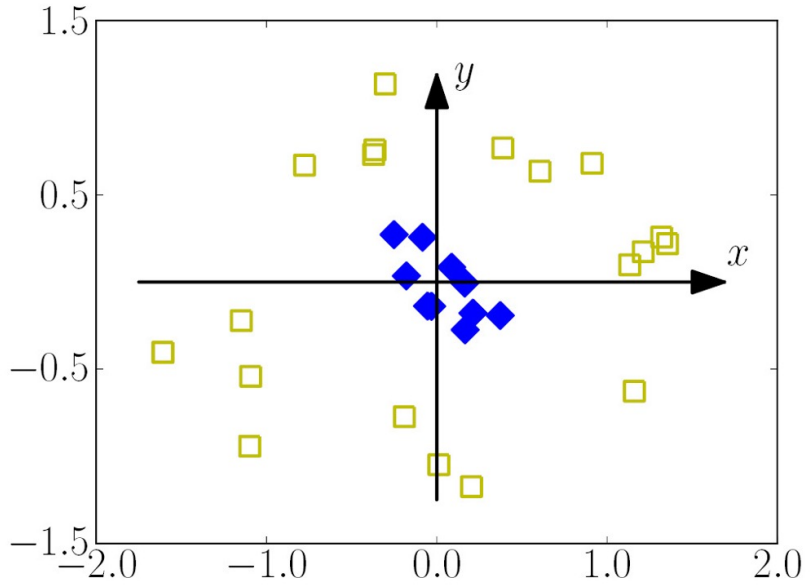
Learning in Feature Space

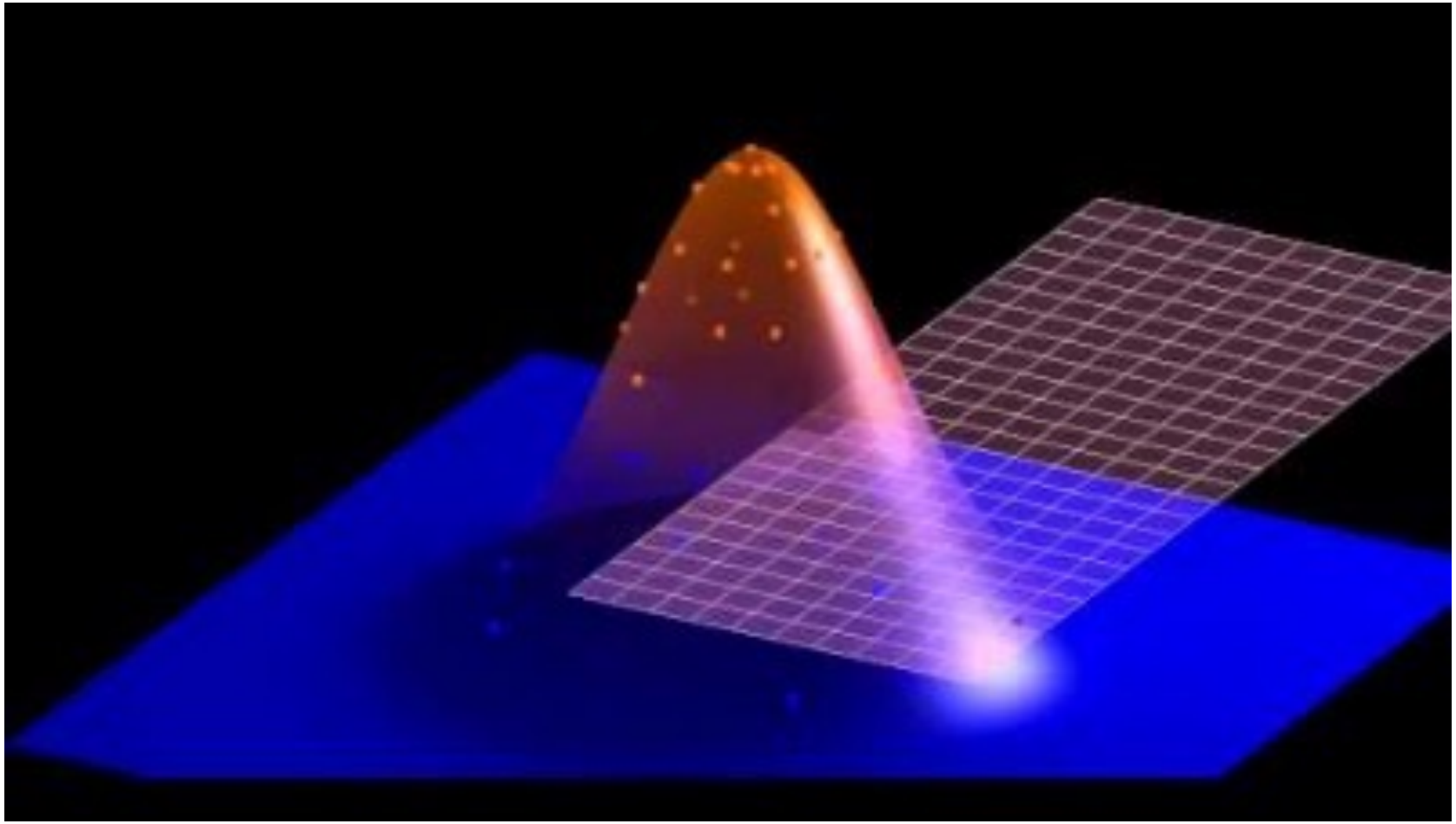
learning in input space: difficult if the input-output relationship is nonlinear

common strategy in ML: using some appropriate function

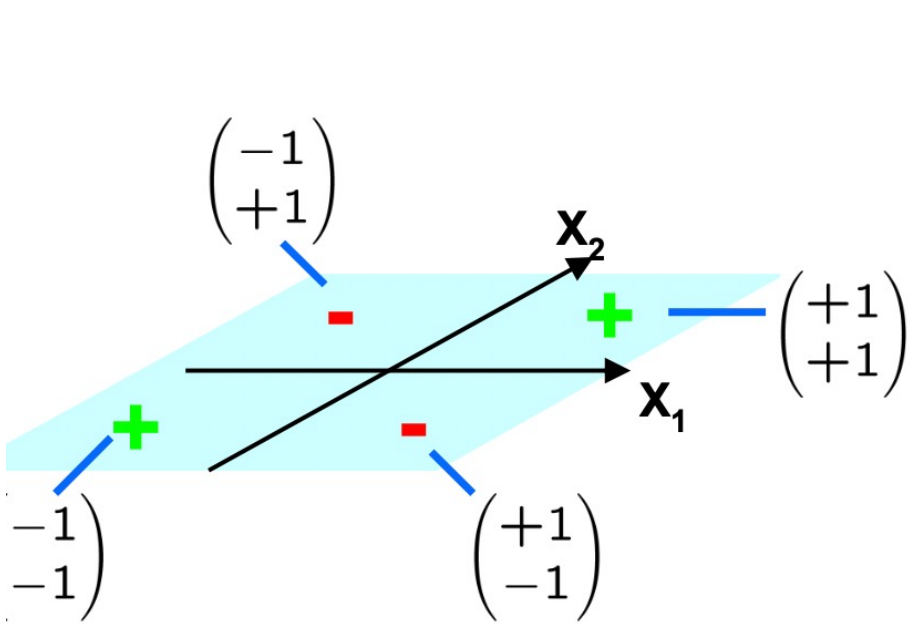
$$\Phi : \mathbb{R}^d \to \mathbb{R}^D$$

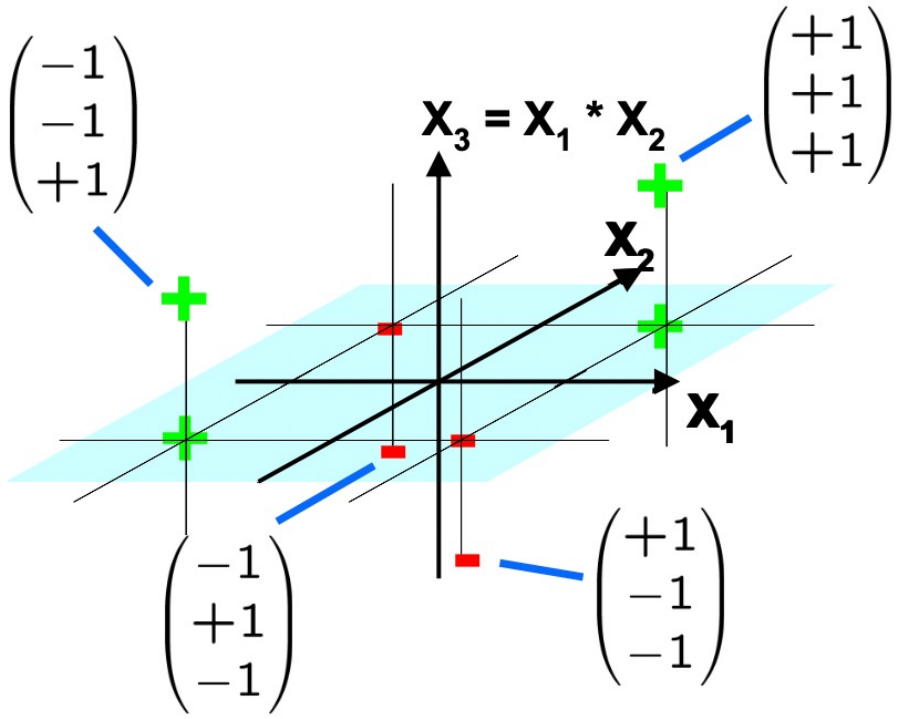
transform your data (in $\mathbb{R}^d$) into another space ($\mathbb{R}^D$), called the **feature space**, in which the relationship becomes linear





Example: XOR



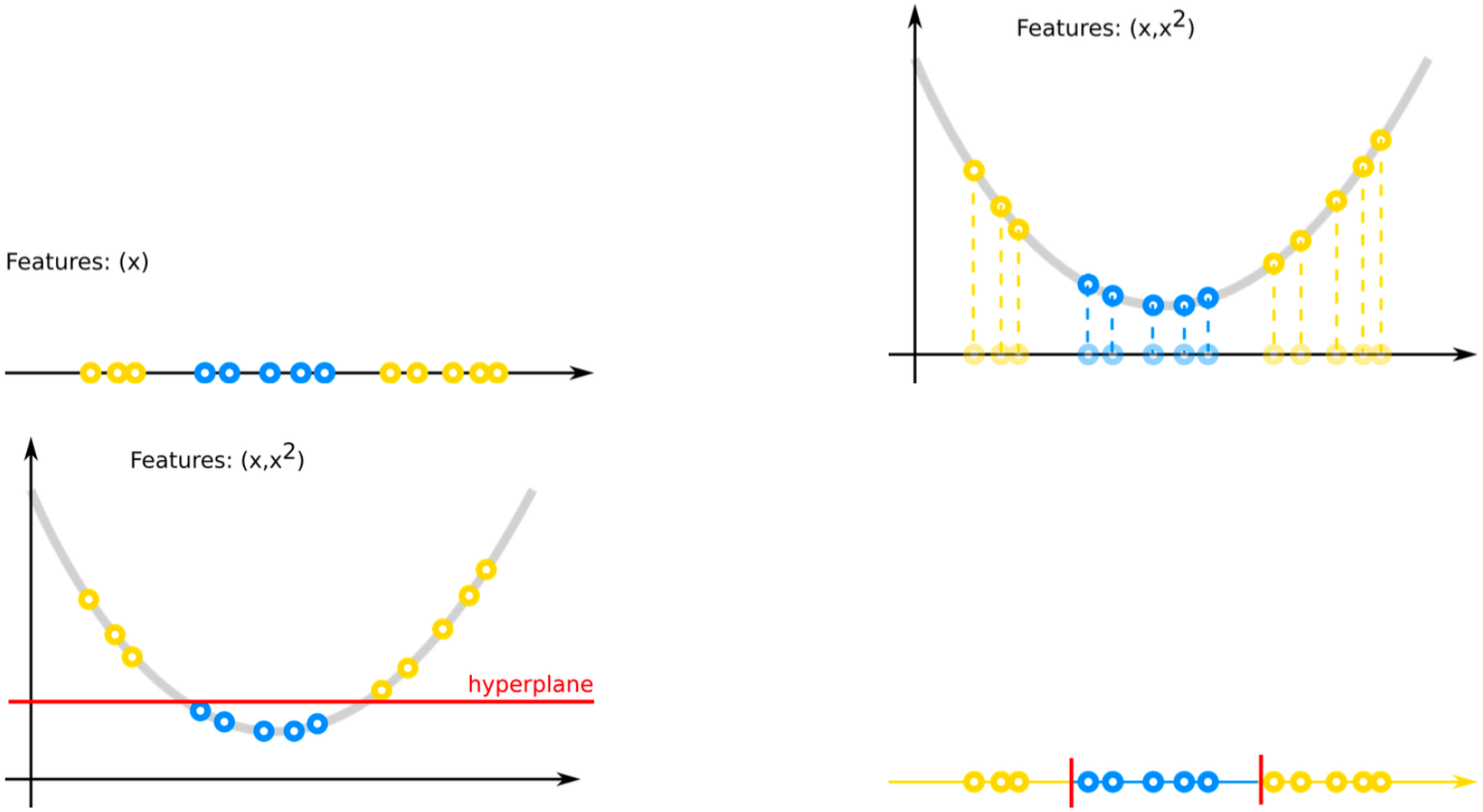


$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

$$\phi(x) = \begin{pmatrix} x_1 \\ x_2 \\ x_1 x_2 \end{pmatrix}$$

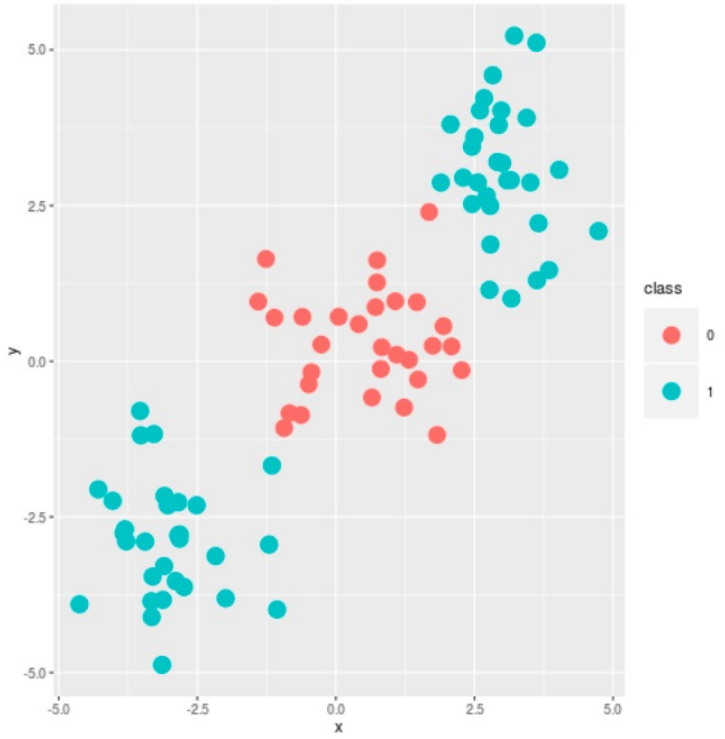
Example: 1D to 2D

Data can become linearly separable in higher-dimensional space.

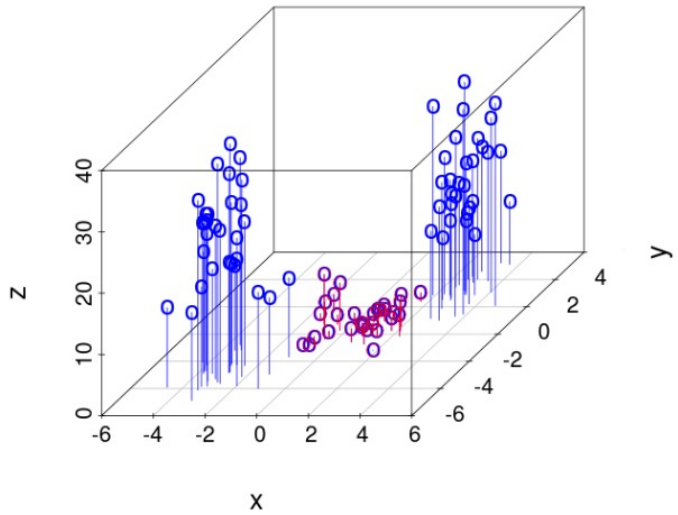


Example: 2D to 3D

Left: Features (x,y)



Right: Features (x,y, x^2+y^2)



Challenges of learning non-linear relationships

- How to choose the transformation such that the relation become linear?

- The transformation increases the features dimension, which increases the computation cost

The Kernel Trick solves both problems

The Kernel Trick

Def.: a **kernel** is a function $X \times X \to \mathbb{R}$ such that for all $x, y \in X$,

$$k(x, y) = \langle \Phi(x), \Phi(y) \rangle$$

for some function Φ mapping X to an *inner product* feature space $\mathcal{H}$

kernel trick: substitute all occurrences of $\langle \cdot, \cdot \rangle$ by a kernel k with

$$k(x, y) = \langle \Phi(x), \Phi(y) \rangle$$

where Φ is the underlying function mapping the input space into the feature space

crucial point: Φ does not have to be calculated; it can be even unknown!

The Kernel Trick

example: let $k : X \times X \to \mathbb{R}$ with $X \subseteq \mathbb{R}^d$ be defined by

$$k(\vec{x}, \vec{y}) = \langle \vec{x}, \vec{y} \rangle^2 \text{ for all } \vec{x}, \vec{y} \in X$$

claim: k is a kernel corresponding to the feature map $\Phi : \mathbb{R}^d \to \mathbb{R}^{d^2}$ defined by

$$\Phi : \vec{z} \mapsto (z_i z_j)_{i,j=1}^d \text{ for all } \vec{z} \in \mathbb{R}^d$$

proof:

$$\begin{aligned} \langle \Phi(\vec{x}), \Phi(\vec{y}) \rangle &= \left\langle (x_i x_j)_{i,j=1}^d, (y_i y_j)_{i,j=1}^d \right\rangle \\ &= \sum_{i,j=1}^d x_i x_j y_i y_j \\ &= \sum_{i=1}^d x_i y_i \sum_{j=1}^d x_j y_j \\ &= \langle \vec{x}, \vec{y} \rangle^2 \end{aligned}$$

Properties of Kernels

let $k : X \times X \to \mathbb{R}$ be a kernel function with $k(x, y) = \langle \Phi(x), \Phi(y) \rangle$ for some Φ and let

$$\alpha(x, y) = \frac{k(x, y)}{\sqrt{k(x, x)}\sqrt{k(y, y)}}$$

then

$$\begin{aligned} \alpha(x, y) &= \frac{k(x, y)}{\sqrt{k(x, x)}\sqrt{k(y, y)}} \\ &= \frac{\langle \Phi(x), \Phi(y) \rangle}{\sqrt{\langle \Phi(x), \Phi(x) \rangle}\sqrt{\langle \Phi(y), \Phi(y) \rangle}} \\ &= \frac{\langle \Phi(x), \Phi(y) \rangle}{\|\Phi(x)\|\|\Phi(y)\|} \\ &= \cos(\Phi(x), \Phi(y)) \end{aligned}$$

$\Rightarrow$ normalized kernels: cosine similarity in the feature space

Construction of Kernel Functions

Kernel Construction

we present some basis kernel functions, as well as show rules for constructing more complex kernels from simple ones

proof techniques used to show these results:

- construct the underlying feature map Φ corresponding to the kernel
- or use Mercer's characterization theorem (will not be discussed in this course)

Kernel Construction

Prop. $k(x, y) = f(x)f(y)$ is a kernel over $X \times X$ for all functions $f : X \to \mathbb{R}$

proof: let $\Phi : X \to \mathbb{R}$ be defined by

$$\Phi : x \mapsto f(x) \text{ for all } x \in X$$

$$\Rightarrow k(x, y) = f(x)f(y) = \langle \Phi(x), \Phi(y) \rangle$$

q.e.d.

Kernel Construction

Prop. Let k_1, k_2 be kernels over $X \times X$. Then for all $\alpha, \beta \geq 0$,

$$k(x, y) = \alpha k_1(x, y) + \beta k_2(x, y)$$

is a kernel.

Prop. Let k_1, k_2 be kernels over $X \times X$. Then

$$k(x, y) = k_1(x, y) k_2(x, y)$$

is a kernel.

Kernel Construction

Prop. Let k_1 be a kernel over $X \times X$ and p be a polynomial with positive coefficients. Then

$$k(x, y) = p(k_1(x, y))$$

is a kernel.

Prop. Let $k_1 : X \times X \to \mathbb{R}$ be a kernel. Then

$$k(x, y) = e^{k_1(x,y)}$$

is a kernel.

Kernel Construction

Prop. The function $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ defined by

$$k(\vec{x}, \vec{y}) = \exp\left(-\frac{\|\vec{x} - \vec{y}\|_2^2}{2\sigma^2}\right) \quad \text{for all } \vec{x}, \vec{y} \in \mathbb{R}^d$$

is a kernel for any d and for any $\sigma \in \mathbb{R}^+$. It is called the **Gaussian** or the **radial basis function (RBF)** kernel.

Common Kernel Functions

common kernel functions over $\mathbb{R}^d \times \mathbb{R}^d$:

linear kernel: $k(\vec{x}, \vec{y}) := \vec{x}^\top \vec{y}$

polynomial kernel: $k(\vec{x}, \vec{y}) := (\vec{x}^\top \vec{y} + c)^k$

Gaussian or RBF kernel: $k(\vec{x}, \vec{y}) = \exp\left(-\frac{\|\vec{x}-\vec{y}\|_2^2}{2\sigma^2}\right)$

Recap: Dual Optimization Problem

solve:

$$\max_{\vec{\alpha}} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \left\langle \vec{x}_i, \vec{x}_j \right\rangle$$

$$\text{s.t.} \quad \sum_{i=1}^{n} \alpha_i y_i = 0 \text{ and}$$

$$\alpha_i \geq 0 \quad i = 1, \ldots, n$$

maximum margin hyperplane: $\left(\vec{w} = \sum_{i=1}^{n} y_i \alpha_i \vec{x}_i \right)$:

$$f(\vec{x}) = \sum_{i=1}^{n} y_i \alpha_i \left\langle \vec{x}_i, \vec{x} \right\rangle + b$$

$$b = -\frac{1}{2} \left(\max_{y_j = -1} \left(\sum_{i=1}^{n} y_i \alpha_i \left\langle \vec{x}_i, \vec{x}_j \right\rangle \right) + \min_{y_i = 1} \left(\sum_{i=1}^{n} y_i \alpha_i \left\langle \vec{x}_i, \vec{x}_j \right\rangle \right) \right)$$

Dual Form: Remark

optimization problem:

$$\max_{\vec{\alpha}} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle \vec{x}_i, \vec{x}_j \rangle$$

$$\text{s.t.} \quad \sum_{i=1}^{n} \alpha_i y_i = 0 \text{ and}$$

$$\alpha_i \geq 0, i = 1, \ldots, n$$

remark 1. input data and new points ($\vec{x}$) are used only through inner products

$\Rightarrow$ **kernel trick is applicable!**

maximum margin hyperplane:

$$f(\vec{x}) = \sum_{i=1}^{n} y_i \alpha_i \langle \vec{x}_i, \vec{x} \rangle + b$$

$$b = -\frac{1}{2} \left(\max_{y_j = -1} \left(\sum_{i=1}^{n} y_i \alpha_i \langle \vec{x}_i, \vec{x}_j \rangle \right) + \min_{y_i = 1} \left(\sum_{i=1}^{n} y_i \alpha_i \langle \vec{x}_i, \vec{x}_j \rangle \right) \right)$$

Dual Form: Remark

optimization problem:

$$\max_{\vec{\alpha}} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j k(\vec{x}_i, \vec{x}_j)$$

$$\text{s.t.} \quad \sum_{i=1}^{n} \alpha_i y_i = 0 \text{ and}$$

$$\alpha_i \geq 0, i = 1, \ldots, n$$

kernel trick: replace the inner products by a kernel function

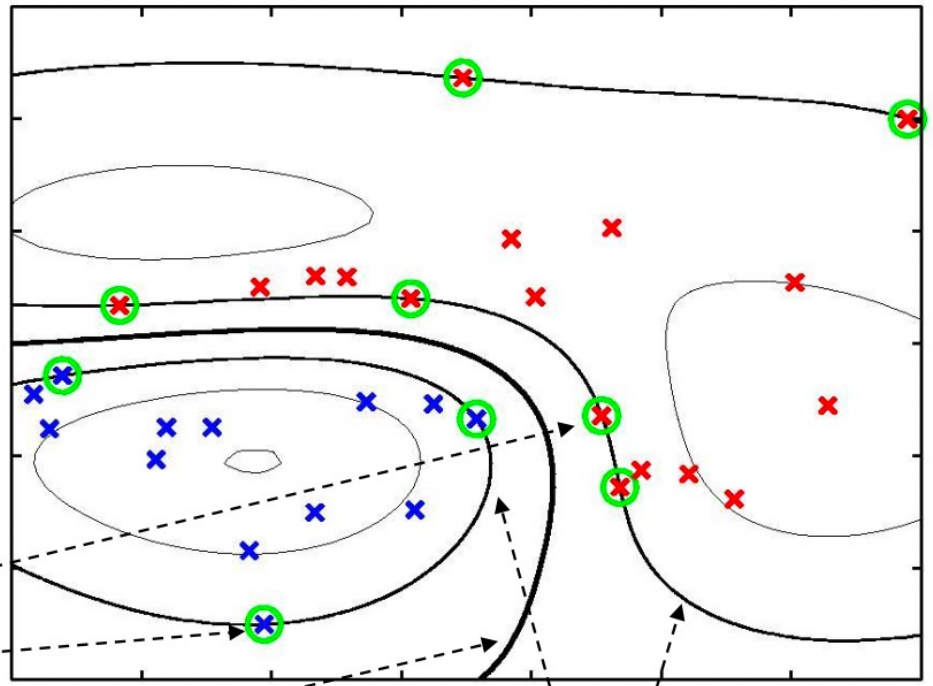
maximum margin hyperplane

$$f(\vec{x}) = \sum_{i=1}^{n} y_i \alpha_i k(\vec{x}_i, \vec{x}) + b$$

$$b = -\frac{1}{2} \left(\max_{y_j=-1} \left(\sum_{i=1}^{n} y_i \alpha_i k(\vec{x}_i, \vec{x}_j) \right) + \min_{y_i=1} \left(\sum_{i=1}^{n} y_i \alpha_i k(\vec{x}_i, \vec{x}_j) \right) \right)$$

Example

- synthetic data $(\mathbb{R}^2)$ with two classes
- Gaussian kernel
- contours: constant $f(\vec{x})$



support vectors

decision boundary
(hyperplane in the feature space)

margin boundaries
(hyperplanes in the feature space)