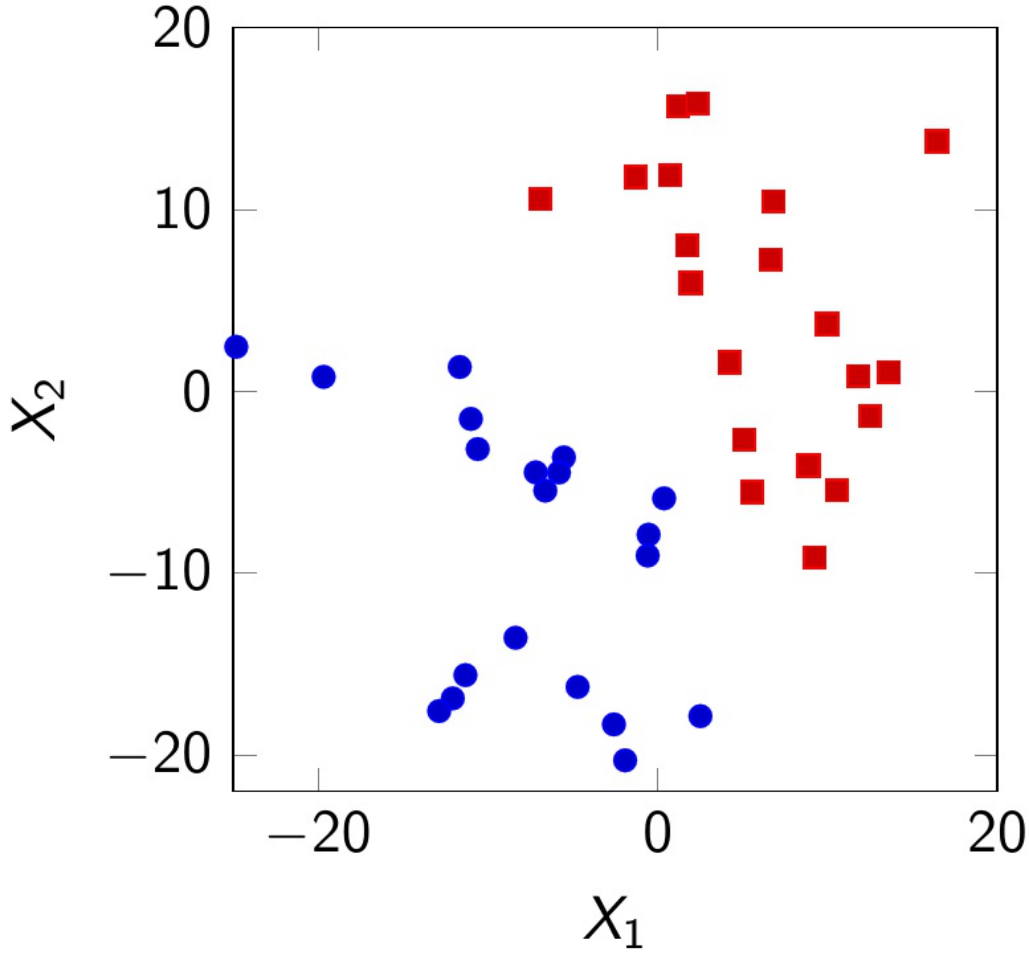
Section 6

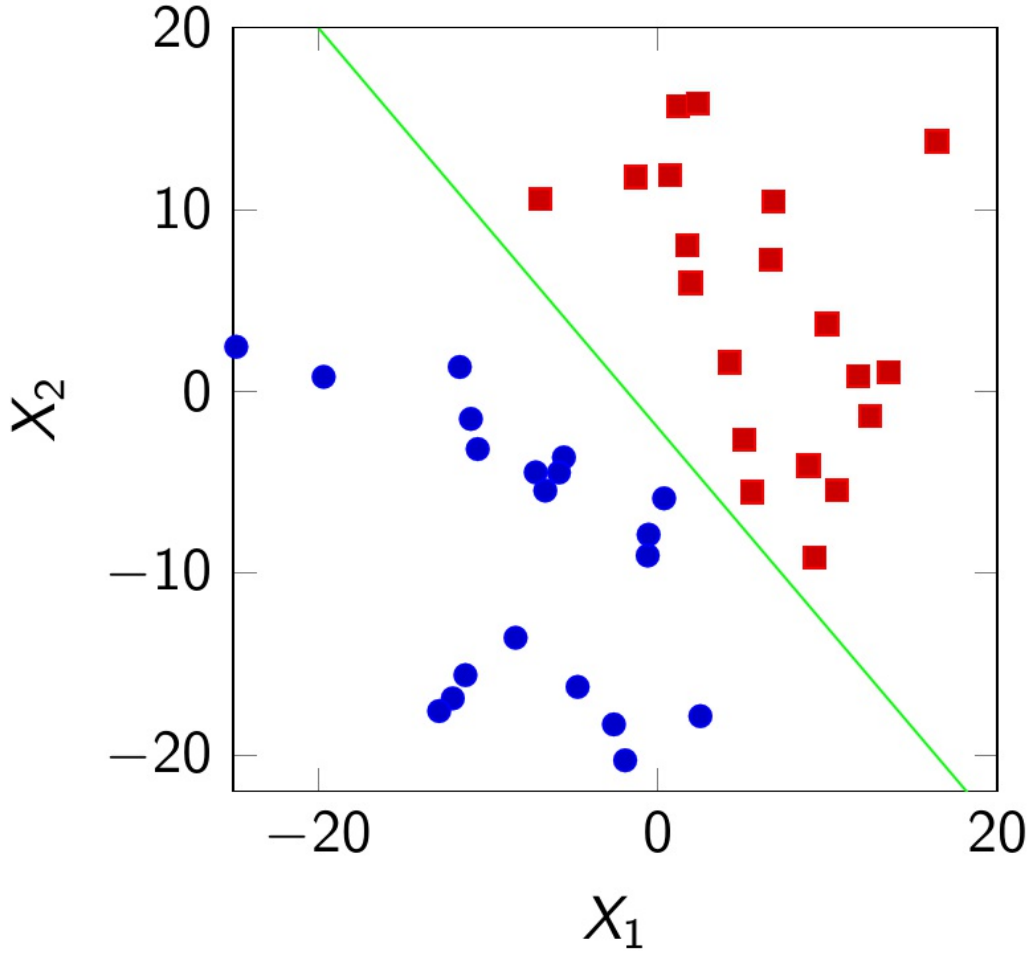
Support Vector Machines

Binary classification



Let's draw a decision boundary!

Binary classification



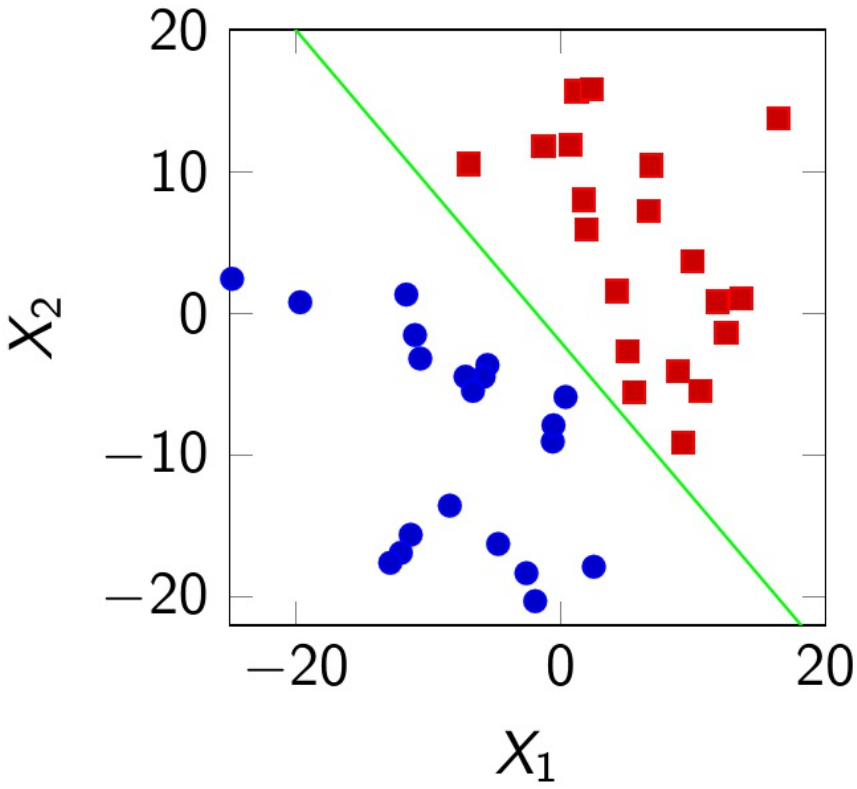
Let's draw a decision boundary!

Hyperplane

- We drew a line:
$$X_2 = mX_1 + q$$
- which can be written also as:
$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0$$
- or, when the feature space is p-dimensional:
$$\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p = 0$$

  the line is a *separating hyperplane*.

Classification with a separating hyperplane

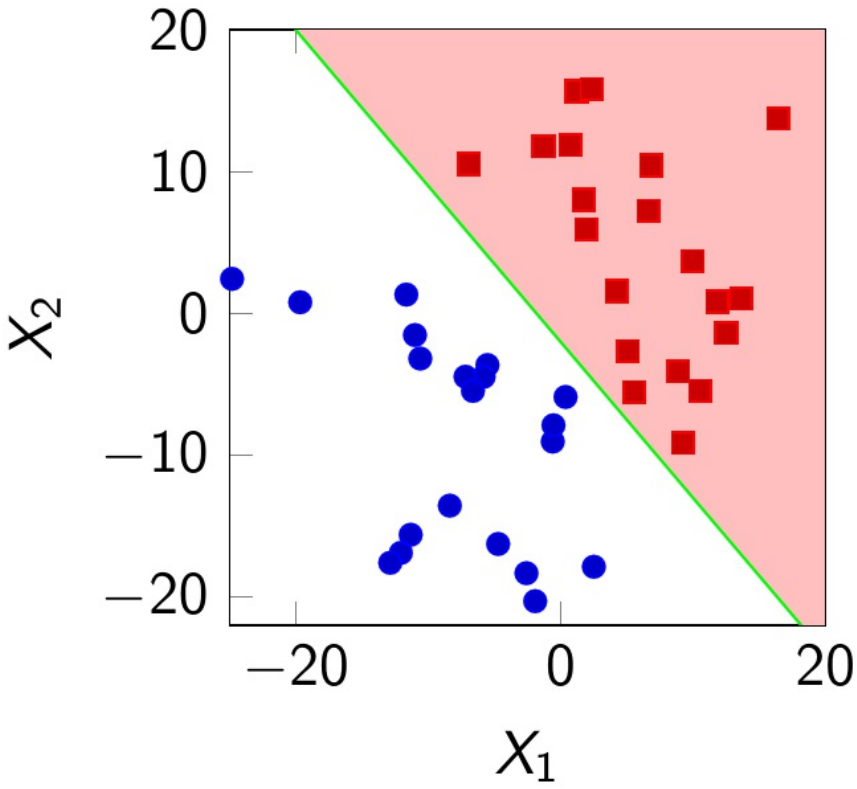


The hyperplane:

$$\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p = 0$$
$$1.1 X_1 + X_2 + 27 = 0$$

Given an observation (x_1, x_2):

Classification with a separating hyperplane



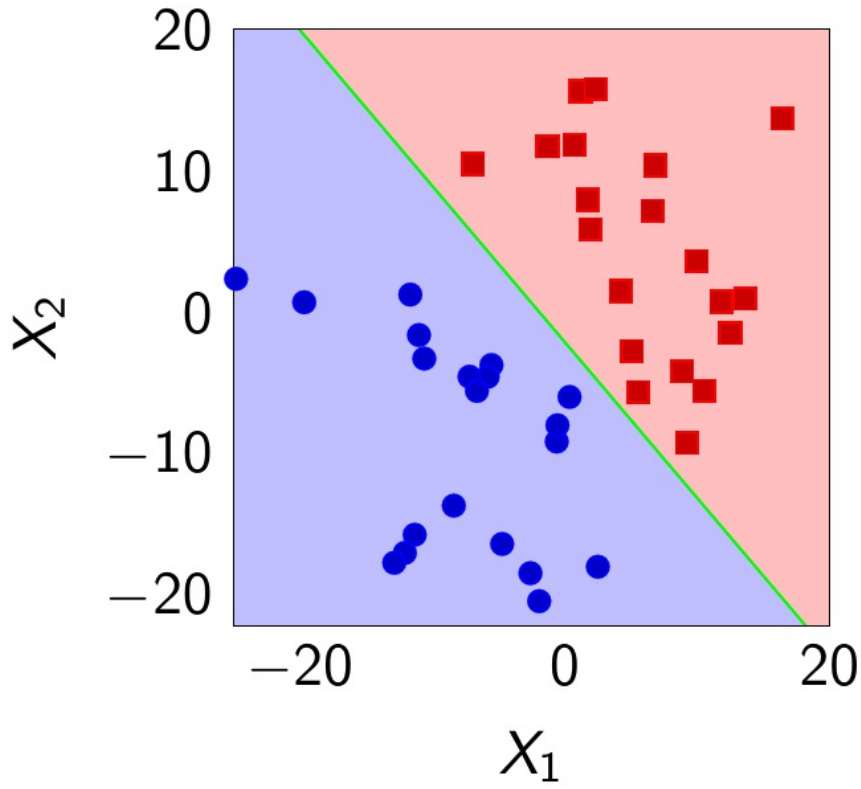
The hyperplane:

$$\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p = 0$$
$$1.1X_1 + X_2 + 27 = 0$$

Given an observation (x_1, x_2):

- if $1.1X_1 + X_2 + 27 > 0$ then
  -

Classification with a separating hyperplane



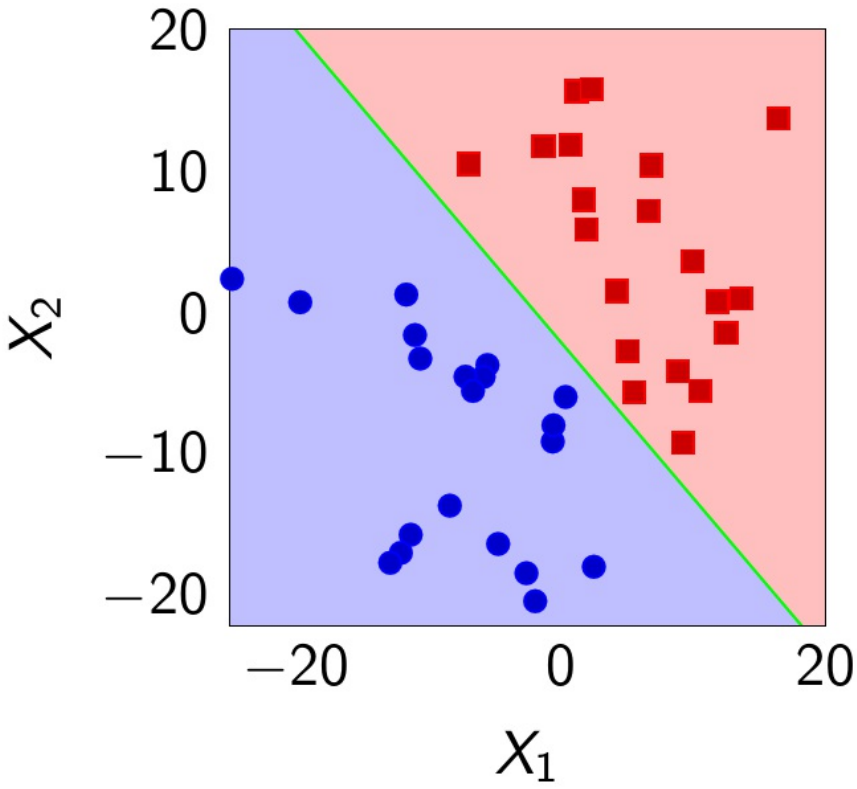
The hyperplane:

$$\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p = 0$$
$$1.1X_1 + X_2 + 27 = 0$$

Given an observation (x_1, x_2):

- if $1.1X_1 + X_2 + 27 > 0$ then
  - ■
- if $1.1X_1 + X_2 + 27 < 0$ then
  - ●

Classification with a separating hyperplane



The hyperplane:

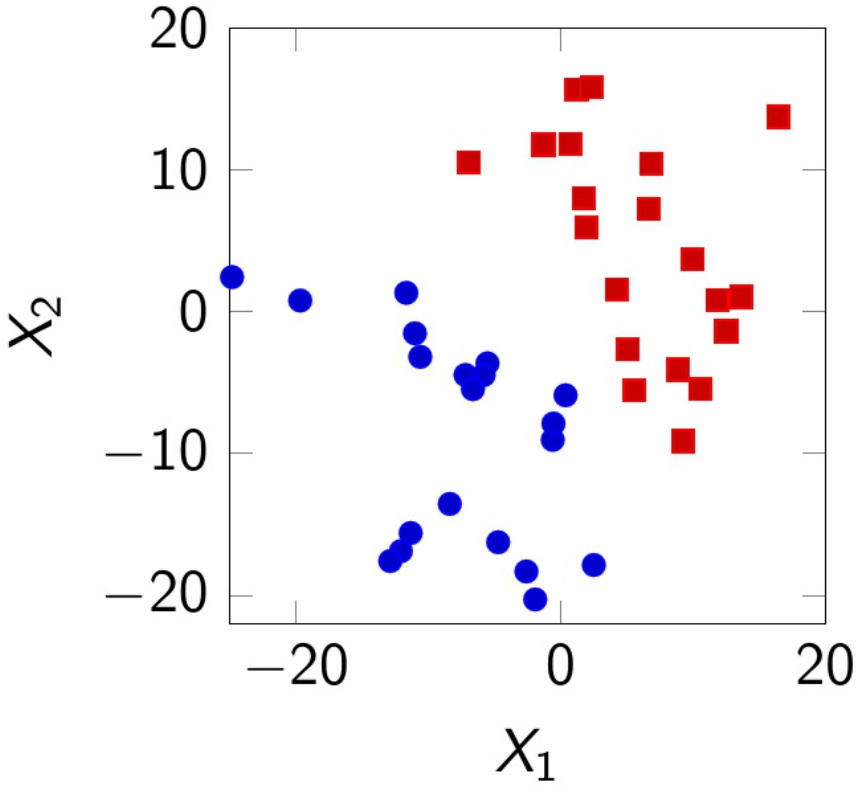
$$\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p = 0$$
$$1.1X_1 + X_2 + 27 = 0$$

Given an observation (x_1, x_2):

- if $1.1X_1 + X_2 + 27 > 0$ then
  
- if $1.1X_1 + X_2 + 27 < 0$ then
  

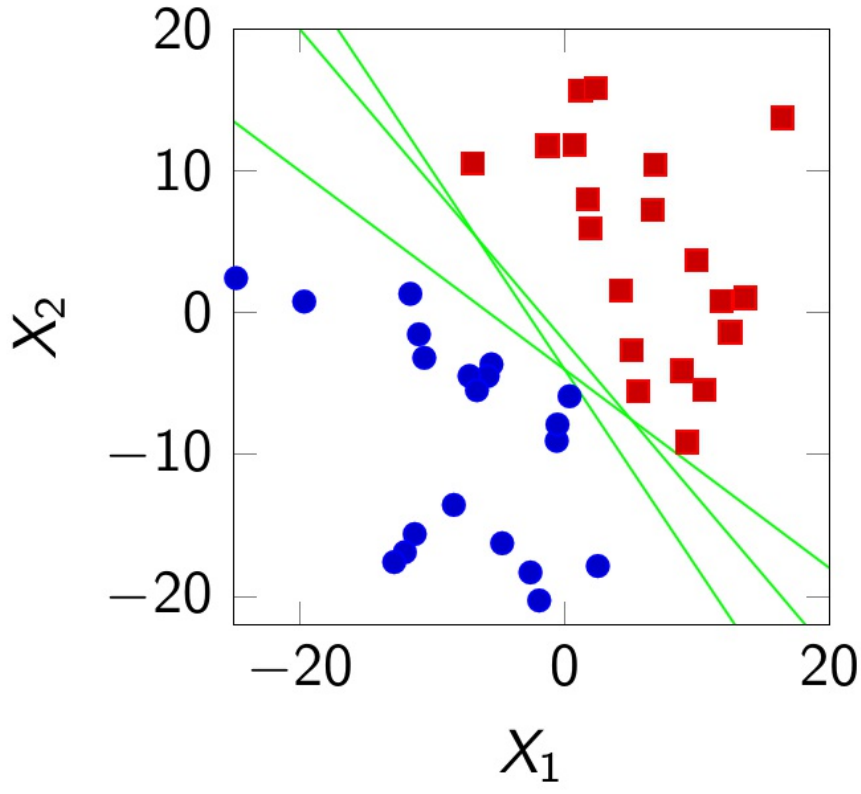
The larger the difference, the stronger the confidence

Learning a separating hyperplane



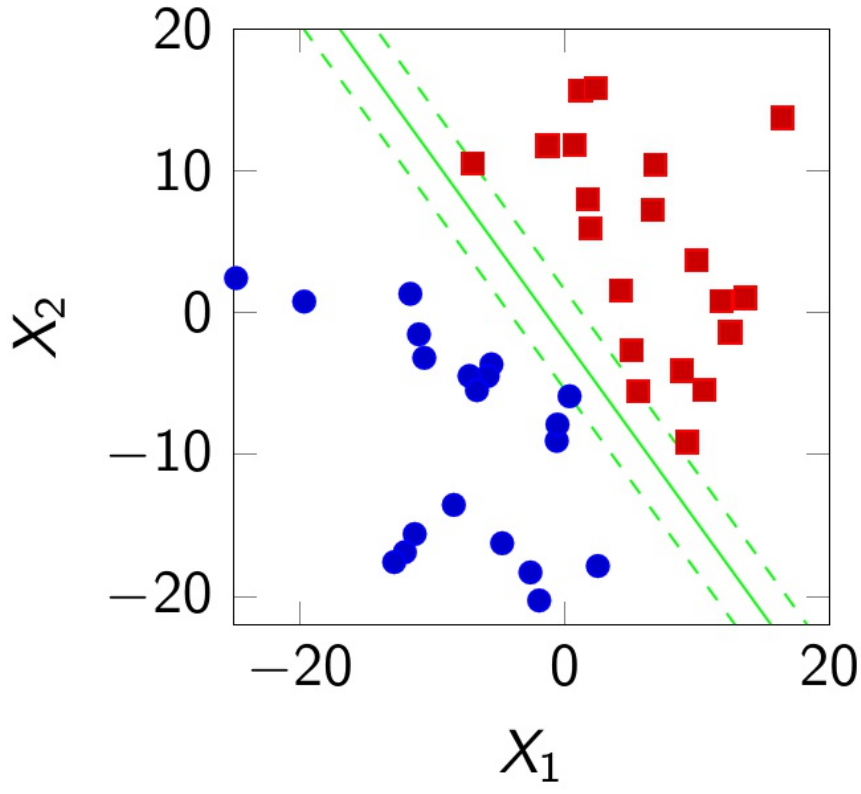
- We want an hyperplane which perfectly separates the learning data...

Learning a separating hyperplane



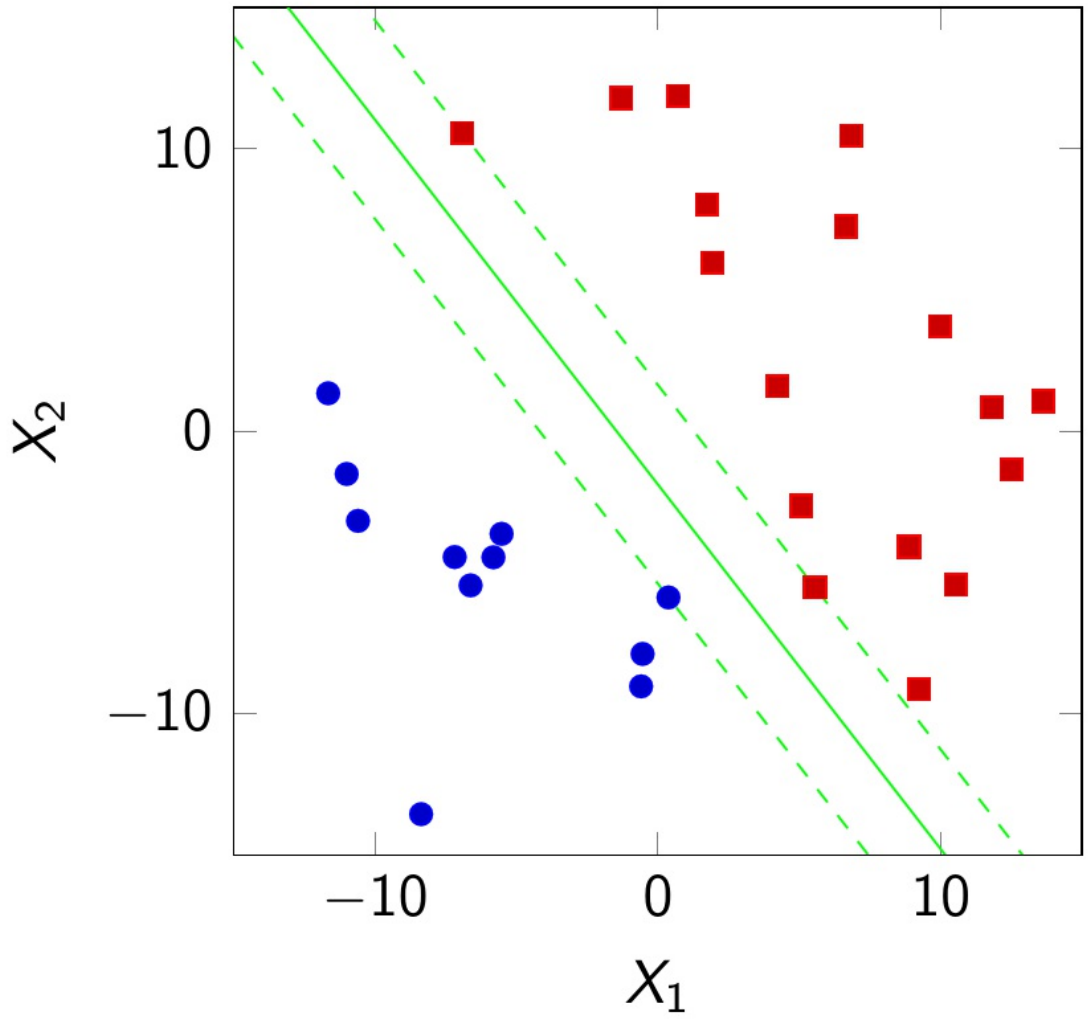
- We want an hyperplane which perfectly separates the learning data...
- ...but there could be many (∞) of them! Which one?

Learning a separating hyperplane

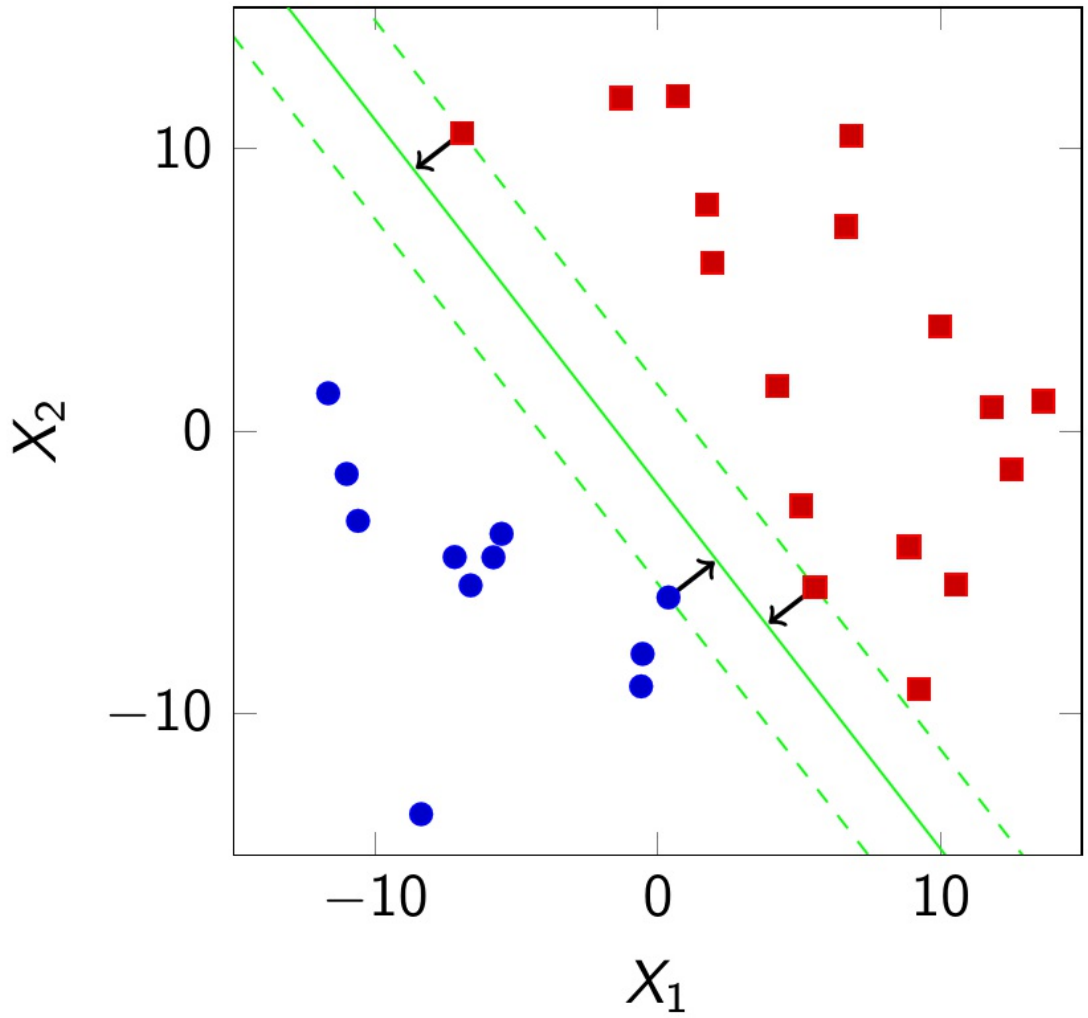


- We want an hyperplane which perfectly separates the learning data...
- ...but there could be many (∞) of them! Which one?
- Idea: the farthest from the learning observations!
  Maximal margin classifier

Maximal margin classifier

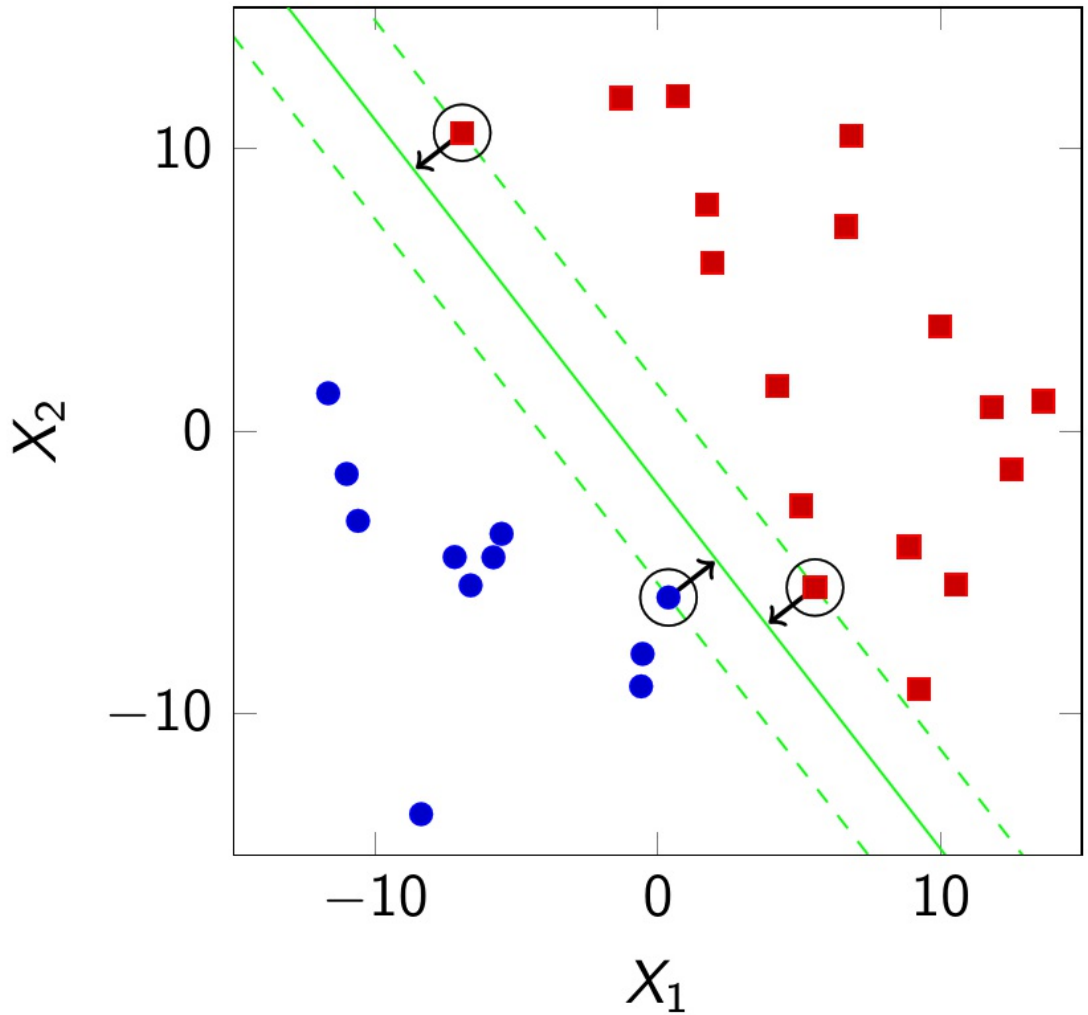


Maximal margin classifier



- margin M

Maximal margin classifier



- margin M
- *support vectors*

Learning the maximal margin classifier

- Find the line which:
  1. perfectly separates learning observations
  2. has the largest margin from support vectors

Looks like an optimization problem...

Learning the maximal margin classifier

$$\max_{\beta_0,\ldots,\beta_p} M$$

under constraints

$$\sum_{j=1}^{p} \beta_j^2 = 1$$

$$\forall i \in \{1,\ldots,n\}, y_i(\beta_0 + \beta_1 x_{i,1} + \cdots + \beta_p x_{i,p}) \geq M$$

Some math tricks:

- if $\sum_{j=1}^{p} \beta_j^2 = 1$, then $|\beta_0 + \beta_1 x_{i,1} + \cdots + \beta_p x_{i,p}|$ is the distance between x_i^T and the hyperplane
- if $y \in \{1, -1\}$, then writing $y_i(\ldots) \geq M$ is like writing $\cdots \geq M, \forall$ ■ and $\cdots \leq M, \forall$ ●

Support vectors

$$\forall i \in \{1, \ldots, n\}, y_i(\beta_0 + \beta_1 x_{i,1} + \cdots + \beta_p x_{i,p}) = M$$

They lie exactly on the margin!

Learning the maximal margin classifier

Looks like an optimization problem...

...which is not hard to be solved.

Maximal margin classifier issues

- What if the learning data is not perfectly separable?
  - cannot learn!
- What if a learning observation (being a support vector) is added/removed?
  - could learn a very different classifier $\rightarrow$ high variance!

High variance of Maximal margin

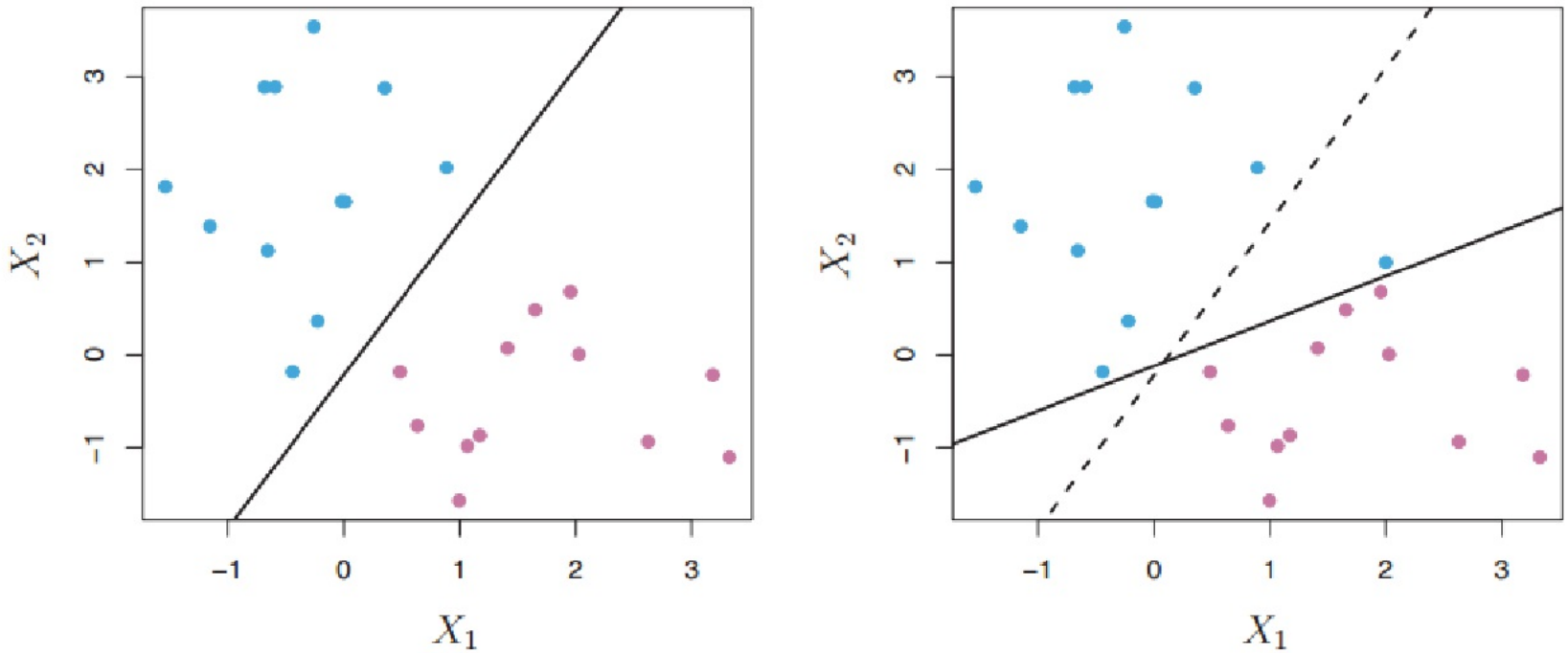


Image from An Introduction to Statistical Learning

Soft margin

How to cope with these issues?

- Idea: be more tolerant!
  - some learning observation may be within the margin
  - some learning observation may be misclassified

Margin can be exceeded $\rightarrow$ *soft margin classifier* or *support vector classifier*

Learning with toleration: support vector classifier

$$\max_{\beta_0,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n} M$$

under constraints

$$\sum_{j=1}^{p} \beta_j^2 = 1$$

$$\forall i \in \{1,\ldots,n\}, y_i(\beta_0 + \beta_1 x_{i,1} + \cdots + \beta_p x_{i,p}) \geq M(1-\epsilon_i)$$

$$\forall i \in \{1,\ldots,n\}, \epsilon_i \geq 0$$

$$\sum_{j=1}^{n} \epsilon_j = C$$

- ϵ_i are positive slack variables
  - if $\epsilon_i \in]0,1[$, then ${x_i}^T$ is within the margin
  - if $\epsilon_i \in [1,\infty[$, then ${x_i}^T$ is misclassified
- C is the toleration budget ($C = 0 \rightarrow$ maximal margin classifier)

Role of the parameter C

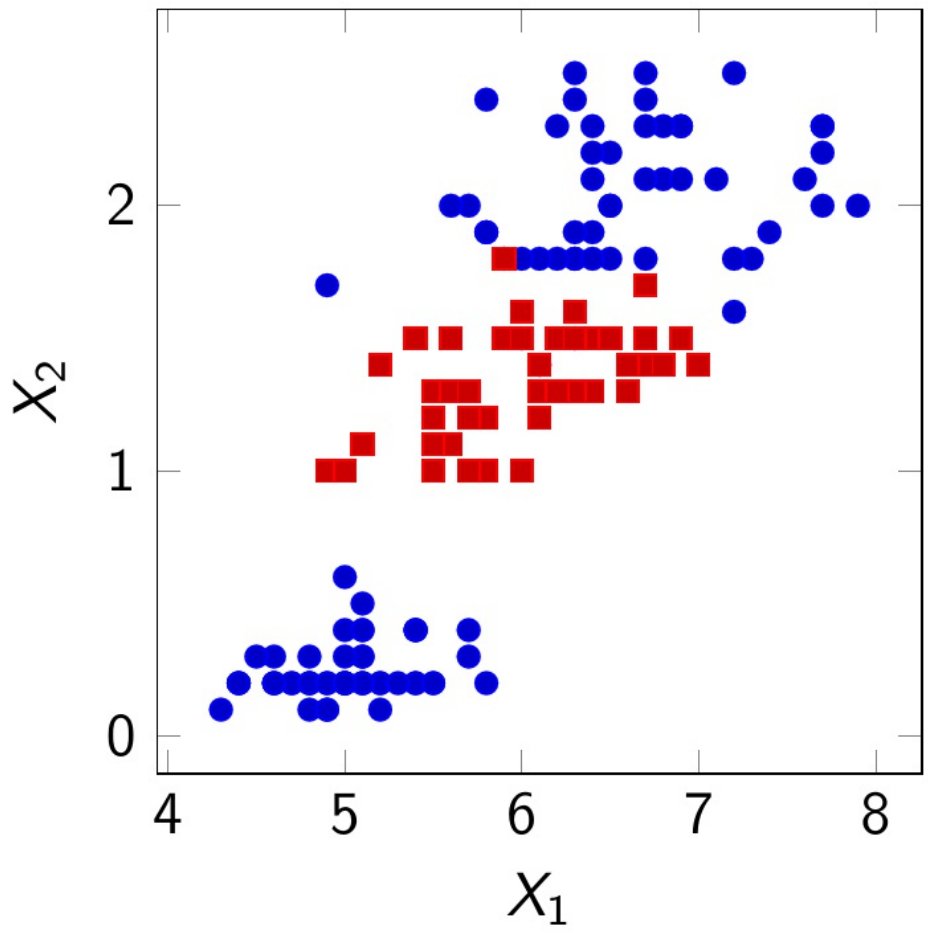
The larger C

- the larger the toleration
- the larger the number of learning observations which can exceed the margin (or be misclassified)
- the larger the number of support vectors
- the lower the variance

Summary

	Maximal margin	Soft margin
fast to learn	▲	▲
variance	▼	▲
robustness to "trivial" observations	▲	▲

Linearity?



Some problems cannot be solved with an hyperplane!

Some math rewriting

Finding values for $\beta_0, \ldots, \beta_p$ involves computing inner products between pair of observations:

$$\langle x_i, x_{i'} \rangle = \sum_{j=1}^{p} x_{i,j} x_{i',j}$$

And we can rewrite:

$$\beta_0 + \sum_{i=1}^{p} \beta_i x_i^\star = f(x^\star) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x^\star, x_i \rangle$$

For non support vectors, $\alpha_i = 0 \Rightarrow x_i$ does not impact on $f(x^\star)$!

Non support vectors

$$f(x^\star) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x^\star, x_i \rangle$$

- $f(x^\star)$ is the distance of $x^\star$ from the decision boundary
- the (position of the) decision boundary depends only on the support vectors
- $\Rightarrow f(x^\star)$ depends only on the support vectors

When predicting:

$$f(x^\star) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i \langle x^\star, x_i \rangle$$

Kernel

Equation for the decision boundary can be generalized

$$f(x^\star) = \beta_0 + \sum_{i=1}^{n} \alpha_i K(x^\star, x_i)$$

Where $K(x^\star, x_i)$ is a function $K : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$, called *kernel* (with some other properties).

Support Vector Machines

Intuition for the kernel

Consider prediction:

$$f(x^\star) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i K(x^\star, x_i)$$

- $x^\star$ is mapped from $\mathbb{R}^p$ to $\mathbb{R}^{p'}$, with $p' \gg p$, using a function Φ: $K(x_i, x_j)$ computes the inner product $\langle \phi(x_i), \phi(x_j) \rangle$ of mapped x_i, x_j without explicitly mapping them (*kernel trick*)
- the α_i define (indirectly) an hyperplane in $\mathbb{R}^{p'}$
- the classification is done by means of a separating hyperplane in the new space, i.e., $f(x^\star)$ measures the distance of *mapped* $x^\star$ from the hyperplane

Kernels

- linear kernel:

$$K(x^\star, x_i) = \langle x_i, x^\star \rangle = \sum_{j=1}^{p} x_{i,j} x_j^\star$$

- polynomial kernel: (d is the degree)

$$K(x^\star, x_i) = \left(1 + \sum_{j=1}^{p} x_{i,j} x_j^\star\right)^d$$

- radial basis function kernel (or radial, or RBF, or Gaussian):

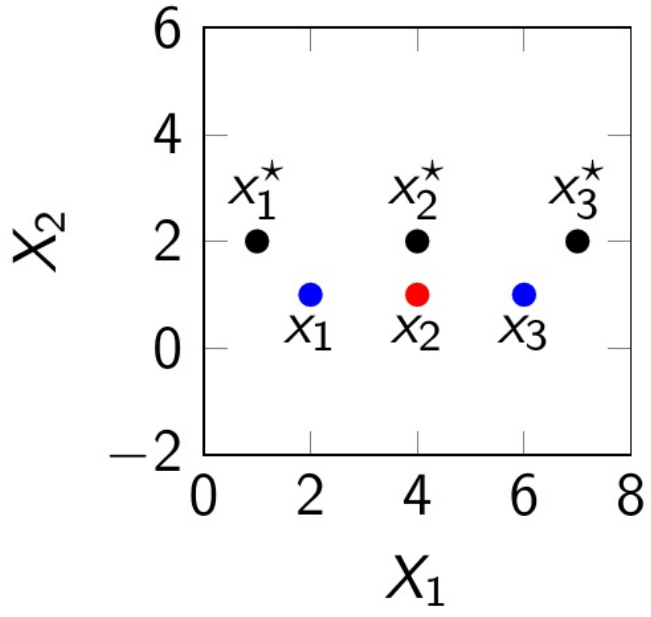
$$K(x^\star, x_i) = \exp\left(-\gamma \sum_{j=1}^{p} (x_{i,j} - x_j^\star)^2\right)$$

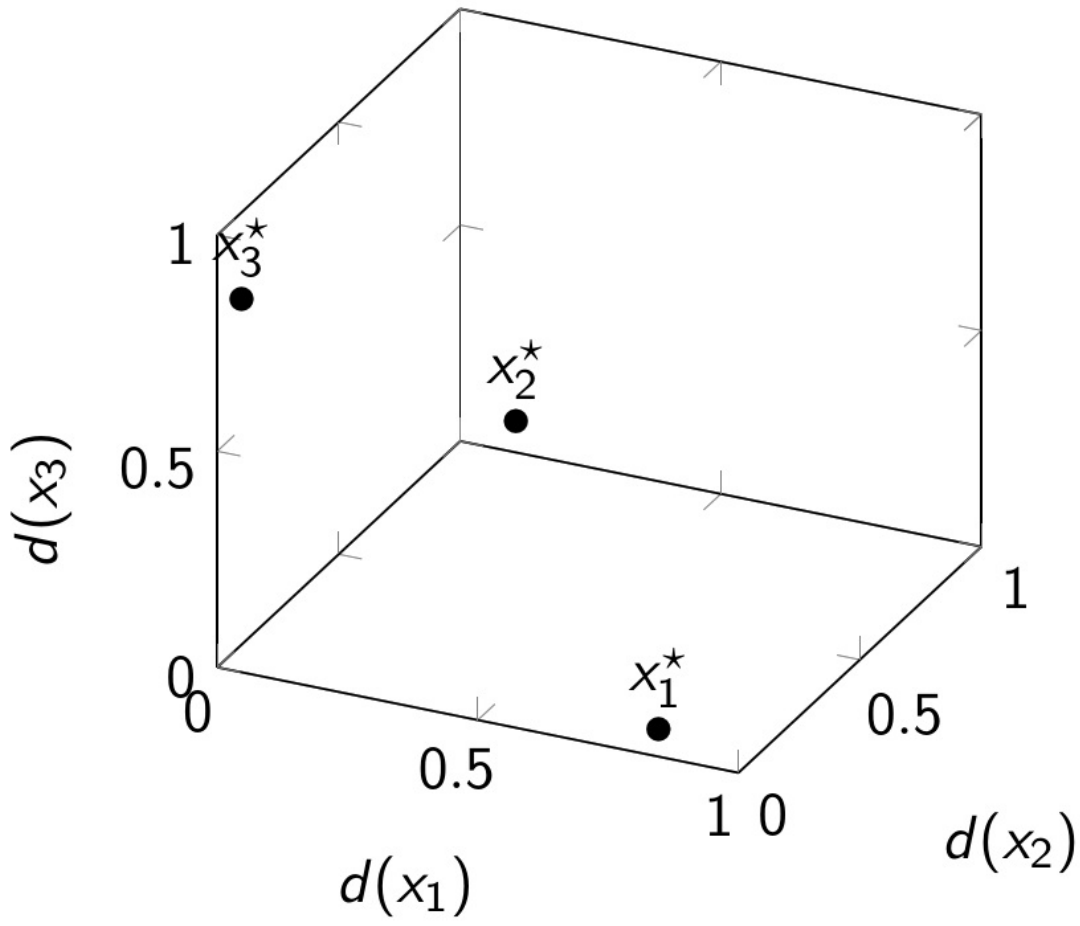
Intuition behind radial kernel

$$K(x^\star, x_i) = \exp\left(-\gamma \sum_{j=1}^{p} (x_{i,j} - x_j^\star)^2\right) = \exp\left(-\gamma ||x_i, x^\star||^2\right)$$

- the coordinates in the new space are related to the distances of $x^\star$ from the support vectors (the closer, the higher the $K(\cdot)$, $K(\cdot) \in]0, 1]$)
- γ determines how fast the coordinate goes to 0, i.e., a support vector becomes irrelevant for classifying $x^\star$

Very raw visual intuition





$d(x_1)$ "means" $\exp\left(-\gamma||x_1, \cdot||^2\right)$

Q: draw a reasonable decision boundary, in the two spaces

Multiclass (> 2) classification with SVM

- one-vs.-one classification
- one-vs.-all classification

and many other proposals...

One-vs.-one SVM

When learning:

1. for each pair $(\mathcal{C}_1, \mathcal{C}_2)$ of classes, learn a binary SVM

When predicting:

1. for each learned SVM, predict class $\hat{y}$
2. choose the most frequently predicted class

$\binom{K}{2} = \frac{K(K-1)}{2}$ binary classifiers

One-vs.-all SVM

When learning:

1. for each class $\mathcal{C}_i$, learn a binary SVM ($\mathcal{C}_i$ vs. all $\mathcal{C}_j$, with $j \neq i$, $\mathcal{C}_i$ coded as $y = +1$)

When predicting:

1. for each learned SVM, get $f(x^\star)$
2. choose the class with the largest $f(x^\star)$

K classifiers