

# Kernel methods for classification

I. Santamaría, S. Van Vaerenbergh

GTAS, Universidad de Cantabria

26 de enero de 2022

Maître Universitario Oficial **Data Science**



# Contents

Statistical Learning

Kernel Methods

Introduction

Linear SVM

Introduction

Optimization problem

Non-linear SVM

Formulation

Kernels

Implementation

Extensions

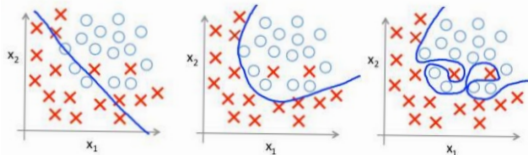
Extensions

Conclusions

Conclusions

# An example

3 classifiers trained over the dataset shown in the figure



Which one will perform better over a different **test** dataset?

There is a tradeoff between:


- ▶ Training error / test error (generalization error, aka out-of-sample error)
- ▶ Bias/variance of the model/classifier

**Statistical Learning Theory** places these ideas in a mathematical framework, characterizing the properties of learning machines

# Statistical Learning

- ▶ Supervised binary classification problem

$$f(\mathbf{x}) : \mathcal{X} \rightarrow \{\pm 1\}$$

- ▶ Training dataset:  $(\mathcal{X}, \mathcal{Y}) = \{(\mathbf{x}_i, y_i)\}$  
- ▶ **Loss function**:  $l(\mathbf{x}, y, f)$  (e.g.,  $l(\mathbf{x}, y, f) = \frac{1}{2}|f(\mathbf{x}) - y|$ )
- ▶ A good classifier should minimize the **risk or test error**

$$R[f] = \int \frac{1}{2}|f(\mathbf{x}) - y|dP(\mathbf{x}, y)$$

- ▶ As we are only given the training data, we can minimize only the **empirical risk or training error**

$$R_{emp}[f] = \sum_{i=1}^n \frac{1}{2}|f(\mathbf{x}_i) - y_i|$$

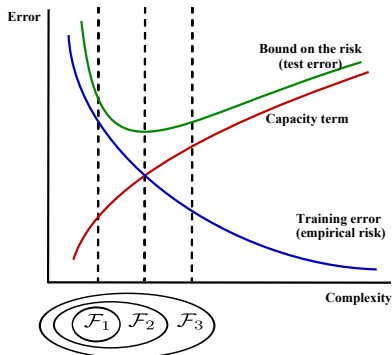


- The test error can be bounded as

$$R[f] \leq R_{emp}[f] + \phi(f)$$

where  $\phi(f)$  is a capacity term that measures the complexity of the set of functions from which  $f$  is chosen

- It is imperative to restrict the set of functions  $f(\mathbf{x})$



**Structural Risk Minimization** o **Regularized Empirical Risk Minimization**: To minimize a regularized version of the training error



$$\text{minimize } R_{\text{emp}}[f] + \lambda \Omega(f),$$

where  $\Omega(f)$  measures the complexity of the classifier (learning machine) and  $\lambda$  is the regularization parameter



- ▶  $\lambda \uparrow$  Simple models/class. boundaries
- ▶  $\lambda \downarrow$  More complex models/class. boundaries (**overfitting** risk)

Typically  $\lambda$  is estimated by cross-validation

# Introducción

- ▶ Many machine learning algorithms (still) need a suitable **feature space** to perform satisfactorily
- ▶ Dimensionality reduction techniques (PCA/LDA) are routinely used in many applications

$$\mathbf{x}_i \in \mathcal{R}^d \longrightarrow \mathbf{W}\mathbf{x}_i \in \mathcal{R}^r, \quad r < d$$

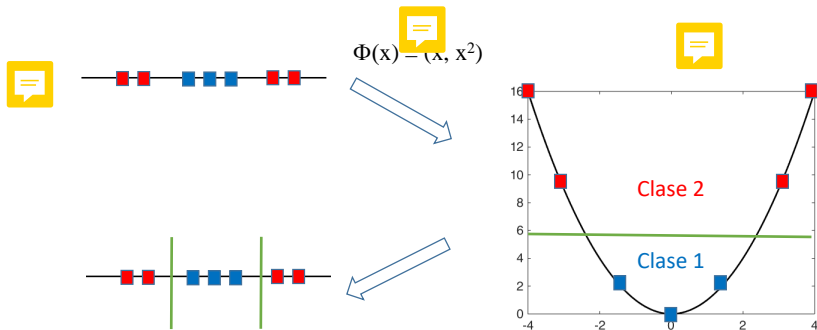
- ▶ Kernel methods follow a different approach: map the data to a higher dimensionality space



$$\mathbf{x}_i \in \mathcal{R}^d \longrightarrow \Phi(\mathbf{x}_i) \in \mathcal{R}^r, \quad r \gg d$$

Why?

- ▶ Let's consider a simple binary 1D classification problem
- ▶ Training dataset: { -4, -3, -1, 0, 1, 3, 4 }

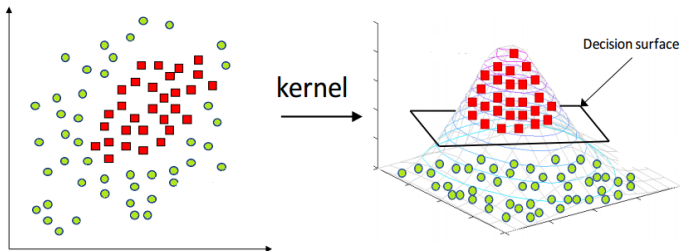


- ▶  $\Phi(x) = [x, x^2]^T$  produces a linearly separable problem in a 2D feature space



- ▶ In practice, there is no need to know the mapping  $\Phi(\mathbf{x})$  explicitly
- ▶ We just need its **kernel** function

$$K(\mathbf{x}, \mathbf{x}') = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle = \Phi(\mathbf{x})^T \Phi(\mathbf{x}')$$



- ▶ Kernel methods obtain a linear solution in the feature space, which becomes a nonlinear solution in the input space

# Support Vector Machine (SVM)



- ▶ The Support Vector Machine **SVM** is the most popular kernel machine for classification
- ▶ It solves a linear classification problem in the feature space applying the SRM principle

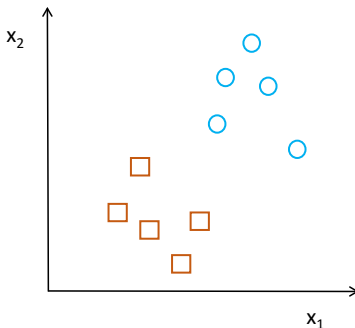
$$\min_{f(\cdot) \in \mathcal{F}} \sum_{i=1}^n \frac{1}{2} |f(\mathbf{x}_i) - y_i| + \lambda \Omega(f)$$

- ▶ Let's start with the linear SVM working in the input space
  - ▶  $f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$  : **Optimal Hyperplane**



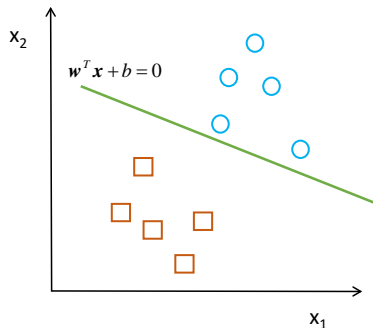
# Linear SVM

- ▶ Binary classification problem:  $\{(\mathbf{x}_i, y_i = \pm 1)\}$
- ▶ Linear classifier:  $f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b = \langle \mathbf{w}, \mathbf{x} \rangle + b$
- ▶ Linearly separable data:  $y_i(\mathbf{w}^T \mathbf{x} + b) \geq 0, i = 1, \dots, n$



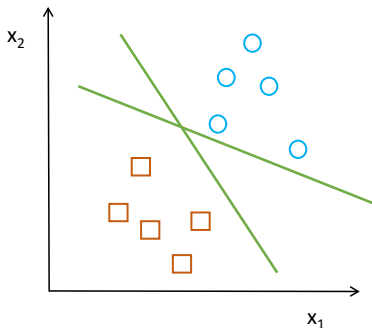
# Linear SVM

- ▶ Binary classification problem:  $\{(\mathbf{x}_i, y_i = \pm 1)\}$
- ▶ Linear classifier:  $f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b = \langle \mathbf{w}, \mathbf{x} \rangle + b$
- ▶ Linearly separable data:  $y_i(\mathbf{w}^T \mathbf{x} + b) \geq 0, i = 1, \dots, n$



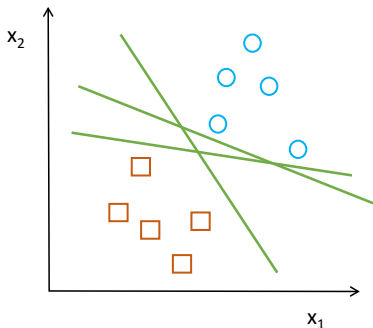
# Linear SVM

- ▶ Binary classification problem:  $\{(\mathbf{x}_i, y_i = \pm 1)\}$
- ▶ Linear classifier:  $f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b = \langle \mathbf{w}, \mathbf{x} \rangle + b$
- ▶ Linearly separable data:  $y_i(\mathbf{w}^T \mathbf{x} + b) \geq 0, i = 1, \dots, n$



# Linear SVM

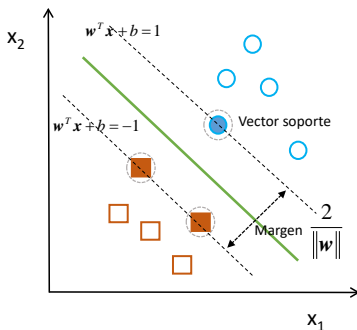
- ▶ Binary classification problem:  $\{(\mathbf{x}_i, y_i = \pm 1)\}$
- ▶ Linear classifier:  $f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b = \langle \mathbf{w}, \mathbf{x} \rangle + b$
- ▶ Linearly separable data:  $y_i(\mathbf{w}^T \mathbf{x} + b) \geq 0, i = 1, \dots, n$



- Scale  $\mathbf{w}$  and  $b$  so that the closest points to the hyperplane satisfy:

$$|\mathbf{w}^T \mathbf{x} + b| = 1 \implies y_i(\mathbf{w}^T \mathbf{x} + b) \geq 1, \forall i$$

- The optimal hyperplane maximizes the **margin**



- The support vectors  $\mathbf{w}^T \mathbf{x}_j + b = \pm 1$  determine the optimal, or maximum margin, hyperplane

## Optimization problem

$$\begin{aligned} \min_{\mathbf{w}, b} \quad & \frac{1}{2} \|\mathbf{w}\|^2 \\ \text{s.t.} \quad & y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1, \quad \forall i \end{aligned}$$

It is a **convex** problem → the solution is unique





# Solution

- The Lagrangian is

$$\mathcal{L}(\mathbf{w}, b, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 + \sum_{i=1}^n \alpha_i \left( 1 - y_i (\mathbf{w}^T \mathbf{x}_i + b) \right)$$

- Strong duality  $\Rightarrow$  KKT optimality

1. The optimal hyperplane is a linear combination of the input patterns

$$\nabla \mathcal{L}_{\mathbf{w}}(\mathbf{w}, b, \alpha) = \mathbf{w} + \sum_{i=1}^n \alpha_i y_i \mathbf{x}_i = 0 \Rightarrow \boxed{\mathbf{w} = - \sum_{i=1}^n \alpha_i y_i \mathbf{x}_i}$$

2. The optimal hyperplane only depends on a few (closest) patterns: the support vectors

$$\alpha_i (1 - y_i (\mathbf{w}^T \mathbf{x}_i + b)) = 0, \forall i \Rightarrow y_i (\mathbf{w}^T \mathbf{x}_i + b) = 1$$

3. The bias  $b$  can be found from any support vector

Substituting  $\mathbf{w} = \sum_{i=1}^n \alpha_i y_i \mathbf{x}_i$  in the Lagrangian, we obtain the **dual problem**, which is the problem we actually solve

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j - \sum_i \alpha_i \\ \text{s.t.} \quad & \sum_i \alpha_i y_i = 0, \\ & \alpha_i \geq 0, \quad \forall i \end{aligned}$$

Defining  $\alpha = (\alpha_1, \dots, \alpha_n)^T$ ,  $\mathbf{1} = (1, \dots, 1)^T$ ,  $\mathbf{Y} = \text{diag}(y_1, \dots, y_n)$  and the  $n \times n$  matrix  $\mathbf{K}$  with elements  $k(i, j) = \mathbf{x}_i^T \mathbf{x}_j = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$ , the problem can be written as

### QP (Quadratic Programming) Problem

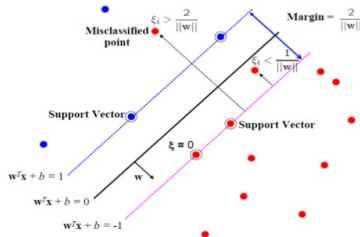
$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \alpha^T \mathbf{Y} \mathbf{K} \mathbf{Y} \alpha - \mathbf{1}^T \alpha \\ \text{s.t.} \quad & \alpha^T \mathbf{y} = 0, \\ & \alpha \geq 0 \end{aligned}$$

# Soft-margin SVM




- ▶ Non-linearly separable classes
- ▶ We introduce **slack variables** into the optimization problem to allow for classification errors:  $\xi_i$
- ▶ Regularization parameter  $C \rightarrow$  penalty
- ▶ Still a QP problem

$$\begin{aligned}
 \min_{\mathbf{w}, b} \quad & \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_i \xi_i \\
 \text{s.t.} \quad & y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i, \quad \forall i \\
 & \xi_i \geq 0 \quad \forall i
 \end{aligned}$$



# Non-linear SVM

- ▶ The input patterns are mapped to a higher dimensionality (probably  $\infty$ ) feature space:  $\mathbf{x}_i \rightarrow \Phi(\mathbf{x}_i)$
- ▶ We solve a linear SVM problem in the feature space
- ▶ Optimal hyperplane in the feature space 

$$\mathbf{w} = \sum_i \alpha_i y_i \Phi(\mathbf{x}_i)$$

- ▶ Same dual problem

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \alpha^T \mathbf{Y} \mathbf{K} \mathbf{Y} \alpha - \mathbf{1}^T \alpha \\ \text{s.t.} \quad & \alpha^T \mathbf{y} = 0, \\ & 0 \leq \alpha \leq C \end{aligned}$$

but now the  $n \times n$  kernel matrix  $\mathbf{K}$  has elements

$$k(i, j) = \Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$$

- ▶ A linear classifier in the feature space

$$f(\mathbf{x}) = \mathbf{w}^T \Phi(\mathbf{x}) + b$$



- ▶ But a nonlinear classifier in the input space
- ▶ The decision function can be expressed in terms of the kernel function



$$\begin{aligned} f(\mathbf{x}) &= \underbrace{\left( \sum_i \alpha_i y_i \Phi(\mathbf{x}_i) \right)^T}_{\mathbf{w}^T} \Phi(\mathbf{x}) + b \\ &= \sum_i \alpha_i y_i \Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}) + b = \sum_i \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) + b \end{aligned}$$

- ▶ This is the **kernel trick**!

## Example: polynomial kernel



- ▶ Consider a problem with 2D patterns  $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$
- ▶ And the following polynomial mapping to a feature 3D space

$$\Phi(\mathbf{x}) = \begin{bmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}x_1x_2 \end{bmatrix}$$



- ▶ The corresponding kernel function is

$$\begin{aligned} k(\mathbf{x}, \mathbf{y}) &= \langle \Phi(\mathbf{x}), \Phi(\mathbf{y}) \rangle = \Phi(\mathbf{x})^T \Phi(\mathbf{y}) = \\ &= x_1^2 y_1^2 + x_2^2 y_2^2 + 2x_1 y_1 x_2 y_2 \end{aligned}$$



# Kernel functions

## Mercer Theorem (informal statement)

Any function  $k(\cdot, \cdot)$  that produces a positive definite kernel matrix  $\mathbf{K}$  for any training dataset

$$\mathbf{x}^T \mathbf{K} \mathbf{x} \geq 0, \quad \forall \mathbf{x},$$

induces an inner product in a Hilbert space (feature space).  
That is,

$$k(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle = \Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}_j)$$

- Note that the mapping  $\Phi(\mathbf{x})$  does not have to be known
- As long as we choose a positive definite kernel  $\Rightarrow$  QP dual problem

# Kernels



► **Linear**

$$k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$$



► **Polynomial** (parameters  $p$  y  $c$ )

$$k(\mathbf{x}_i, \mathbf{x}_j) = \left( \mathbf{x}_i^T \mathbf{x}_j + c \right)^p$$

► **Gaussian** (parameter  $\sigma^2$ , or  $\gamma = \frac{1}{2\sigma^2}$ )

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp \left( -\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2} \right)$$

► Let  $k_1(\mathbf{x}, \mathbf{y})$  and  $k_2(\mathbf{x}, \mathbf{y})$  be kernels, then the following functions are also kernels

1.  $k_1(\mathbf{x}, \mathbf{y}) + k_2(\mathbf{x}, \mathbf{y})$  
2.  $k_1(\mathbf{x}, \mathbf{y})k_2(\mathbf{x}, \mathbf{y})$
3.  $\exp(k_1(\mathbf{x}, \mathbf{y}))$  

► The sigmoid function  $\tanh(\mathbf{x}^T \mathbf{y} + b)$  is not a valid kernel



# String kernel



It is also possible to define kernel functions over non-vectorial or non-Euclidean spaces (e.g., text strings)

- ▶ Given two sequences

$s$  = *statistics*

$t$  = *computation*

- ▶ Generate all substrings of a given length (e.g., 3)

$s \rightarrow \{sta, tat, ati, tis, ist, sti, tic, ics\}$

$t \rightarrow \{com, omp, mpu, put, uta, tat, ati, tio, ion\}$

- ▶ A string kernel can be defined counting the number of common substrings

$$k(s, t) = 2$$



Other kernels can be defined over structured data: text (bag of words), graphs, times series, etc

# Kernel matrix



The input to any kernel method is the kernel matrix

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & k(\mathbf{x}_1, \mathbf{x}_2) & \cdots & k(\mathbf{x}_1, \mathbf{x}_n) \\ k(\mathbf{x}_2, \mathbf{x}_1) & k(\mathbf{x}_2, \mathbf{x}_2) & \cdots & k(\mathbf{x}_2, \mathbf{x}_n) \\ \vdots & \ddots & \ddots & \vdots \\ k(\mathbf{x}_n, \mathbf{x}_1) & k(\mathbf{x}_n, \mathbf{x}_2) & \cdots & k(\mathbf{x}_n, \mathbf{x}_n) \end{bmatrix}$$

- ▶ It is a **Gramian matrix**: matrix of inner products
- ▶  $k(\mathbf{x}_i, \mathbf{x}_j)$  measures the similarity between patterns
- ▶  $n \times n$  matrix: storage and computational complexities when  $n \uparrow \uparrow$

# The Gaussian kernel



- The Gaussian kernel is an inner product in an infinite-dimensional feature space



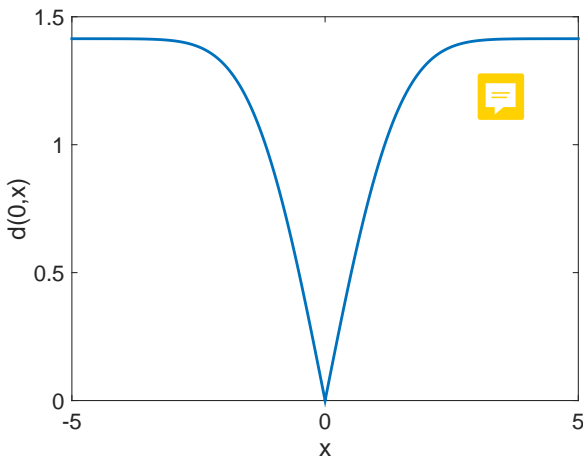
$$k(\mathbf{x}, \mathbf{y}) = \Phi(\mathbf{x})^T \Phi(\mathbf{y}) = e^{-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma^2}}$$



- The distance between  $\Phi(\mathbf{x})$  and  $\Phi(\mathbf{y})$  is

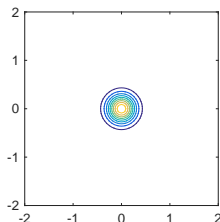
$$\begin{aligned} d(\Phi(\mathbf{x}), \Phi(\mathbf{y})) &= \|\Phi(\mathbf{x}) - \Phi(\mathbf{y})\| = \sqrt{2 \left( 1 - e^{-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma^2}} \right)} \\ &= \sqrt{2(1 - k(\mathbf{x}, \mathbf{y}))} \end{aligned}$$

# Example 1D, $\sigma^2 = 1$

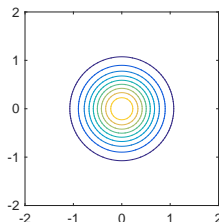


# Example 2D

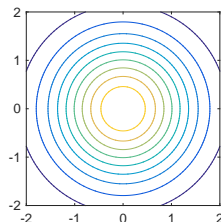
$$\sigma^2 = 0,2$$



$$\sigma^2 = 0,5$$



$$\sigma^2 = 1$$



- ▶  $\sigma^2 \downarrow$  local distance: all points beyond a given radius are at the same distance (equally far apart)
- ▶  $\sigma^2 \uparrow$  global distance: like a linear kernel

# Hyperparameters

- ▶ SVM with Gaussian kernel

$$\min_{\alpha} \quad \frac{1}{2} \alpha^T \mathbf{Y} \mathbf{K} \mathbf{Y} \alpha - \mathbf{1}^T \alpha$$

$$\text{s.t.} \quad \alpha^T \mathbf{y} = 0,$$

$$0 \leq \alpha \leq \mathbf{C}$$

$$k(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma \|\mathbf{x} - \mathbf{y}\|^2}$$

$$\text{where } \gamma = \frac{1}{2\sigma^2}$$

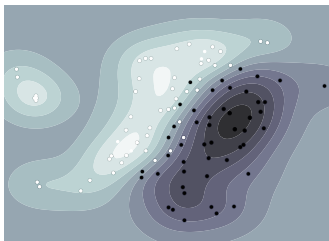
- ▶ Choosing suitable hyperparameters  $\gamma$  and  $\mathbf{C}$  is essential to get good performance
- ▶ Typically, there are selected by **cross-validation**

# Regularization parameter: $C$

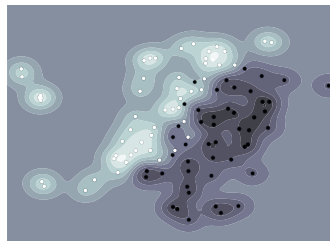
- ▶  $C$  establishes a compromise between training error and model complexity
- ▶  $C \downarrow$  simple model, large training error, smooth decision boundary
- ▶  $C \uparrow$  complex model, small training error, non-smooth decision boundary, **overfitting** risk

# Example

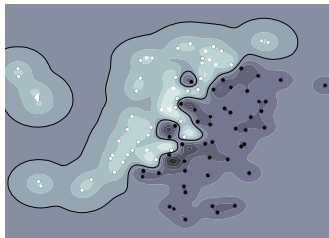
$C = 0.001$



$C = 0.01$



$C = 100$





# Kernel size: $\gamma$

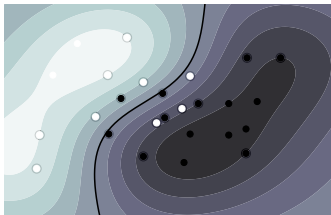
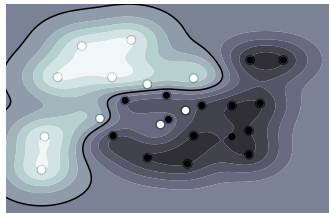
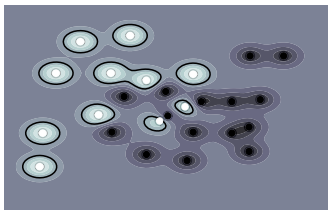


- ▶ The kernel size  $\gamma$  (a.k.a. bandwidth) controls how fast  $k(\mathbf{x}, \mathbf{y}) \rightarrow 0$  as a function of the pairwise distance
- ▶ Recall that the SVM decision function for a new pattern  $\mathbf{x}$  is

$$f(\mathbf{x}) = \sum_i \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) + b \stackrel{C_1}{\underset{C_0}{\geq}} 0$$

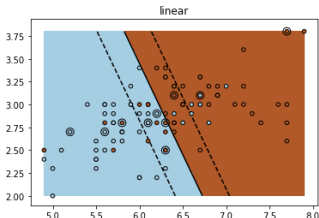
- ▶  $\gamma \downarrow$  large overlap among Gaussians, smooth decision boundary
- ▶  $\gamma \uparrow$  all patterns tend to be orthogonal to each other  
**overfitting**

# Example

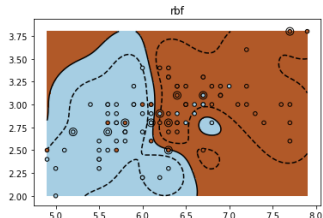
 $\gamma = 0.001$  $\gamma = 0.01$  $\gamma = 100$ 

# Kernel comparison

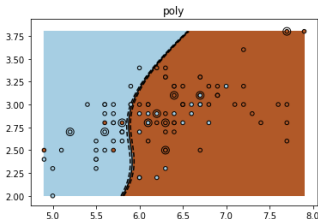
Linear  $C = 1$



Gaussian  $C = 1, \gamma = 10$



Polynomial  $C = 1, \text{order} = 10$



# SVM solvers

- ▶ QP problem → **Interior Point Methods**
  1. Memory requirements for  $\mathbf{K}$ :  $\mathcal{O}(n^2)$
  2. Slow convergence, computational complexity  $\mathcal{O}(n^3)$
- ▶ More efficient (and scalable) algorithms exist
- ▶ **Sequential Minimal Optimization (SMO)**: it solves a sequence of smaller subproblems
- ▶ **LIBSVM**:
  - ▶ Standard SVM package
  - ▶ It applies a version of SMO
  - ▶ Interfaces in R, Matlab, Python,...

# Multi-class SVM

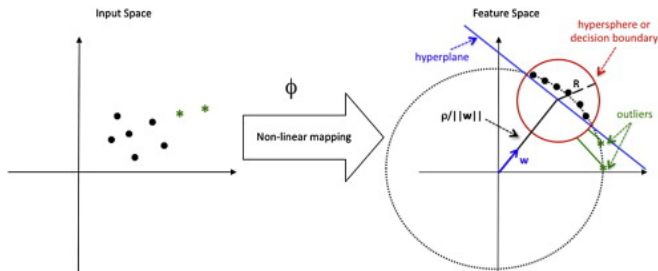
- ▶ Standard methodology: **One-Versus-All**
- ▶ For a problem with  $K$  classes we solve  $K$  independent binary problems
- ▶ Each SVM is trained to separate one class from the others
- ▶ With a new test pattern,  $\mathbf{x}$ , the  $k$ -th SVM outputs a score

$$f^k(\mathbf{x}) = \sum_i \alpha_i^k y_i^k k(\mathbf{x}_i^k, \mathbf{x}) + b^k, \quad k = 1, \dots, K$$

- ▶ The class finally assigned to  $\mathbf{x}$  is the one providing a highest score

$$k^* = \operatorname{argmax}_k f^k(\mathbf{x})$$

# One-class SVM



- ▶ Goal: to find an SVM that encloses most of the data
- ▶ Outlier/Novelty detection
- ▶ We can separate normal data from outliers in the feature space through
  - ▶ A hyperplane (see figure)
  - ▶ A hypersphere

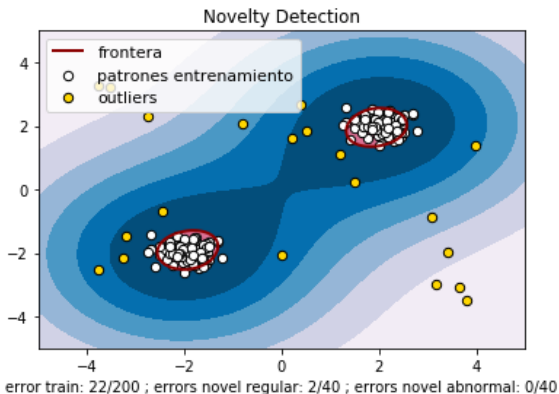
## One-class SVM

$$\begin{aligned} \min_{\mathbf{w}, \xi_i, \rho} \quad & \frac{1}{2} \|\mathbf{w}\|^2 + \frac{1}{\nu n} \sum_{i=1}^n \xi_i - \rho \\ \text{s.t.} \quad & \mathbf{w}^T \Phi(\mathbf{x}_i) \geq \rho - \xi_i, \quad \forall i \\ & \xi_i \geq 0 \quad \forall i \end{aligned}$$

- ▶ Dual problem equivalent to that of a conventional SVM
- ▶ The parameter  $\nu$  characterizes the solution  $\rightarrow$   **$\nu$ -SVM**
  - ▶ An upper bound on the fraction of outliers
  - ▶ A lower bound on the fraction of support vectors

# Example

- $\nu$ -SVM, Gaussian kernel,  $\gamma = 0,1$ ,  $\nu = 0,1$





# Conclusions



- ▶ SVM: one of the most popular learning machines
- ▶ Derived from the Structural Risk Minimization principle
- ▶ QP problem: unique minimum, well-defined problem
- ▶ A kernel (measuring the similarity between patterns)+ regularization parameter + hyperparameters
- ▶ Sparse solution: it admits an expansion in terms of a few patterns (support vectors)
- ▶ Still competitive results in a number of applications