

# Kernel Methods in Machine Learning

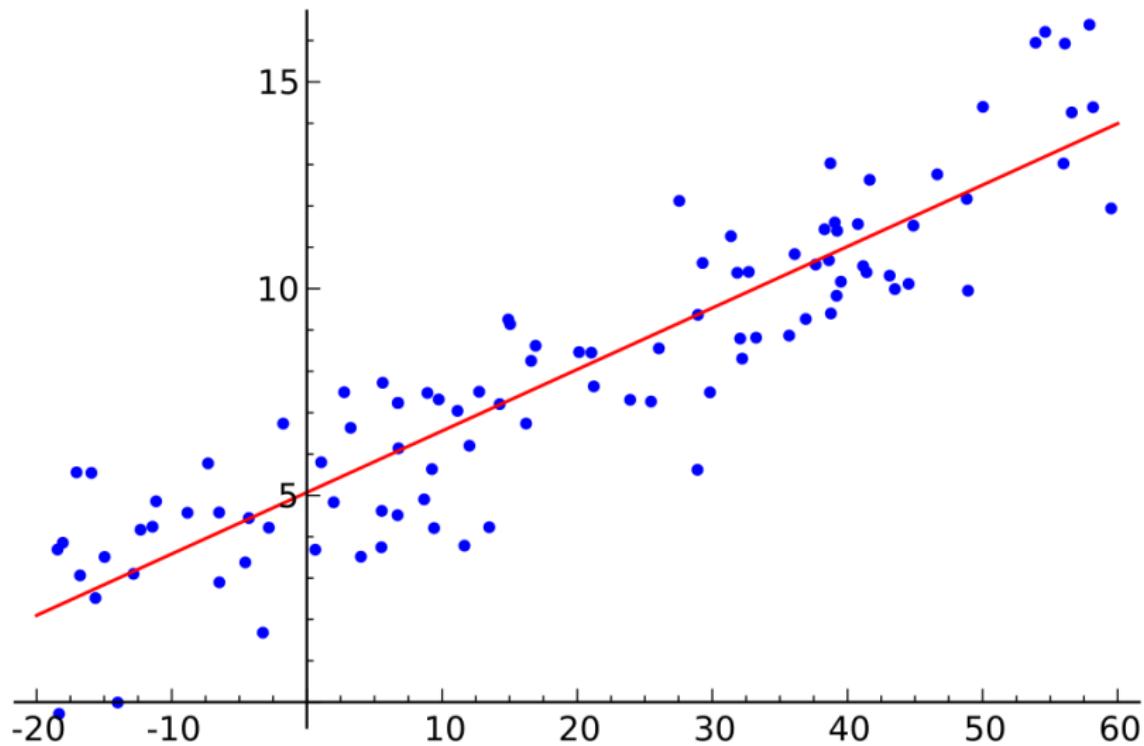
Julien Mairal & Jean-Philippe Vert

Romain Ménégaux (T.A.)

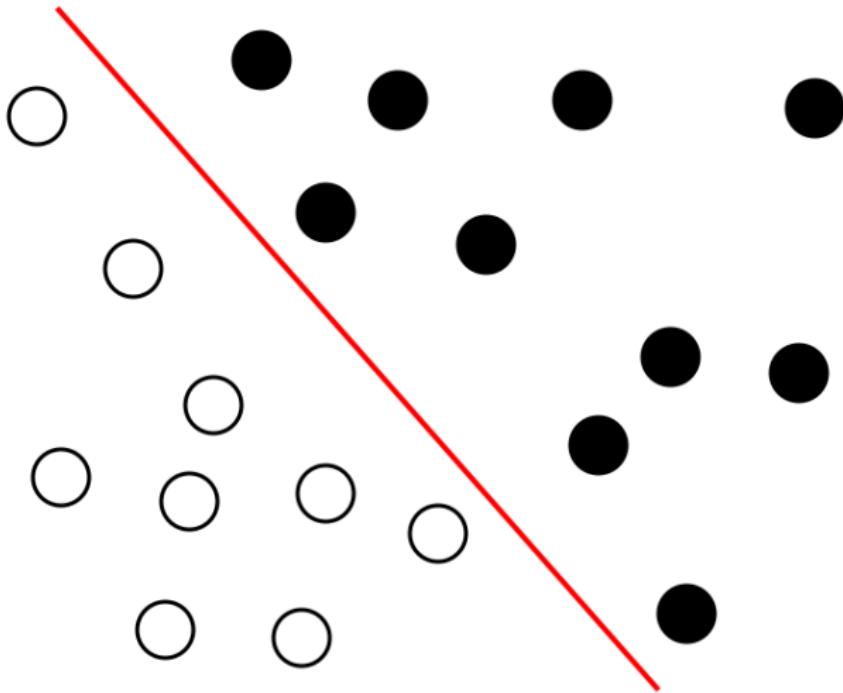
`firstname.lastname@m4x.org`



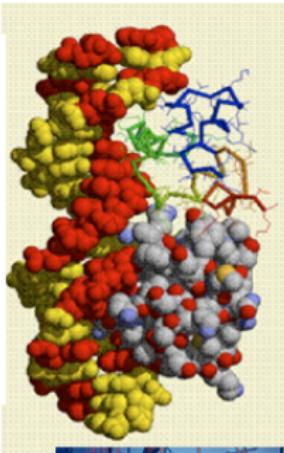
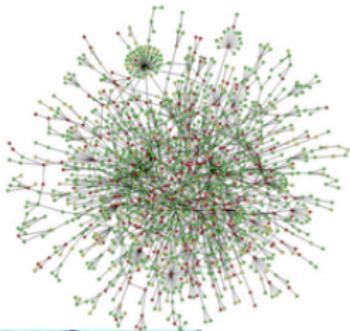
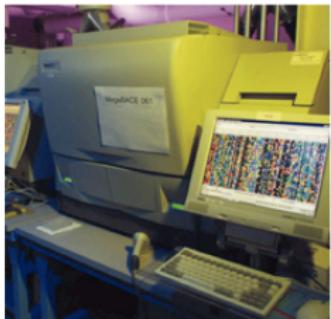
## What we know how to solve



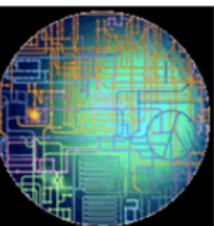
Or



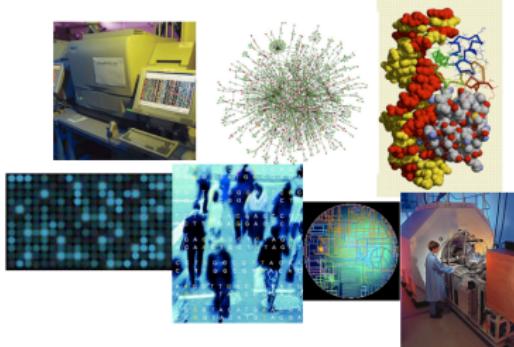
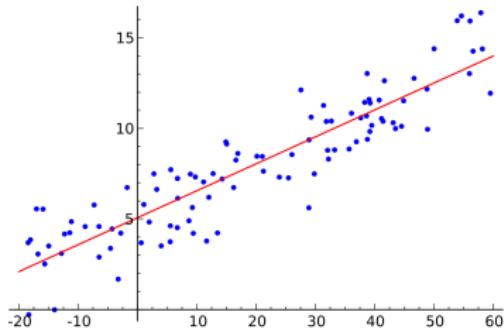
But real data are often more complicated...



A G T T C G G C G A G A  
A G G T T T A T G C C G A A  
C G T T T C G G G A T T C C C  
T C T T T C G G G A T T C C C  
T C T T T C G G G A T T C C C  
G A G C T G T G T C T A G A T  
G A A C T G G T T A T A G E T  
C C T T A G G C G T T A C A A  
C A T T T G G C G T T A C A C  
A A G G T T T G G C S R A C G  
A G G P T T A G C C C G A A C G  
C C A G T A C A T G A A C G A  
C C G G T A C A T G T A C G A



# Main goal of this course



Extend  
well-understood, linear statistical learning techniques  
to  
real-world, complicated, structured, high-dimensional data  
based on  
a rigorous mathematical framework  
leading to  
practical modelling tools and algorithms

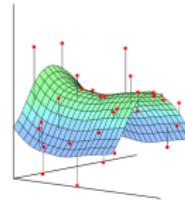
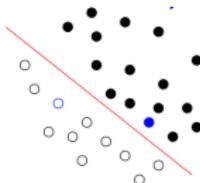
# Outline

- 1 Learning in high dimension
- 2 Learning with  $\ell_2$  regularization
  - Ridge regression
  - Ridge logistic regression
  - Linear hard-margin SVM
  - Interlude: quick notes on constrained optimization
  - Back to hard-margin SVM
  - Soft-margin SVM
  - Large-margin classifiers
- 3 Learning with kernels
  - Kernel methods
  - Positive definite kernels and RKHS
  - Kernel examples
  - Multiple Kernel Learning (MKL)
- 4 Conclusion

# Outline

- 1 Learning in high dimension
- 2 Learning with  $\ell_2$  regularization
  - Ridge regression
  - Ridge logistic regression
  - Linear hard-margin SVM
  - Interlude: quick notes on constrained optimization
  - Back to hard-margin SVM
  - Soft-margin SVM
  - Large-margin classifiers
- 3 Learning with kernels
  - Kernel methods
  - Positive definite kernels and RKHS
  - Kernel examples
  - Multiple Kernel Learning (MKL)
- 4 Conclusion

# General learning framework



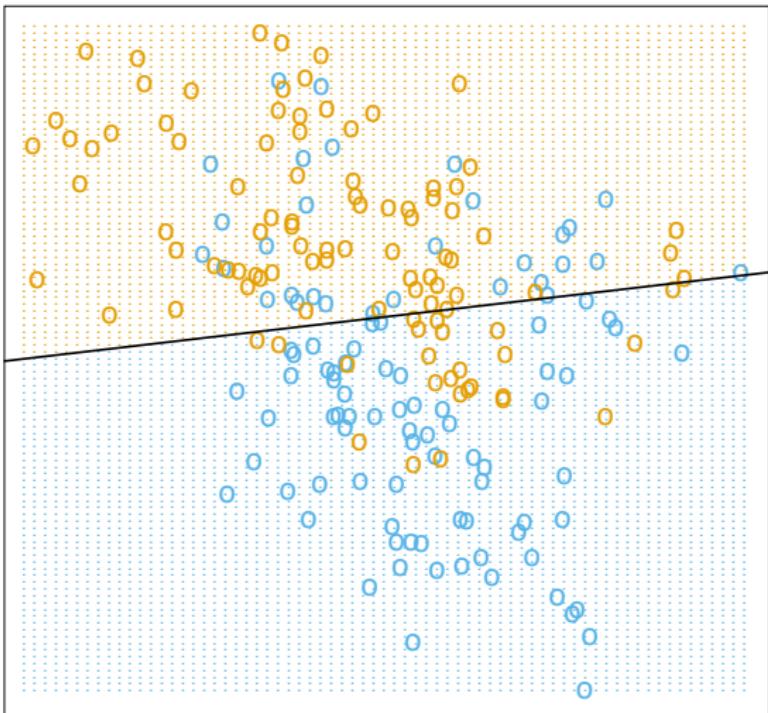
## Input

- $\mathcal{X}$  the space of **patterns** or **data** (typically,  $\mathcal{X} = \mathbb{R}^p$ )
- $\mathcal{Y}$  the space of **response** or **labels**
  - Classification or pattern recognition :  $\mathcal{Y} = \{-1, 1\}$
  - Regression :  $\mathcal{Y} = \mathbb{R}$
  - Structured output:  $\mathcal{Y}$  general
- $\mathcal{S} = \{(x_1, y_1), \dots, (x_n, y_n)\}$  a **training set** in  $(\mathcal{X} \times \mathcal{Y})^n$

## Output

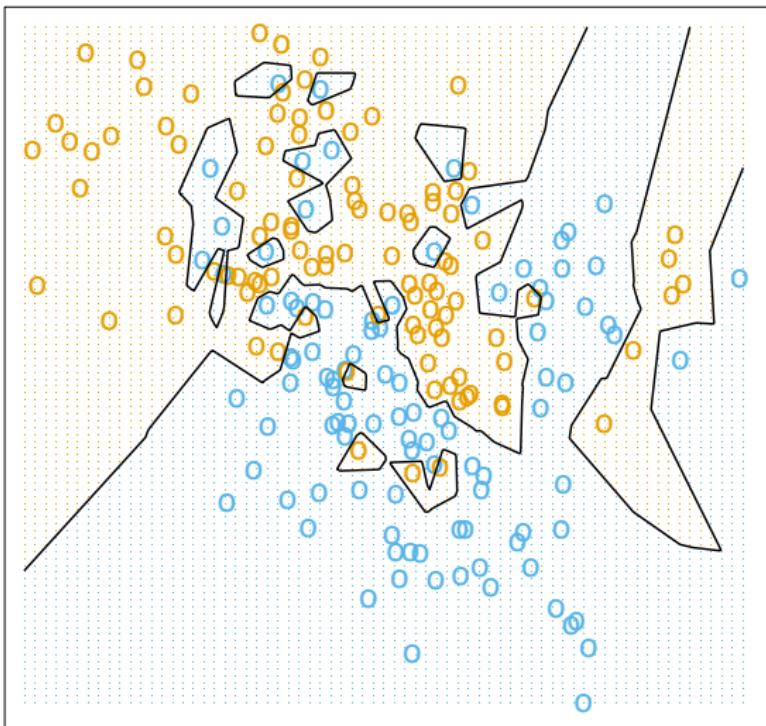
- A **function**  $f : \mathcal{X} \rightarrow \mathcal{Y}$  to predict the output associated to any new pattern  $x \in \mathcal{X}$  by  $f(x)$

# Simple example 1 : ordinary least squares (OLS)



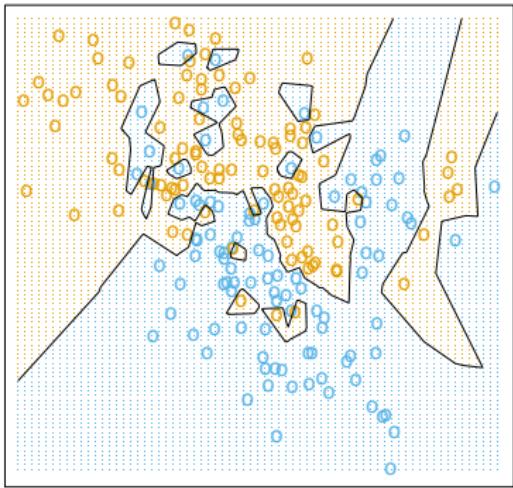
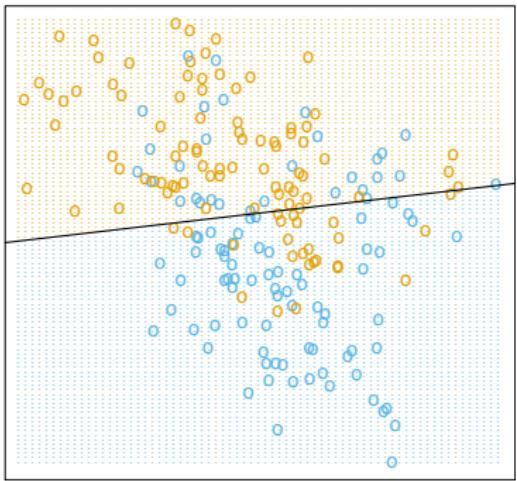
(Hastie et al. *The elements of statistical learning*. Springer, 2001.)

## Simple example 2 : 1-nearest neighbor (1-NN)



(Hastie et al. *The elements of statistical learning*. Springer, 2001.)

# What's wrong?



- OLS: the linear separation is not appropriate = "large bias"
- 1-NN: the classifier seems too unstable = "large variance"

## The fundamental "bias-variance" trade-off

- Assume  $Y = f(X) + \epsilon$ , where  $\epsilon$  is some noise
- From the training set  $\mathcal{S}$  we estimate the predictor  $\hat{f}$
- On a new point  $x_0$ , we predict  $\hat{f}(x_0)$  but the "true" observation will be  $Y_0 = f(x_0) + \epsilon$
- On average, we make an error of:

$$\begin{aligned} & E_{\epsilon, \mathcal{S}} \left( Y_0 - \hat{f}(x_0) \right)^2 \\ &= E_{\epsilon, \mathcal{S}} \left( f(x_0) + \epsilon - \hat{f}(x_0) \right)^2 \\ &= E\epsilon^2 + E_{\mathcal{S}} \left( f(x_0) - \hat{f}(x_0) \right)^2 \\ &= E\epsilon^2 + \left( f(x_0) - E_{\mathcal{S}} \hat{f}(x_0) \right)^2 + E_{\mathcal{S}} \left( \hat{f}(x_0) - E_{\mathcal{S}} \hat{f}(x_0) \right)^2 \\ &= \text{noise} + \text{bias}^2 + \text{variance} \end{aligned}$$

## Important message

Future prediction error = noise + bias<sup>2</sup> + variance

- The "noise" part can not be avoided
- By choosing a learning model, we should consider both "bias" and "variance" if we want to make good predictions
- Intuitively, a **more realistic, more complex model** with more parameters to estimate has **smaller bias but larger variance**
- If variance dominates bias (eg, in high dimension), then having more complex, more realistic models can hurt performance
- In other words, **a wrong but simple model can work better than a more realistic but more complex model**
- In many applications, domain experts (non-statisticians) often ignore the cost of complexity and prefer complex models, which can lead to disappointing results. You can help them!

## Back to OLS

- Linear model with parameter  $\beta \in \mathbb{R}^p$ :

$$\forall x \in \mathbb{R}^p, \quad f_\beta(x) = \beta^\top x \quad \left(= \sum_{i=1}^p \beta_i x_i\right)$$

- Estimate  $\hat{\beta}^{OLS}$  from training data to minimize the mean sum of squares (MSE):

$$\text{MSE}(\beta) = \frac{1}{n} \sum_{i=1}^n (y_i - f_\beta(x_i))^2$$

## Back to OLS (cont.)

- Let's use matrix notations:
  - $Y = (y_1, \dots, y_n)^\top \in \mathbb{R}^n$  the vector of outcomes
  - $X = (x_1, \dots, x_n)^\top \in \mathbb{R}^{n \times p}$  the matrix ( $n$  rows=samples,  $p$  columns=features)
- We can rewrite MSE as

$$\text{MSE}(\beta) = \frac{1}{n} (Y - X\beta)^\top (Y - X\beta)$$

- $\text{MSE}(\beta)$  is a quadratic convex function; we minimize it by setting its gradient to 0:

$$\nabla_\beta \text{MSE}(\beta) = \frac{2}{n} X^\top (X\beta - Y) = 0$$

- If  $X^\top X$  is non-singular, the minimum is reached at

$$\hat{\beta}^{OLS} = \underset{\beta}{\operatorname{argmin}} \text{MSE}(\beta) = (X^\top X)^{-1} X^\top Y$$

# Properties of OLS

## Bias and variance of OLS

- Assume  $Y = X\beta^* + \epsilon$ , where  $E\epsilon = 0$  and  $E\epsilon\epsilon^\top = \sigma^2 I$ .
- Then the least squares estimator

$$\hat{\beta}^{OLS} = (X^\top X)^{-1} X^\top Y$$

satisfies

$$\begin{cases} E(\hat{\beta}^{OLS}) = \beta^*, \\ Var(\hat{\beta}^{OLS}) = E(\hat{\beta}^{OLS} - \beta^*)(\hat{\beta}^{OLS} - \beta^*)^\top = \sigma^2 (X^\top X)^{-1}. \end{cases}$$

Proof: exercice

# Optimality of OLS

## Gauss-Markov theorem

- Assume  $Y = X\beta^* + \epsilon$ , where  $E\epsilon = 0$  and  $E\epsilon\epsilon^\top = \sigma^2 I$ .
- Then the least squares estimator  $\hat{\beta}^{OLS}$  is **BLUE** (best linear unbiased estimator), i.e., for any other estimator  $\tilde{\beta} = CY$  with  $E\tilde{\beta} = \beta^*$ ,

$$\text{Var}(\hat{\beta}^{OLS}) \preccurlyeq \text{Var}(\tilde{\beta}) \quad \left( \Leftrightarrow \forall x_0, \text{Var}(x_0^\top \hat{\beta}^{OLS}) \leq \text{Var}(x_0^\top \tilde{\beta}) \right)$$

Proof: exercice

- For  $\tilde{\beta} = CY$ , define  $D = C - (X^\top X)^{-1}X^\top$
- Show that if  $\tilde{\beta}$  is unbiased, then  $DX = 0$
- Show that  $\text{Var}(\tilde{\beta}) = \text{Var}(\hat{\beta}^{OLS}) + \sigma^2 DD^\top$

# Is BLUE good enough?

## OLS mean prediction variance

Suppose  $(x_i, y_i)_{i=0, \dots, n}$  are i.i.d. random pairs that follow a distribution  $(x, y)$  with

$$Ex = 0, \quad Exx^\top = C,$$

$$y = x^\top \beta^* + \epsilon, \quad E\epsilon = 0, \quad E\epsilon^2 = 1,$$

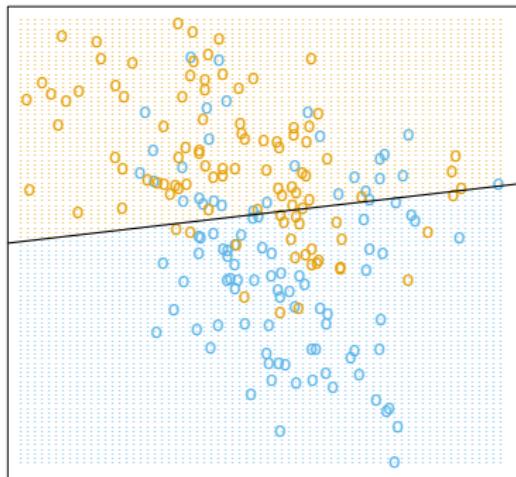
then

$$E_{S, x_0} \text{Var}(f(x_0)) = E_{S, x_0} \text{Var}(x_0^\top \hat{\beta}^{OLS}) = \frac{\sigma^2 p}{n}.$$

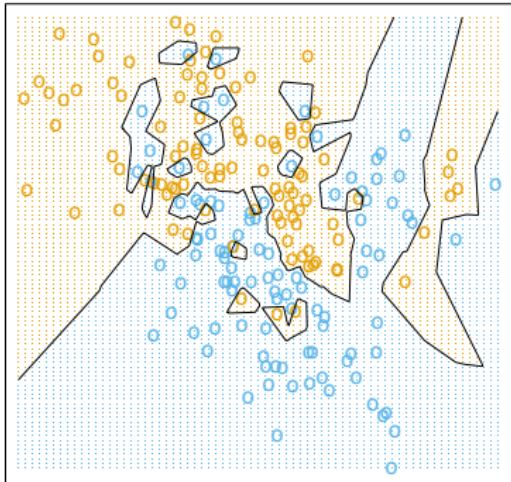
Proof:

$$\begin{aligned} E_{S, x_0} \text{Var}(x_0^\top \hat{\beta}^{OLS}) &= \sigma^2 E_{S, x_0} [x_0^\top (X^\top X)^{-1} x_0] \\ &= \sigma^2 E_{S, x_0} \text{Trace} [(X^\top X)^{-1} x_0 x_0^\top] \\ &= \sigma^2 \text{Trace} [E_S (X^\top X)^{-1} \times E_{x_0} x_0 x_0^\top] \\ &= \sigma^2 \text{Trace} [(nC)^{-1} \times C] \\ &= \sigma^2 p/n \end{aligned}$$

# The curse of dimensionality



Small dimension



Large dimension

- In high dimensions, **variance dominates**, even for simple linear estimators.
- **BLUE estimators are therefore useless.**
- In that case, we may have smaller total risk by **increasing bias and decreasing variance**

## A solution: shrinkage estimators

- ① Define a large family of "candidate classifiers", e.g., **linear predictors**:

$$f_{\beta}(x) = \beta^T x \quad \text{for } x \in \mathbb{R}^p$$

## A solution: shrinkage estimators

- ① Define a large family of "candidate classifiers", e.g., **linear predictors**:

$$f_{\beta}(x) = \beta^T x \quad \text{for } x \in \mathbb{R}^p$$

- ② For any candidate classifier  $f_{\beta}$ , quantify how "good" it is on the training set with some **empirical risk**, e.g.:

$$R(\beta) = \frac{1}{n} \sum_{i=1}^n (f_{\beta}(x_i) - y_i)^2.$$

## A solution: shrinkage estimators

- ① Define a large family of "candidate classifiers", e.g., **linear predictors**:

$$f_{\beta}(x) = \beta^T x \quad \text{for } x \in \mathbb{R}^p$$

- ② For any candidate classifier  $f_{\beta}$ , quantify how "good" it is on the training set with some **empirical risk**, e.g.:

$$R(\beta) = \frac{1}{n} \sum_{i=1}^n (f_{\beta}(x_i) - y_i)^2.$$

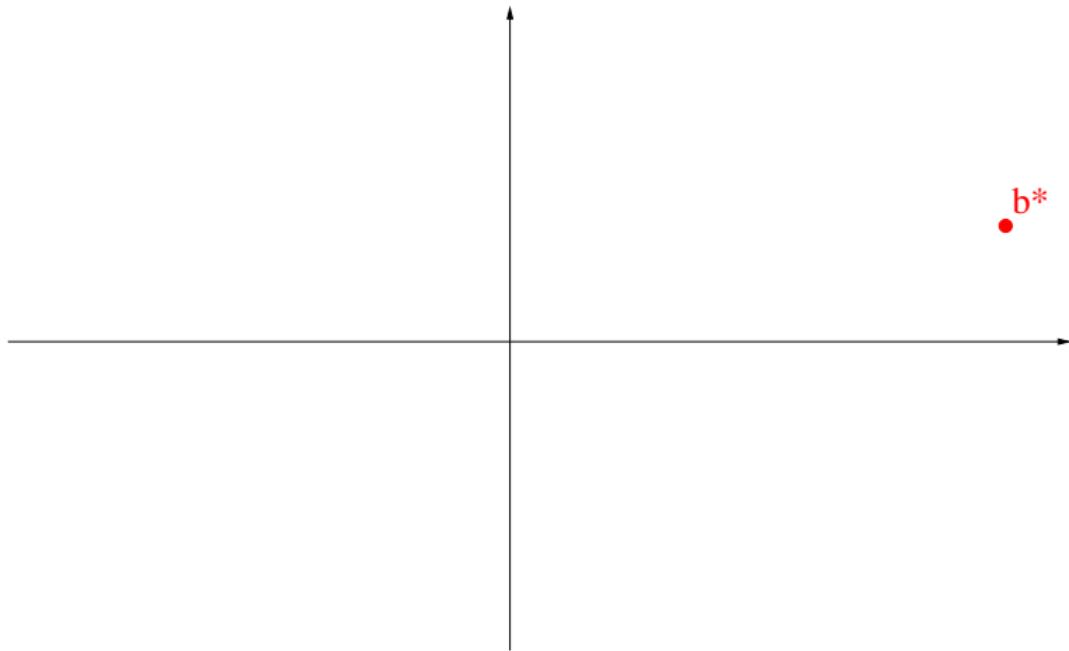
- ③ Choose  $\beta$  that achieves the minimum empirical risk, subject to some **constraint**:

$$\min_{\beta} R(\beta) \quad \text{subject to} \quad \Omega(\beta) \leq C,$$

for some **penalty function**  $\Omega : \mathbb{R}^p \rightarrow \mathbb{R}^+$  and  $C \geq 0$ .

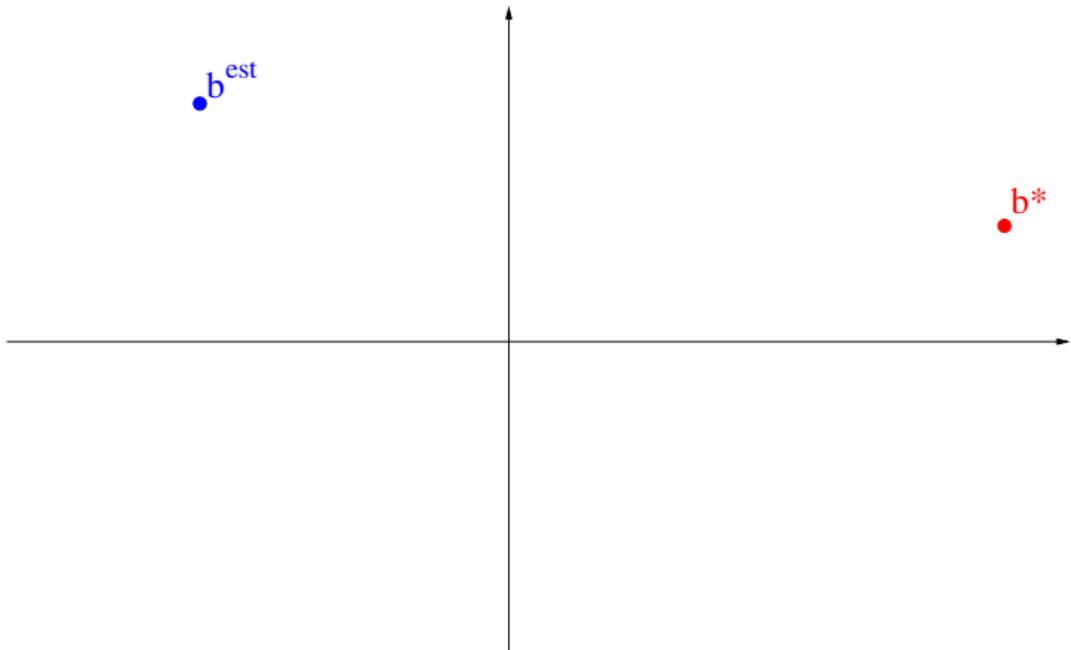
# Why skrinkage classifiers?

$$\min_{\beta} R(\beta) \quad \text{subject to} \quad \Omega(\beta) \leq C .$$



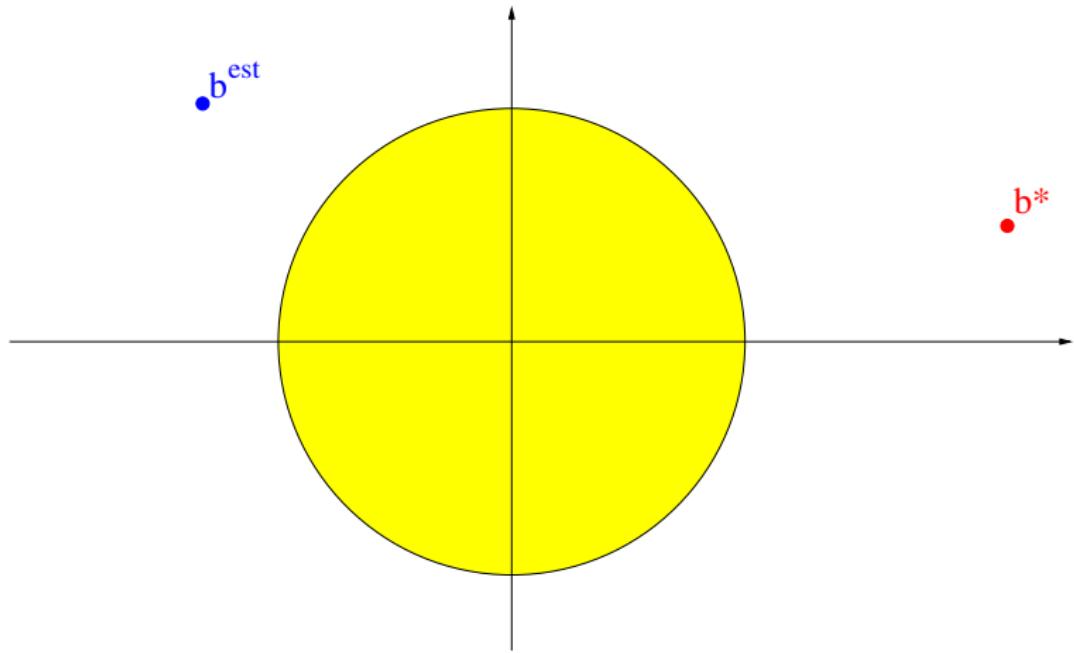
# Why skrinkage classifiers?

$$\min_{\beta} R(\beta) \quad \text{subject to} \quad \Omega(\beta) \leq C .$$



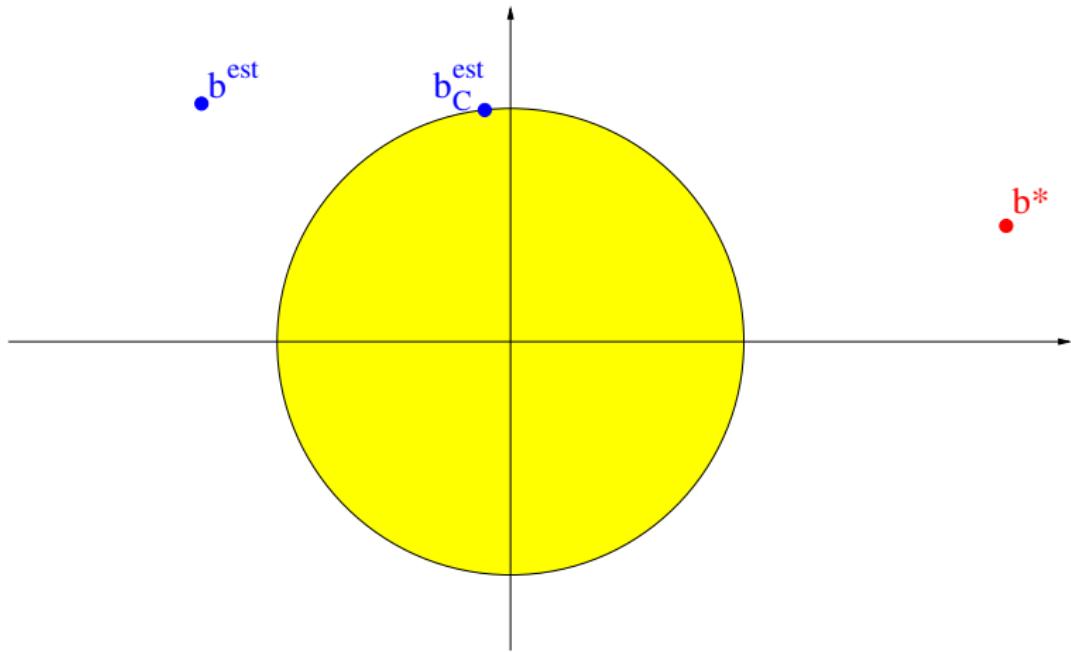
# Why shrinkage classifiers?

$$\min_{\beta} R(\beta) \quad \text{subject to} \quad \Omega(\beta) \leq C .$$



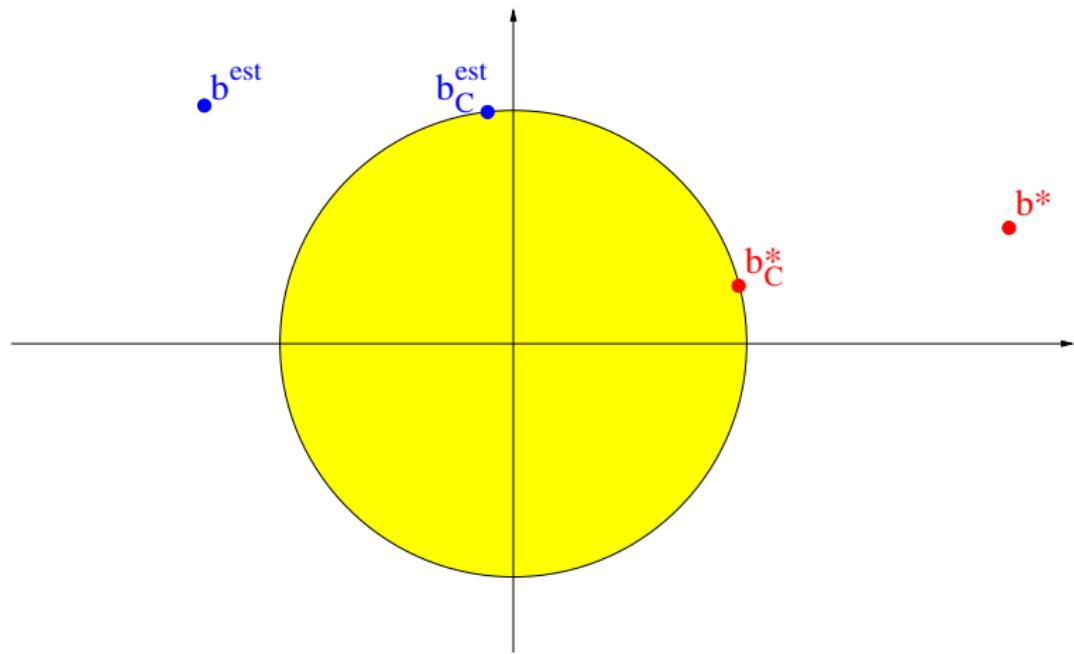
# Why skrinkage classifiers?

$$\min_{\beta} R(\beta) \quad \text{subject to} \quad \Omega(\beta) \leq C .$$



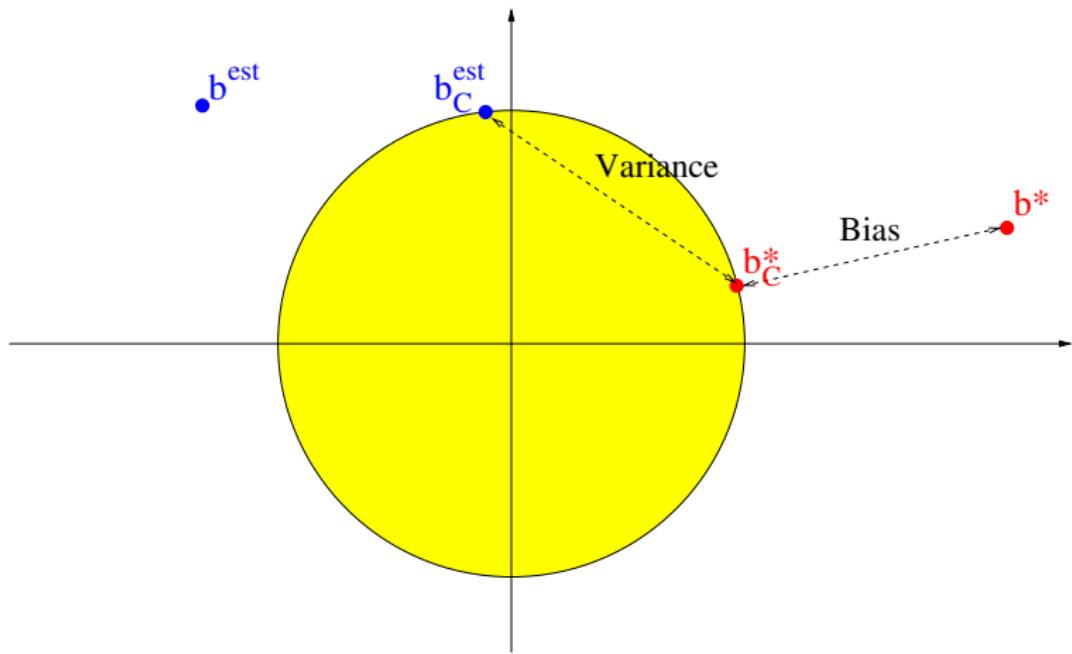
# Why skrinkage classifiers?

$$\min_{\beta} R(\beta) \quad \text{subject to} \quad \Omega(\beta) \leq C .$$



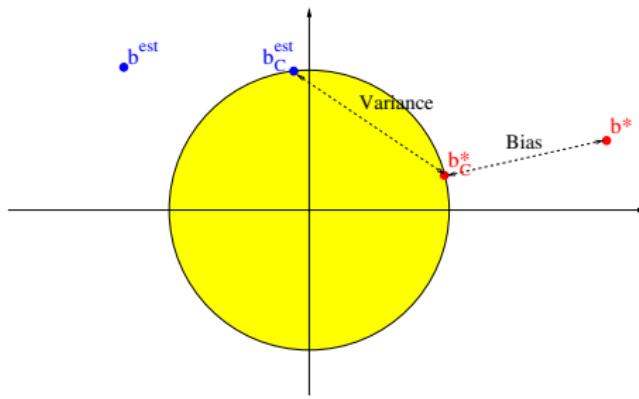
# Why skrinkage classifiers?

$$\min_{\beta} R(\beta) \quad \text{subject to} \quad \Omega(\beta) \leq C .$$



# Why shrinkage classifiers?

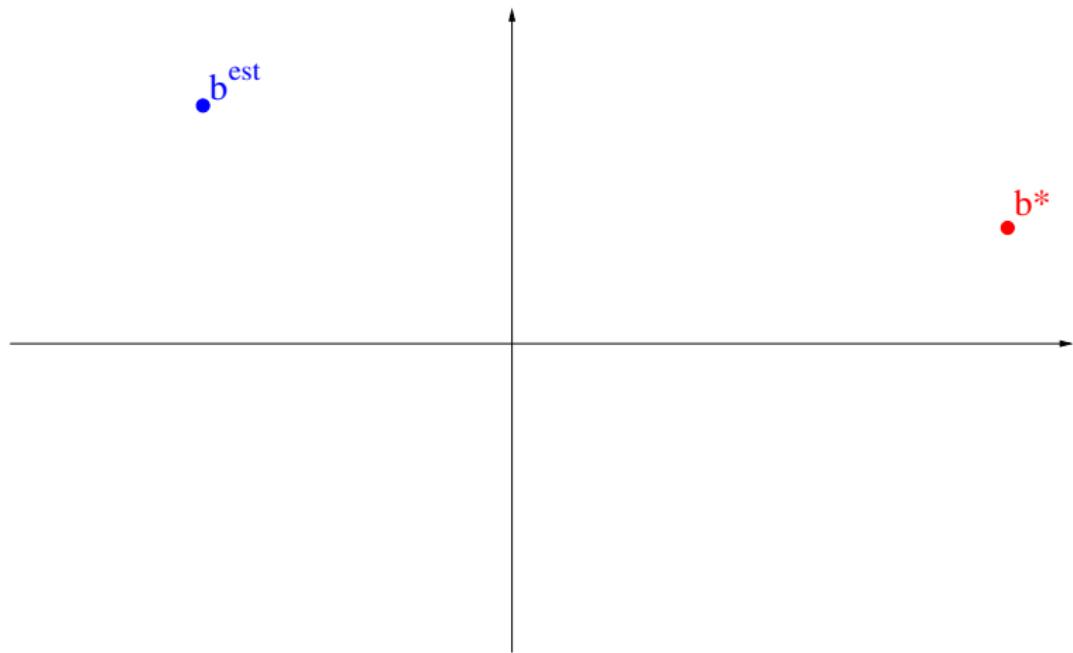
$$\min_{\beta} R(\beta) \quad \text{subject to} \quad \Omega(\beta) \leq C .$$



"Increases bias and decreases variance"

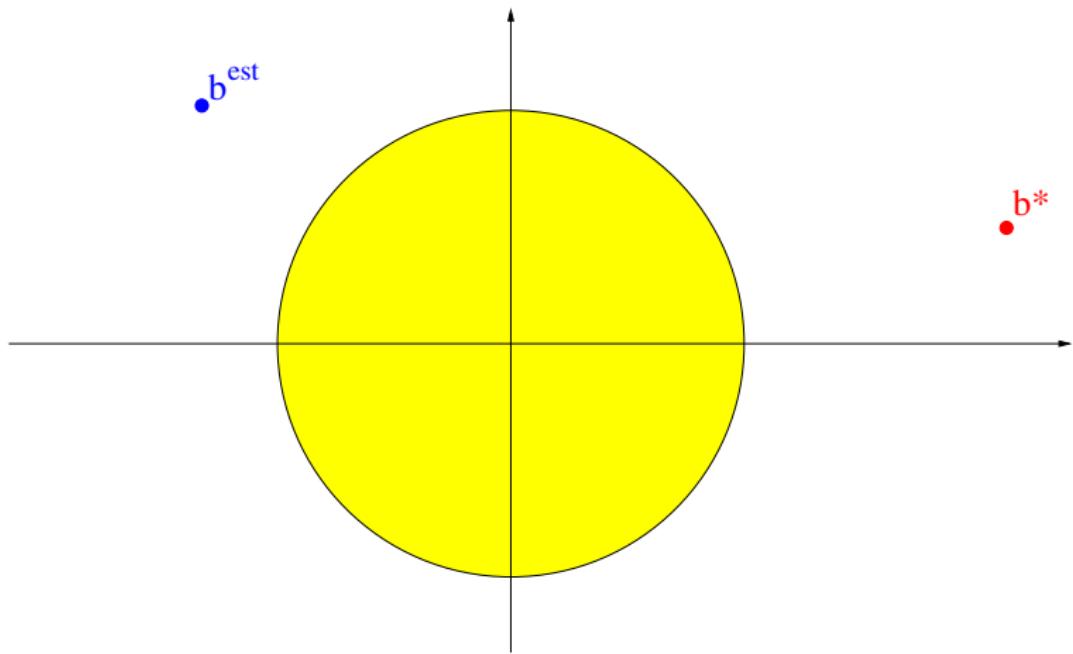
Choice of  $\Omega$  can decrease the bias

$$\min_{\beta} R(\beta) \quad \text{subject to} \quad \Omega(\beta) \leq C .$$



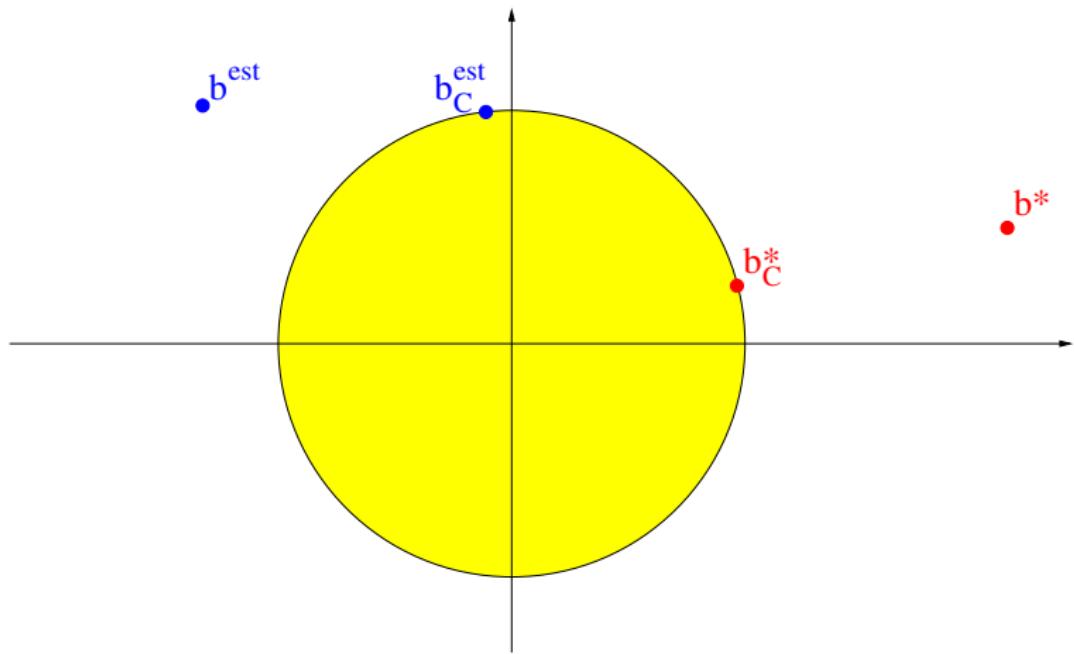
Choice of  $\Omega$  can decrease the bias

$$\min_{\beta} R(\beta) \quad \text{subject to} \quad \Omega(\beta) \leq C .$$



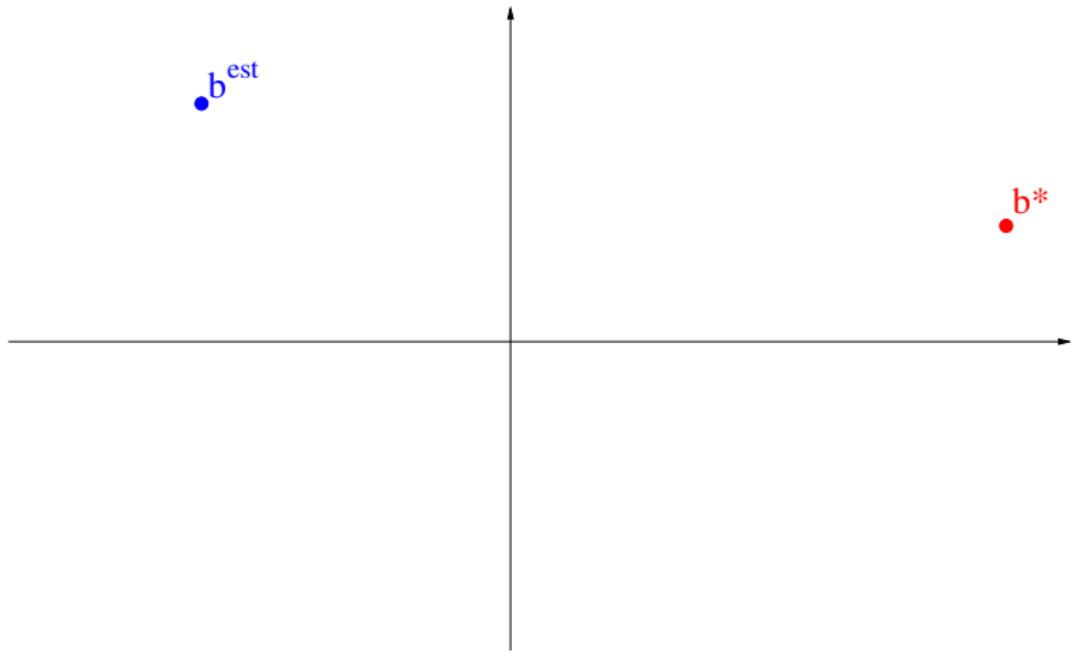
Choice of  $\Omega$  can decrease the bias

$$\min_{\beta} R(\beta) \quad \text{subject to} \quad \Omega(\beta) \leq C .$$



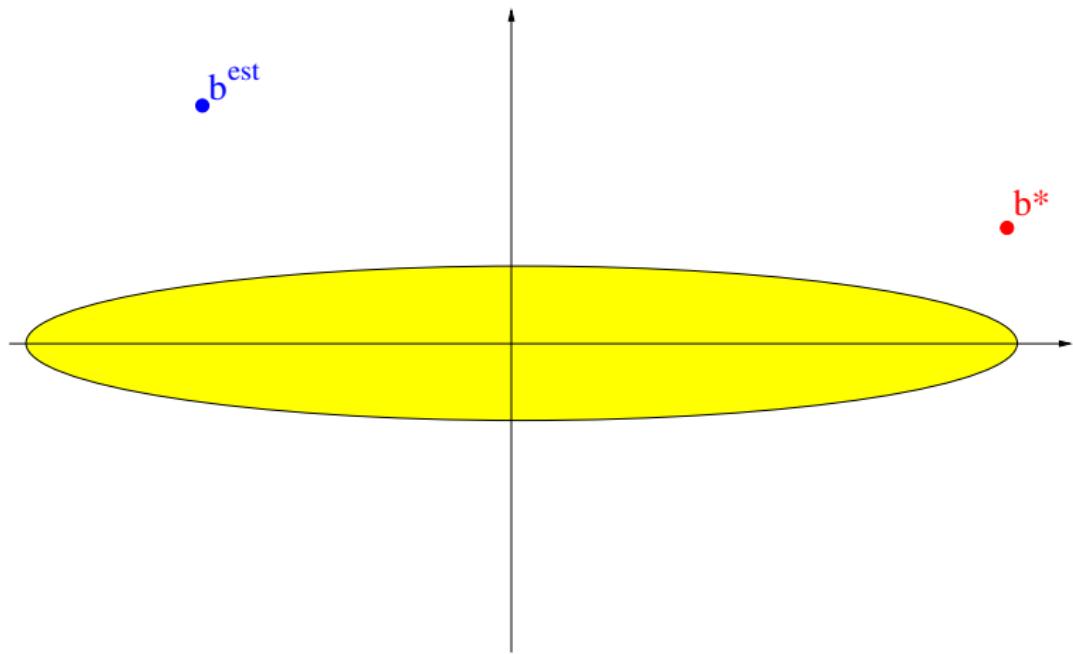
Choice of  $\Omega$  can decrease the bias

$$\min_{\beta} R(\beta) \quad \text{subject to} \quad \Omega(\beta) \leq C .$$



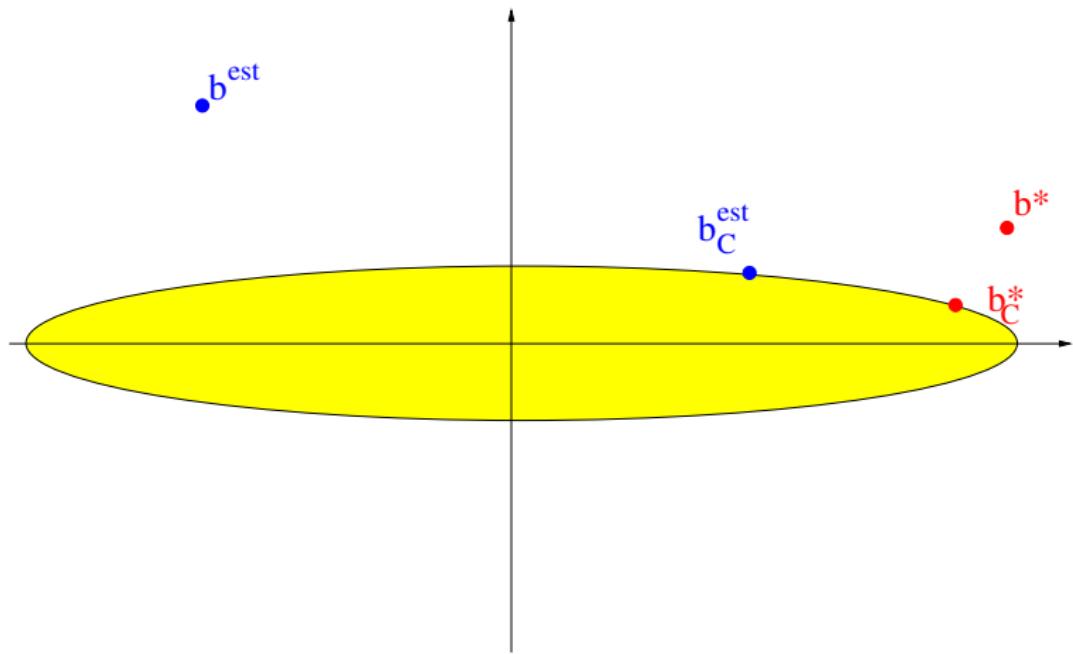
Choice of  $\Omega$  can decrease the bias

$$\min_{\beta} R(\beta) \quad \text{subject to} \quad \Omega(\beta) \leq C .$$



Choice of  $\Omega$  can decrease the bias

$$\min_{\beta} R(\beta) \quad \text{subject to} \quad \Omega(\beta) \leq C .$$



## Equivalent formulation

$$\min_{\beta} R(\beta) \quad \text{subject to} \quad \Omega(\beta) \leq C$$

is equivalent to

$$\min_{\beta} R(\beta) + \lambda \Omega(\beta)$$

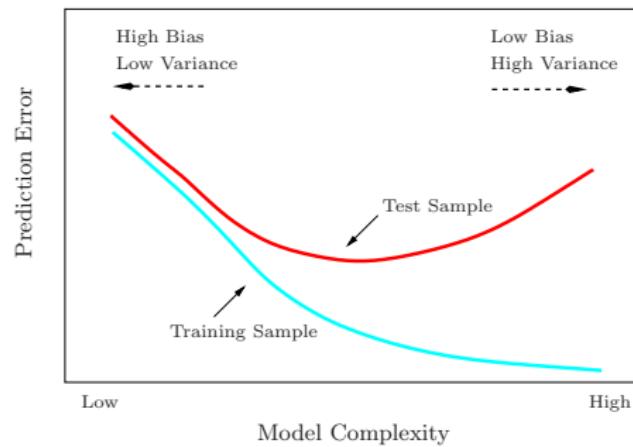
- There exists a (not necessarily unique) correspondance between  $C$  and  $\lambda$  such that the solutions to both problems are the same.
- If  $C$  increase,  $\lambda$  decreases
- The formulation with  $\lambda$  is often preferred to implement the algorithm
- Proof: using Lagrangian duality (only true under some assumptions, eg,  $R$  and  $\Omega$  convex + Slater conditions, see later)

# Choice of $C$ or $\lambda$

- Choose a grid of values  $\Lambda$  for  $\lambda$  (or  $C$ )
- For each  $\lambda \in \Lambda$  (or  $C$ ) estimate the best model

$$\hat{\beta}_\lambda \in \operatorname{argmin}_{\beta} R(\beta) + \lambda \Omega(\beta)$$

- Select  $\hat{\beta} = \hat{\beta}_{\hat{\lambda}}$  to **minimize the bias-variance tradeoff**.



## Cross-validation

A simple and systematic procedure to estimate the risk (and to optimize the model's parameters)

- ① Randomly divide the training set (of size  $n$ ) into  $K$  (almost) equal portions, each of size  $K/n$
- ② For each portion, fit the model with different parameters on the  $K - 1$  other groups and test its performance on the left-out group
- ③ Average performance over the  $K$  groups, and take the parameter with the smallest average performance.

Taking  $K = 5$  or  $10$  is recommended as a good default choice.

## Summary

- ① Many problems in modern machine learning involve models with many parameters (i.e., high dimension)
- ② The total prediction error of a learning system is the sum of a **bias** and a **variance** error
- ③ In **high dimension**, the **variance** term often dominates
- ④ **Shrinkage methods** allow us to control the bias/variance trade-off
- ⑤ The choice of the **penalty** is where we can put **prior knowledge** to decrease bias
- ⑥ The parameter to control the bias-variance trade-off ( $C$  or  $\lambda$ ) is typically chosen by **cross-validation**, to minimize the test error.

# Outline

## 1 Learning in high dimension

### 2 Learning with $\ell_2$ regularization

- Ridge regression
- Ridge logistic regression
- Linear hard-margin SVM
- Interlude: quick notes on constrained optimization
- Back to hard-margin SVM
- Soft-margin SVM
- Large-margin classifiers

### 3 Learning with kernels

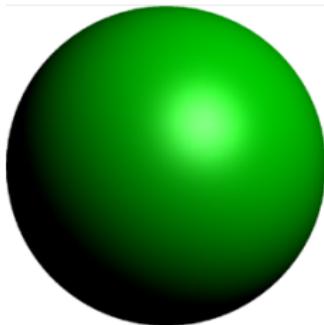
- Kernel methods
- Positive definite kernels and RKHS
- Kernel examples
- Multiple Kernel Learning (MKL)

### 4 Conclusion

# Overview

- We focus on a simple penalty function: the squared Euclidean norm

$$\Omega(\beta) = \|\beta\|^2 \quad \left( = \beta^\top \beta = \sum_{i=1}^p \beta_i^2 \right)$$



- This will allow us to derive many state-of-the-art linear methods:
  - Ridge regression
  - Ridge logistic regression
  - SVM and large-margin classifiers
- This will allow us to extend these linear methods to nonlinear models, using kernels

# Outline

- 1 Learning in high dimension
- 2 Learning with  $\ell_2$  regularization
  - Ridge regression
  - Ridge logistic regression
  - Linear hard-margin SVM
  - Interlude: quick notes on constrained optimization
  - Back to hard-margin SVM
  - Soft-margin SVM
  - Large-margin classifiers
- 3 Learning with kernels
  - Kernel methods
  - Positive definite kernels and RKHS
  - Kernel examples
  - Multiple Kernel Learning (MKL)
- 4 Conclusion

## Ridge regression (?)

- ① Consider the set of linear predictors:

$$\forall \beta \in \mathbb{R}^p, \quad f_\beta(x) = \beta^\top x \quad \text{for } x \in \mathbb{R}^p.$$

# Ridge regression (?)

- ① Consider the set of **linear predictors**:

$$\forall \beta \in \mathbb{R}^p, \quad f_\beta(x) = \beta^\top x \quad \text{for } x \in \mathbb{R}^p.$$

- ② Consider the MSE as empirical risk:

$$R(\beta) = \frac{1}{n} \sum_{i=1}^n (f_\beta(x_i) - y_i)^2.$$

# Ridge regression (?)

- ① Consider the set of **linear predictors**:

$$\forall \beta \in \mathbb{R}^p, \quad f_\beta(x) = \beta^\top x \quad \text{for } x \in \mathbb{R}^p.$$

- ② Consider the MSE as empirical risk:

$$R(\beta) = \frac{1}{n} \sum_{i=1}^n (f_\beta(x_i) - y_i)^2.$$

- ③ Consider the **squared Euclidean norm** as a penalty:

$$\Omega(\beta) = \|\beta\|^2.$$

# Solution

- The penalized risk can be written in matrix form:

$$\begin{aligned} R(\beta) + \lambda\Omega(\beta) &= \frac{1}{n} \sum_{i=1}^n (f_\beta(x_i) - y_i)^2 + \lambda \sum_{i=1}^p \beta_i^2 \\ &= \frac{1}{n} (Y - X\beta)^\top (Y - X\beta) + \lambda \beta^\top \beta. \end{aligned}$$

# Solution

- The penalized risk can be written in matrix form:

$$\begin{aligned} R(\beta) + \lambda\Omega(\beta) &= \frac{1}{n} \sum_{i=1}^n (f_\beta(x_i) - y_i)^2 + \lambda \sum_{i=1}^p \beta_i^2 \\ &= \frac{1}{n} (Y - X\beta)^\top (Y - X\beta) + \lambda \beta^\top \beta. \end{aligned}$$

- Unique minimizer (by setting the gradient to 0):

$$\hat{\beta}_\lambda^{\text{ridge}} = \arg \min_{\beta \in \mathbb{R}^p} \{R(\beta) + \lambda\Omega(\beta)\} = (X^\top X + \lambda n I)^{-1} X^\top Y.$$

# Performance of ridge regression

## Lemma

Assume that:

- $Y = X\beta^* + \epsilon$ , where  $E\epsilon = 0$  and  $E\epsilon\epsilon^\top = \sigma^2 I$ .
- $X^\top X = nI_p$  (orthogonal design)

Then:

$$\begin{cases} bias(\hat{\beta}_\lambda^{\text{ridge}}) = E(\hat{\beta}_\lambda^{\text{ridge}}) - \beta^* = -\frac{\lambda}{1+\lambda}\beta^*, \\ Var(\hat{\beta}_\lambda^{\text{ridge}}) = \frac{\sigma^2}{n(1+\lambda)^2} I_p = \frac{1}{(1+\lambda)^2} Var(\hat{\beta}^{OLS}). \end{cases}$$

Proof: exercice

# Performance of ridge regression

## Corollary

For any  $\lambda \geq 0$  let

$$f(\lambda) = E_{S,x_0} \left[ \text{bias}^2 \left( x_0^\top \hat{\beta}_\lambda^{\text{ridge}} \right) + \text{Var} \left( x_0^\top \hat{\beta}_\lambda^{\text{ridge}} \right) \right]$$

where  $E x_0 = 0$ ,  $E x_0 x_0^\top = I_p$ . Then  $f(\lambda)$  is minimum for

$$\lambda^* = \frac{\sigma^2 p}{n \|\beta^*\|^2}$$

and

$$f(\lambda^*) = \frac{f(0)f(+\infty)}{f(0) + f(+\infty)} \leq \min \{f(0), f(\infty)\}$$

where

$$f(0) = \sigma^2 p/n, \quad f(\infty) = \|\beta^*\|^2$$

Proof: exercice

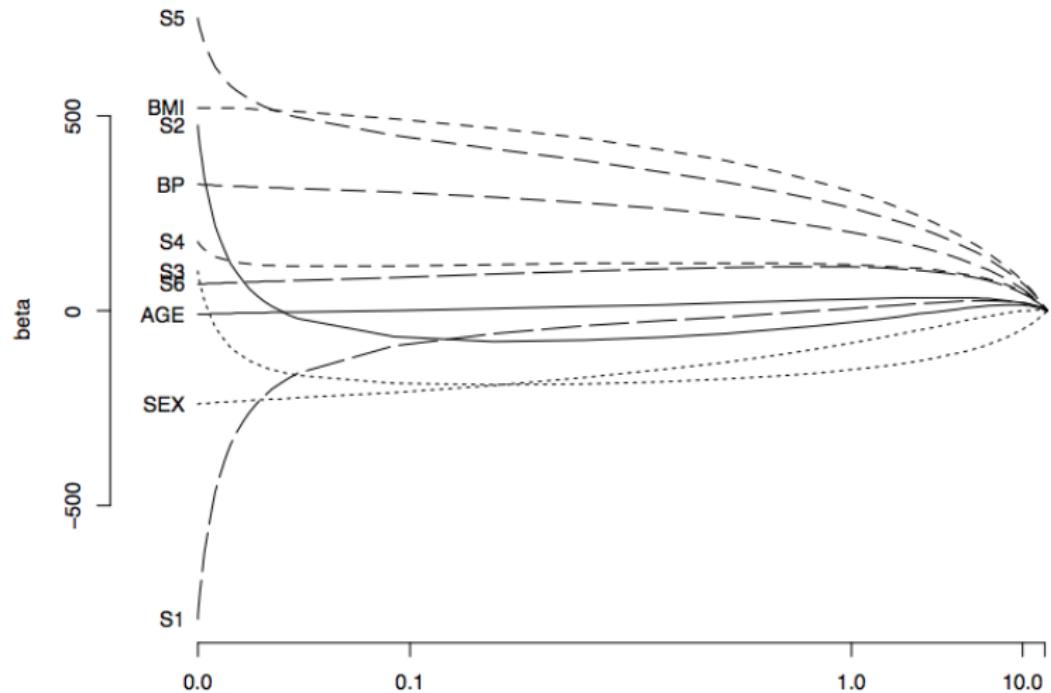
## Limit cases

$$\hat{\beta}_\lambda^{\text{ridge}} = (X^\top X + \lambda n I)^{-1} X^\top Y$$

### Corollary

- As  $\lambda \rightarrow 0$ ,  $\hat{\beta}_\lambda^{\text{ridge}} \rightarrow \hat{\beta}^{\text{OLS}}$  (low bias, high variance).
- As  $\lambda \rightarrow +\infty$ ,  $\hat{\beta}_\lambda^{\text{ridge}} \rightarrow 0$  (high bias, low variance).

# Ridge regression example



(From ?)

## Ridge regression with correlated features

Ridge regression is particularly useful in the presence of correlated features:

```
> library(MASS) # for the lm.ridge command
> x1 <- rnorm(20)
> x2 <- rnorm(20,mean=x1,sd=.01)
> y <- rnorm(20,mean=3+x1+x2)
> lm(y~x1+x2)$coef
(Intercept)          x1          x2
 3.070699    25.797872   -23.748019
> lm.ridge(y~x1+x2,lambda=1)
            x1          x2
3.066027 1.015862 0.956560
```

## Generalization: $\ell_2$ -regularized learning

- A general  $\ell_2$ -penalized estimator is of the form

$$\min_{\beta} \{ R(\beta) + \lambda \|\beta\|^2 \}, \quad (1)$$

where

$$R(\beta) = \frac{1}{n} \sum_{i=1}^n \ell(f_{\beta}(x_i), y_i)$$

for some general loss functions  $\ell$ .

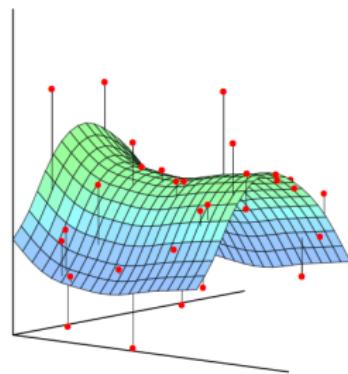
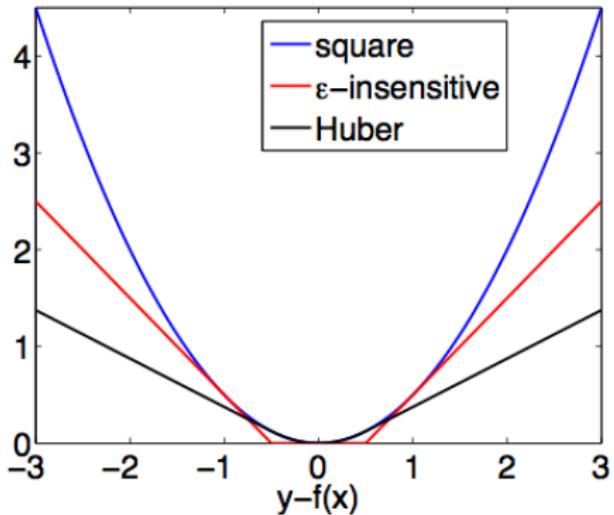
- Ridge regression corresponds to the particular loss

$$\ell(u, y) = (u - y)^2.$$

- For general, **convex** losses, the problem (??) is strictly convex and has a **unique global minimum**, which can usually be found by **numerical algorithms** for convex optimization.

# Losses for regression

- Square loss :  $\ell(u, y) = (u - y)^2$
- $\epsilon$ -insensitive loss :  $\ell(u, y) = (|u - y| - \epsilon)_+$
- Huber loss : mixed quadratic/linear



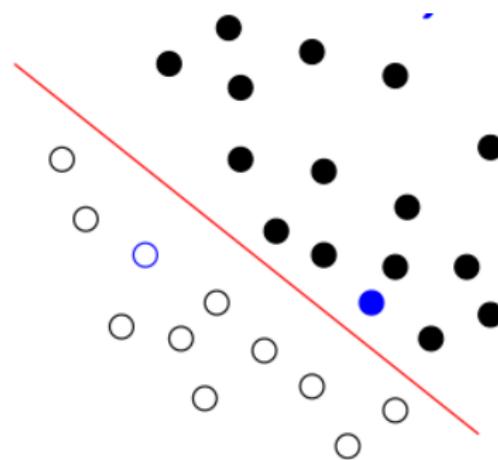
# Outline

- 1 Learning in high dimension
- 2 Learning with  $\ell_2$  regularization
  - Ridge regression
  - Ridge logistic regression
  - Linear hard-margin SVM
  - Interlude: quick notes on constrained optimization
  - Back to hard-margin SVM
  - Soft-margin SVM
  - Large-margin classifiers
- 3 Learning with kernels
  - Kernel methods
  - Positive definite kernels and RKHS
  - Kernel examples
  - Multiple Kernel Learning (MKL)
- 4 Conclusion

# Binary classification

## Setting

- $\mathcal{X} = \mathbb{R}^p$  set of inputs
- $\mathcal{Y} = \{-1, 1\}$  binary outputs
- $\mathcal{S} = \{(x_1, y_1), \dots, (x_n, y_n)\}$  a training set in  $(\mathcal{X} \times \mathcal{Y})^n$
- Goal: Estimate a function  $f : \mathcal{X} \rightarrow \mathbb{R}$  to predict  $y$  by  $\text{sign}(f(x))$



## The 0/1 loss

- The 0/1 loss measures if a prediction is correct or not:

$$\ell_{0/1}(f(x), y) = \mathbf{1}(yf(x) < 0) = \begin{cases} 0 & \text{if } y = \text{sign}(f(x)) \\ 1 & \text{otherwise.} \end{cases}$$

- It is then tempting to learn  $f_\beta(x) = \beta^\top x$  by solving:

$$\min_{\beta \in \mathbb{R}^p} \underbrace{\frac{1}{n} \sum_{i=1}^n \ell_{0/1}(f_\beta(x_i), y_i)}_{\text{misclassification rate}} + \underbrace{\lambda \|\beta\|^2}_{\text{regularization}}$$

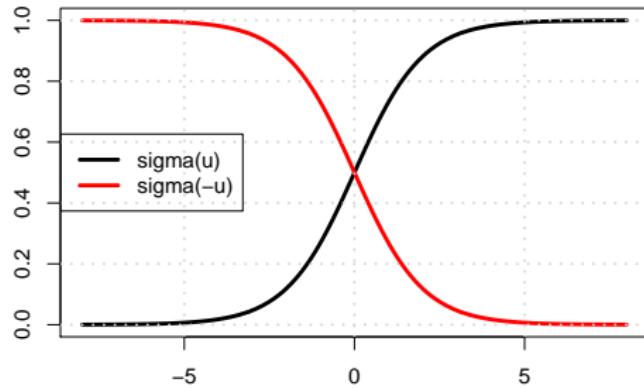
- However:

- The problem is non-smooth, and typically NP-hard to solve
- The regularization has **no effect** since the 0/1 loss is invariant by scaling of  $\beta$
- In fact, no function achieves the minimum when  $\lambda > 0$  (*why?*)

# The logistic loss

- An alternative is to define a probabilistic model of  $y$  parametrized by  $f(x)$ , e.g.:

$$\forall y \in \{-1, 1\}, \quad p(y | f(x)) = \frac{1}{1 + e^{-yf(x)}} = \sigma(yf(x))$$



- The **logistic loss** is the negative conditional likelihood:

$$\ell_{logistic}(f(x), y) = -\ln p(y | f(x)) = \ln \left( 1 + e^{-yf(x)} \right)$$

## Ridge logistic regression (?)

$$\min_{\beta \in \mathbb{R}^p} J(\beta) = \frac{1}{n} \sum_{i=1}^n \ln \left( 1 + e^{-y_i \beta^\top x_i} \right) + \lambda \|\beta\|^2$$

- Can be interpreted as a regularized conditional maximum likelihood estimator
- No explicit solution, but smooth convex optimization problem that can be solved numerically

# Solving ridge logistic regression

$$\min_{\beta} J(\beta) = \frac{1}{n} \sum_{i=1}^n \ln \left( 1 + e^{-y_i \beta^\top x_i} \right) + \lambda \|\beta\|^2$$

No explicit solution, but convex problem with:

$$\begin{aligned}\nabla_{\beta} J(\beta) &= -\frac{1}{n} \sum_{i=1}^n \frac{y_i x_i}{1 + e^{y_i \beta^\top x_i}} + 2\lambda\beta \\ &= -\frac{1}{n} \sum_{i=1}^n y_i [1 - P_{\beta}(y_i | x_i)] x_i + 2\lambda\beta\end{aligned}$$

$$\begin{aligned}\nabla_{\beta}^2 J(\beta) &= \frac{1}{n} \sum_{i=1}^n \frac{x_i x_i^\top e^{y_i \beta^\top x_i}}{(1 + e^{y_i \beta^\top x_i})^2} + 2\lambda I \\ &= \frac{1}{n} \sum_{i=1}^n P_{\beta}(1 | x_i) (1 - P_{\beta}(1 | x_i)) x_i x_i^\top + 2\lambda I\end{aligned}$$

## Solving ridge logistic regression (cont.)

$$\min_{\beta} J(\beta) = \frac{1}{n} \sum_{i=1}^n \ln \left( 1 + e^{-y_i \beta^\top x_i} \right) + \lambda \|\beta\|^2$$

- The solution can then be found by Newton-Raphson iterations:

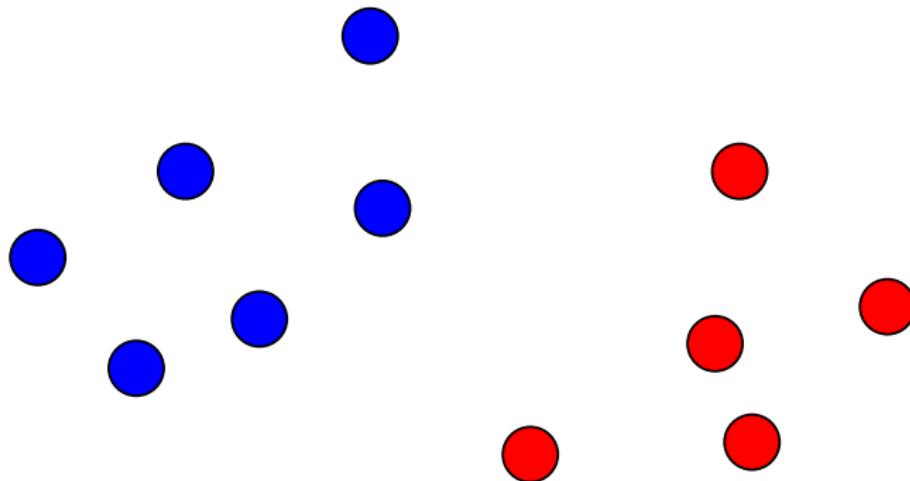
$$\beta^{new} \leftarrow \beta^{old} - \left[ \nabla_{\beta}^2 J(\beta^{old}) \right]^{-1} \nabla_{\beta} J(\beta^{old}).$$

- Each step is equivalent to solving a weighted ridge regression problem (*left as exercise*)
- This method is therefore called **iteratively reweighted least squares (IRLS)**.

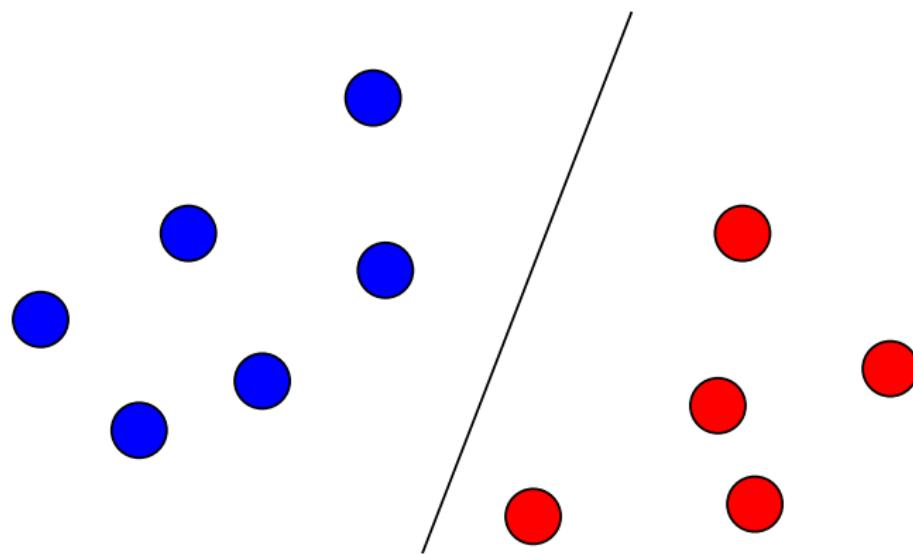
# Outline

- 1 Learning in high dimension
- 2 Learning with  $\ell_2$  regularization
  - Ridge regression
  - Ridge logistic regression
  - Linear hard-margin SVM
  - Interlude: quick notes on constrained optimization
  - Back to hard-margin SVM
  - Soft-margin SVM
  - Large-margin classifiers
- 3 Learning with kernels
  - Kernel methods
  - Positive definite kernels and RKHS
  - Kernel examples
  - Multiple Kernel Learning (MKL)
- 4 Conclusion

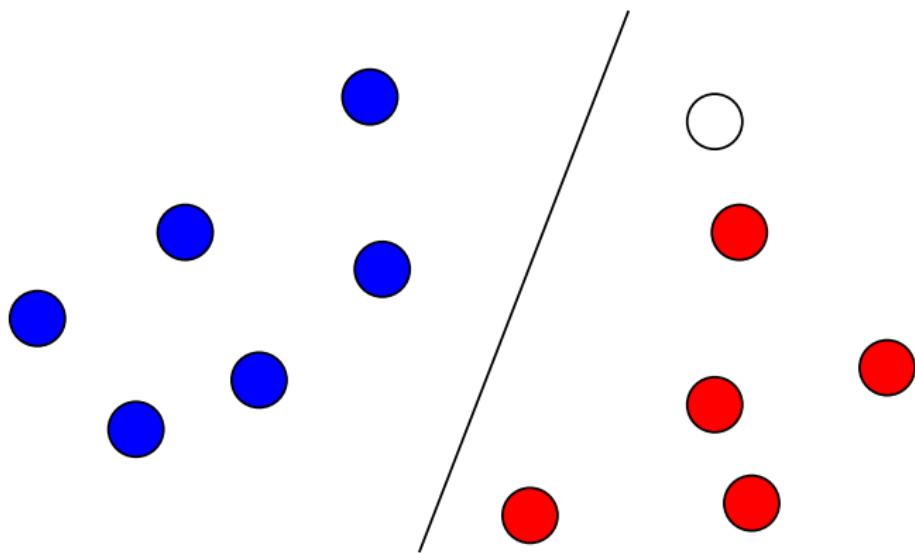
## Linear classifier



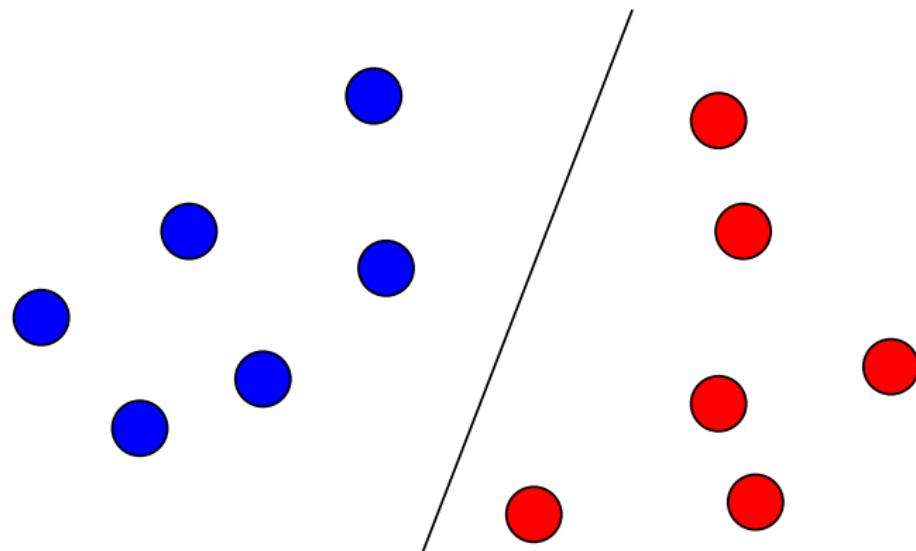
## Linear classifier



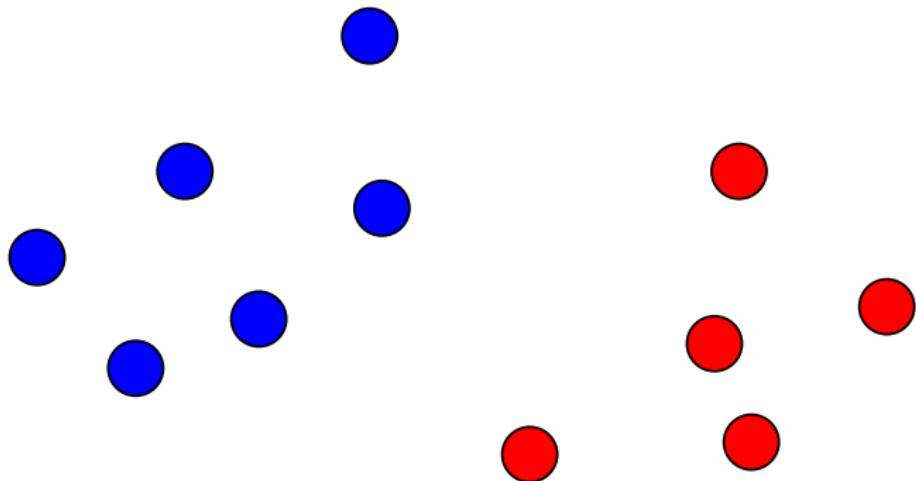
## Linear classifier



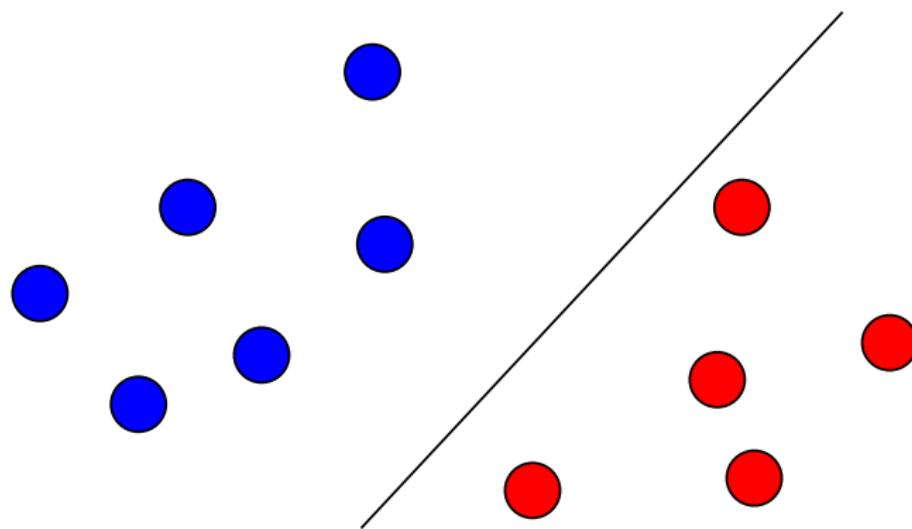
## Linear classifier



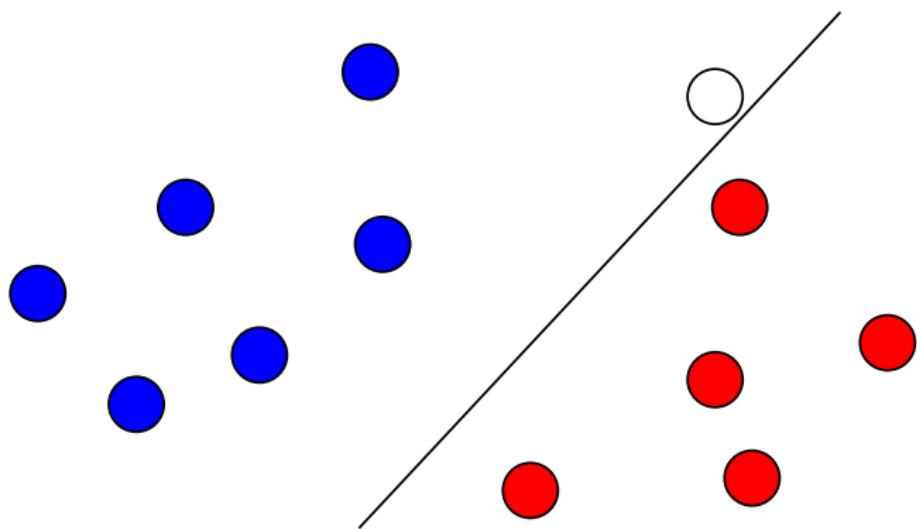
## Linear classifier



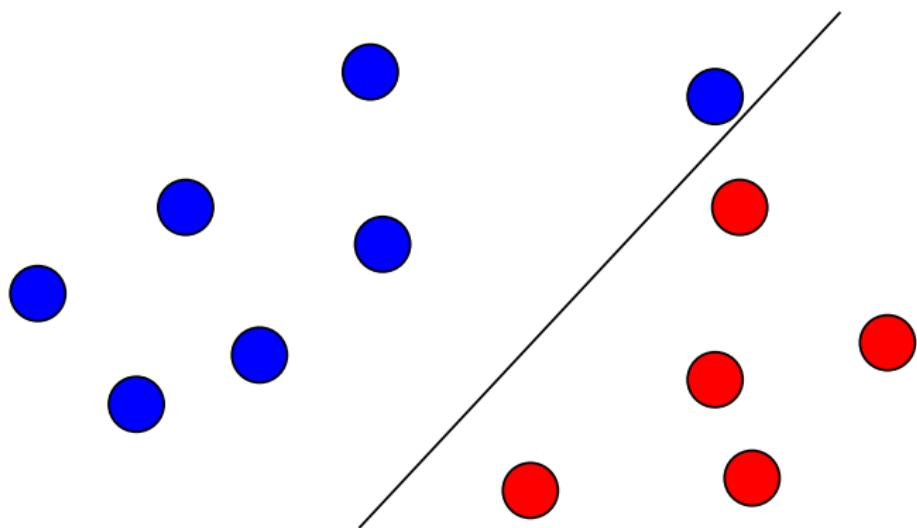
## Linear classifier



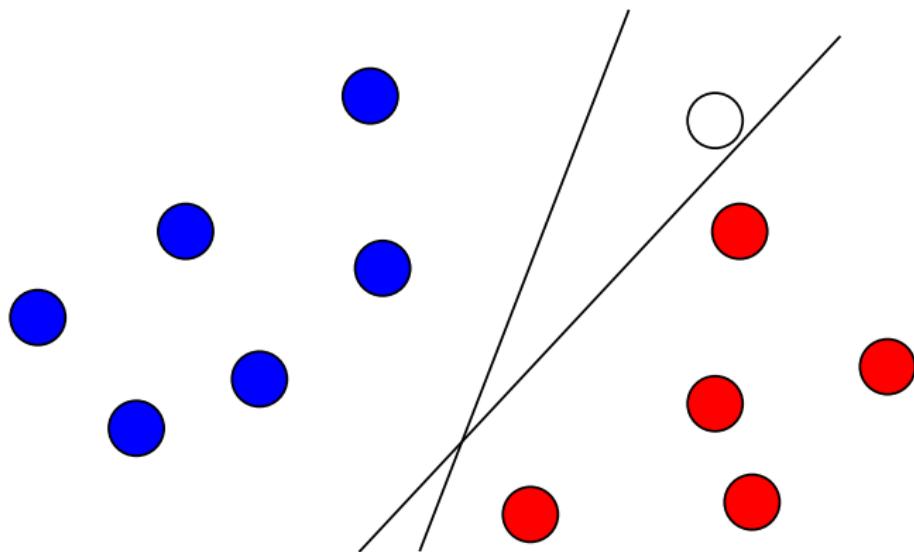
## Linear classifier



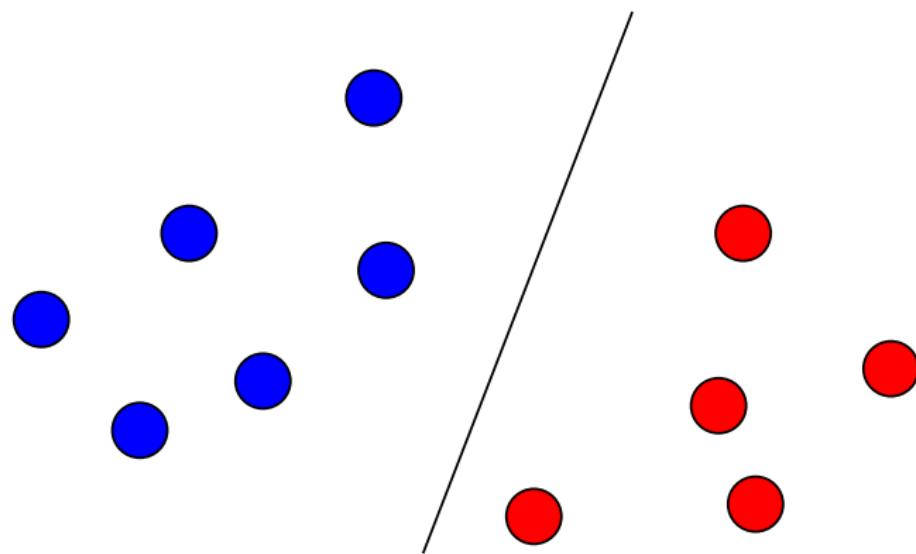
## Linear classifier



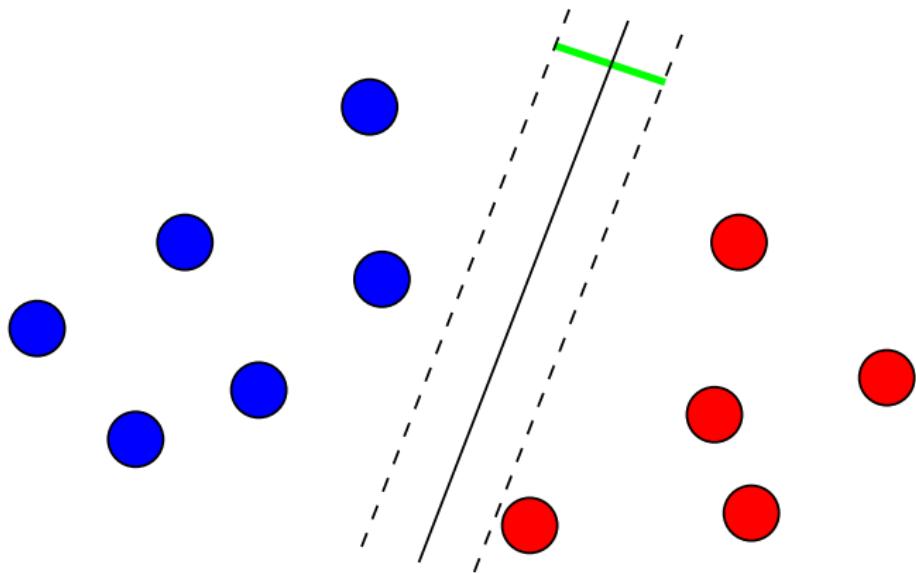
Which one is better?



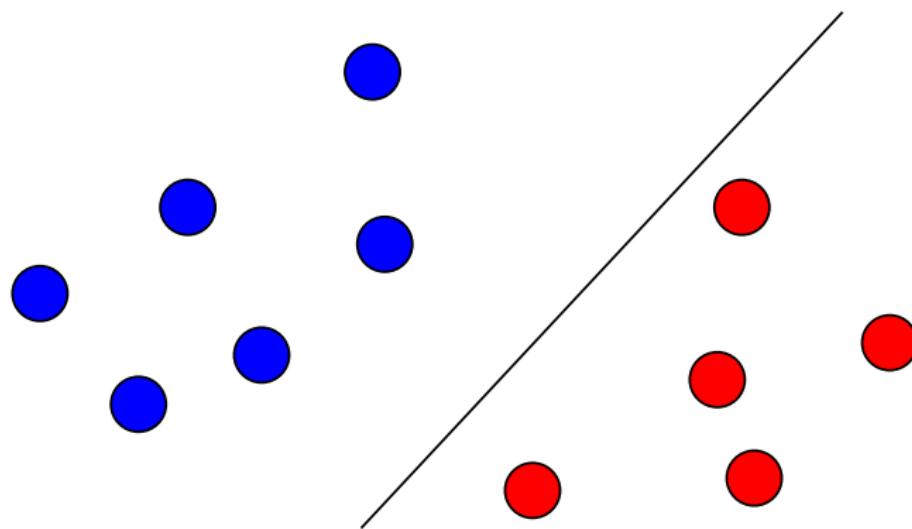
## The margin of a linear classifier



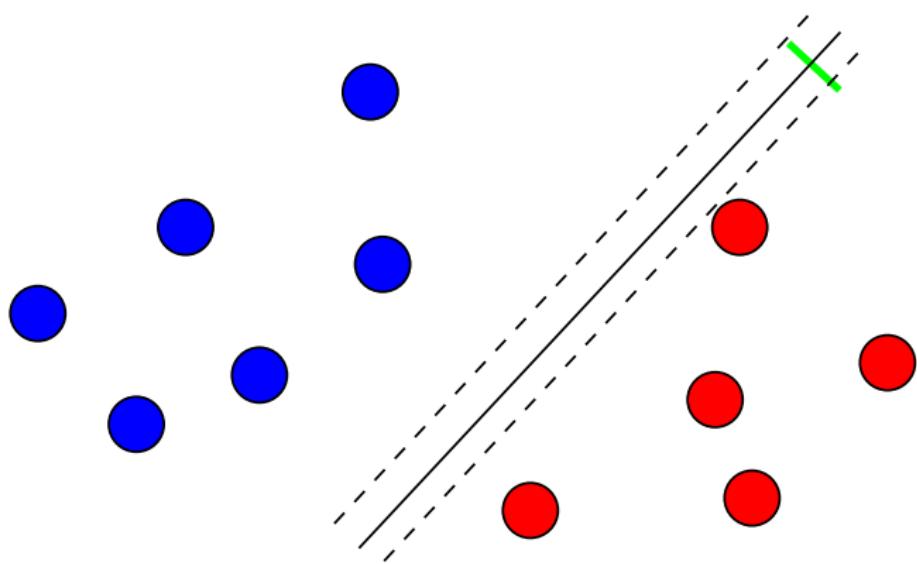
## The margin of a linear classifier



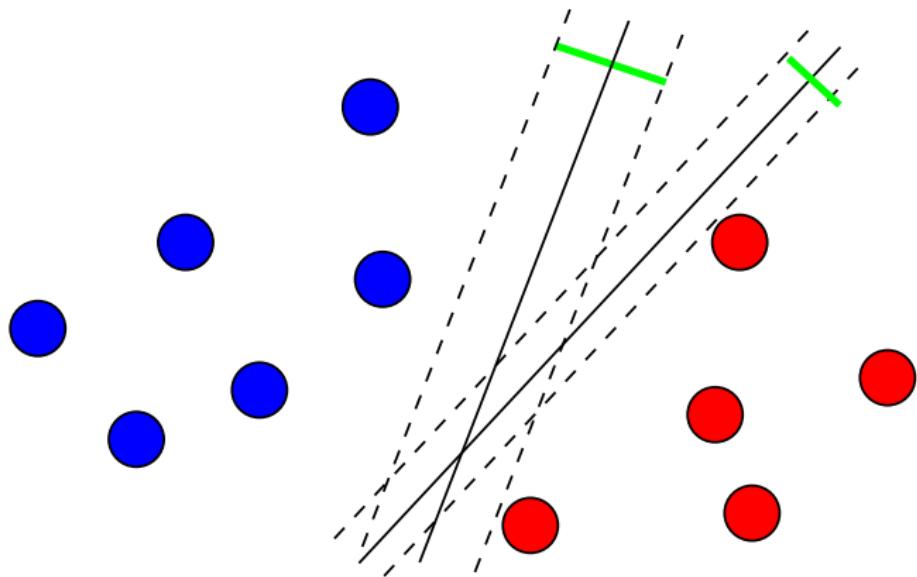
## The margin of a linear classifier



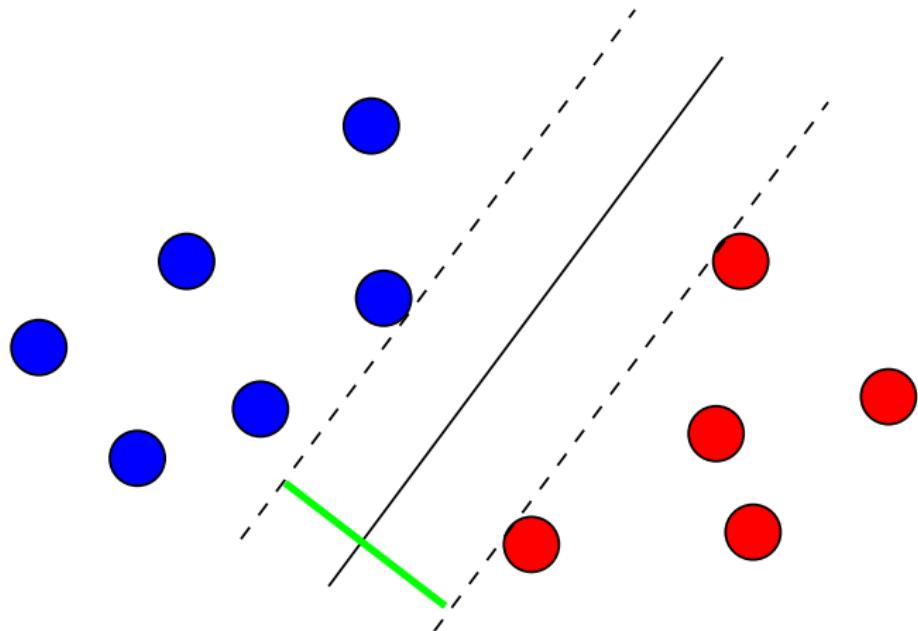
## The margin of a linear classifier



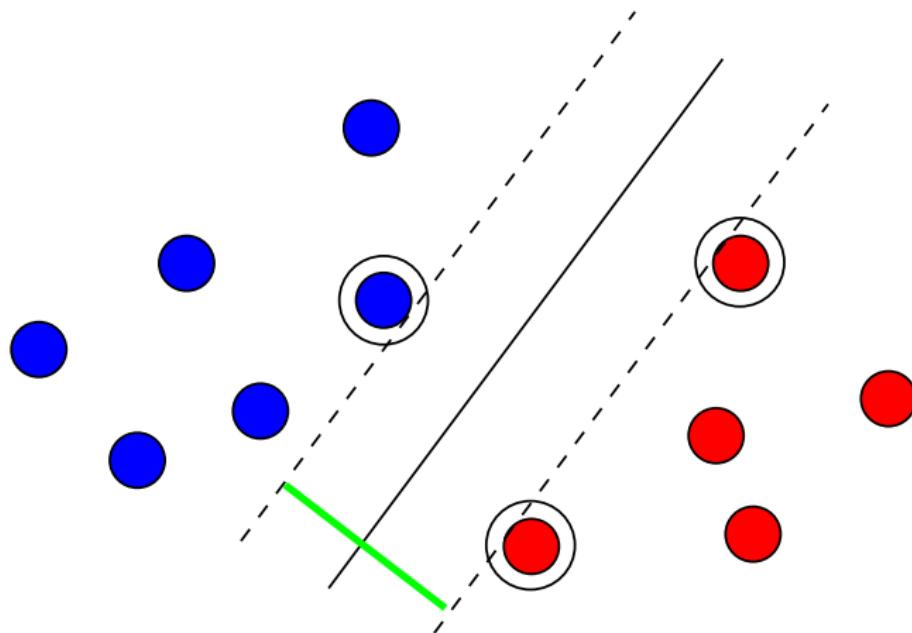
## The margin of a linear classifier



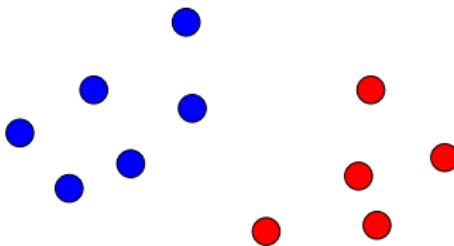
## Largest margin classifier (*hard-margin SVM*)



## Support vectors



## More formally



- The **training set** is a finite set of  $n$  data/class pairs:

$$\mathcal{S} = \{(x_1, y_1), \dots, (x_n, y_n)\},$$

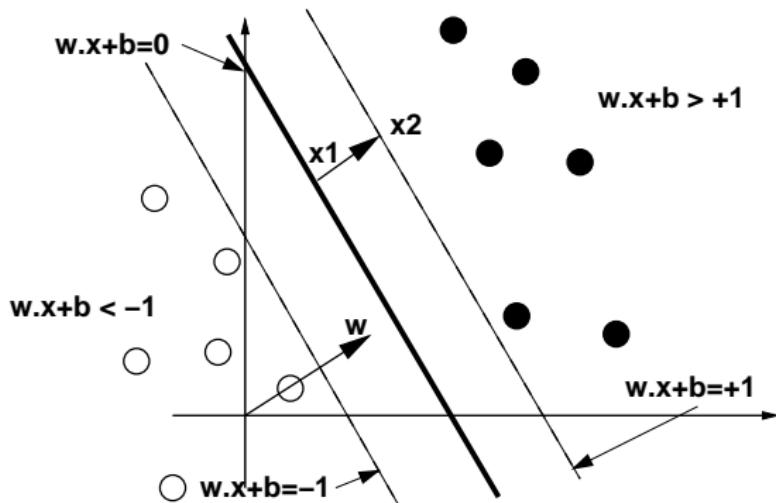
where  $x_i \in \mathbb{R}^p$  and  $y_i \in \{-1, 1\}$ .

- We assume (for the moment) that the data are **linearly separable**, i.e., that there exists  $(w, b) \in \mathbb{R}^p \times \mathbb{R}$  such that:

$$\begin{cases} w^\top x_i + b > 0 & \text{if } y_i = 1, \\ w^\top x_i + b < 0 & \text{if } y_i = -1. \end{cases}$$

# How to find the largest separating hyperplane?

For a given linear classifier  $f(x) = w^\top x + b$  consider the "tube" defined by the values  $-1$  and  $+1$  of the decision function:



The margin is  $2/\| w \|$

Indeed, the points  $x_1$  and  $x_2$  satisfy:

$$\begin{cases} w^\top x_1 + b = 0, \\ w^\top x_2 + b = 1. \end{cases}$$

By subtracting we get

$$w^\top (x_2 - x_1) = 1 = \| w \| \times \| x_2 - x_1 \|,$$

and therefore:

$$\gamma = 2\| x_2 - x_1 \|_2 = \frac{2}{\| w \|}.$$

All training points should be on the correct side of the dotted line

For positive examples ( $y_i = 1$ ) this means:

$$w^\top x_i + b \geq 1.$$

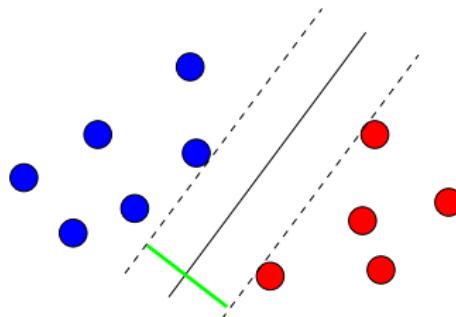
For negative examples ( $y_i = -1$ ) this means:

$$w^\top x_i + b \leq -1.$$

Both cases are summarized by:

$$\forall i = 1, \dots, n, \quad y_i (w^\top x_i + b) \geq 1.$$

## Finding the optimal hyperplane



Find  $(w, b)$  which minimize:

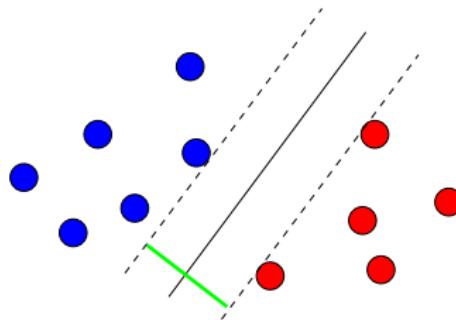
$$\|w\|^2$$

under the constraints:

$$\forall i = 1, \dots, n, \quad y_i (w^\top x_i + b) - 1 \geq 0.$$

This is a classical quadratic program on  $\mathbb{R}^{p+1}$ .

## Another view of hard-margin SVM



$$\min_{w,b} \left\{ \sum_{i=1}^n \ell_{hard-margin} (w^\top x_i + b, y_i) + \lambda \|w\|^2 \right\},$$

for the hard-margin loss function:

$$\ell_{hard-margin}(u, y) = \begin{cases} 0 & \text{if } yu \geq 1, \\ +\infty & \text{otherwise.} \end{cases}$$

# Outline

- 1 Learning in high dimension
- 2 Learning with  $\ell_2$  regularization
  - Ridge regression
  - Ridge logistic regression
  - Linear hard-margin SVM
  - Interlude: quick notes on constrained optimization
    - Back to hard-margin SVM
    - Soft-margin SVM
    - Large-margin classifiers
- 3 Learning with kernels
  - Kernel methods
  - Positive definite kernels and RKHS
  - Kernel examples
  - Multiple Kernel Learning (MKL)
- 4 Conclusion

# Optimization problems

## Setting

- We consider an equality and inequality constrained optimization problem over a variable  $x \in \mathcal{X}$ :

$$\text{minimize } f(x)$$

$$\text{subject to } h_i(x) = 0, \quad i = 1, \dots, m,$$

$$g_j(x) \leq 0, \quad j = 1, \dots, r,$$

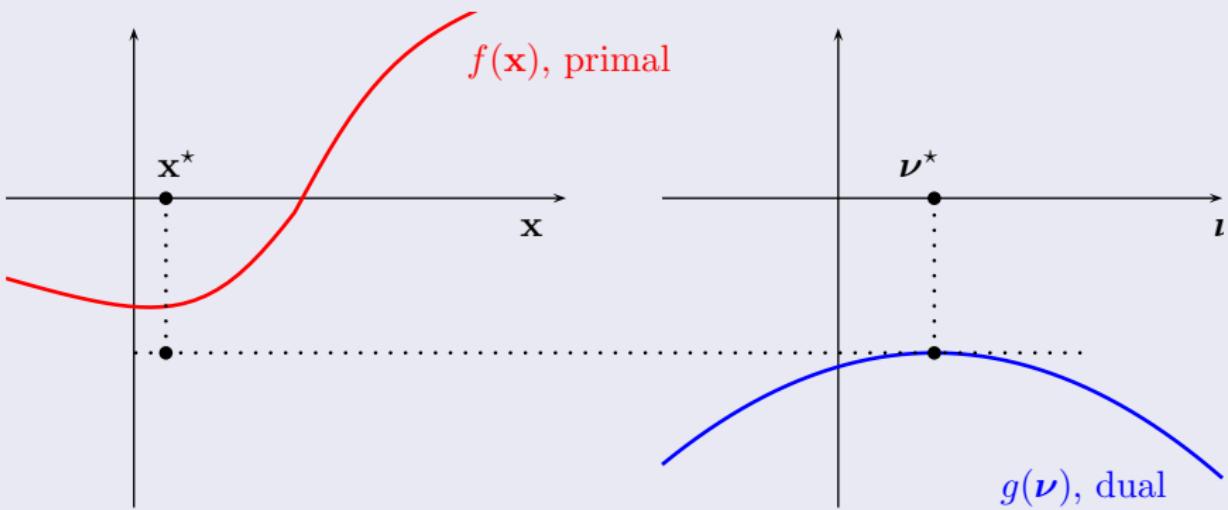
making **no assumption** of  $f$ ,  $g$  and  $h$ .

- Let us denote by  $f^*$  the optimal value of the decision function under the constraints, i.e.,  $f^* = f(x^*)$  if the minimum is reached at a global minimum  $x^*$ .

# Duality

Duality provides of a concave *dual* function  $q$  from a *primal* function  $f$  such that for all pair  $x, \nu$ , the value  $f(x)$  is greater than  $q(\nu)$ .

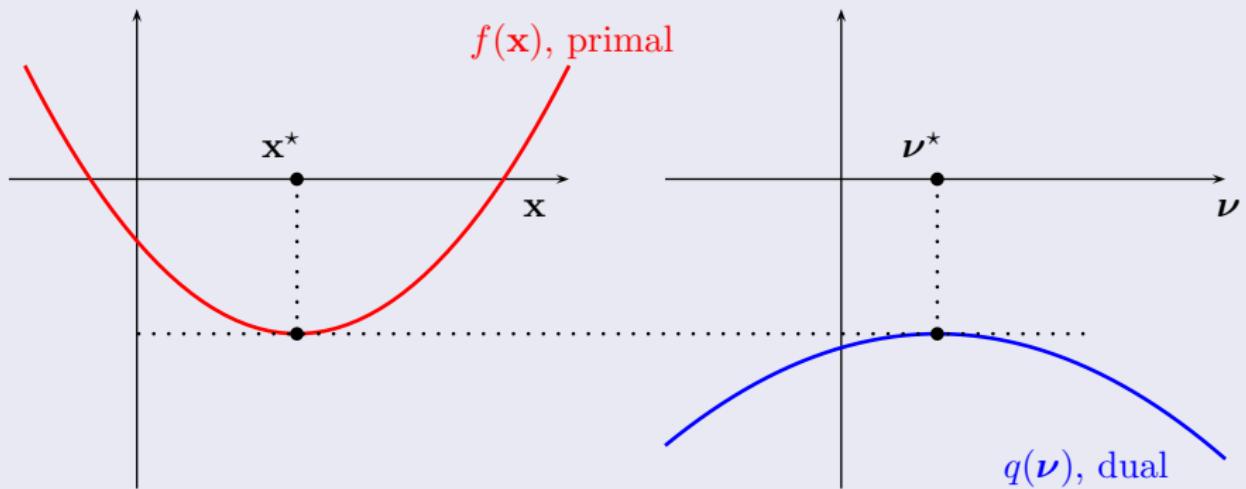
## Weak Duality



# Duality

For most “reasonable” convex problems, duality comes with an additional property called strong duality, which is very useful.

## Strong Duality

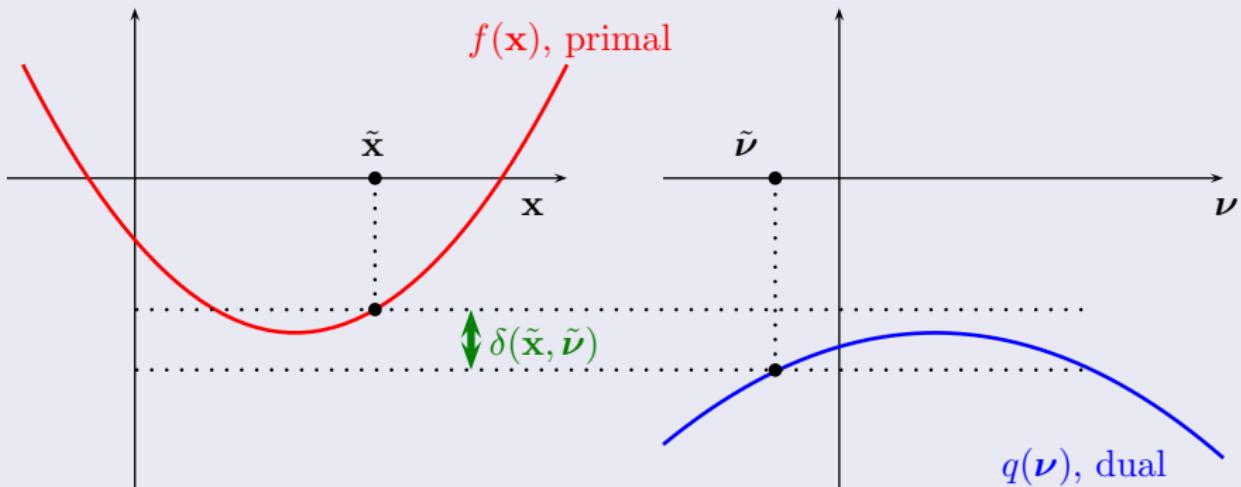


- Strong duality means that  $\max_{\nu} q(\nu) = \min_{\mathbf{x}} f(\mathbf{x})$

# Duality

For instance, we have the concept of duality gap, which are very useful to stop an optimization algorithm.

## Duality gaps



- The duality gap guarantees us that  $0 \leq f(\tilde{\mathbf{x}}) - f(\mathbf{x}^*) \leq \delta(\tilde{\mathbf{x}}, \tilde{\boldsymbol{\nu}})$ .
- Dual problems are often obtained by Lagrangian or Fenchel duality.

# Optimization problems

## Setting

- We consider an equality and inequality constrained optimization problem over a variable  $x \in \mathcal{X}$ :

$$\begin{aligned} & \text{minimize} && f(x) \\ & \text{subject to} && h_i(x) = 0, \quad i = 1, \dots, m, \\ & && g_j(x) \leq 0, \quad j = 1, \dots, r, \end{aligned}$$

making **no assumption** of  $f$ ,  $g$  and  $h$ .

- Let us denote by  $f^*$  the optimal value of the decision function under the constraints, i.e.,  $f^* = f(x^*)$  if the minimum is reached at a global minimum  $x^*$ .

# Lagrangian and dual function

## Lagrangian

The **Lagrangian** of this problem is the function  $L : \mathcal{X} \times \mathbb{R}^m \times \mathbb{R}^r \rightarrow \mathbb{R}$  defined by:

$$L(x, \lambda, \mu) = f(x) + \sum_{i=1}^m \lambda_i h_i(x) + \sum_{j=1}^r \mu_j g_j(x).$$

## Lagrangian dual function

The **Lagrange dual function**  $g : \mathbb{R}^m \times \mathbb{R}^r \rightarrow \mathbb{R}$  is:

$$\begin{aligned} q(\lambda, \mu) &= \inf_{x \in \mathcal{X}} L(x, \lambda, \mu) \\ &= \inf_{x \in \mathcal{X}} \left( f(x) + \sum_{i=1}^m \lambda_i h_i(x) + \sum_{j=1}^r \mu_j g_j(x) \right). \end{aligned}$$

## Properties of the dual function

- $q$  is concave in  $(\lambda, \mu)$ , even if the original problem is not convex.
- The dual function yields lower bounds on the optimal value  $f^*$  of the original problem when  $\mu$  is nonnegative:

$$q(\lambda, \mu) \leq f^* , \quad \forall \lambda \in \mathbb{R}^m, \forall \mu \in \mathbb{R}^r, \mu \geq 0 .$$

## Proofs

- For each  $x$ , the function  $(\lambda, \mu) \mapsto L(x, \lambda, \mu)$  is linear, and therefore both convex and concave in  $(\lambda, \mu)$ . The pointwise minimum of concave functions is concave, therefore  $q$  is concave.
- Let  $\bar{x}$  be any feasible point, i.e.,  $h(\bar{x}) = 0$  and  $g(\bar{x}) \leq 0$ . Then we have, for any  $\lambda$  and  $\mu \geq 0$ :

$$\sum_{i=1}^m \lambda_i h_i(\bar{x}) + \sum_{i=1}^r \mu_i g_i(\bar{x}) \leq 0 ,$$

$$\implies L(\bar{x}, \lambda, \mu) = f(\bar{x}) + \sum_{i=1}^m \lambda_i h_i(\bar{x}) + \sum_{i=1}^r \mu_i g_i(\bar{x}) \leq f(\bar{x}) ,$$

$$\implies q(\lambda, \mu) = \inf_x L(x, \lambda, \mu) \leq L(\bar{x}, \lambda, \mu) \leq f(\bar{x}) , \quad \forall \bar{x} . \quad \square$$

# Dual problem

## Definition

For the (primal) problem:

$$\begin{aligned} & \text{minimize} && f(x) \\ & \text{subject to} && h(x) = 0 , \quad g(x) \leq 0 , \end{aligned}$$

the Lagrange dual problem is:

$$\begin{aligned} & \text{maximize} && q(\lambda, \mu) \\ & \text{subject to} && \mu \geq 0 , \end{aligned}$$

where  $q$  is the (concave) Lagrange dual function and  $\lambda$  and  $\mu$  are the Lagrange multipliers associated to the constraints  $h(x) = 0$  and  $g(x) \leq 0$ .

## Weak duality

- Let  $d^*$  the optimal value of the Lagrange dual problem. Each  $q(\lambda, \mu)$  is an lower bound for  $f^*$  and by definition  $d^*$  is the best lower bound that is obtained. The following **weak duality inequality** therefore always hold:

$$d^* \leq f^* .$$

- This inequality holds when  $d^*$  or  $f^*$  are infinite. The difference  $d^* - f^*$  is called the **optimal duality gap** of the original problem.

## Strong duality

- We say that **strong duality** holds if the optimal duality gap is zero, i.e.:

$$d^* = f^* .$$

- If strong duality holds, then the best lower bound that can be obtained from the Lagrange dual function is **tight**
- Strong duality does **not hold** for general nonlinear problems.
- It usually holds for **convex problems**.
- Conditions that ensure strong duality for convex problems are called **constraint qualification**.

## Slater's constraint qualification

Strong duality holds for a **convex** problem:

$$\begin{aligned} & \text{minimize} && f(x) \\ & \text{subject to} && g_j(x) \leq 0, \quad j = 1, \dots, r, \\ & && Ax = b, \end{aligned}$$

if it is **strictly feasible**, i.e., there exists at least one **feasible point** that satisfies:

$$g_j(x) < 0, \quad j = 1, \dots, r, \quad Ax = b.$$

## Remarks

- Slater's conditions also ensure that the maximum  $d^*$  (if  $> -\infty$ ) is attained, i.e., there exists a point  $(\lambda^*, \mu^*)$  with

$$q(\lambda^*, \mu^*) = d^* = f^*$$

- They can be sharpened. For example, strict feasibility is not required for affine constraints.
- There exist many other types of constraint qualifications

## Dual optimal pairs

Suppose that strong duality holds,  $x^*$  is primal optimal,  $(\lambda^*, \mu^*)$  is dual optimal. Then we have:

$$\begin{aligned} f(x^*) &= q(\lambda^*, \mu^*) \\ &= \inf_{x \in \mathbb{R}^n} \left\{ f(x) + \sum_{i=1}^m \lambda_i^* h_i(x) + \sum_{j=1}^r \mu_j^* g_j(x) \right\} \\ &\leq f(x^*) + \sum_{i=1}^m \lambda_i^* h_i(x^*) + \sum_{j=1}^r \mu_j^* g_j(x^*) \\ &\leq f(x^*) \end{aligned}$$

Hence both inequalities are in fact **equalities**.

## Complimentary slackness

The first equality shows that:

$$L(x^*, \lambda^*, \mu^*) = \inf_{x \in \mathbb{R}^n} L(x, \lambda^*, \mu^*) ,$$

showing that  $x^*$  minimizes the Lagrangian at  $(\lambda^*, \mu^*)$ . The second equality shows that:

$$\mu_j g_j(x^*) = 0 , \quad j = 1, \dots, r .$$

This property is called complementary slackness:

the  $i$ th optimal Lagrange multiplier is zero unless the  $i$ th constraint is active at the optimum.

# Outline

- 1 Learning in high dimension
- 2 Learning with  $\ell_2$  regularization

- Ridge regression
- Ridge logistic regression
- Linear hard-margin SVM
- Interlude: quick notes on constrained optimization
- Back to hard-margin SVM
- Soft-margin SVM
- Large-margin classifiers

- 3 Learning with kernels

- Kernel methods
- Positive definite kernels and RKHS
- Kernel examples
- Multiple Kernel Learning (MKL)

- 4 Conclusion

# Lagrangian

In order to minimize:

$$\frac{1}{2} \| w \|^2$$

under the constraints:

$$\forall i = 1, \dots, n, \quad y_i (w^\top x_i + b) - 1 \geq 0,$$

we introduce one dual variable  $\alpha_i$  for each constraint, i.e., for each training point. The Lagrangian is:

$$L(w, b, \alpha) = \frac{1}{2} \| w \|^2 - \sum_{i=1}^n \alpha_i (y_i (w^\top x_i + b) - 1).$$

## Lagrangian

- $L(w, b, \alpha)$  is convex quadratic in  $w$ . It is minimize for:

$$\nabla_w L = w - \sum_{i=1}^n \alpha_i y_i x_i = 0 \quad \Rightarrow \quad w = \sum_{i=1}^n \alpha_i y_i x_i.$$

- $L(w, b, \alpha)$  is affine in  $b$ . Its minimum is  $-\infty$  except if:

$$\nabla_b L = \sum_{i=1}^n \alpha_i y_i = 0.$$

## Dual function

- We therefore obtain the **Lagrange dual function**:

$$\begin{aligned} q(\alpha) &= \inf_{w \in \mathbb{R}^p, b \in \mathbb{R}} L(w, b, \alpha) \\ &= \begin{cases} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i \alpha_j x_i \cdot x_j & \text{if } \sum_{i=1}^n \alpha_i y_i = 0, \\ -\infty & \text{otherwise.} \end{cases} \end{aligned}$$

- The dual problem is:

$$\begin{aligned} &\text{maximize} && q(\alpha) \\ &\text{subject to} && \alpha \geq 0. \end{aligned}$$

## Dual problem

Find  $\alpha^* \in \mathbb{R}^n$  which maximizes

$$L(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j x_i^\top x_j,$$

under the (simple) constraints  $\alpha_i \geq 0$  (for  $i = 1, \dots, n$ ), and

$$\sum_{i=1}^n \alpha_i y_i = 0.$$

*This is a quadratic program on  $\mathbb{R}^N$ , with "box constraints".  $\alpha^*$  can be found efficiently using dedicated optimization softwares.*

## Complementary slackness conditions

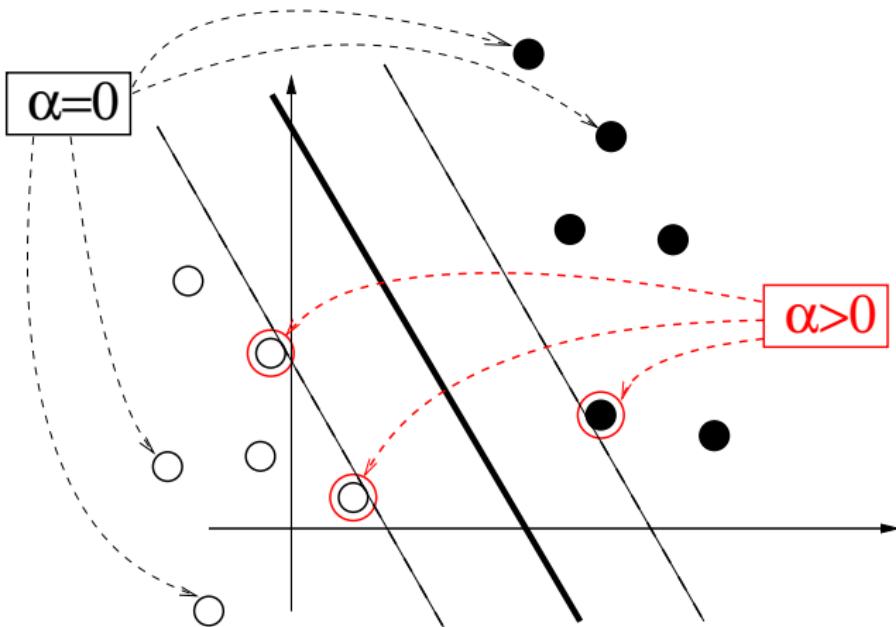
- At the optimal, the complementary slackness conditions must hold:

$$\forall i = 1, \dots, n \quad \alpha_i^* \left( y_i \left( w^{*\top} x_i + b^* \right) - 1 \right) = 0.$$

- This implies that:

- If  $\alpha_i^* > 0$  then  $y_i \left( w^{*\top} x_i + b^* \right) = 1$
- If  $y_i \left( w^{*\top} x_i + b^* \right) > 1$  then  $\alpha_i^* = 0$

## Interpretation: support vectors



# Recovering the optimal hyperplane

- Once  $\alpha^*$  is found, we recover  $w^*$  by:

$$w^* = \underset{w}{\operatorname{argmin}} L(w, b, \alpha^*) = \sum_{i=1}^n \alpha_i y_i x_i$$

- To recover  $b$  we can not just minimize  $L(w, b, \alpha^*)$ , since it does not depend on  $b$ . Instead, we use the complementary slackness condition: if  $i$  is such that  $\alpha_i > 0$ , then

$$y_i (w^{*\top} x_i + b^*) = 1 \implies b^* = y_i - w^{*\top} x_i$$

- The **decision function** is therefore:

$$f^*(x) = w^{*\top} x + b^*$$

$$= \sum_{i=1}^n \alpha_i y_i x_i^\top x + b^*.$$

## Primal (for large $n$ ) vs dual (for large $p$ ) optimization

- ① Find  $(w, b) \in \mathbb{R}^{p+1}$  which minimize:

$$\frac{1}{2} \| w \|^2$$

under the constraints:

$$\forall i = 1, \dots, n, \quad y_i (w^\top x_i + b) - 1 \geq 0.$$

- ② Find  $\alpha^* \in \mathbb{R}^n$  which maximizes

$$L(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j x_i \cdot x_j,$$

under the (simple) constraints  $\alpha_i \geq 0$  (for  $i = 1, \dots, n$ ), and

$$\sum_{i=1}^n \alpha_i y_i = 0.$$

# Outline

- 1 Learning in high dimension
- 2 Learning with  $\ell_2$  regularization

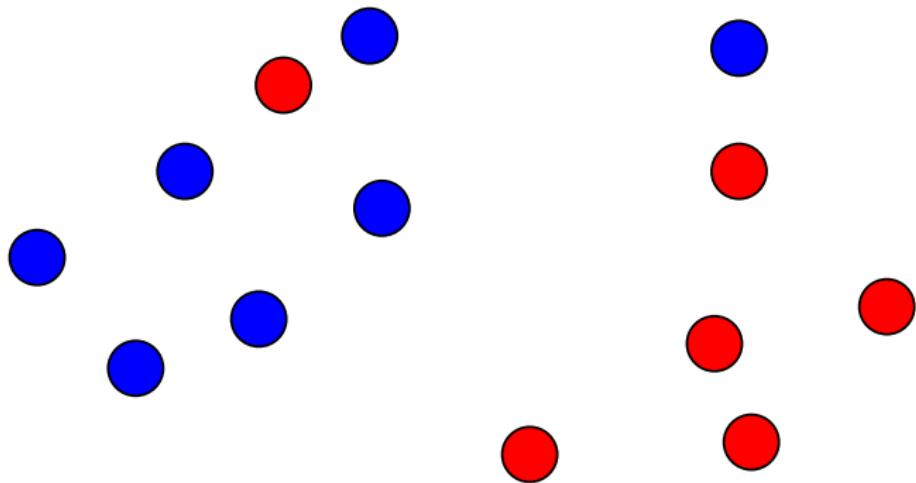
- Ridge regression
- Ridge logistic regression
- Linear hard-margin SVM
- Interlude: quick notes on constrained optimization
- Back to hard-margin SVM
- Soft-margin SVM
- Large-margin classifiers

- 3 Learning with kernels

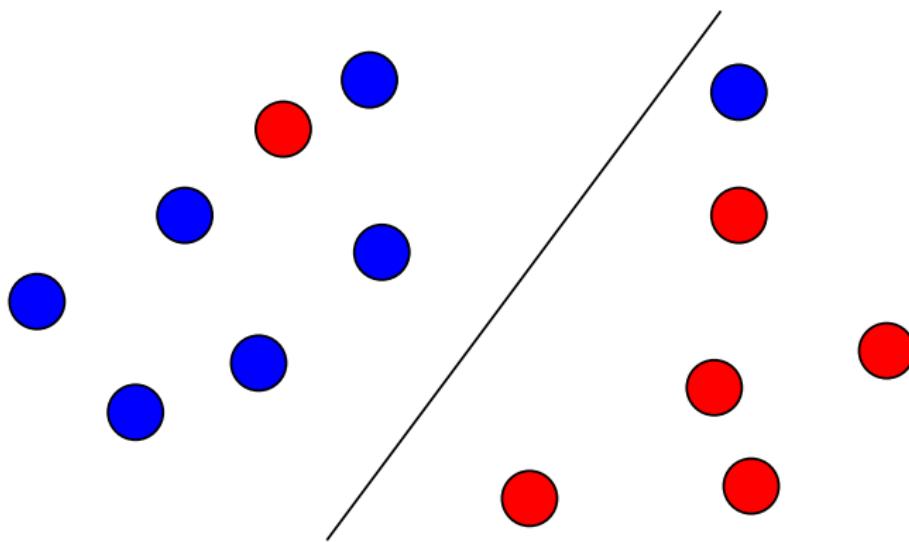
- Kernel methods
- Positive definite kernels and RKHS
- Kernel examples
- Multiple Kernel Learning (MKL)

- 4 Conclusion

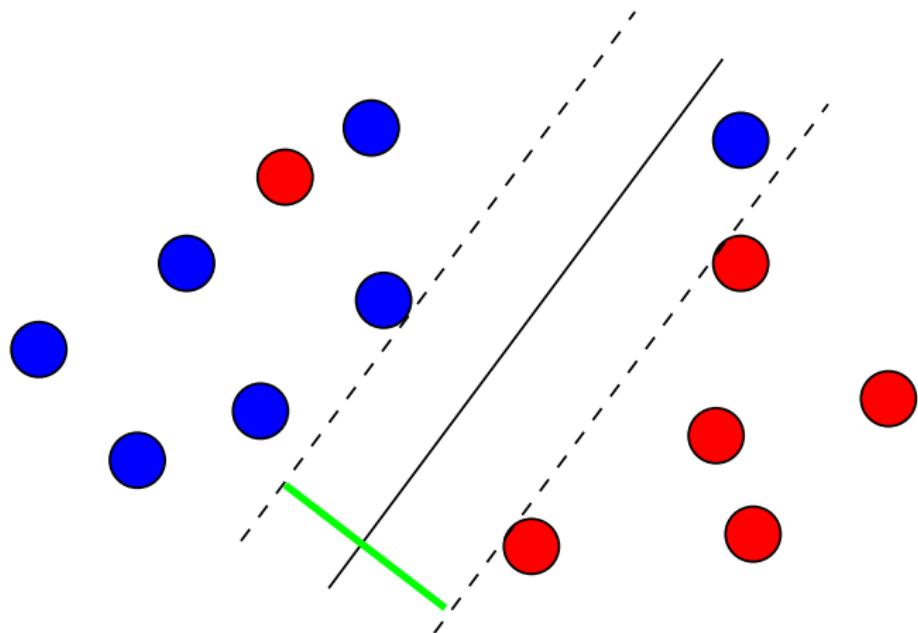
What if data are not linearly separable?



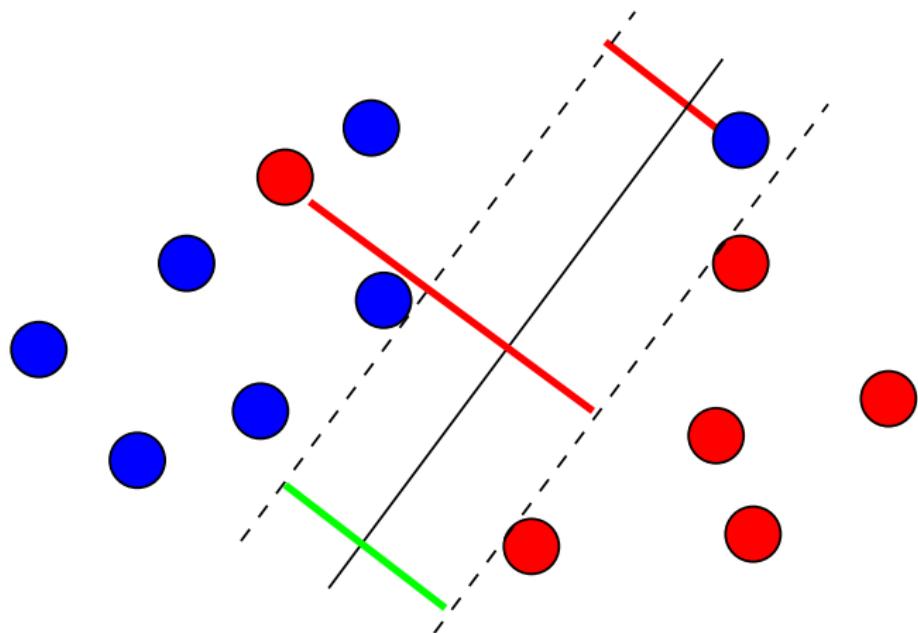
What if data are not linearly separable?



What if data are not linearly separable?



What if data are not linearly separable?

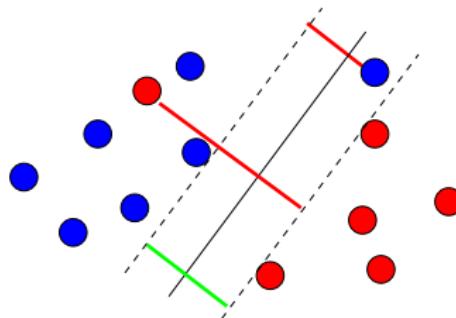


# Soft-margin SVM

- Find a trade-off between **large margin** and **few errors**.
- Mathematically:

$$\min_f \left\{ \frac{1}{margin(f)} + C \times errors(f) \right\}$$

- $C$  is a parameter



## Soft-margin SVM formulation

- The **margin** of a labeled point  $(x, y)$  is

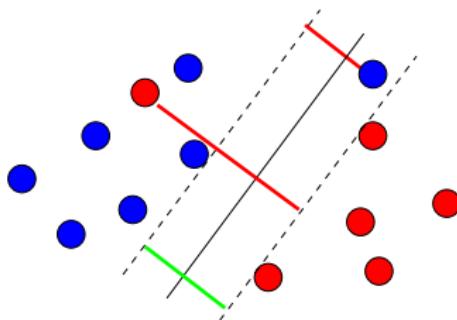
$$\text{margin}(x, y) = y \left( w^\top x + b \right)$$

- The **error** is

- 0 if  $\text{margin}(x, y) > 1$ ,
- $1 - \text{margin}(x, y)$  otherwise.

- The soft margin SVM solves:

$$\min_{w,b} \left\{ \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \max \left( 0, 1 - y_i (w^\top x_i + b) \right) \right\}$$

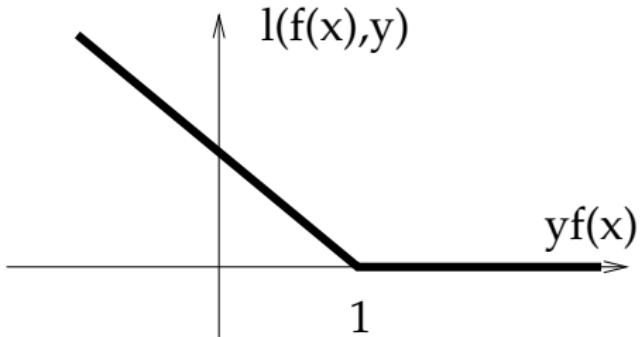
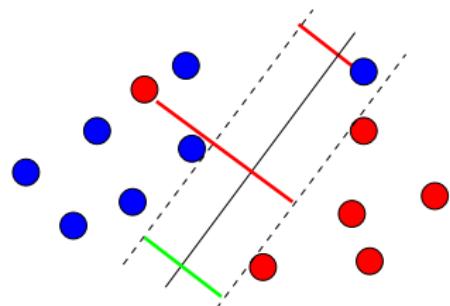


# Soft-margin SVM and hinge loss

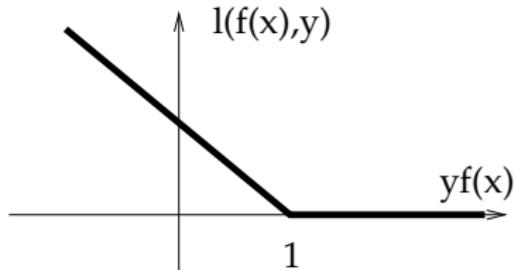
$$\min_{w,b} \left\{ \frac{1}{n} \sum_{i=1}^n \ell_{\text{hinge}}(w^\top x_i + b, y_i) + \lambda \|w\|^2 \right\},$$

for  $\lambda = 1/2nC$  and the hinge loss function:

$$\ell_{\text{hinge}}(u, y) = \max(1 - yu, 0) = \begin{cases} 0 & \text{if } yu \geq 1, \\ 1 - yu & \text{otherwise.} \end{cases}$$



## Reformulation as a QP



- Note that for any  $u \in \mathbb{R}$ ,

$$\varphi_{\text{hinge}}(u) = \min_{\xi \in \mathbb{R}} \xi \quad \text{such that} \quad \begin{cases} \xi \geq 0 \\ \xi \geq 1 - u \end{cases}$$

- Therefore SVM solves the QP

$$\min_{w, b, \xi} \left\{ \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \xi_i \right\}, \text{ s.t. } \forall i \in [1, n], \begin{cases} \xi_i \geq 0 \\ \xi_i \geq 1 - y_i (w^\top x_i + b) \end{cases}$$

# Lagrangian

Form the Lagrangian:

$$L(w, b, \xi, \alpha, \gamma) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \xi_i - \sum_{i=1}^n \alpha_i (y_i(x_i^\top w + b) + \xi_i - 1) - \gamma^\top \xi$$

Minimize in the primal variables  $(w, b, \xi)$ :

$$\nabla_w L = w - \sum_{i=1}^n \alpha_i y_i x_i \implies w = \sum_{i=1}^n \alpha_i y_i x_i$$

$$\nabla_b L = - \sum_{i=1}^n \alpha_i y_i \implies \sum_{i=1}^n \alpha_i y_i = 0$$

$$\nabla_{\xi_i} L = C - \alpha_i - \gamma_i \implies \alpha_i + \gamma_i = C, \forall i \in [1, n]$$

## Dual function

- We therefore obtain the **Lagrange dual function**:

$$q(\alpha; \gamma) = \inf_{w \in \mathbb{R}^p, b \in \mathbb{R}} L(w, b, \alpha, \gamma)$$
$$= \begin{cases} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i \alpha_j x_i \cdot x_j & \text{if } \sum_{i=1}^n \alpha_i y_i = 0 \\ & \text{and } \alpha_i + \gamma_i = C, \forall i, \\ -\infty & \text{otherwise.} \end{cases}$$

- The dual problem is:

$$\begin{aligned} & \text{maximize} && q(\alpha, \gamma) \\ & \text{subject to} && \alpha \geq 0 \text{ and } \gamma \geq 0. \end{aligned}$$

- Since  $\gamma$  only appears in the constraints, this is equivalent to maximizing the dual function in  $\alpha$  under the constraint  $0 \leq \alpha \leq C$

## Dual formulation

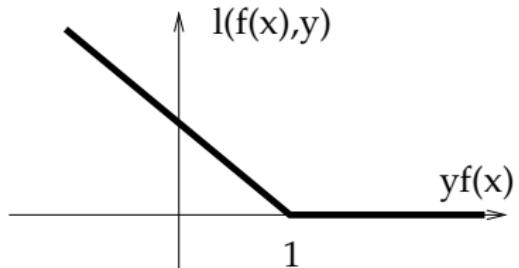
$$\max_{\alpha \in \mathbb{R}^n} \left\{ \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j x_i^\top x_j \right\}$$

under the constraints:

$$\begin{cases} 0 \leq \alpha_i \leq C, & \text{for } i = 1, \dots, n \\ \sum_{i=1}^n \alpha_i y_i = 0. \end{cases}$$

*Remark: we recover hard-margin SVM with  $C = +\infty$*

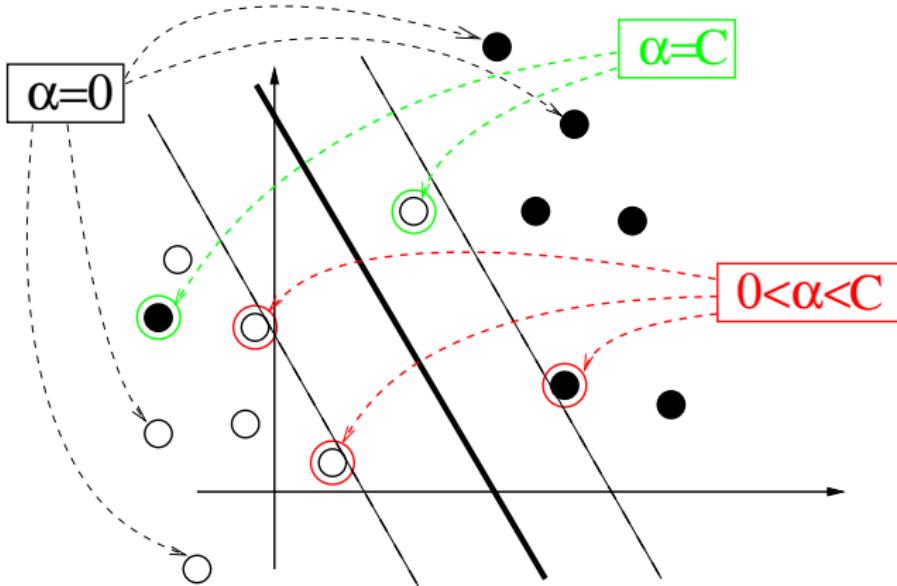
# KKT



Let  $m_i = y_i(w^\top x_i + b)$  be the margin of the  $i$ -th point

- $\alpha_i = 0 \implies \gamma_i > 0 \implies \varphi_{hinge}(m_i) = 0 \implies m_i \geq 1$
- $\alpha_i = C > 0 \implies \varphi_{hinge}(m_i) = 1 - m_i \implies m_i \leq 1$
- $0 < \alpha < C \implies (\alpha_i > 0 \text{ and } \gamma_i > 0) \implies \varphi_{hinge}(m_i) = 1 - m_i = 0 \implies m_i = 1$

# Interpretation: bounded and unbounded support vectors



# Outline

- 1 Learning in high dimension
- 2 Learning with  $\ell_2$  regularization

- Ridge regression
- Ridge logistic regression
- Linear hard-margin SVM
- Interlude: quick notes on constrained optimization
- Back to hard-margin SVM
- Soft-margin SVM
- Large-margin classifiers

- 3 Learning with kernels

- Kernel methods
- Positive definite kernels and RKHS
- Kernel examples
- Multiple Kernel Learning (MKL)

- 4 Conclusion

# Loss functions for classifications

We already saw 3 loss functions for binary classification problems

- The 0/1 loss  $\ell_{0/1}(f(x), y) = \mathbf{1}(yf(x) < 0)$
- The logistic loss  $\ell_{logistic}(f(x), y) = \ln(1 + e^{-yf(x)})$
- The hinge loss  $\ell_{hinge}(f(x), y) = \max(0, 1 - yf(x))$

## Definition

In binary classification ( $\mathcal{Y} = \{-1, 1\}$ ), the **margin** of the function  $f$  for a pair  $(x, y)$  is:

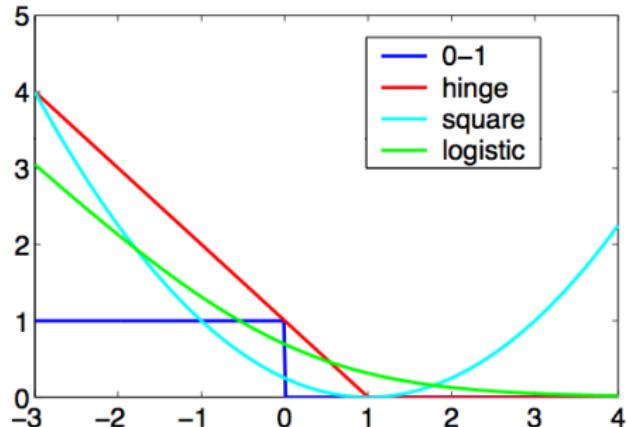
$$yf(x).$$

In all cases the loss is a decreasing function of the margin, i.e.,

$$\ell(f(x), y) = \varphi(yf(x)), \quad \text{with } \varphi \text{ non-increasing}$$

What about other similar loss functions?

# Loss function examples



Method	$\varphi(u)$
Logistic regression	$\log(1 + e^{-u})$
Support vector machine (1-SVM)	$\max(1 - u, 0)$
Support vector machine (2-SVM)	$\max(1 - u, 0)^2$
Boosting	$e^{-u}$

# Large-margin classifiers

## Definition

Given a non-increasing function  $\varphi : \mathbb{R} \rightarrow \mathbb{R}_+$ , a large-margin linear classifier is an algorithm that estimates a function  $f_\beta(x) = \beta^\top x$  by solving

$$\min_{\beta} \frac{1}{n} \sum_{i=1}^n \varphi(y_i f_\beta(x_i)) + \lambda \|\beta\|_2^2$$

Hence, ridge logistic regression and SVM are large-margin classifier, corresponding to  $\varphi(u) = \ln(1 + e^{-u})$  and  $\varphi(u) = \max(0, 1 - u)$ , respectively. Many more are possible.

## Questions:

- ① Can we solve the optimization problem for other  $\varphi$ 's?
- ② Is it a good idea to optimize this objective function, if at the end of the day we are interested in the  $\ell_{0/1}$  loss, i.e., learning models that make few errors?

# Solving large-margin classifiers

$$\min_{\beta} \frac{1}{n} \sum_{i=1}^n \varphi(y_i \beta^\top x_i) + \lambda \|\beta\|_2^2$$

- When  $\varphi$  is convex, this is a strictly convex function of  $\beta$
- It can then be solved numerically by generic or specific algorithms for **convex optimization**, e.g., Newton's or gradient method
- When  **$n$  is large**, stochastic optimization is particularly useful (at each step, only approximate the gradient with one or a batch of examples)

# A tiny bit of learning theory

## Assumptions and notations

- Let  $\mathbb{P}$  be an (unknown) distribution on  $\mathcal{X} \times \mathcal{Y}$ , and  $\eta(x) = \mathbb{P}(Y = 1 | X = x)$  a measurable version of the conditional distribution of  $Y$  given  $X$
- Assume the training set  $\mathcal{S}_n = (X_i, Y_i)_{i=1,\dots,n}$  are i.i.d. random variables according to  $\mathbb{P}$ .
- The **risk** of a classifier  $f : \mathcal{X} \rightarrow \mathbb{R}$  is  $R(f) = \mathbb{P}(\text{sign}(f(X)) \neq Y)$
- The **Bayes risk** is

$$R^* = \inf_{f \text{ measurable}} R(f)$$

which is attained for  $f^*(x) = \eta(x) - 1/2$

- The **empirical risk** of a classifier  $f : \mathcal{X} \rightarrow \mathbb{R}$  is

$$R^n(f) = \frac{1}{n} \sum_{i=1}^n \mathbf{1}(\text{sign}(f(X_i)) \neq Y_i)$$

## $\varphi$ -risk

- Let the **empirical  $\varphi$ -risk** be the empirical risk optimized by a large-margin classifier:

$$R_\varphi^n(f) = \frac{1}{n} \sum_{i=1}^n \varphi(Y_i f(X_i))$$

- It is the empirical version of the  **$\varphi$ -risk**

$$R_\varphi(f) = \mathbb{E}[\varphi(Y f(X))]$$

- Can we hope to have a small risk  $R(f)$  if we focus instead on the  $\varphi$ -risk  $R_\varphi(f)$ ?

# A small $\varphi$ -risk ensures a small 0/1 risk

## Theorem (?)

Let  $\varphi : \mathbb{R} \rightarrow \mathbb{R}_+$  be convex, non-increasing, differentiable at 0 with  $\varphi'(0) < 0$ . Let  $f : \mathcal{X} \rightarrow \mathbb{R}$  measurable such that

$$R_\varphi(f) = \min_{g \text{ measurable}} R_\varphi(g) = R_\varphi^*.$$

Then

$$R(f) = \min_{g \text{ measurable}} R(g) = R^*.$$

Remarks:

- This tells us that, if we know  $\mathbb{P}$ , then minimizing the  $\varphi$ -risk is a good idea even if our focus is on the classification error.
- The assumptions on  $\varphi$  can be relaxed; it works for the broader class of *classification-calibrated* loss functions (?).
- More generally, we can show that if  $R_\varphi(f) - R_\varphi^*$  is small, then  $R(f) - R^*$  is small too (?).

# A small $\varphi$ -risk ensures a small 0/1 risk

Proof sketch:

Condition on  $X = x$ :

$$R_\varphi(f | X = x) = \mathbb{E}[\varphi(Yf(X)) | X = x] = \eta(x)\varphi(f(x)) + (1 - \eta(x))\varphi(-f(x))$$

$$R_\varphi(-f | X = x) = \mathbb{E}[\varphi(-Yf(X)) | X = x] = \eta(x)\varphi(-f(x)) + (1 - \eta(x))\varphi(f(x))$$

Therefore:

$$R_\varphi(f | X = x) - R_\varphi(-f | X = x) = [2\eta(x) - 1] \times [\varphi(f(x)) - \varphi(-f(x))]$$

This must be a.s.  $\leq 0$  because  $R_\varphi(f) \leq R_\varphi(-f)$ , which implies:

- if  $\eta(x) > \frac{1}{2}$ ,  $\varphi(f(x)) \leq \varphi(-f(x)) \implies f(x) \geq 0$
- if  $\eta(x) < \frac{1}{2}$ ,  $\varphi(f(x)) \geq \varphi(-f(x)) \implies f(x) \leq 0$

These inequalities are in fact strict thanks to the assumptions we made on  $\varphi$  (*left as exercice*).  $\square$

# Empirical risk minimization (ERM)

To find a function with a small  $\varphi$ -risk, the following is a good candidate:

## Definition

The **ERM estimator** on a functional class  $\mathcal{F}$  is the solution (when it exists) of:

$$\hat{f}_n = \operatorname{argmin}_{f \in \mathcal{F}} R_\varphi^n(f).$$

# Empirical risk minimization (ERM)

To find a function with a small  $\varphi$ -risk, the following is a good candidate:

## Definition

The **ERM estimator** on a functional class  $\mathcal{F}$  is the solution (when it exists) of:

$$\hat{f}_n = \operatorname{argmin}_{f \in \mathcal{F}} R_\varphi^n(f).$$

## Questions:

- ① Is  $R_\varphi^n(f)$  a good estimate of the true risk  $R_\varphi(f)$ ?
- ② Is  $R_\varphi(\hat{f}_n)$  small?

# Class capacity

## Motivations

- The ERM principle gives a good solution if  $R_\varphi(\hat{f}_n)$  is similar to the minimum achievable risk  $\inf_{f \in \mathcal{F}} R_\varphi(f)$ .
- This can be ensured if  $\mathcal{F}$  is not “too large”.
- We need a measure of the “capacity” of  $\mathcal{F}$ .

## Definition: Rademacher complexity

The **Rademacher complexity** of a class of functions  $\mathcal{F}$  is:

$$\text{Rad}_n(\mathcal{F}) = \mathbb{E}_{X,\sigma} \left[ \sup_{f \in \mathcal{F}} \left| \frac{2}{n} \sum_{i=1}^n \sigma_i f(X_i) \right| \right],$$

where the expectation is over  $(X_i)_{i=1,\dots,n}$  and the independent uniform  $\{\pm 1\}$ -valued (Rademacher) random variables  $(\sigma_i)_{i=1,\dots,n}$ .

# Basic learning bounds

## Theorem

Suppose  $\varphi$  is **Lipschitz** with constant  $L_\varphi$ :

$$\forall u, u' \in \mathbb{R}, \quad |\varphi(u) - \varphi(u')| \leq L_\varphi |u - u'|.$$

Then the  $\varphi$ -risk of the ERM estimator satisfies (on average over the sampling of training set)

$$\underbrace{\mathbb{E}_{\mathcal{S}_n} R_\varphi(\hat{f}_n) - R_\varphi^*}_{\text{Excess } \varphi\text{-risk}} \leq \underbrace{4L_\varphi \text{Rad}_n(\mathcal{F})}_{\text{Estimation error}} + \underbrace{\inf_{f \in \mathcal{F}} R_\varphi(f) - R_\varphi^*}_{\text{Approximation error}}$$

This quantifies a trade-off between:

- $\mathcal{F}$  "large" = **overfitting** (approximation error small, estimation error large)
- $\mathcal{F}$  "small" = **underfitting** (estimation error small, approximation error large)

## ERM for bounded linear classifiers

Consider the set of linear functions  $f_\beta(x) = \beta^\top x$  where  $\beta$  is bounded:

$$\mathcal{F}_B = \{f_\beta : \|\beta\|_2 \leq B\}.$$

### Theorem

$$\text{Rad}_n(\mathcal{F}_B) \leq \frac{2B\sqrt{\mathbb{E}\|X\|_2^2}}{\sqrt{n}}.$$

## Proof (1/2)

$$\begin{aligned}\text{Rad}_n(\mathcal{F}_B) &= \mathbb{E}_{X,\sigma} \left[ \sup_{f \in \mathcal{F}_B} \left| \frac{2}{n} \sum_{i=1}^n \sigma_i f(X_i) \right| \right] \\ &= \mathbb{E}_{X,\sigma} \left[ \sup_{\|\beta\| \leq B} \left| \left\langle \beta, \frac{2}{n} \sum_{i=1}^n \sigma_i X_i \right\rangle \right| \right] \quad (\text{linearity}) \\ &= \mathbb{E}_{X,\sigma} \left[ B \left\| \frac{2}{n} \sum_{i=1}^n \sigma_i X_i \right\|_2 \right] \quad (\text{Cauchy-Schwarz}) \\ &= \frac{2B}{n} \mathbb{E}_{X,\sigma} \left[ \sqrt{\left\| \sum_{i=1}^n \sigma_i X_i \right\|_2^2} \right] \\ &\leq \frac{2B}{n} \sqrt{\mathbb{E}_{X,\sigma} \left[ \sum_{i,j=1}^n \sigma_i \sigma_j X_i^\top X_j \right]} \quad (\text{Jensen})\end{aligned}$$

## Proof (2/2)

But  $\mathbb{E}_\sigma [\sigma_i \sigma_j]$  is 1 if  $i = j$ , 0 otherwise. Therefore:

$$\begin{aligned}\text{Rad}_n(\mathcal{F}_B) &\leq \frac{2B}{n} \sqrt{\mathbb{E}_X \left[ \sum_{i,j=1}^n \mathbb{E}_\sigma [\sigma_i \sigma_j] X_i^\top X_j \right]} \\ &\leq \frac{2B}{n} \sqrt{\mathbb{E}_X \sum_{i=1}^n \|X_i\|_2^2} \\ &= \frac{2B \sqrt{\mathbb{E}_X \|X\|_2^2}}{\sqrt{n}}.\end{aligned}$$

□

# Basic learning bounds in RKHS balls

## Corollary

Suppose  $\|X\| \leq \kappa$  a.s. Then the ERM estimator in  $\mathcal{F}_B$  satisfies

$$\mathbb{E}R_\varphi(\hat{f}_n) - R_\varphi^* \leq \frac{8L_\varphi\kappa B}{\sqrt{n}} + \left[ \inf_{f \in \mathcal{F}_B} R_\varphi(f) - R_\varphi^* \right].$$

## Remarks

- $B$  controls the trade-off between approximation and estimation error
- The bound on expression error is independent of  $\mathcal{P}$  and decreases with  $n$
- The approximation error is harder to analyze in general
- In practice,  $B$  (or  $\lambda$ , next slide) is tuned by cross-validation

# ERM as penalized risk minimization

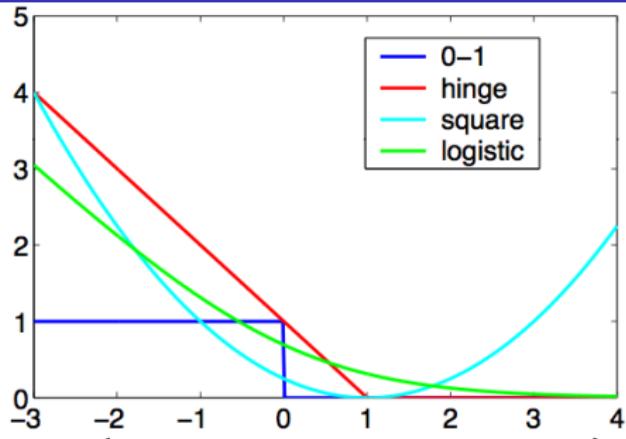
- ERM over  $\mathcal{F}_B$  solves the **constrained minimization problem**:

$$\begin{cases} \min_{\beta} \frac{1}{n} \sum_{i=1}^n \varphi(y_i f_\beta(x_i)) \\ \text{subject to } \|\beta\|_2 \leq B. \end{cases}$$

- To make this practical we assume that  $\varphi$  is **convex**.
- The problem is then a **convex problem** in  $\beta$  for which **strong duality holds**. In particular  $\beta$  solves the problem if and only if it solves for some dual parameter  $\lambda$  the **unconstrained problem**:

$$\min_{\beta \in \mathbb{R}^p} \left\{ \frac{1}{n} \sum_{i=1}^n \varphi(y_i f_\beta(x_i)) + \lambda \|\beta\|_2^2 \right\}.$$

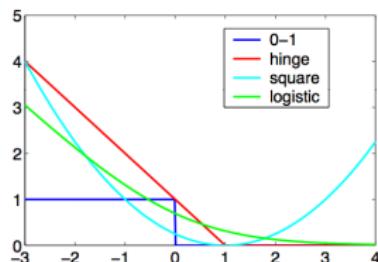
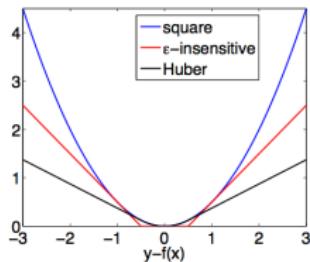
## Summary: large margin classifiers



$$\min_{f \in \mathcal{H}} \left\{ \frac{1}{n} \sum_{i=1}^n \varphi(y_i f(x_i)) + \lambda \|f\|_{\mathcal{H}}^2 \right\}$$

- $\varphi$  calibrated (e.g., decreasing,  $\varphi'(0) < 0$ )  $\implies$  good proxy for classification error
- $\varphi$  convex + representer theorem  $\implies$  efficient algorithms
- $\varphi$  smooth (Lipschitz) +  $\ell_2$  regularization  $\implies$  good learning ability

# Summary: $\ell_2$ -regularized linear methods



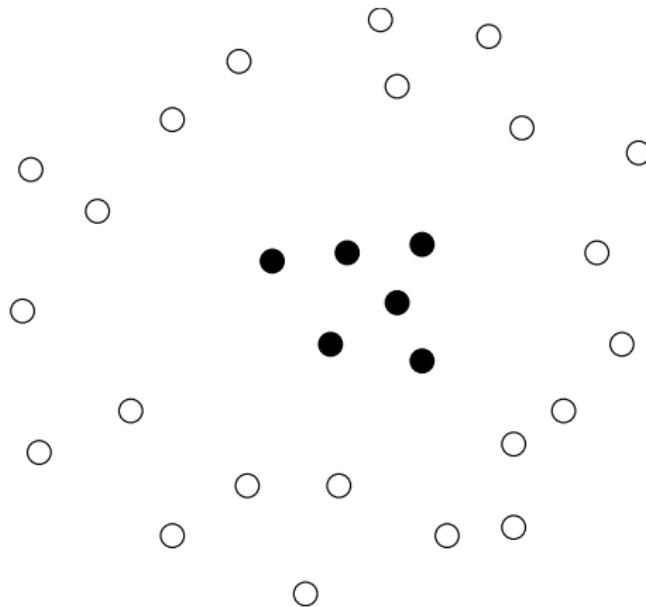
$$f_{\beta}(x) = \beta^T x, \quad \min_{\beta} \left\{ \frac{1}{n} \sum_{i=1}^n \ell(f_{\beta}(x_i), y_i) + \lambda \|\beta\|_2^2 \right\}$$

- Many popular methods for regression and classification are obtained by changing the loss function: ridge regression, logistic regression, SVM...
- Needs to solve numerically a convex optimization problem, well adapted to large datasets (stochastic gradient...)
- In practice, very similar performance between the different variants in general

# Outline

- 1 Learning in high dimension
- 2 Learning with  $\ell_2$  regularization
  - Ridge regression
  - Ridge logistic regression
  - Linear hard-margin SVM
  - Interlude: quick notes on constrained optimization
  - Back to hard-margin SVM
  - Soft-margin SVM
  - Large-margin classifiers
- 3 Learning with kernels
  - Kernel methods
  - Positive definite kernels and RKHS
  - Kernel examples
  - Multiple Kernel Learning (MKL)
- 4 Conclusion

# Motivation

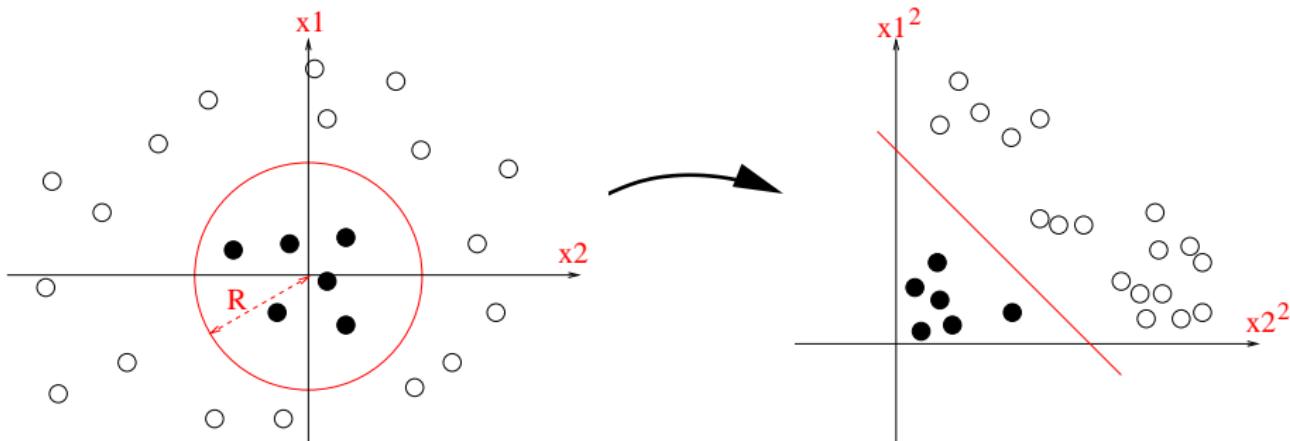


- Sometimes linear models are not interesting...
- Kernels will allow to solve nonlinear problems with linear methods!

# Outline

- 1 Learning in high dimension
- 2 Learning with  $\ell_2$  regularization
  - Ridge regression
  - Ridge logistic regression
  - Linear hard-margin SVM
  - Interlude: quick notes on constrained optimization
  - Back to hard-margin SVM
  - Soft-margin SVM
  - Large-margin classifiers
- 3 Learning with kernels
  - Kernel methods
    - Positive definite kernels and RKHS
    - Kernel examples
    - Multiple Kernel Learning (MKL)
- 4 Conclusion

"Linear" depends on the representation you choose



For  $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$  let  $\Phi(x) = \begin{pmatrix} x_1^2 \\ x_2^2 \end{pmatrix}$ . The decision function is:

$$f(x) = x_1^2 + x_2^2 - R^2 = \beta^\top \Phi(x) + b$$

with  $\beta = (1, 1)^\top$  and  $b = -R^2$

# Kernel = inner product in the feature space

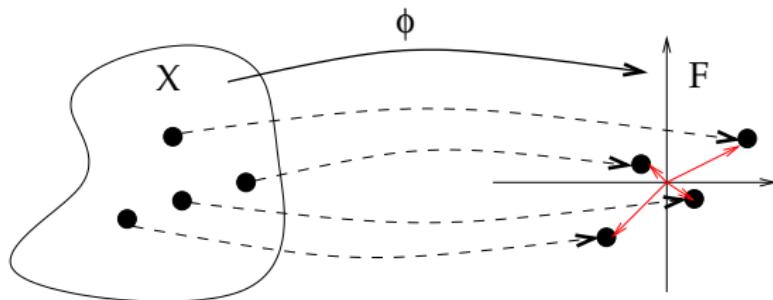
## Definition

For a given mapping

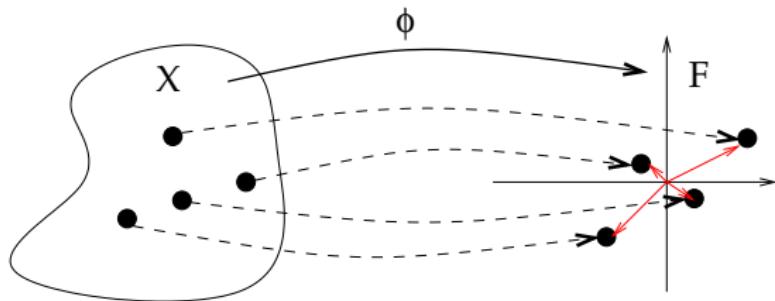
$$\Phi : \mathcal{X} \mapsto \mathcal{H}$$

from the space of data  $\mathcal{X}$  to some feature space  $\mathcal{H}$ , the **kernel** between two objects  $x$  and  $x'$  is the inner product of their images:

$$\forall x, x' \in \mathcal{X}, \quad K(x, x') = \Phi(x)^\top \Phi(x').$$



## Example

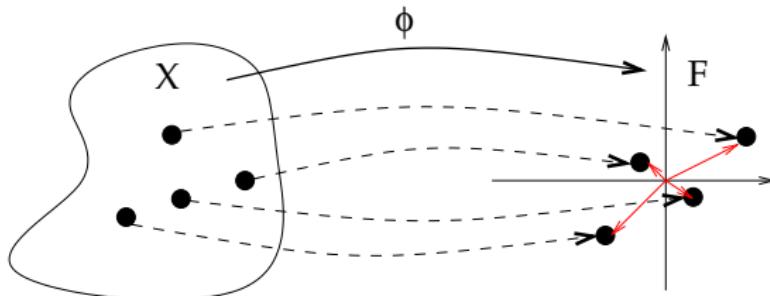


Let  $\mathcal{X} = \mathcal{H} = \mathbb{R}^2$  and for  $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$  let  $\Phi(x) = \begin{pmatrix} x_1^2 \\ x_2^2 \end{pmatrix}$

Then the kernel is:

$$K(x, x') = \Phi(x)^\top \Phi(x') = (x_1)^2 (x'_1)^2 + (x_2)^2 (x'_2)^2.$$

# The kernel tricks



## 2 tricks

- ① Many linear algorithms (in particular  $\ell_2$ -regularized methods) can be performed in the feature space of  $\Phi(x)$  without explicitly computing the images  $\Phi(x)$ , but instead by computing kernels  $K(x, x')$ .
- ② It is sometimes possible to easily compute kernels which correspond to complex large-dimensional feature spaces:  $K(x, x')$  is often much simpler to compute than  $\Phi(x)$  and  $\Phi(x')$

## Trick 1 illustration: SVM in the original space

- Train the SVM by maximizing

$$\max_{\alpha \in \mathbb{R}^n} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j ,$$

under the constraints:

$$\begin{cases} 0 \leq \alpha_i \leq C, & \text{for } i = 1, \dots, n \\ \sum_{i=1}^n \alpha_i y_i = 0. \end{cases}$$

- Predict with the decision function

$$f(x) = \sum_{i=1}^n \alpha_i y_i \mathbf{x}_i^\top \mathbf{x} + b^* .$$

## Trick 1 illustration: SVM in the feature space

- Train the SVM by maximizing

$$\max_{\alpha \in \mathbb{R}^n} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j \Phi(x_i)^\top \Phi(x_j),$$

under the constraints:

$$\begin{cases} 0 \leq \alpha_i \leq C, & \text{for } i = 1, \dots, n \\ \sum_{i=1}^n \alpha_i y_i = 0. \end{cases}$$

- Predict with the decision function

$$f(x) = \sum_{i=1}^n \alpha_i y_i \Phi(x_i)^\top \Phi(x) + b^*.$$

## Trick 1 illustration: SVM in the feature space with a kernel

- Train the SVM by maximizing

$$\max_{\alpha \in \mathbb{R}^n} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j K(x_i, x_j) ,$$

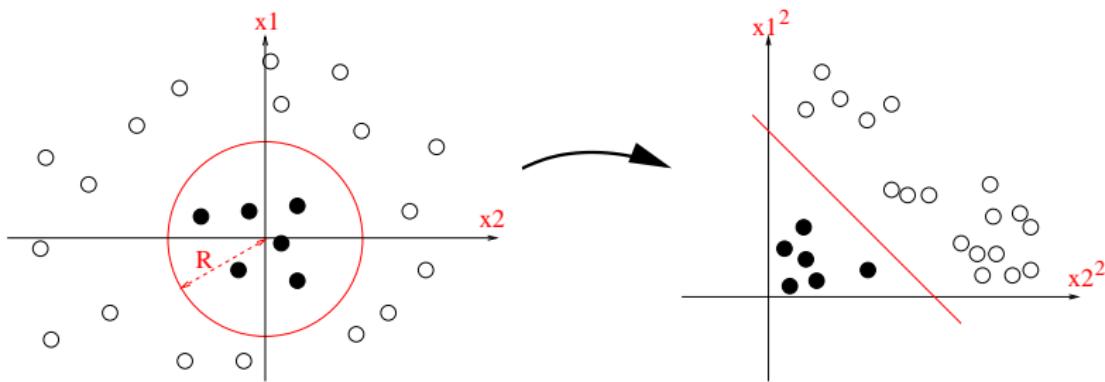
under the constraints:

$$\begin{cases} 0 \leq \alpha_i \leq C, & \text{for } i = 1, \dots, n \\ \sum_{i=1}^n \alpha_i y_i = 0. \end{cases}$$

- Predict with the decision function

$$f(x) = \sum_{i=1}^n \alpha_i y_i K(x_i, x) + b^* .$$

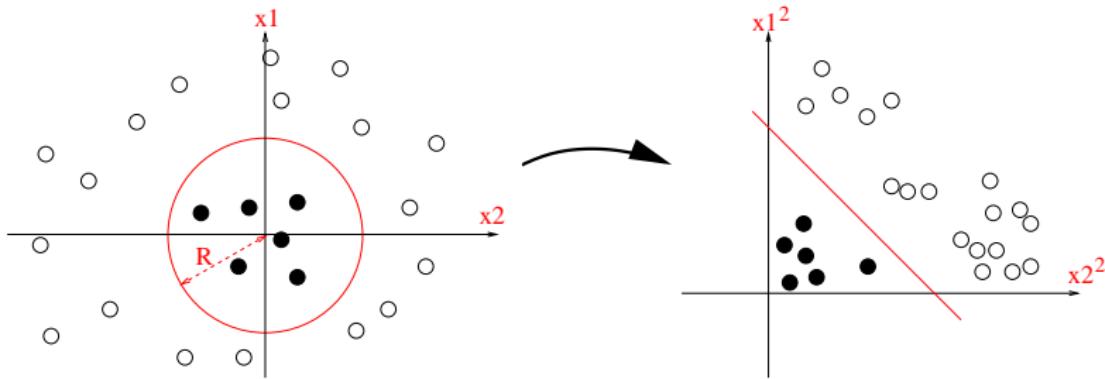
## Trick 2 illustration: polynomial kernel



For  $x = (x_1, x_2)^\top \in \mathbb{R}^2$ , let  $\Phi(x) = (x_1^2, \sqrt{2}x_1x_2, x_2^2) \in \mathbb{R}^3$ :

$$\begin{aligned} K(x, x') &= x_1^2 x'^2 + 2x_1 x_2 x'_1 x'_2 + x_2^2 x'^2 \\ &= (x_1 x'_1 + x_2 x'_2)^2 \\ &= (x^\top x')^2. \end{aligned}$$

## Trick 2 illustration: polynomial kernel



More generally, for  $x, x' \in \mathbb{R}^p$ ,

$$K(x, x') = (x^\top x' + 1)^d$$

is an inner product in a feature space of all monomials of degree up to  $d$   
*(left as exercise.)*

# Combining tricks: learn a polynomial discrimination rule with SVM

- Train the SVM by maximizing

$$\max_{\alpha \in \mathbb{R}^n} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j (x_i^\top x_j + 1)^d,$$

under the constraints:

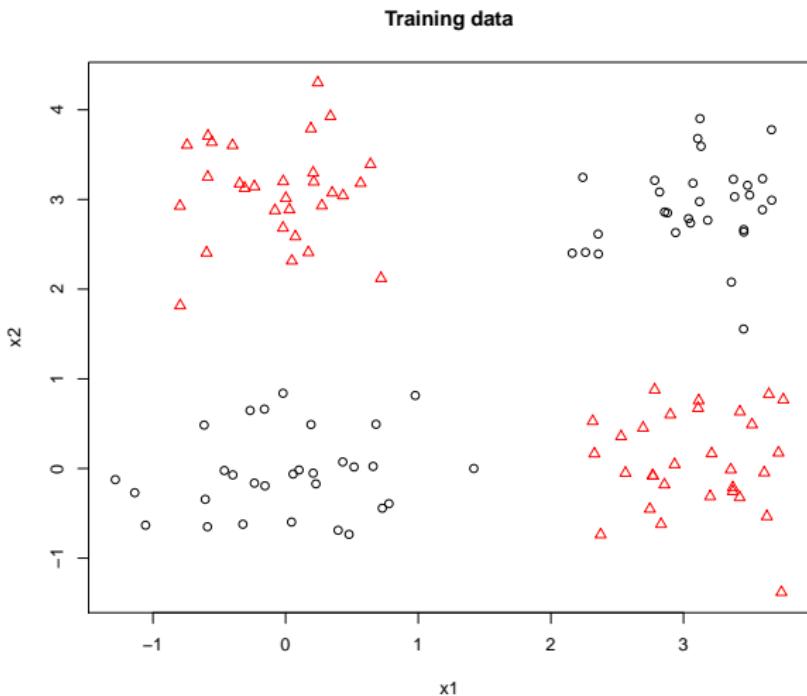
$$\begin{cases} 0 \leq \alpha_i \leq C, & \text{for } i = 1, \dots, n \\ \sum_{i=1}^n \alpha_i y_i = 0. \end{cases}$$

- Predict with the decision function

$$f(x) = \sum_{i=1}^n \alpha_i y_i (x_i^\top x + 1)^d + b^*.$$

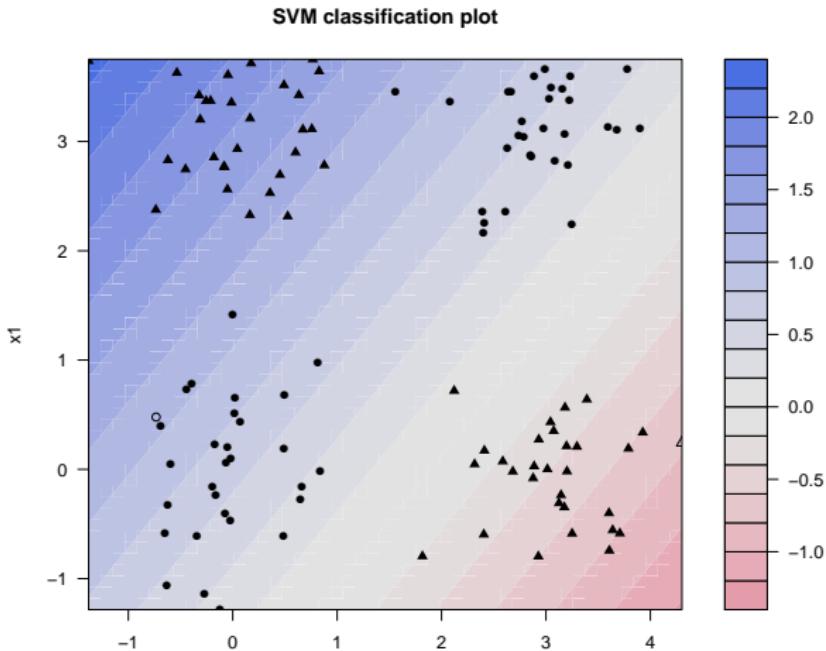
## Illustration: toy nonlinear problem

```
> plot(x,col=ifelse(y>0,1,2),pch=ifelse(y>0,1,2))
```



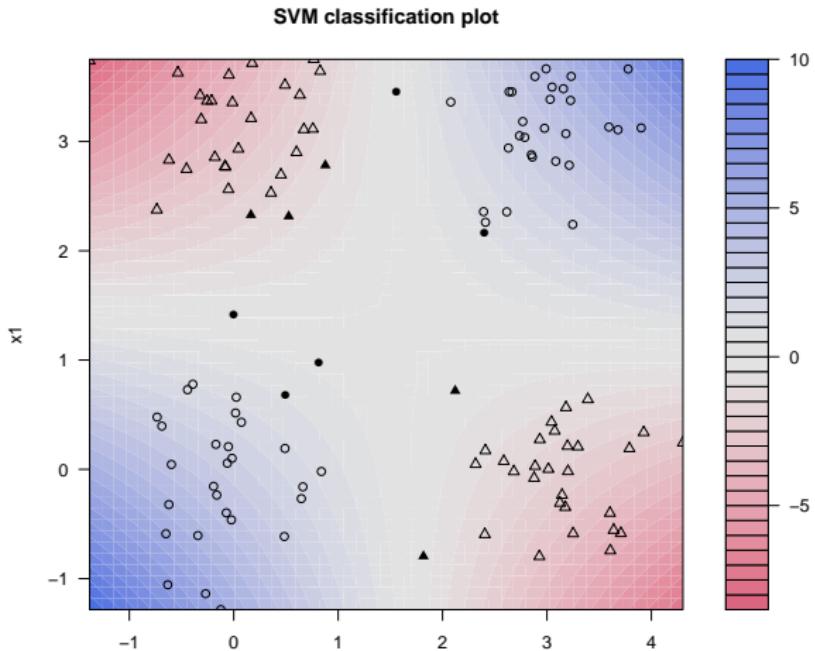
## Illustration: toy nonlinear problem, linear SVM

```
> library(kernlab)  
> svp <- ksvm(x,y,type="C-svc",kernel='vanilladot')  
> plot(svp,data=x)
```



## Illustration: toy nonlinear problem, polynomial SVM

```
> svp <- ksvm(x,y,type="C-svc", ...
                 kernel=polydot(degree=2))
> plot(svp,data=x)
```



## More generally: trick 1 for $\ell_2$ -regularized linear models

### Representer theorem

Let  $f_\beta(x) = \beta^\top \Phi(x)$ . Then any solution  $\hat{f}_\beta$  of

$$\min_{\beta} \frac{1}{n} \sum_{i=1}^n \ell(f_\beta(x_i), y_i) + \lambda \|\beta\|_2^2$$

can be expanded as

$$\hat{f}_\beta(x) = \sum_{i=1}^n \alpha_i K(x_i, x),$$

where  $\alpha \in \mathbb{R}^n$  is a solution of:

$$\min_{\alpha \in \mathbb{R}^n} \frac{1}{n} \sum_{i=1}^n \ell \left( \sum_{j=1}^n \alpha_j K(x_i, x_j), y_i \right) + \lambda \sum_{i,j=1}^n \alpha_i \alpha_j K(x_i, x_j).$$

## Representer theorem: proof

- For any  $\beta \in \mathbb{R}^p$ , decompose  $\beta = \beta_S + \beta_{\perp}$  where  $\beta_S \in \text{span}(\Phi(x_1), \dots, \Phi(x_n))$  and  $\beta_{\perp}$  is orthogonal to it.
- On any point  $x_i$  of the training set, we have:

$$f_{\beta}(x_i) = \beta^T \Phi(x_i) = \beta_S^T \Phi(x_i) + \beta_{\perp}^T \Phi(x_i) = \beta_S^T \Phi(x_i) = f_{\beta_S}(x_i).$$

- On the other hand, we have  $\|\beta\|_2^2 = \|\beta_S\|_2^2 + \|\beta_{\perp}\|_2^2 \geq \|\beta_S\|_2^2$ , with strict inequality if  $\beta_{\perp} \neq 0$ .
- Consequently,  $\beta_S$  is always as good as  $\beta$  in terms of objective function, and strictly better if  $\beta_{\perp} \neq 0$ . This implies that at any minimum,  $\beta_{\perp} = 0$  and therefore  $\beta = \beta_S = \sum_{i=1}^n \alpha_i \Phi(x_i)$  for some  $\alpha \in \mathbb{R}^n$ .
- We then just replace  $\beta$  by this expression in the objective function, noting that

$$\|\beta\|_2^2 = \left\| \sum_{i=1}^n \alpha_i \Phi(x_i) \right\|_2^2 = \sum_{i,j=1}^n \alpha_i \alpha_j \Phi(x_i)^T \Phi(x_j) = \sum_{i,j=1}^n \alpha_i \alpha_j K(x_i, x_j).$$

## Example: kernel ridge regression

- Let  $\Phi : \mathcal{X} \rightarrow \mathbb{R}^P$  be a feature mapping from the space of data to a Euclidean or Hilbert space.
- Let  $f_\beta(x) = \beta^\top \Phi(x)$  and  $K$  the corresponding kernel.
- By the representer theorem, any solution of:

$$\hat{f} = \arg \min_{f_\beta} \frac{1}{n} \sum_{i=1}^n (y_i - f_\beta(x_i))^2 + \lambda \|\beta\|_2^2$$

can be expanded as:

$$\hat{f} = \sum_{i=1}^n \alpha_i K(x_i, x).$$

## Example: kernel ridge regression

- Let  $Y = (y_1, \dots, y_n)^\top \in \mathbb{R}^n$  the vector of response variables.
- Let  $\alpha = (\alpha_1, \dots, \alpha_n)^\top \in \mathbb{R}^n$  the unknown coefficients.
- Let  $K$  be the  $n \times n$  Gram matrix:  $K_{i,j} = K(x_i, x_j)$ .
- We can then write in matrix form:

$$(\hat{f}(x_1), \dots, \hat{f}(x_n))^\top = K\alpha,$$

- Moreover,

$$\|\beta\|_2^2 = \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j K(x_i, x_j) = \alpha^\top K \alpha.$$

## Example: kernel ridge regression

- The problem is therefore equivalent to:

$$\arg \min_{\alpha \in \mathbb{R}^n} \frac{1}{n} (K\alpha - Y)^\top (K\alpha - Y) + \lambda \alpha^\top K \alpha.$$

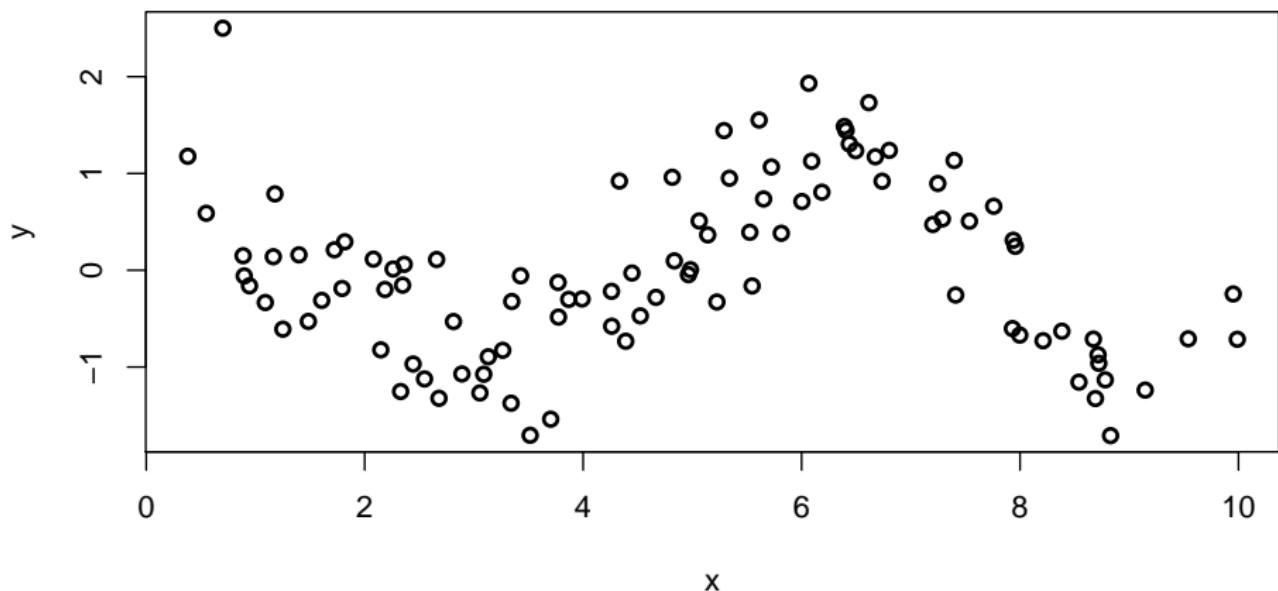
- This is a convex and differentiable function of  $\alpha$ . Its minimum can therefore be found by setting the gradient in  $\alpha$  to zero:

$$\begin{aligned} 0 &= \frac{2}{n} K (K\alpha - Y) + 2\lambda K \alpha \\ &= K [(K + \lambda n I) \alpha - Y] \end{aligned}$$

- For  $\lambda > 0$ ,  $K + \lambda n I$  is invertible (because  $K$  is positive semidefinite) so one solution is to take:

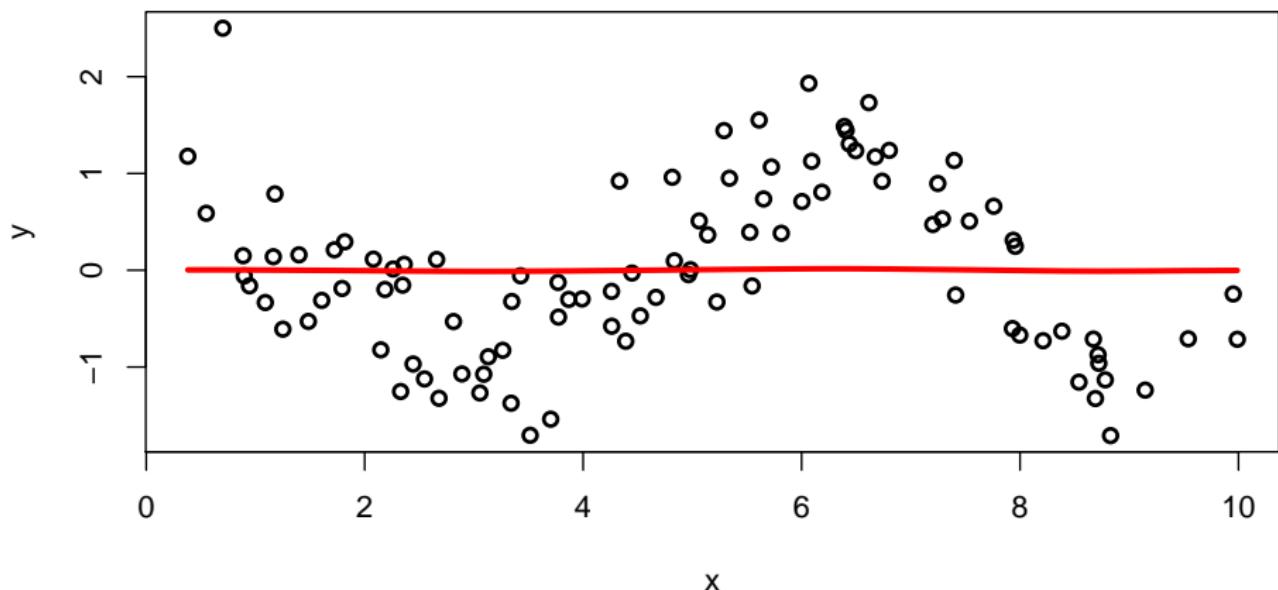
$$\alpha = (K + \lambda n I)^{-1} Y.$$

## Example (KRR with Gaussian RBF kernel)



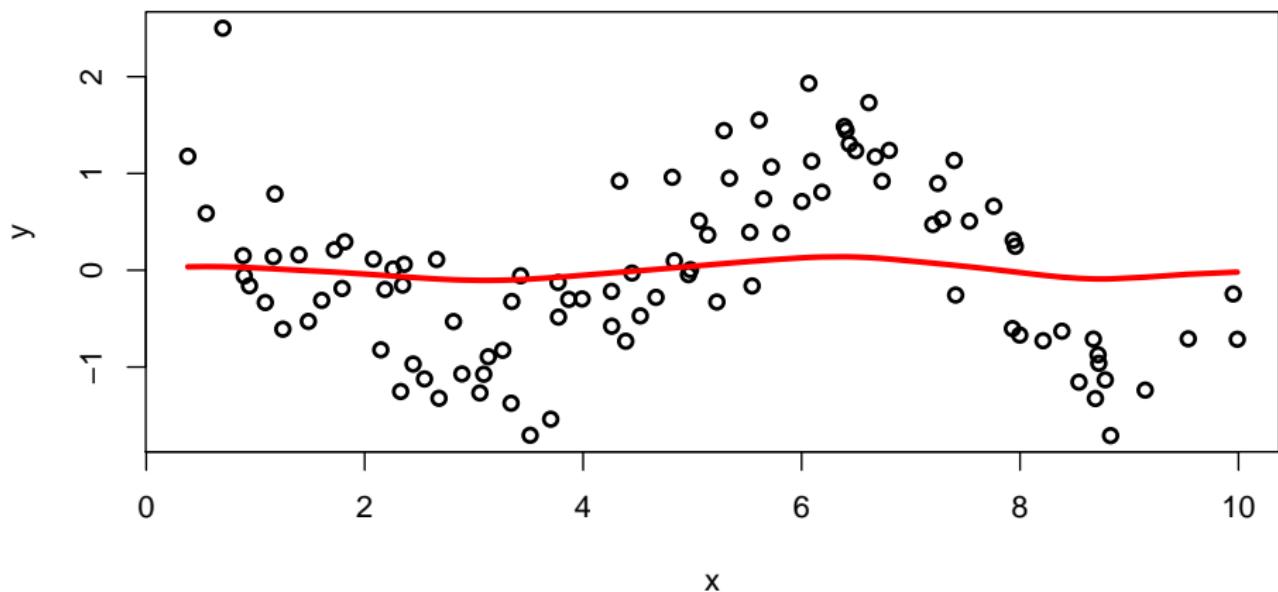
# Example (KRR with Gaussian RBF kernel)

**lambda = 1000**



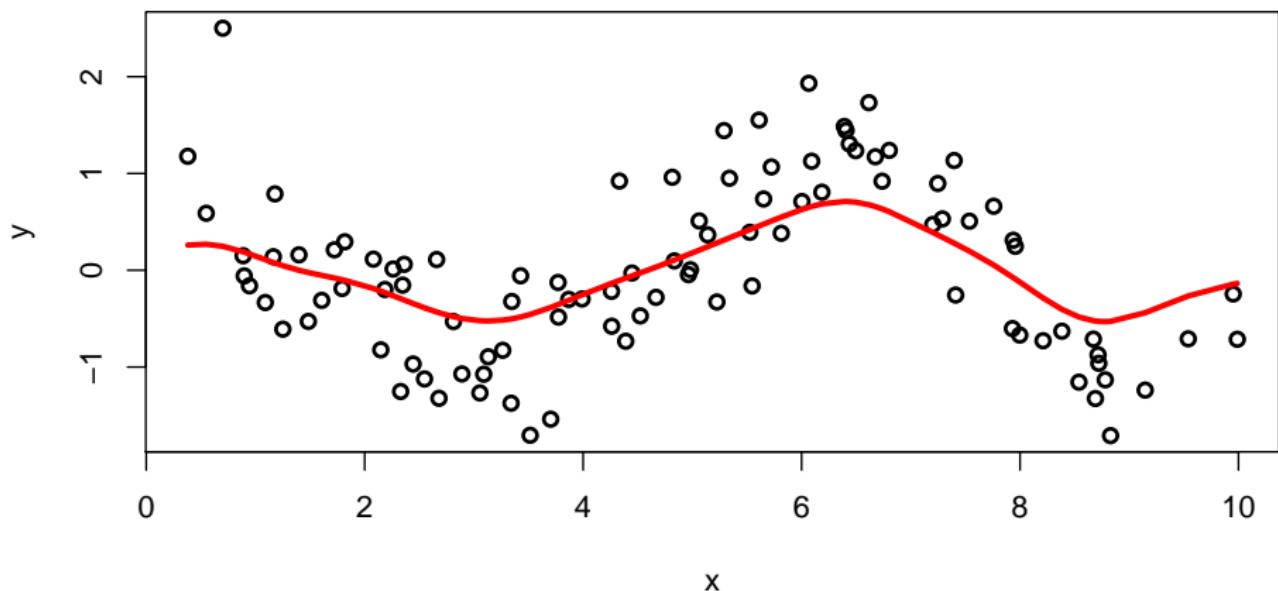
# Example (KRR with Gaussian RBF kernel)

**lambda = 100**



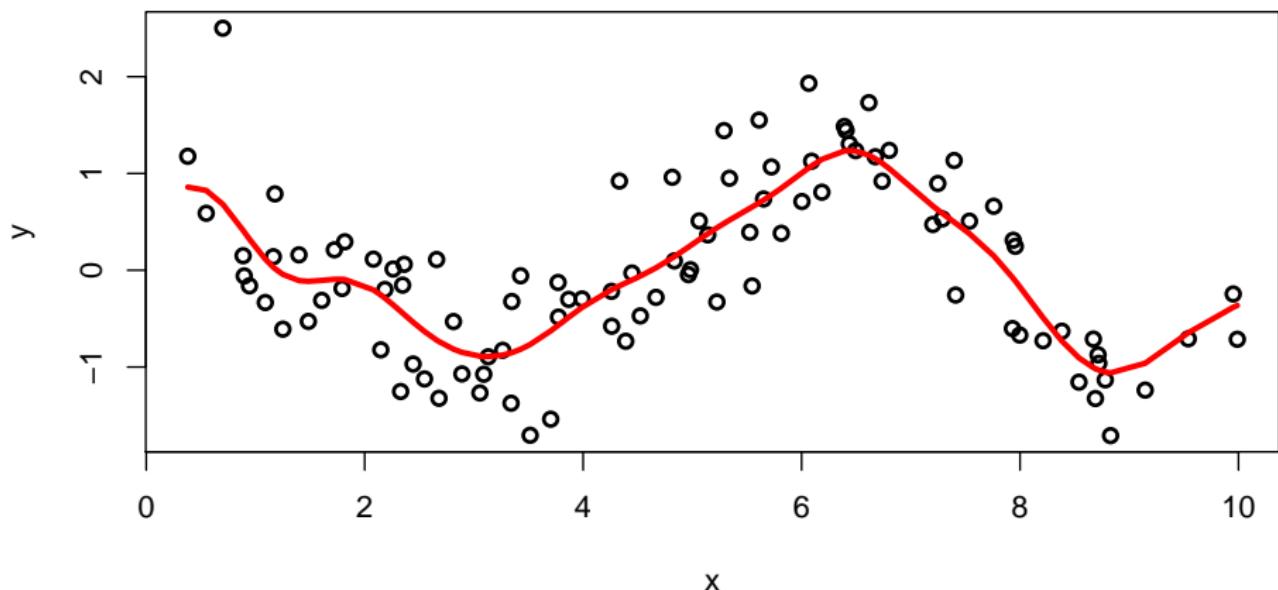
# Example (KRR with Gaussian RBF kernel)

**lambda = 10**



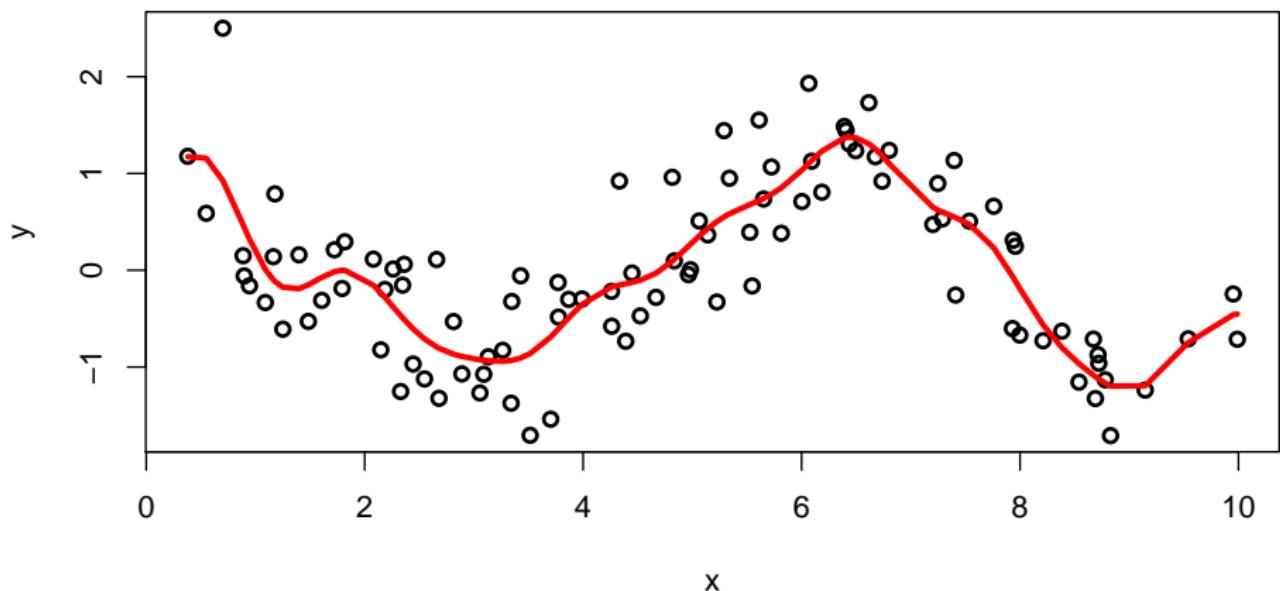
# Example (KRR with Gaussian RBF kernel)

**lambda = 1**



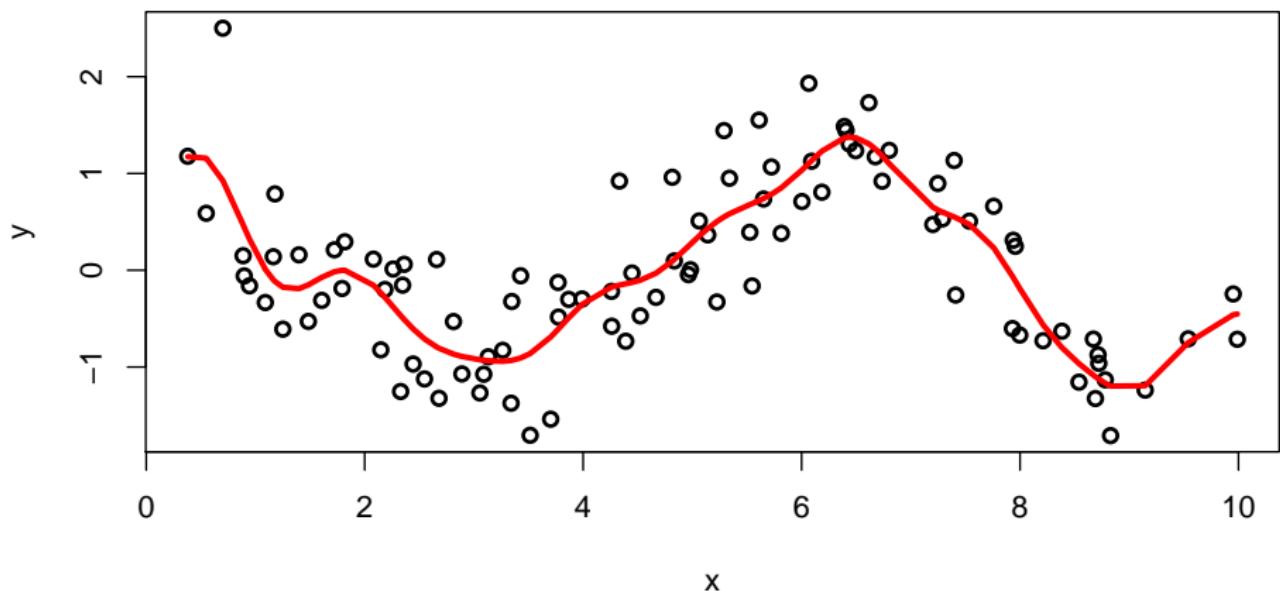
# Example (KRR with Gaussian RBF kernel)

**lambda = 0.1**



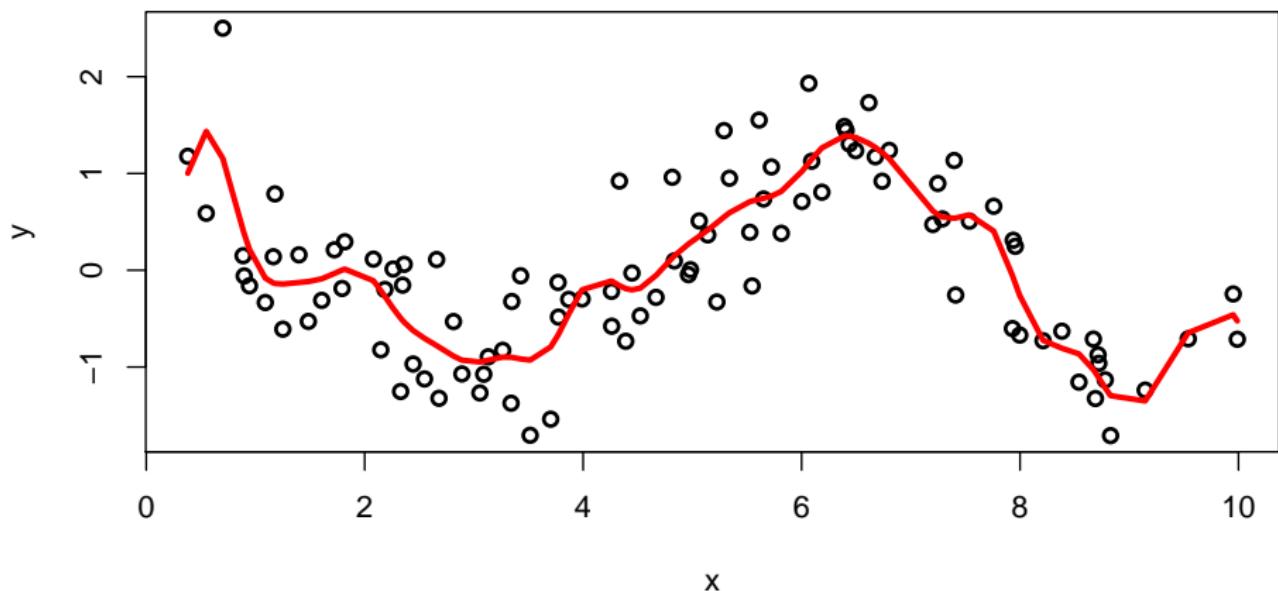
# Example (KRR with Gaussian RBF kernel)

**lambda = 0.01**



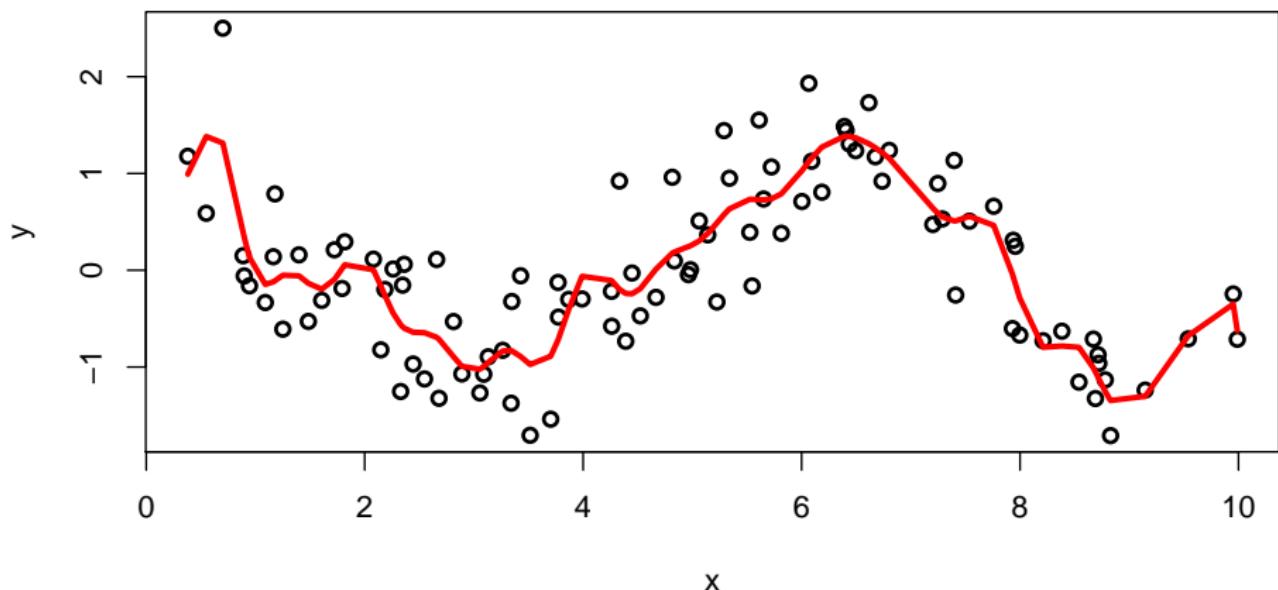
# Example (KRR with Gaussian RBF kernel)

**lambda = 0.001**



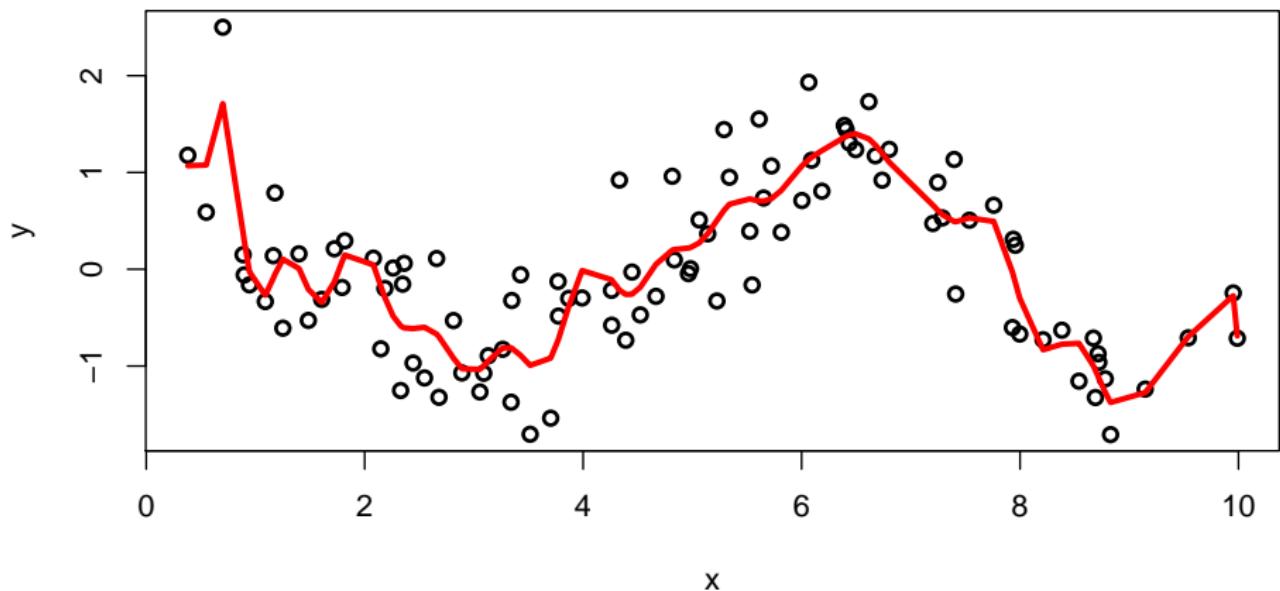
# Example (KRR with Gaussian RBF kernel)

**lambda = 0.0001**



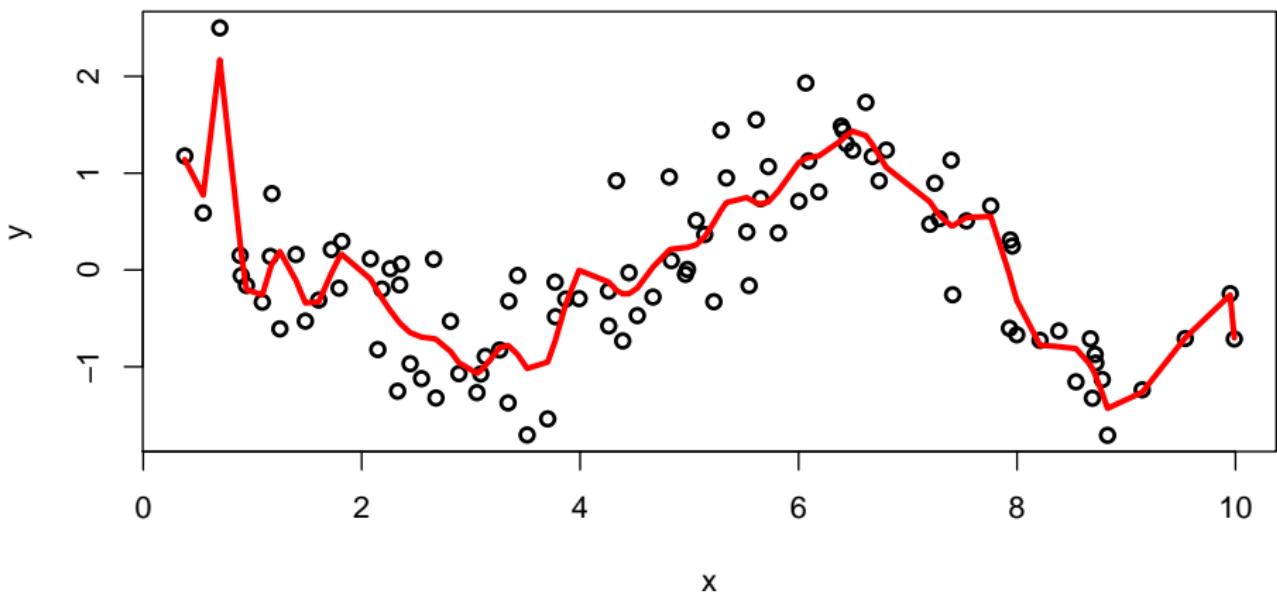
# Example (KRR with Gaussian RBF kernel)

**lambda = 0.00001**



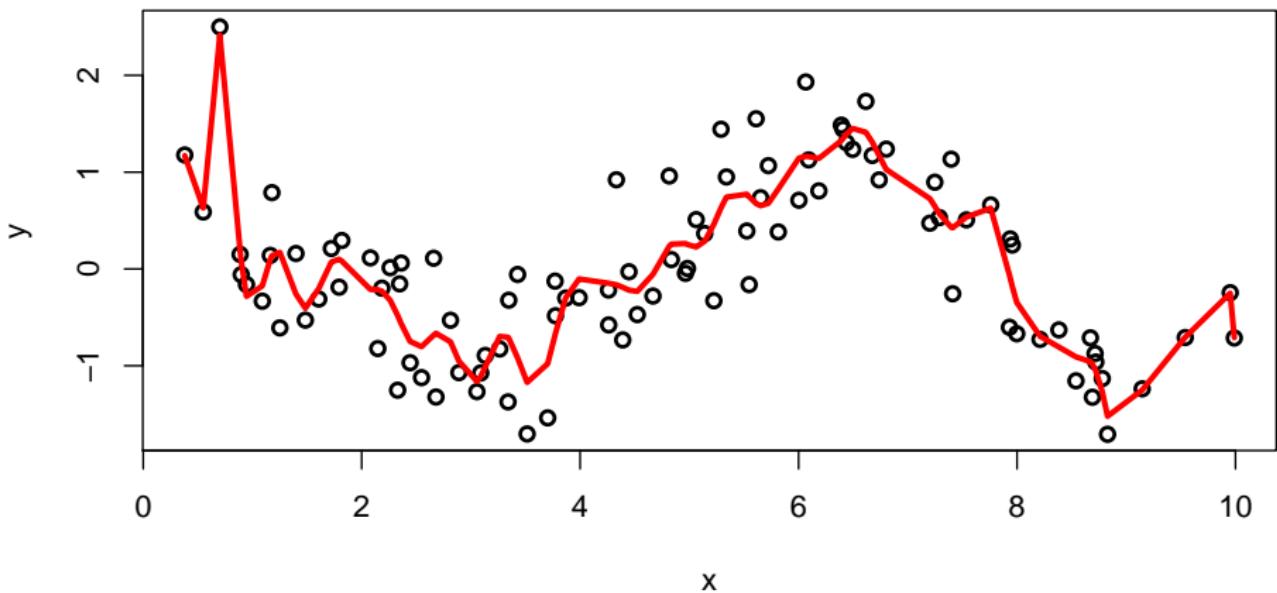
# Example (KRR with Gaussian RBF kernel)

**lambda = 0.000001**



# Example (KRR with Gaussian RBF kernel)

**lambda = 0.0000001**



## Remark: uniqueness of the solution

Let us find *all*  $\alpha$ 's that solve

$$K [(K + \lambda nI) \alpha - Y] = 0$$

- $K$  being a symmetric matrix, it can be diagonalized in an orthonormal basis and  $\text{Ker}(K) \perp \text{Im}(K)$ .
- In this basis we see that  $(K + \lambda nI)^{-1}$  leaves  $\text{Im}(K)$  and  $\text{Ker}(K)$  invariant.
- The problem is therefore equivalent to:

$$\begin{aligned} & (K + \lambda nI) \alpha - Y \in \text{Ker}(K) \\ \Leftrightarrow & \alpha - (K + \lambda nI)^{-1} Y \in \text{Ker}(K) \\ \Leftrightarrow & \alpha = (K + \lambda nI)^{-1} Y + \epsilon, \text{ with } K\epsilon = 0. \end{aligned}$$

- However, if  $\alpha' = \alpha + \epsilon$  with  $K\epsilon = 0$ , then:

$$\|\beta - \beta'\|_2^2 = (\alpha - \alpha')^\top K (\alpha - \alpha') = 0,$$

therefore  $\beta = \beta'$ . KRR has a unique solution  $\beta$ , which can possibly be expressed by several  $\alpha$ 's if  $K$  is singular.

## Comparison with "standard" ridge regression

- Let  $X$  the  $n \times p$  data matrix,  $K = XX^\top$  the kernel Gram matrix.
- In "standard" ridge regression, we have  $\hat{f}(x) = \hat{\beta}^\top x$  with

$$\hat{\beta} = (X^\top X + n\lambda I)^{-1} X^\top Y.$$

- In "kernel" ridge regression, we have  $\tilde{f}(x) = \sum_{i=1}^n \alpha_i x_i^\top x = \tilde{\beta}^\top x$  with

$$\tilde{\beta} = \sum_{i=1}^n \alpha_i x_i = X^\top \alpha = X^\top (X X^\top + \lambda n I)^{-1} Y.$$

- Oups... which one is correct?

## Comparison with "standard" ridge regression

### Matrix inversion lemma

For any matrices  $B$  and  $C$ , and  $\gamma > 0$  the following holds (when it makes sense):

$$B(CB + \gamma I)^{-1} = (BC + \gamma I)^{-1}B$$

We deduce that (of course...):

$$\hat{\beta} = \underbrace{(X^\top X + n\lambda I)^{-1}}_{p \times p} X^\top Y = X^\top \underbrace{(XX^\top + \lambda nI)^{-1}}_{n \times n} Y = \tilde{\beta}$$

Computationally, inverting the matrix is the expensive part, which suggest to implement:

- KRR when  $p > n$  (high dimension)
- RR when  $p < n$  (many points)

## Generalization

- We learn the function  $f(x) = \sum_{i=1}^n \alpha_i K(x_i, x)$  by solving in  $\alpha$  the following optimization problem, with adequate loss function  $\ell$ :

$$\min_{\alpha \in \mathbb{R}^n} \frac{1}{n} \sum_{i=1}^n \ell \left( \sum_{j=1}^n \alpha_j K(x_i, x_j), y_i \right) + \lambda \sum_{i,j=1}^n \alpha_i \alpha_j K(x_i, x_j).$$

- No explicit solution, but **convex** optimization problem
- Note that the **dimension** of the problem is now  $n$  instead of  $p$  (useful when  $n < p$ )

# The case of SVM

- Soft-margin SVM with a kernel solves:

$$\min_{\alpha \in \mathbb{R}^n, b \in \mathbb{R}} \left\{ \sum_{i=1}^n \ell_{\text{hinge}} \left( \sum_{j=1}^n \alpha_j K(x_i, x_j), y_i \right) + \lambda \sum_{i,j=1}^n \alpha_i \alpha_j K(x_i, x_j) \right\}.$$

- By Lagrange duality we saw that this is equivalent to

$$\max_{\alpha \in \mathbb{R}^n} L(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j K(x_i, x_j) + b,$$

under the constraints:

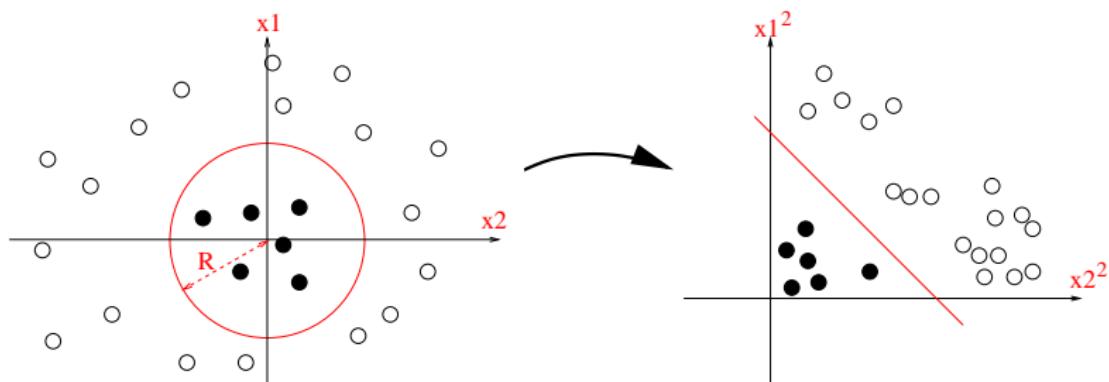
$$\begin{cases} 0 \leq \alpha_i \leq C, & \text{for } i = 1, \dots, n \\ \sum_{i=1}^n \alpha_i y_i = 0. \end{cases}$$

- This is not a surprise, both problems are also dual to each other (exercise).

# Outline

- 1 Learning in high dimension
- 2 Learning with  $\ell_2$  regularization
  - Ridge regression
  - Ridge logistic regression
  - Linear hard-margin SVM
  - Interlude: quick notes on constrained optimization
  - Back to hard-margin SVM
  - Soft-margin SVM
  - Large-margin classifiers
- 3 Learning with kernels
  - Kernel methods
  - Positive definite kernels and RKHS
  - Kernel examples
  - Multiple Kernel Learning (MKL)
- 4 Conclusion

Remember: polynomial kernel



$$\forall x, x' \in \mathbb{R}^p, \quad K(x, x') = (x^\top x' + 1)^d$$

is an inner product in a feature space of all monomials of degree up to  $d$

# Which functions $K(x, x')$ are kernels?

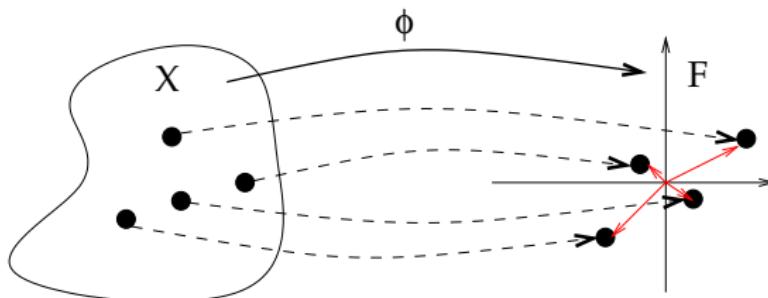
## Definition

A function  $K(x, x')$  defined on a set  $\mathcal{X}$  is a **kernel** if and only if there exists a features space (Hilbert space)  $\mathcal{H}$  and a mapping

$$\Phi : \mathcal{X} \mapsto \mathcal{H},$$

such that, for any  $x, x'$  in  $\mathcal{X}$ :

$$K(x, x') = \langle \Phi(x), \Phi(x') \rangle_{\mathcal{H}}.$$



## Reminder ...

- An **inner product** on an  $\mathbb{R}$ -vector space  $\mathcal{H}$  is a mapping  $(f, g) \mapsto \langle f, g \rangle_{\mathcal{H}}$  from  $\mathcal{H}^2$  to  $\mathbb{R}$  that is **bilinear, symmetric** and such that  $\langle f, f \rangle > 0$  for all  $f \in \mathcal{H} \setminus \{0\}$ .
- A vector space endowed with an inner product is called **pre-Hilbert**. It is endowed with a norm defined by the inner product as
$$\|f\|_{\mathcal{H}} = \langle f, f \rangle_{\mathcal{H}}^{\frac{1}{2}}.$$
- A **Hilbert space** is a pre-Hilbert space **complete** for the norm defined by the inner product.

## Kernel examples

- **Polynomial** (on  $\mathbb{R}^d$ ):

$$K(x, x') = (x \cdot x' + 1)^d$$

- **Gaussian radial basis function (RBF)** (on  $\mathbb{R}^d$ )

$$K(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right)$$

- **Laplace kernel** (on  $\mathbb{R}$ )

$$K(x, x') = \exp(-\gamma|x - x'|)$$

- **Min kernel** (on  $\mathbb{R}_+$ )

$$K(x, x') = \min(x, x')$$

## Example: SVM with a Gaussian kernel

- Training:

$$\min_{\alpha \in \mathbb{R}^n} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j \exp\left(-\frac{\|\vec{x}_i - \vec{x}_j\|^2}{2\sigma^2}\right)$$

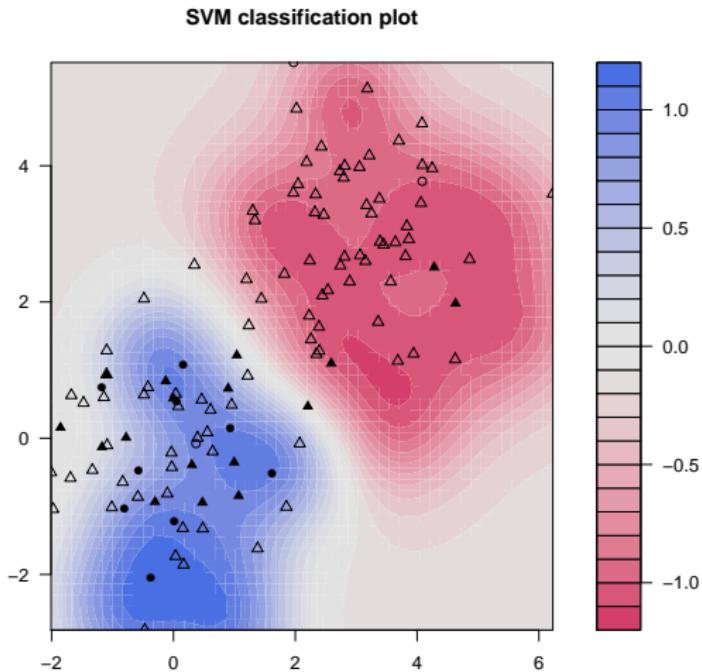
$$\text{s.t. } 0 \leq \alpha_i \leq C, \quad \text{and} \quad \sum_{i=1}^n \alpha_i y_i = 0.$$

- Prediction

$$f(\vec{x}) = \sum_{i=1}^n \alpha_i \exp\left(-\frac{\|\vec{x} - \vec{x}_i\|^2}{2\sigma^2}\right)$$

# Example: SVM with a Gaussian kernel

$$f(\vec{x}) = \sum_{i=1}^n \alpha_i \exp\left(-\frac{\|\vec{x} - \vec{x}_i\|^2}{2\sigma^2}\right)$$



# Positive Definite (p.d.) functions

## Definition

A **positive definite (p.d.) function** on the set  $\mathcal{X}$  is a function  $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$  **symmetric**:

$$\forall (x, x') \in \mathcal{X}^2, \quad K(x, x') = K(x', x),$$

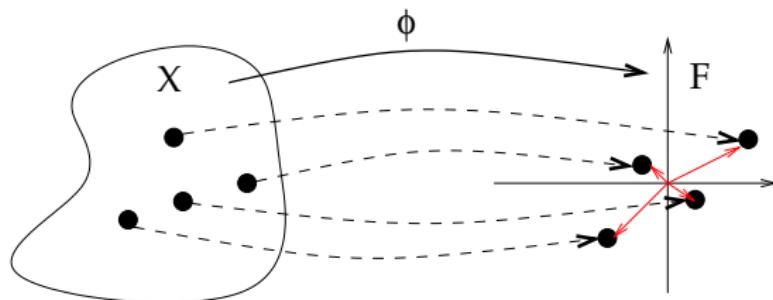
and which satisfies, for all  $N \in \mathbb{N}$ ,  $(x_1, x_2, \dots, x_N) \in \mathcal{X}^N$  et  $(a_1, a_2, \dots, a_N) \in \mathbb{R}^N$ :

$$\sum_{i=1}^N \sum_{j=1}^N a_i a_j K(x_i, x_j) \geq 0.$$

# Kernels are p.d. functions

Theorem (Aronszajn, 1950)

*K is a kernel if and only if it is a positive definite function.*



Proof: kernel  $\implies$  p.d. (easy)

Let

$$K(x, x') = \langle \Phi(x), \Phi(x') \rangle_{\mathcal{H}}$$

be a kernel. It is p.d. because:

- $K(x, x') = \langle \Phi(x), \Phi(x') \rangle_{\mathcal{H}} = \langle \Phi(x'), \Phi(x) \rangle_{\mathcal{H}} = K(x', x)$  ,
- $\sum_{i=1}^N \sum_{j=1}^N a_i a_j \langle \Phi(x_i), \Phi(x_j) \rangle_{\mathcal{H}} = \| \sum_{i=1}^N a_i \Phi(x_i) \|_{\mathcal{H}}^2 \geq 0$  .

Proof: p.d.  $\implies$  kernel when  $\mathcal{X}$  is finite

- Suppose  $\mathcal{X} = \{x_1, x_2, \dots, x_N\}$  is finite of size  $N$ .
- Any p.d. kernel  $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$  is entirely defined by the  $N \times N$  symmetric positive semidefinite matrix  $[K]_{ij} := K(x_i, x_j)$ .
- It can therefore be diagonalized on an orthonormal basis of eigenvectors  $(u_1, u_2, \dots, u_N)$ , with non-negative eigenvalues  $0 \leq \lambda_1 \leq \dots \leq \lambda_N$ , i.e.,

$$K(x_i, x_j) = \left[ \sum_{l=1}^N \lambda_l u_l u_l^\top \right]_{ij} = \sum_{l=1}^N \lambda_l u_l(i) u_l(j) = \langle \Phi(x_i), \Phi(x_j) \rangle_{\mathbb{R}^N},$$

with

$$\Phi(x_i) = \begin{pmatrix} \sqrt{\lambda_1} u_1(i) \\ \vdots \\ \sqrt{\lambda_N} u_N(i) \end{pmatrix}. \quad \square$$

Proof: p.d.  $\implies$  kernel in the general case

- Mercer (1909) for  $\mathcal{X} = [a, b] \subset \mathbb{R}$  (more generally  $\mathcal{X}$  compact) and  $K$  continuous (the so-called **Mercer kernels**).
- Kolmogorov (1941) for  $\mathcal{X}$  countable.
- Aronszajn (1944, 1950) for the general case, using the theory of RKHS.

## Definition

Let  $\mathcal{X}$  be a set and  $\mathcal{H} \subset \mathbb{R}^{\mathcal{X}}$  be a class of functions forming a (real) Hilbert space with inner product  $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ . The function  $K : \mathcal{X}^2 \mapsto \mathbb{R}$  is called a reproducing kernel (r.k.) of  $\mathcal{H}$  if

- ①  $\mathcal{H}$  contains all functions of the form

$$\forall x \in \mathcal{X}, \quad K_x : t \mapsto K(x, t).$$

- ② For every  $x \in \mathcal{X}$  and  $f \in \mathcal{H}$  the reproducing property holds:

$$f(x) = \langle f, K_x \rangle_{\mathcal{H}}.$$

If a r.k. exists, then  $\mathcal{H}$  is called a reproducing kernel Hilbert space (RKHS).

# An equivalent definition of RKHS

## Theorem

The Hilbert space  $\mathcal{H} \subset \mathbb{R}^{\mathcal{X}}$  is a RKHS if and only if for any  $x \in \mathcal{X}$ , the mapping:

$$\begin{aligned} F : \quad & \mathcal{H} \rightarrow \mathbb{R} \\ f & \mapsto f(x) \end{aligned}$$

is **continuous**.

# An equivalent definition of RKHS

## Theorem

The Hilbert space  $\mathcal{H} \subset \mathbb{R}^{\mathcal{X}}$  is a RKHS if and only if for any  $x \in \mathcal{X}$ , the mapping:

$$\begin{aligned} F : \quad & \mathcal{H} \rightarrow \mathbb{R} \\ f & \mapsto f(x) \end{aligned}$$

is **continuous**.

## Corollary

Convergence in a RKHS implies pointwise convergence, i.e., if  $(f_n)_{n \in \mathbb{N}}$  converges to  $f$  in  $\mathcal{H}$ , then  $(f_n(x))_{n \in \mathbb{N}}$  converges to  $f(x)$  for any  $x \in \mathcal{X}$ .

## Proof

If  $\mathcal{H}$  is a RKHS then  $f \mapsto f(x)$  is continuous

If a r.k.  $K$  exists, then for any  $(x, f) \in \mathcal{X} \times \mathcal{H}$ :

$$\begin{aligned}|f(x)| &= |\langle f, K_x \rangle_{\mathcal{H}}| \\&\leq \|f\|_{\mathcal{H}} \cdot \|K_x\|_{\mathcal{H}} \text{ (Cauchy-Schwarz)} \\&\leq \|f\|_{\mathcal{H}} \cdot K(x, x)^{\frac{1}{2}},\end{aligned}$$

because  $\|K_x\|_{\mathcal{H}}^2 = \langle K_x, K_x \rangle_{\mathcal{H}} = K(x, x)$ . Therefore  $f \in \mathcal{H} \mapsto f(x) \in \mathbb{R}$  is a continuous linear mapping.  $\square$

## Proof (Converse)

If  $f \mapsto f(x)$  is continuous then  $\mathcal{H}$  is a RKHS

Conversely, let us assume that for any  $x \in \mathcal{X}$  the linear form  $f \in \mathcal{H} \mapsto f(x)$  is continuous.

Then by Riesz representation theorem there (general property of Hilbert spaces) there exists a unique  $g_x \in \mathcal{H}$  such that:

$$f(x) = \langle f, g_x \rangle_{\mathcal{H}}$$

The function  $K(x, y) = g_x(y)$  is then a r.k. for  $\mathcal{H}$ .  $\square$

# Unicity of r.k. and RKHS

## Theorem

- If  $\mathcal{H}$  is a RKHS, then it has a unique r.k.
- Conversely, a function  $K$  can be the r.k. of at most one RKHS.

# Unicity of r.k. and RKHS

## Theorem

- If  $\mathcal{H}$  is a RKHS, then it has a unique r.k.
- Conversely, a function  $K$  can be the r.k. of at most one RKHS.

## Consequence

This shows that we can talk of "the" kernel of a RKHS, or "the" RKHS of a kernel.

## Proof

If a r.k. exists then it is unique

Let  $K$  and  $K'$  be two r.k. of a RKHS  $\mathcal{H}$ . Then for any  $x \in \mathcal{X}$ :

$$\begin{aligned}\|K_x - K'_x\|_{\mathcal{H}}^2 &= \langle K_x - K'_x, K_x - K'_x \rangle_{\mathcal{H}} \\ &= \langle K_x - K'_x, K_x \rangle_{\mathcal{H}} - \langle K_x - K'_x, K'_x \rangle_{\mathcal{H}} \\ &= K_x(x) - K'_x(x) - K_x(x) + K'_x(x) \\ &= 0.\end{aligned}$$

This shows that  $K_x = K'_x$  as functions, i.e.,  $K_x(y) = K'_x(y)$  for any  $y \in \mathcal{X}$ .  
In other words,  $\mathbf{K}=\mathbf{K}'$ .  $\square$

## Proof

If a r.k. exists then it is unique

Let  $K$  and  $K'$  be two r.k. of a RKHS  $\mathcal{H}$ . Then for any  $x \in \mathcal{X}$ :

$$\begin{aligned}\|K_x - K'_x\|_{\mathcal{H}}^2 &= \langle K_x - K'_x, K_x - K'_x \rangle_{\mathcal{H}} \\ &= \langle K_x - K'_x, K_x \rangle_{\mathcal{H}} - \langle K_x - K'_x, K'_x \rangle_{\mathcal{H}} \\ &= K_x(x) - K'_x(x) - K_x(x) + K'_x(x) \\ &= 0.\end{aligned}$$

This shows that  $K_x = K'_x$  as functions, i.e.,  $K_x(y) = K'_x(y)$  for any  $y \in \mathcal{X}$ .  
In other words,  $\mathbf{K}=\mathbf{K}'$ .  $\square$

The RKHS of a r.k.  $K$  is unique

Left as exercise.

# An important result

## Theorem

A function  $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$  is p.d. if and only if it is a r.k.

Proof: r.k.  $\implies$  p.d.

- ① A r.k. is **symmetric** because, for any  $(x, y) \in \mathcal{X}^2$ :

$$K(x, y) = \langle K_x, K_y \rangle_{\mathcal{H}} = \langle K_y, K_x \rangle_{\mathcal{H}} = K(y, x).$$

- ② It is **p.d.** because for any  $N \in \mathbb{N}, (x_1, x_2, \dots, x_N) \in \mathcal{X}^N$ , and  $(a_1, a_2, \dots, a_N) \in \mathbb{R}^N$ :

$$\begin{aligned} \sum_{i,j=1}^N a_i a_j K(x_i, x_j) &= \sum_{i,j=1}^N a_i a_j \langle K_{x_i}, K_{x_j} \rangle_{\mathcal{H}} \\ &= \| \sum_{i=1}^N a_i K_{x_i} \|_{\mathcal{H}}^2 \\ &\geq 0. \quad \square \end{aligned}$$

Proof: p.d.  $\implies$  r.k. (1/4)

- Let  $\mathcal{H}_0$  be the vector subspace of  $\mathbb{R}^{\mathcal{X}}$  spanned by the functions  $\{K_x\}_{x \in \mathcal{X}}$ .
- For any  $f, g \in \mathcal{H}_0$ , given by:

$$f = \sum_{i=1}^m a_i K_{x_i}, \quad g = \sum_{j=1}^n b_j K_{y_j},$$

let:

$$\langle f, g \rangle_{\mathcal{H}_0} := \sum_{i,j} a_i b_j K(x_i, y_j).$$

Proof: p.d.  $\implies$  r.k. (2/4)

- $\langle f, g \rangle_{\mathcal{H}_0}$  does not depend on the expansion of  $f$  and  $g$  because:

$$\langle f, g \rangle_{\mathcal{H}_0} = \sum_{i=1}^m a_i g(x_i) = \sum_{j=1}^n b_j f(y_j).$$

- This also shows that  $\langle \cdot, \cdot \rangle_{\mathcal{H}_0}$  is a symmetric bilinear form.
- This also shows that for any  $x \in \mathcal{X}$  and  $f \in \mathcal{H}_0$ :

$$\langle f, K_x \rangle_{\mathcal{H}_0} = f(x).$$

Proof: p.d.  $\implies$  r.k. (3/4)

- $K$  is assumed to be p.d., therefore:

$$\|f\|_{\mathcal{H}_0}^2 = \sum_{i,j=1}^m a_i a_j K(x_i, x_j) \geq 0.$$

In particular Cauchy-Schwarz is valid with  $\langle \cdot, \cdot \rangle_{\mathcal{H}_0}$ .

- By Cauchy-Schwarz we deduce that  $\forall x \in \mathcal{X}$ :

$$|f(x)| = |\langle f, K_x \rangle_{\mathcal{H}_0}| \leq \|f\|_{\mathcal{H}_0} \cdot K(x, x)^{\frac{1}{2}},$$

therefore  $\|f\|_{\mathcal{H}_0} = 0 \implies f = 0$ .

- $\mathcal{H}_0$  is therefore a **pre-Hilbert space** endowed with the inner product  $\langle \cdot, \cdot \rangle_{\mathcal{H}_0}$ .

## Proof: p.d. $\implies$ r.k. (4/4)

- For any Cauchy sequence  $(f_n)_{n \geq 0}$  in  $(\mathcal{H}_0, \langle \cdot, \cdot \rangle_{\mathcal{H}_0})$ , we note that:

$$\forall (x, m, n) \in \mathcal{X} \times \mathbb{N}^2, \quad |f_m(x) - f_n(x)| \leq \|f_m - f_n\|_{\mathcal{H}_0} K(x, x)^{\frac{1}{2}}.$$

Therefore for any  $x$  the sequence  $(f_n(x))_{n \geq 0}$  is Cauchy in  $\mathbb{R}$  and has therefore a limit.

- If we add to  $\mathcal{H}_0$  the functions defined as the pointwise limits of Cauchy sequences, then the space becomes complete and is therefore a Hilbert space, with  $K$  as r.k. (up to a few technicalities, left as exercice).  $\square$

# Application: back to Aronszajn's theorem

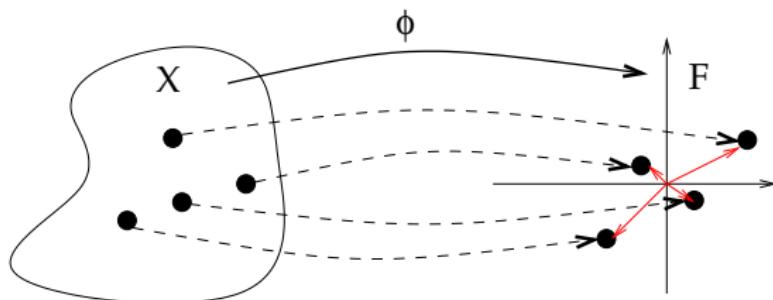
## Theorem (Aronszajn, 1950)

$K$  is a p.d. kernel on the set  $\mathcal{X}$  if and only if there exists a Hilbert space  $\mathcal{H}$  and a mapping

$$\Phi : \mathcal{X} \mapsto \mathcal{H},$$

such that, for any  $x, x'$  in  $\mathcal{X}$ :

$$K(x, x') = \langle \Phi(x), \Phi(x') \rangle_{\mathcal{H}}.$$



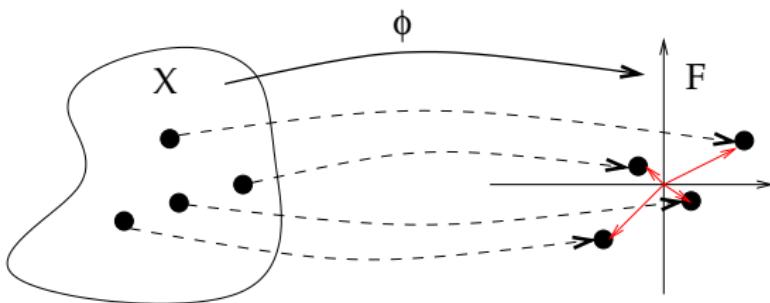
## Proof of Aronzsajn's theorem: p.d. $\implies$ kernel

- If  $K$  is p.d. over a set  $\mathcal{X}$  then it is the r.k. of a Hilbert space  $\mathcal{H} \subset \mathbb{R}^{\mathcal{X}}$ .
- Let the mapping  $\Phi : \mathcal{X} \rightarrow \mathcal{H}$  defined by:

$$\forall x \in \mathcal{X}, \quad \Phi(x) = K_x.$$

- By the reproducing property we have:

$$\forall (x, y) \in \mathcal{X}^2, \quad \langle \Phi(x), \Phi(y) \rangle_{\mathcal{H}} = \langle K_x, K_y \rangle_{\mathcal{H}} = K(x, y). \quad \square$$



## RKHS of the linear kernel

- Let  $\mathcal{X} = \mathbb{R}^d$  and  $K(x, y) = \langle x, y \rangle_{\mathbb{R}^d}$  be the linear kernel
- The corresponding RKHS consists of functions:

$$x \in \mathbb{R}^d \mapsto f(x) = \sum_i a_i \langle x_i, x \rangle_{\mathbb{R}^d} = \langle w, x \rangle_{\mathbb{R}^d},$$

with  $w = \sum_i a_i x_i$ .

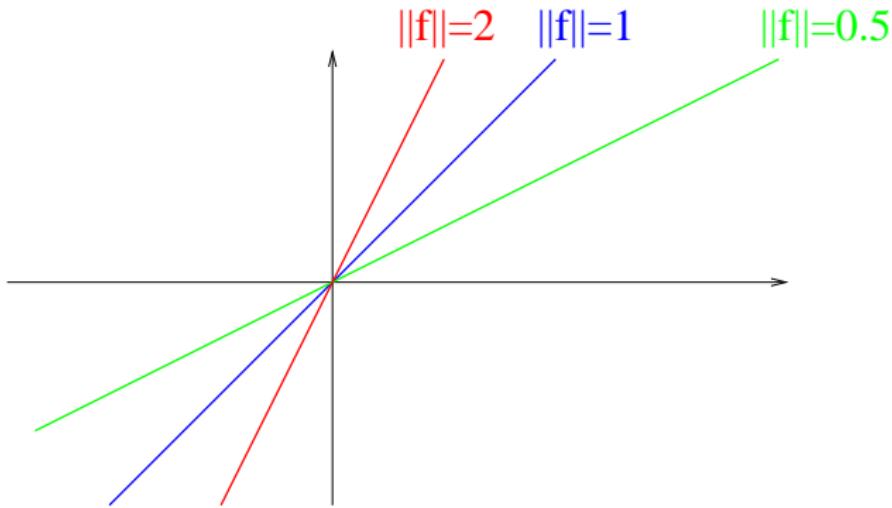
- The RKHS is therefore the set of linear forms endowed with the following inner product:

$$\langle f, g \rangle_{\mathcal{H}_K} = \langle w, v \rangle_{\mathbb{R}^d},$$

when  $f(x) = w^\top x$  and  $g(x) = v^\top x$ .

## RKHS of the linear kernel (cont.)

$$\begin{cases} K_{lin}(x, x') &= x^\top x' . \\ f(x) &= w^\top x , \\ \|f\|_{\mathcal{H}} &= \|w\|_2 . \end{cases}$$



## $\ell_2$ -regularized methods in RKHS

$$f_\beta(x) = \beta^\top \Phi(x), \quad \min_{\beta \in \mathbb{R}^p} \left\{ \frac{1}{n} \sum_{i=1}^n \ell(f_\beta(x_i), y_i) + \lambda \|\beta\|_2^2 \right\}$$

is equivalent to

$$\min_{f \in \mathcal{H}} \left\{ \frac{1}{n} \sum_{i=1}^n \ell(f(x_i), y_i) + \lambda \|f\|_{\mathcal{H}}^2 \right\}$$

where  $\mathcal{H}$  is the RKHS of the kernel  $K(x, x') = \Phi(x)^\top \Phi(x')$ .

# Smoothness functional

## A simple inequality

- By Cauchy-Schwarz we have, for any function  $f \in \mathcal{H}$  and any two points  $x, x' \in \mathcal{X}$ :

$$\begin{aligned} |f(x) - f(x')| &= |\langle f, K_x - K_{x'} \rangle_{\mathcal{H}}| \\ &\leq \|f\|_{\mathcal{H}} \times \|K_x - K_{x'}\|_{\mathcal{H}} \\ &= \|f\|_{\mathcal{H}} \times d_K(x, x') . \end{aligned}$$

- The norm of a function in the RKHS controls **how fast** the function varies over  $\mathcal{X}$  with respect to the **geometry defined by the kernel** (Lipschitz with constant  $\|f\|_{\mathcal{H}}$ ).

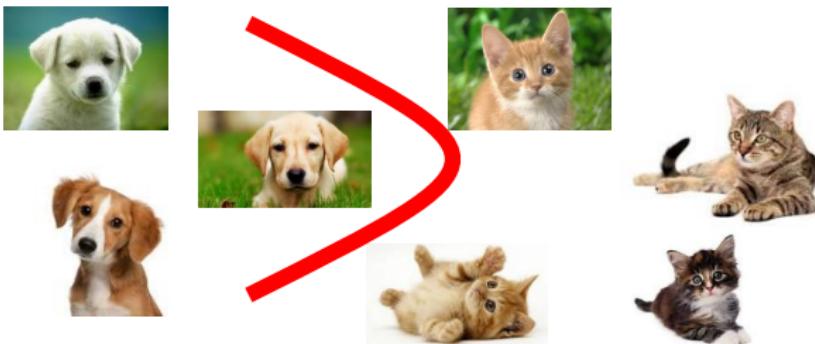
## Important message

Small norm  $\implies$  slow variations.

# Kernels and RKHS : Summary for supervised learning

The goal is to learn a **prediction function**  $f : \mathcal{X} \rightarrow \mathcal{Y}$  given labeled training data  $(x_i, y_i)_{i=1,\dots,n}$  with  $x_i$  in  $\mathcal{X}$ , and  $y_i$  in  $\mathcal{Y}$ :

$$\min_{f \in \mathcal{F}} \underbrace{\frac{1}{n} \sum_{i=1}^n L(y_i, f(x_i))}_{\text{empirical risk, data fit}} + \underbrace{\lambda \Omega(f)}_{\text{regularization}}.$$



# Kernels and RKHS : Summary for supervised learning

The goal is to learn a **prediction function**  $f : \mathcal{X} \rightarrow \mathcal{Y}$  given labeled training data  $(x_i, y_i)_{i=1,\dots,n}$  with  $x_i$  in  $\mathcal{X}$ , and  $y_i$  in  $\mathcal{Y}$ :

$$\min_{f \in \mathcal{F}} \underbrace{\frac{1}{n} \sum_{i=1}^n L(y_i, f(x_i))}_{\text{empirical risk, data fit}} + \underbrace{\lambda \Omega(f)}_{\text{regularization}}.$$

The labels  $y_i$  are in

- $\{-1, +1\}$  for **binary** classification problems.
- $\{1, \dots, K\}$  for **multi-class** classification problems.
- $\mathbb{R}$  for **regression** problems.
- $\mathbb{R}^k$  for **multivariate regression** problems.

# Kernels and RKHS : Summary for supervised learning

The goal is to learn a **prediction function**  $f : \mathcal{X} \rightarrow \mathcal{Y}$  given labeled training data  $(x_i, y_i)_{i=1,\dots,n}$  with  $x_i$  in  $\mathcal{X}$ , and  $y_i$  in  $\mathcal{Y}$ :

$$\min_{f \in \mathcal{F}} \underbrace{\frac{1}{n} \sum_{i=1}^n L(y_i, f(x_i))}_{\text{empirical risk, data fit}} + \underbrace{\lambda \Omega(f)}_{\text{regularization}}.$$

Example with linear models: logistic regression, SVMs, etc.

- assume there exists a linear relation between  $y$  and features  $x$  in  $\mathbb{R}^p$ .
- $f(x) = w^\top x + b$  is parametrized by  $w, b$  in  $\mathbb{R}^{p+1}$ ;
- $L$  is often a **convex** loss function;
- $\Omega(f)$  is often the squared  $\ell_2$ -norm  $\|w\|^2$ .

# Kernels and RKHS : Summary for supervised learning

The goal is to learn a **prediction function**  $f : \mathcal{X} \rightarrow \mathcal{Y}$  given labeled training data  $(x_i, y_i)_{i=1,\dots,n}$  with  $x_i$  in  $\mathcal{X}$ , and  $y_i$  in  $\mathcal{Y}$ :

$$\min_{f \in \mathcal{F}} \underbrace{\frac{1}{n} \sum_{i=1}^n L(y_i, f(x_i))}_{\text{empirical risk, data fit}} + \underbrace{\lambda \Omega(f)}_{\text{regularization}}.$$

## Remark about multilayer neural networks

- The “neural network” space  $\mathcal{F}$  is explicitly parametrized by:

$$f(\mathbf{x}) = \sigma_k(\mathbf{A}_k \sigma_{k-1}(\mathbf{A}_{k-1} \dots \sigma_2(\mathbf{A}_2 \sigma_1(\mathbf{A}_1 \mathbf{x})) \dots)).$$

- Linear operations are either unconstrained (fully connected) or involve parameter sharing (e.g., convolutions).
- Finding the optimal  $\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_k$  yields a **non-convex** optimization problem.

# Kernels and RKHS : Summary for supervised learning

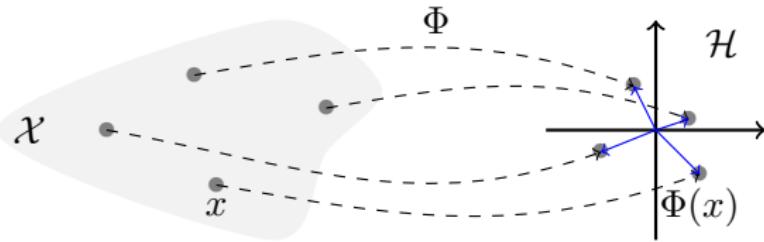
A classical kernel formulation for supervised learning

$$\min_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^n L(y_i, f(x_i)) + \lambda \|f\|_{\mathcal{H}}^2.$$

- map data  $x$  in  $\mathcal{X}$  to a Hilbert space and work with **linear forms**:

$$\Phi : \mathcal{X} \rightarrow \mathcal{H} \quad \text{and} \quad f(x) = \langle \Phi(x), f \rangle_{\mathcal{H}}.$$

- This is done implicitly with a positive definite kernel  $K(x, x') = \langle \Phi(x), \Phi(x') \rangle$ !



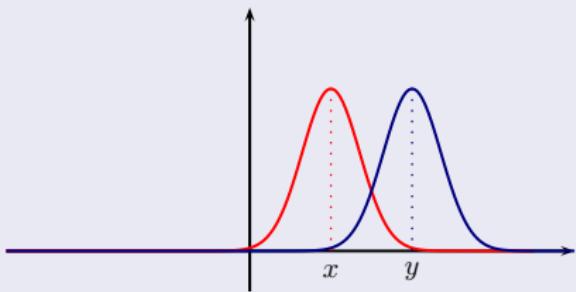
# Kernels and RKHS : Summary for supervised learning

What does it mean to map a data point to a function?

Ex: if  $x, y$  in  $\mathbb{R}$  and  $K(x, y) = e^{-\frac{1}{\sigma^2}(x-y)^2}$  is the Gaussian kernel,

$$\Phi(x) : t \mapsto e^{-\frac{1}{\sigma^2}(x-t)^2}$$

$$\Phi(y) : t \mapsto e^{-\frac{1}{\sigma^2}(y-t)^2}$$



- Data points are mapped to Gaussian functions living in a Hilbert space  $\mathcal{H}$ .
- But  $\mathcal{H}$  is much richer and contains much more than Gaussian functions!
- Prediction functions  $f$  live in  $\mathcal{H}$ :  $f(x) = \langle f, \Phi(x) \rangle$ .

# Kernels and RKHS : Summary

- P.d. kernels can be thought of as **inner product** after embedding the data space  $\mathcal{X}$  in some Hilbert space. As such a p.d. kernel defines a **metric** on  $\mathcal{X}$ .
- A realization of this embedding is the **RKHS**, valid without restriction on the space  $\mathcal{X}$  nor on the kernel.
- The RKHS is a space of functions over  $\mathcal{X}$ . The **norm** of a function in the RKHS is related to its degree of **smoothness** w.r.t. the metric defined by the kernel on  $\mathcal{X}$ .
- $\ell_2$ -regularized learning in the feature space can be formulated in the RKHS

$$\min_{f \in \mathcal{H}} \left\{ \frac{1}{n} \sum_{i=1}^n \ell(f(x_i), y_i) + \lambda \|f\|_{\mathcal{H}}^2 \right\}$$

# Outline

- 1 Learning in high dimension
- 2 Learning with  $\ell_2$  regularization

- Ridge regression
- Ridge logistic regression
- Linear hard-margin SVM
- Interlude: quick notes on constrained optimization
- Back to hard-margin SVM
- Soft-margin SVM
- Large-margin classifiers

- 3 Learning with kernels

- Kernel methods
- Positive definite kernels and RKHS
- Kernel examples
- Multiple Kernel Learning (MKL)

- 4 Conclusion

## Kernel examples

- **Polynomial** (on  $\mathbb{R}^d$ ):

$$K(x, x') = (x \cdot x' + 1)^d$$

- **Gaussian radial basis function (RBF)** (on  $\mathbb{R}^d$ )

$$K(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right)$$

- **Laplace kernel** (on  $\mathbb{R}$ )

$$K(x, x') = \exp(-\gamma|x - x'|)$$

- **Min kernel** (on  $\mathbb{R}_+$ )

$$K(x, x') = \min(x, x')$$

### Exercice

*Exercice: for each kernel, find a Hilbert space  $\mathcal{H}$  and a mapping  $\Phi : \mathcal{X} \rightarrow \mathcal{H}$  such that  $K(x, x') = \langle \Phi(x), \Phi(x') \rangle$*

## Example: SVM with a Gaussian kernel

- Training:

$$\min_{\alpha \in \mathbb{R}^n} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j \exp\left(-\frac{\|\vec{x}_i - \vec{x}_j\|^2}{2\sigma^2}\right)$$

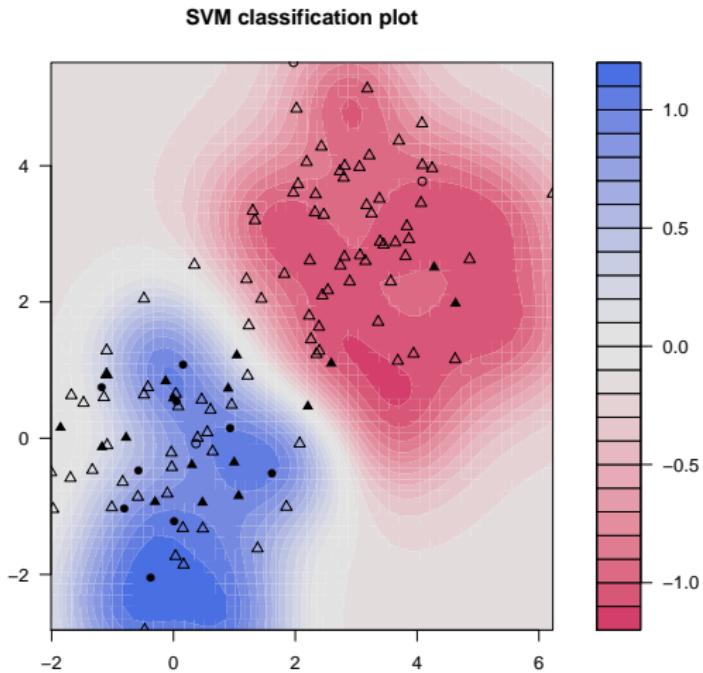
$$\text{s.t. } 0 \leq \alpha_i \leq C, \quad \text{and} \quad \sum_{i=1}^n \alpha_i y_i = 0.$$

- Prediction

$$f(\vec{x}) = \sum_{i=1}^n \alpha_i \exp\left(-\frac{\|\vec{x} - \vec{x}_i\|^2}{2\sigma^2}\right)$$

# Example: SVM with a Gaussian kernel

$$f(\vec{x}) = \sum_{i=1}^n \alpha_i \exp\left(-\frac{\|\vec{x} - \vec{x}_i\|^2}{2\sigma^2}\right)$$



# How to choose or make a kernel?

- Design **features**
- Design a **distance** or **similarity** measure
- Design a **regularizer** on  $f$

## Example: Sobolev norm as regularizer

### Theorem

Let  $\mathcal{X} = [0, 1]$  and the kernel:

$$\forall (x, y) \in [0, 1]^2, \quad K(x, y) = \min(x, y).$$

Then the RKHS is

$$\mathcal{H} = \{f : [0, 1] \mapsto \mathbb{R}, \text{absolutely continuous, } f' \in L^2([0, 1]), f(0) = 0\}.$$

and the regularizer is a Sobolev norm

$$\Omega(f) = \|f\|_{\mathcal{H}}^2 = \int_0^1 f'(u)^2 du = \|f'\|_{L^2([0,1])}^2.$$

## Proof (1/5)

### Sketch

We need to show that

- $\mathcal{H}$  is a Hilbert space
- $\forall x \in [0, 1], K_x \in \mathcal{H},$
- $\forall (x, f) \in [0, 1] \times \mathcal{H}, \langle f, K_x \rangle_{\mathcal{H}} = f(x).$

## Proof (2/5)

$\mathcal{H}$  is a pre-Hilbert space

- $f$  absolutely continuous implies differentiable almost everywhere, and

$$\forall x \in [0, 1], \quad f(x) = f(0) + \int_0^x f'(u)du.$$

- For any  $f \in \mathcal{H}$ ,  $f(0) = 0$  implies by Cauchy-Schwarz:

$$|f(x)| = \left| \int_0^x f'(u)du \right| \leq \sqrt{x} \left( \int_0^1 f'(u)^2 du \right)^{\frac{1}{2}} = \sqrt{x} \|f\|_{\mathcal{H}}.$$

Therefore,  $\|f\|_{\mathcal{H}} = 0 \implies f = 0$ , showing that  $\langle \cdot, \cdot \rangle_{\mathcal{H}}$  is an inner product.  $\mathcal{H}$  is thus a pre-Hilbert space.

## Proof (3/5)

$\mathcal{H}$  is a Hilbert space

- To show that  $\mathcal{H}$  is complete, let  $(f_n)_{n \in \mathbb{N}}$  a Cauchy sequence in  $\mathcal{H}$
- $(f'_n)_{n \in \mathbb{N}}$  is a Cauchy sequence in  $L^2[0, 1]$ , thus converges to  $g \in L^2[0, 1]$
- By the previous inequality,  $(f_n(x))_{n \in \mathbb{N}}$  is a Cauchy sequence and thus converges to a real number  $f(x)$ , for any  $x \in [0, 1]$ . Moreover:

$$f(x) = \lim_n f_n(x) = \lim_n \int_0^x f'_n(u) du = \int_0^x g(u) du,$$

showing that  $f$  is absolutely continuous and  $f' = g$  almost everywhere; in particular,  $f' \in L^2[0, 1]$ .

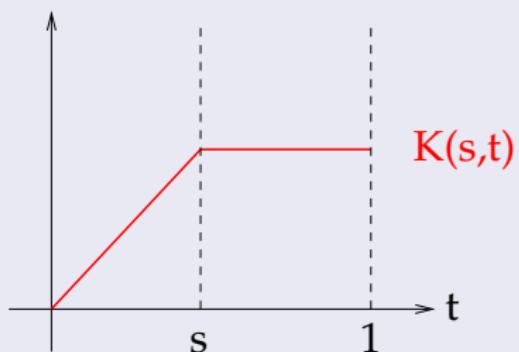
- Finally,  $f(0) = \lim_n f_n(0) = 0$ , therefore  $f \in \mathcal{H}$  and

$$\lim_n \|f_n - f\|_{\mathcal{H}} = \|f' - g\|_{L^2[0,1]} = 0.$$

## Proof (4/5)

$\forall x \in [0, 1], K_x \in \mathcal{H}$

Let  $K_x(y) = K(x, y) = \min(x, y)$  sur  $[0, 1]^2$ :



$K_x$  is differentiable except at  $s$ , has a square integrable derivative, and  $K_x(0) = 0$ , therefore  $K_x \in \mathcal{H}$  for all  $x \in [0, 1]$ .  $\square$

## Proof (5/5)

For all  $x, f$ ,  $\langle f, K_x \rangle_{\mathcal{H}} = f(x)$

For any  $x \in [0, 1]$  and  $f \in \mathcal{H}$  we have:

$$\langle f, K_x \rangle_{\mathcal{H}} = \int_0^1 f'(u) K'_x(u) du = \int_0^x f'(u) du = f(x),$$

which shows that  $K$  is the r.k. associated to  $\mathcal{H}$ .  $\square$

# Generalization

## Theorem

Let  $\mathcal{X} = \mathbb{R}^d$  and  $D$  a differential operator on a class of functions  $\mathcal{H}$  such that, endowed with the inner product:

$$\forall (f, g) \in \mathcal{H}^2, \quad \langle f, g \rangle_{\mathcal{H}} = \langle Df, Dg \rangle_{L^2(\mathcal{X})},$$

it is a Hilbert space.

Then  $\mathcal{H}$  is a RKHS that admits as r.k. the Green function of the operator  $D^*D$ , where  $D^*$  denotes the adjoint operator of  $D$ .

In case of...

## Green functions

Let the differential equation on  $\mathcal{H}$ :

$$f = Dg ,$$

where  $g$  is unknown. In order to solve it we can look for  $g$  of the form:

$$g(x) = \int_{\mathcal{X}} k(x, y) f(y) dy$$

for some function  $k : \mathcal{X}^2 \mapsto \mathbb{R}$ .  $k$  must then satisfy, for all  $x \in \mathcal{X}$ ,

$$f(x) = Dg(x) = \langle Dk_x, f \rangle_{L^2(\mathcal{X})} .$$

$k$  is called the **Green function** of the operator  $D$ .

## Proof

Let  $\mathcal{H}$  be a Hilbert space endowed with the inner product:

$$\langle f, g \rangle_{\mathcal{X}} = \langle Df, Dg \rangle_{L^2(\mathcal{X})},$$

and  $K$  be the Green function of the operator  $D^*D$ . For all  $x \in \mathcal{X}$ ,  $K_x \in \mathcal{H}$  because:

$$\langle DK_x, DK_x \rangle_{L^2(\mathcal{X})} = \langle D^*DK_x, K_x \rangle_{L^2(\mathcal{X})} = K_x(x) < \infty.$$

Moreover, for all  $f \in \mathcal{H}$  and  $x \in \mathcal{X}$ , we have:

$$f(x) = \langle D^*DK_x, f \rangle_{L^2(\mathcal{X})} = \langle DK_x, Df \rangle_{L^2(\mathcal{X})} = \langle K_x, f \rangle_{\mathcal{H}},$$

which shows that  $\mathcal{H}$  is a RKHS with  $K$  as r.k.  $\square$

# Translation invariant kernels

## Definition

A kernel  $K : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}$  is called **translation invariant** (t.i.) if it only depends on the difference between its arguments, i.e.:

$$\forall (x, y) \in \mathbb{R}^{2d}, \quad K(x, y) = \kappa(x - y).$$

## Theorem (Bochner)

A real-valued function  $\kappa(x - y)$  on  $\mathbb{R}^d$  is positive definite if and only if it is the Fourier transform of a **symmetric, positive, and finite** Borel measure.

# RKHS of translation invariant kernels

## Theorem

Let  $K$  be a translation invariant p.d. kernel, such that  $\kappa$  is integrable on  $\mathbb{R}^d$  as well as its Fourier transform  $\hat{\kappa}$ . The subset  $\mathcal{H}_K$  of  $L_2(\mathbb{R}^d)$  that consists of integrable and continuous functions  $f$  such that:

$$\|f\|_K^2 := \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} \frac{|\hat{f}(\omega)|^2}{\hat{\kappa}(\omega)} d\omega < +\infty,$$

endowed with the inner product:

$$\langle f, g \rangle := \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} \frac{\hat{f}(\omega) \hat{g}(\omega)^*}{\hat{\kappa}(\omega)} d\omega$$

is a RKHS with  $K$  as r.k.

## Example: Gaussian RBF kernel

$$K(x, y) = e^{-\frac{(x-y)^2}{2\sigma^2}}$$

corresponds to:

$$\hat{\kappa}(\omega) = e^{-\frac{\sigma^2 \omega^2}{2}}$$

and

$$\|f\|_{\mathcal{H}}^2 = \int \left| \hat{f}(\omega) \right|^2 e^{\frac{\sigma^2 \omega^2}{2}} d\omega.$$

In particular, all functions in  $\mathcal{H}$  are **infinitely differentiable** with all derivatives in  $L^2$ .

## Example: Laplace kernel

$$K(x, y) = \frac{1}{2} e^{-\gamma|x-y|}$$

corresponds to:

$$\hat{\kappa}(\omega) = \frac{\gamma}{\gamma^2 + \omega^2}$$

and

$$\|f\|_{\mathcal{H}}^2 = \int \left| \hat{f}(\omega) \right|^2 \frac{(\gamma^2 + \omega^2)}{\gamma} d\omega.$$

The RKHS is the set of functions  $L^2$  differentiable with derivatives in  $L^2$  (Sobolev space).

## Example: sinc kernel

$$K(x, y) = \frac{\sin(\Omega(x - y))}{\pi(x - y)}$$

corresponds to:

$$\hat{\kappa}(\omega) = \mathbf{1}(-\Omega \leq \omega \leq \Omega).$$

The RKHS is the set of functions whose spectrum is included in  $[-\Omega, \Omega]$ :

$$\mathcal{H} = \left\{ f : \int_{|\omega| > \Omega} |\hat{f}(\omega)|^2 d\omega = 0 \right\},$$

and

$$\|f\|_{\mathcal{H}}^2 = \int_{|\omega| \leq \Omega} |\hat{f}(\omega)|^2 = \int_{\omega \in \mathbb{R}} |\hat{f}(\omega)|^2 = (2\pi)^d \int_{x \in \mathbb{R}} |f(x)|^2 dx.$$

# Supervised sequence classification

## Data (training)

- Secreted proteins:

MASKATLLLAFATLLFATCIARHQQRQQQNQCQLQNIEA...

MARSSLFTFLCLAVFINGCLSQIEQQSPWEFQGSEVW...

MALHTVLIMLSLLPMLEAQNPEHANITIGEPITNETLGWL...

...

- Non-secreted proteins:

MAPPSSVFAEVPQAQPVLVFKLIADFREDPDPRKVN LGVG...

MAHTLGLTQPNSTEPHKISFTAKEIDVIEWKG DILVVG...

MSISESYAKEIKTAFRQFTDFPIEGEQFEDFLPIIGNP..

...

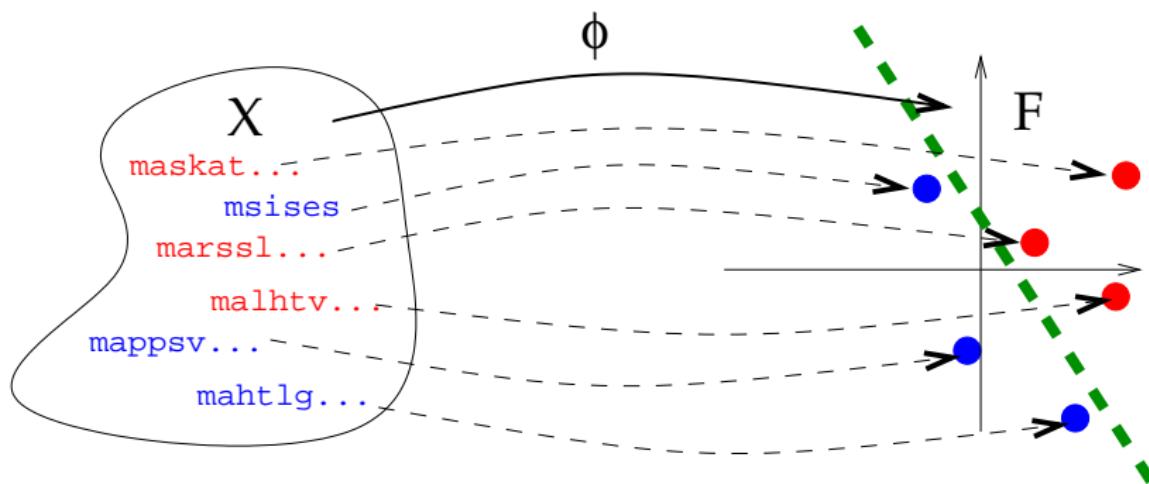
## Goal

- Build a **classifier** to **predict** whether new proteins are secreted or not.

# String kernels

## The idea

- Map each string  $x \in \mathcal{X}$  to a vector  $\Phi(x) \in \mathcal{F}$ .
- Train a **classifier for vectors** on the images  $\Phi(x_1), \dots, \Phi(x_n)$  of the training set (nearest neighbor, linear perceptron, logistic regression, support vector machine...)



## Example: substring indexation

### The approach

Index the feature space by fixed-length strings, i.e.,

$$\Phi(x) = (\Phi_u(x))_{u \in \mathcal{A}^k}$$

where  $\Phi_u(x)$  can be:

- the number of occurrences of  $u$  in  $x$  (without gaps) : **spectrum kernel** (Leslie et al., 2002)
- the number of occurrences of  $u$  in  $x$  up to  $m$  mismatches (without gaps) : **mismatch kernel** (Leslie et al., 2004)
- the number of occurrences of  $u$  in  $x$  allowing gaps, with a weight decaying exponentially with the number of gaps : **substring kernel** (Lohdi et al., 2002)

## Spectrum kernel (1/2)

### Kernel definition

- The 3-spectrum of

$$x = \text{CGGSLIAMMWFGV}$$

is:

$$(\text{CGG}, \text{GGS}, \text{GSL}, \text{SLI}, \text{LIA}, \text{IAM}, \text{AMM}, \text{MMW}, \text{MWF}, \text{WFG}, \text{FGV}) .$$

- Let  $\Phi_u(x)$  denote the number of occurrences of  $u$  in  $x$ . The  $k$ -spectrum kernel is:

$$K(x, x') := \sum_{u \in \mathcal{A}^k} \Phi_u(x) \Phi_u(x') .$$

## Spectrum kernel (2/2)

### Implementation

- The computation of the kernel is formally a sum over  $|\mathcal{A}|^k$  terms, but at most  $|x| - k + 1$  terms are non-zero in  $\Phi(x) \implies$  Computation in  $O(|x| + |x'|)$  with pre-indexation of the strings.
- Fast classification of a sequence  $x$  in  $O(|x|)$ :

$$f(x) = w \cdot \Phi(x) = \sum_u w_u \Phi_u(x) = \sum_{i=1}^{|x|-k+1} w_{x_i \dots x_{i+k-1}}.$$

### Remarks

- Work with any string (natural language, time series...)
- Fast and scalable, a good default method for string classification.
- Variants allow matching of  $k$ -mers up to  $m$  mismatches.

## Local alignment kernel (Saigo et al., 2004)

CGGSLIAMM-----WFGV  
| . . . | | | | . . . | | | |  
C----LIVMMNRLMWFGV

$$s_{S,g}(\pi) = S(C, C) + S(L, L) + S(I, I) + S(A, V) + 2S(M, M) + S(W, W) + S(F, F) + S(G, G) + S(V, V) - g(3) - g(4)$$

$SW_{S,g}(x, y) := \max_{\pi \in \Pi(x, y)} s_{S,g}(\pi)$  is not a kernel

$K_{LA}^{(\beta)}(x, y) = \sum_{\pi \in \Pi(x, y)} \exp(\beta s_{S,g}(x, y, \pi))$  is a kernel

## LA kernel is p.d.: proof (1/2)

Definition: Convolution kernel (Haussler, 1999)

Let  $K_1$  and  $K_2$  be two p.d. kernels for strings. The convolution of  $K_1$  and  $K_2$ , denoted  $K_1 \star K_2$ , is defined for any  $x, x' \in \mathcal{X}$  by:

$$K_1 \star K_2(x, y) := \sum_{x_1 x_2 = x, y_1 y_2 = y} K_1(x_1, y_1) K_2(x_2, y_2).$$

Lemma

If  $K_1$  and  $K_2$  are p.d. then  $K_1 \star K_2$  is p.d..

## LA kernel is p.d.: proof (2/2)

$$K_{LA}^{(\beta)} = \sum_{n=0}^{\infty} K_0 \star \left( K_a^{(\beta)} \star K_g^{(\beta)} \right)^{(n-1)} \star K_a^{(\beta)} \star K_0 ,$$

with

- The constant kernel:

$$K_0(x, y) := 1 .$$

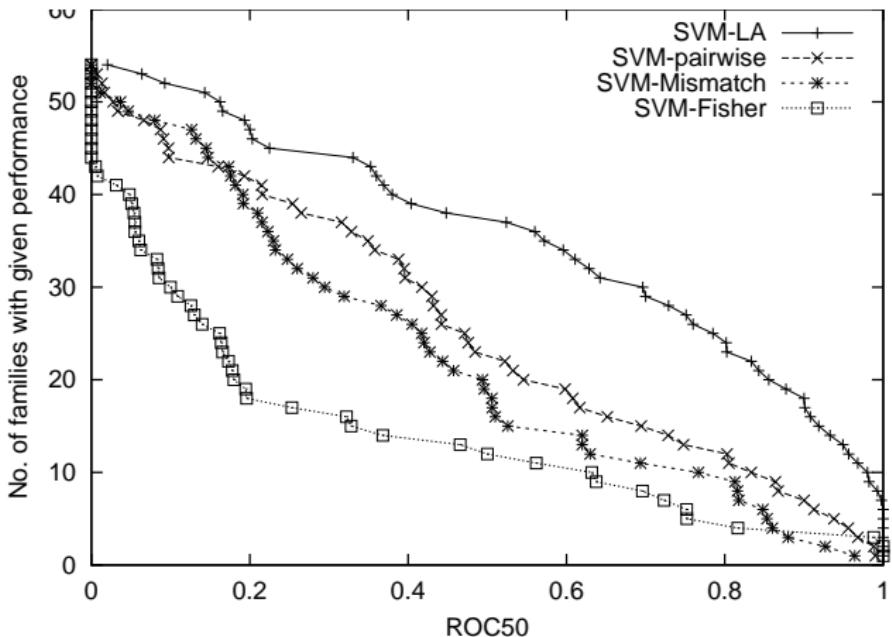
- A kernel for letters:

$$K_a^{(\beta)}(x, y) := \begin{cases} 0 & \text{if } |x| \neq 1 \text{ where } |y| \neq 1 , \\ \exp(\beta S(x, y)) & \text{otherwise .} \end{cases}$$

- A kernel for gaps:

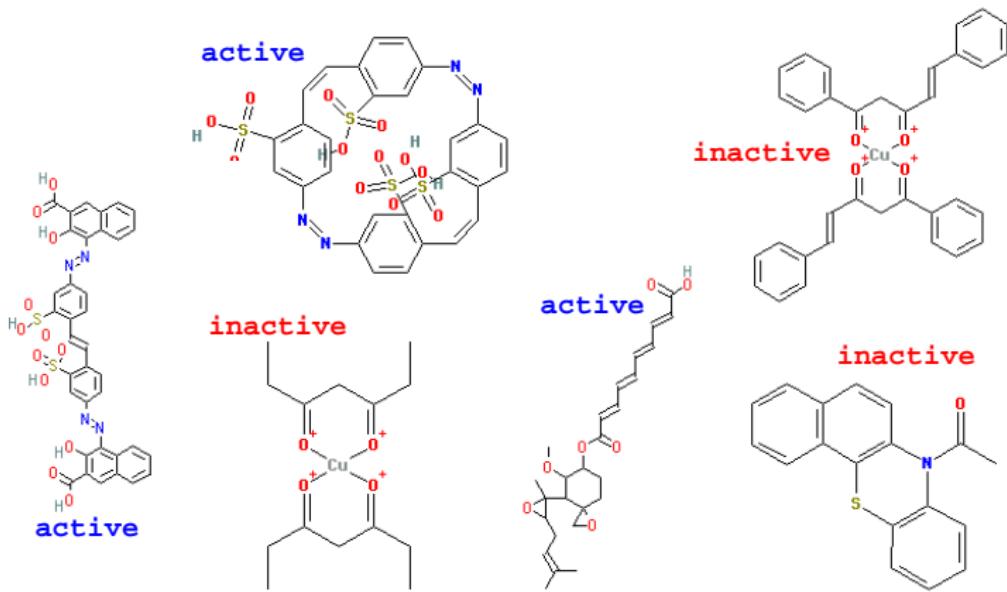
$$K_g^{(\beta)}(x, y) = \exp [\beta (g(|x|) + g(|y|))] .$$

# The choice of kernel matters



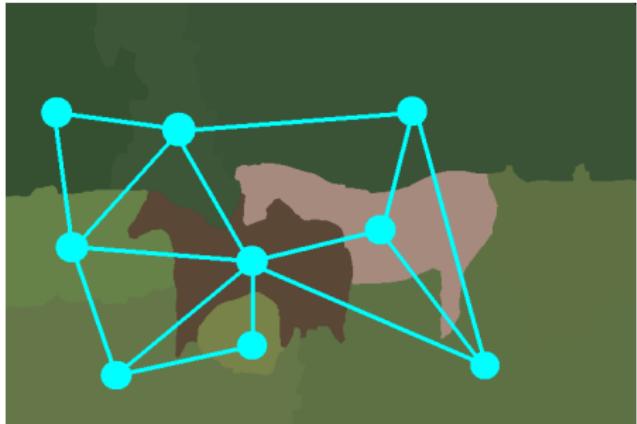
Performance on the SCOP superfamily recognition benchmark (from Saigo et al., 2004).

# Virtual screening for drug discovery



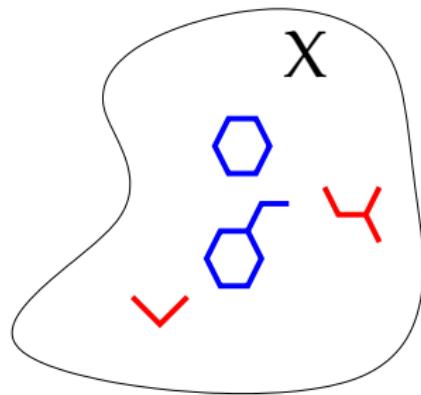
NCI AIDS screen results (from <http://cactus.nci.nih.gov>).

# Image retrieval and classification



*From Harchaoui and Bach (2007).*

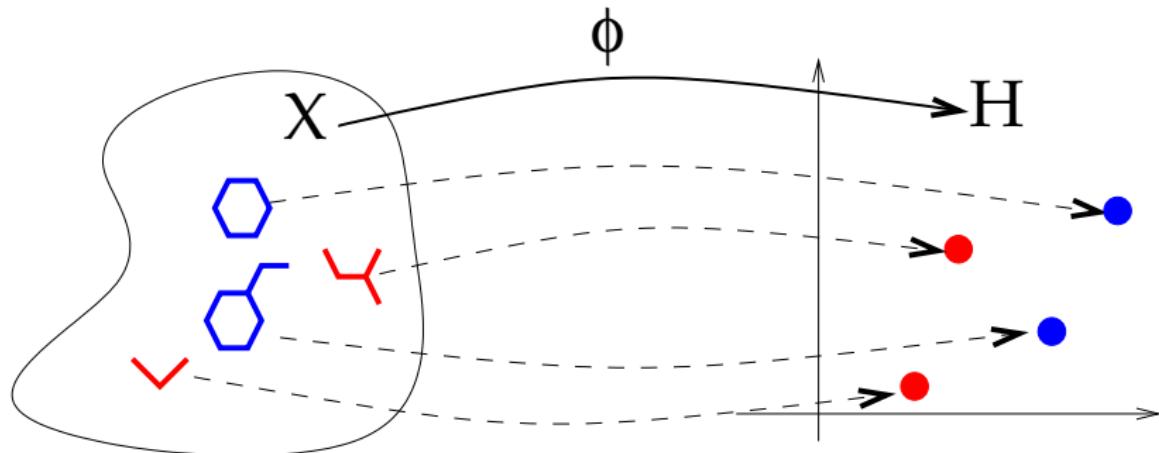
# Graph kernels



# Graph kernels

- ① Represent each graph  $x$  by a vector  $\Phi(x) \in \mathcal{H}$ , either **explicitly** or **implicitly** through the kernel

$$K(x, x') = \Phi(x)^\top \Phi(x').$$

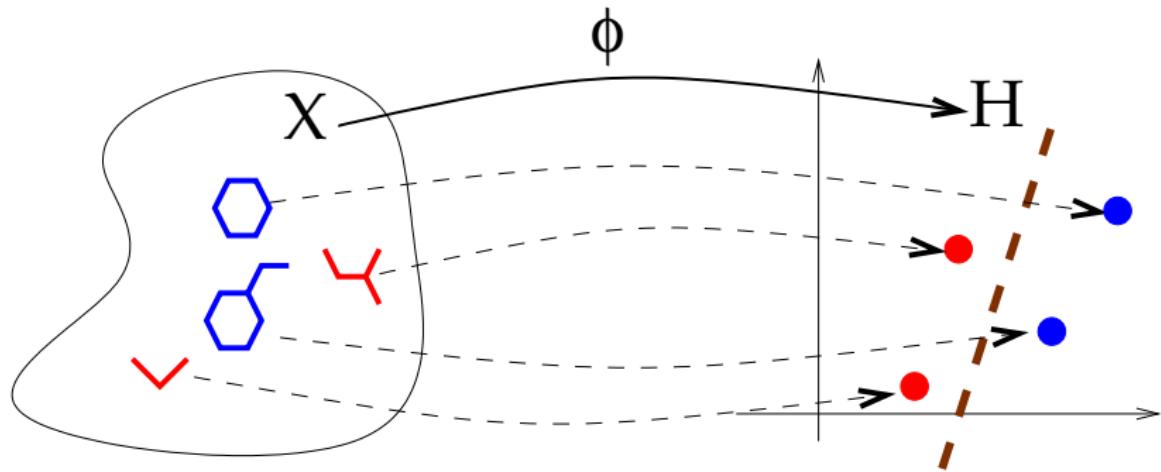


# Graph kernels

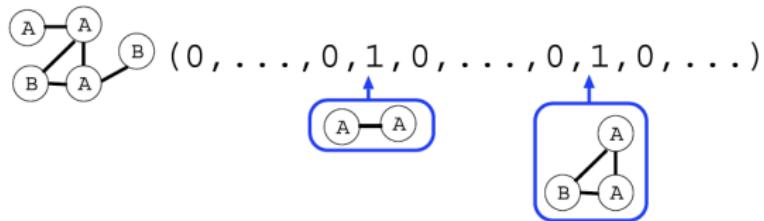
- ① Represent each graph  $x$  by a vector  $\Phi(x) \in \mathcal{H}$ , either **explicitly** or **implicitly** through the kernel

$$K(x, x') = \Phi(x)^\top \Phi(x').$$

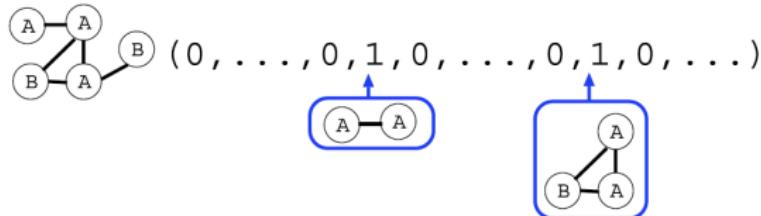
- ② Use a linear method for classification in  $\mathcal{H}$ .



# Indexing by all subgraphs?



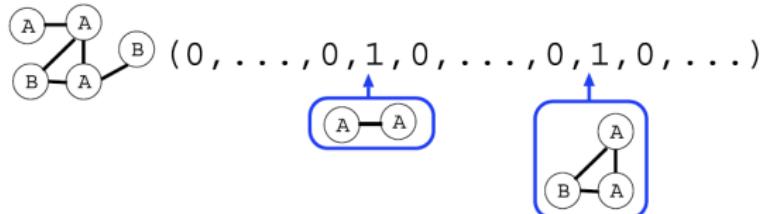
# Indexing by all subgraphs?



## Theorem

*Computing all subgraph occurrences is **NP-hard**.*

# Indexing by all subgraphs?



## Theorem

*Computing all subgraph occurrences is NP-hard.*

## Proof.

- The linear graph of size  $n$  is a subgraph of a graph  $X$  with  $n$  vertices iff  $X$  has an Hamiltonian path
- The decision problem whether a graph has a Hamiltonian path is NP-complete.



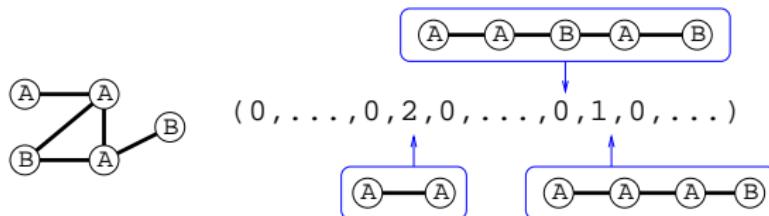
# Indexing by specific subgraphs

## Substructure selection

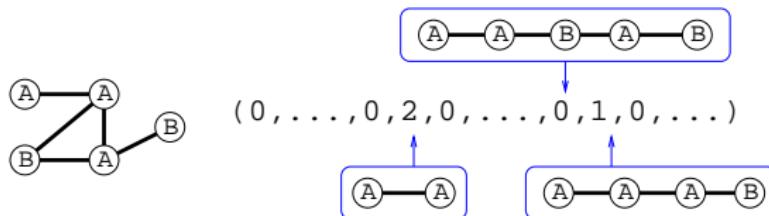
We can imagine more limited sets of substructures that lead to more computationnally efficient indexing (non-exhaustive list)

- substructures selected by **domain knowledge** (MDL fingerprint)
- all path **up to length  $k$**  (Openeye fingerprint, Nicholls 2005)
- all **shortest paths** (Borgwardt and Kriegel, 2005)
- all subgraphs **up to  $k$  vertices** (graphlet kernel, Sherashidze et al., 2009)
- all **frequent** subgraphs in the database (Helma et al., 2004)

## Example : Indexing by all shortest paths



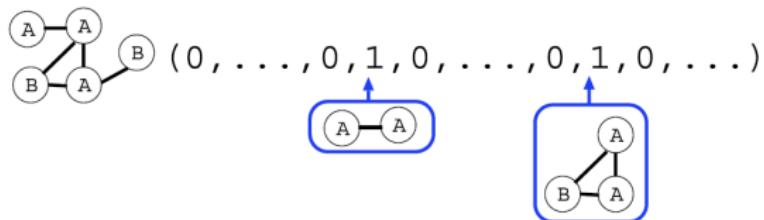
## Example : Indexing by all shortest paths



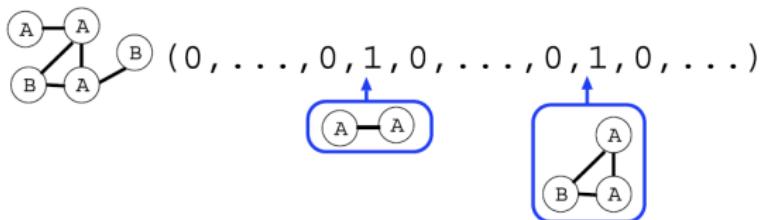
### Properties (Borgwardt and Kriegel, 2005)

- There are  $O(n^2)$  shortest paths.
- The vector of counts can be computed in  $O(n^4)$  with the Floyd-Warshall algorithm.

## Example : Indexing by all subgraphs up to $k$ vertices



## Example : Indexing by all subgraphs up to $k$ vertices



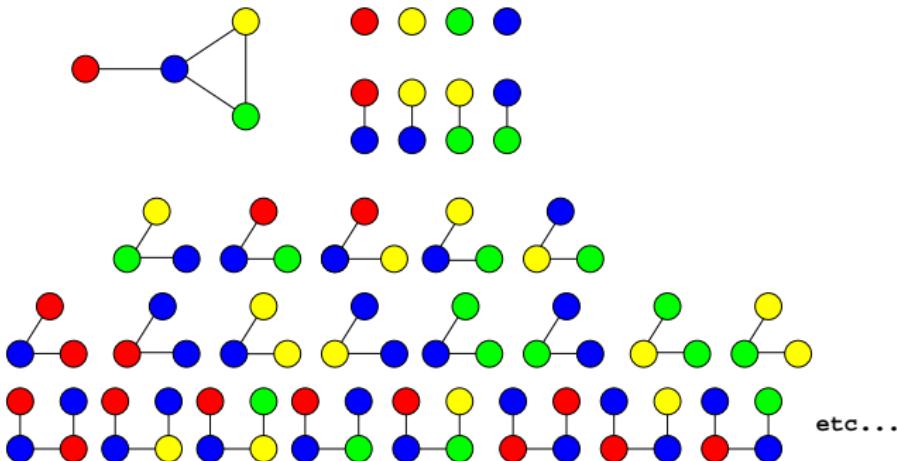
### Properties (Shervashidze et al., 2009)

- Naive enumeration scales as  $O(n^k)$ .
- Enumeration of connected graphlets in  $O(nd^{k-1})$  for graphs with degree  $\leq d$  and  $k \leq 5$ .
- Randomly sample subgraphs if enumeration is infeasible.

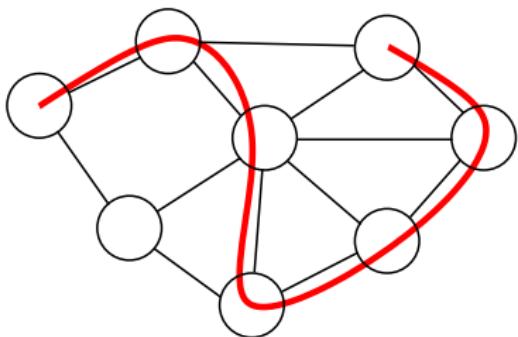
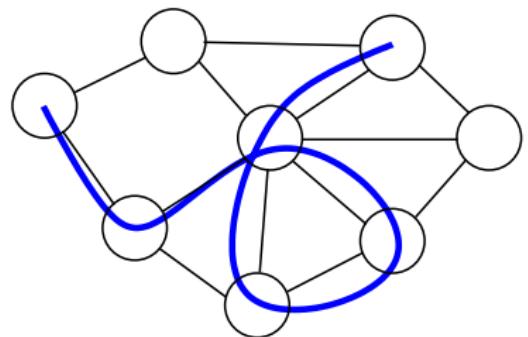
# Walks

## Definition

- A **walk** of a graph  $(V, E)$  is sequence of  $v_1, \dots, v_n \in V$  such that  $(v_i, v_{i+1}) \in E$  for  $i = 1, \dots, n - 1$ .
- We note  $\mathcal{W}_n(G)$  the set of walks with  $n$  vertices of the graph  $G$ , and  $\mathcal{W}(G)$  the set of all walks.



Walks  $\neq$  paths



# Walk kernel

## Definition

- Let  $\mathcal{S}_n$  denote the set of all possible **label sequences** of walks of length  $n$  (including vertices and edges labels), and  $\mathcal{S} = \cup_{n \geq 1} \mathcal{S}_n$ .
- For any graph  $\mathcal{X}$  let a **weight**  $\lambda_G(w)$  be associated to each walk  $w \in \mathcal{W}(G)$ .
- Let the feature vector  $\Phi(G) = (\Phi_s(G))_{s \in \mathcal{S}}$  be defined by:

$$\Phi_s(G) = \sum_{w \in \mathcal{W}(G)} \lambda_G(w) \mathbf{1} (s \text{ is the label sequence of } w).$$

# Walk kernel

## Definition

- Let  $\mathcal{S}_n$  denote the set of all possible **label sequences** of walks of length  $n$  (including vertices and edges labels), and  $\mathcal{S} = \cup_{n \geq 1} \mathcal{S}_n$ .
- For any graph  $\mathcal{X}$  let a **weight**  $\lambda_G(w)$  be associated to each walk  $w \in \mathcal{W}(G)$ .
- Let the feature vector  $\Phi(G) = (\Phi_s(G))_{s \in \mathcal{S}}$  be defined by:

$$\Phi_s(G) = \sum_{w \in \mathcal{W}(G)} \lambda_G(w) \mathbf{1} (s \text{ is the label sequence of } w).$$

- A walk kernel is a graph kernel defined by:

$$K_{\text{walk}}(G_1, G_2) = \sum_{s \in \mathcal{S}} \Phi_s(G_1) \Phi_s(G_2).$$

## Walk kernel examples

- The  $n$ th-order walk kernel is the walk kernel with  $\lambda_G(w) = 1$  if the length of  $w$  is  $n$ , 0 otherwise. It compares two graphs through their common walks of length  $n$ .

## Walk kernel examples

- The *n*th-order walk kernel is the walk kernel with  $\lambda_G(w) = 1$  if the length of  $w$  is  $n$ , 0 otherwise. It compares two graphs through their common walks of length  $n$ .
- The random walk kernel is obtained with  $\lambda_G(w) = P_G(w)$ , where  $P_G$  is a Markov random walk on  $G$ . In that case we have:

$$K(G_1, G_2) = P(\text{label}(W_1) = \text{label}(W_2)),$$

where  $W_1$  and  $W_2$  are two independant random walks on  $G_1$  and  $G_2$ , respectively (Kashima et al., 2003).

## Walk kernel examples

- The *n*th-order walk kernel is the walk kernel with  $\lambda_G(w) = 1$  if the length of  $w$  is  $n$ , 0 otherwise. It compares two graphs through their common walks of length  $n$ .
- The random walk kernel is obtained with  $\lambda_G(w) = P_G(w)$ , where  $P_G$  is a Markov random walk on  $G$ . In that case we have:

$$K(G_1, G_2) = P(\text{label}(W_1) = \text{label}(W_2)),$$

where  $W_1$  and  $W_2$  are two independant random walks on  $G_1$  and  $G_2$ , respectively (Kashima et al., 2003).

- The geometric walk kernel is obtained (when it converges) with  $\lambda_G(w) = \beta^{\text{length}(w)}$ , for  $\beta > 0$ . In that case the feature space is of infinite dimension (Gärtner et al., 2003).

# Computation of walk kernels

## Proposition

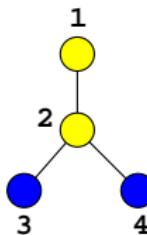
These three kernels ( $n$ th-order, random and geometric walk kernels) can be computed efficiently in **polynomial time**.

# Product graph

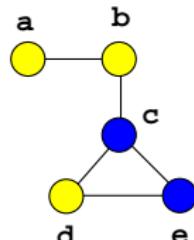
## Definition

Let  $G_1 = (V_1, E_1)$  and  $G_2 = (V_2, E_2)$  be two graphs with labeled vertices. The **product graph**  $G = G_1 \times G_2$  is the graph  $G = (V, E)$  with:

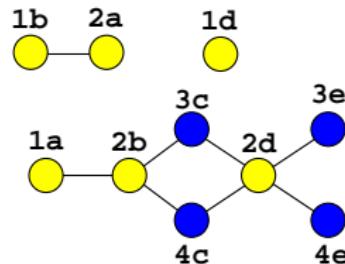
- ①  $V = \{(v_1, v_2) \in V_1 \times V_2 : v_1 \text{ and } v_2 \text{ have the same label}\}$  ,
- ②  $E = \{((v_1, v_2), (v'_1, v'_2)) \in V \times V : (v_1, v'_1) \in E_1 \text{ and } (v_2, v'_2) \in E_2\}$ .



**G1**



**G2**



**$G_1 \times G_2$**

# Walk kernel and product graph

## Lemma

There is a **bijection** between:

- ① The **pairs of walks**  $w_1 \in \mathcal{W}_n(G_1)$  and  $w_2 \in \mathcal{W}_n(G_2)$  with the **same label sequences**,
- ② The **walks on the product graph**  $w \in \mathcal{W}_n(G_1 \times G_2)$ .

# Walk kernel and product graph

## Lemma

There is a **bijection** between:

- ① The **pairs of walks**  $w_1 \in \mathcal{W}_n(G_1)$  and  $w_2 \in \mathcal{W}_n(G_2)$  with the **same label sequences**,
- ② The **walks on the product graph**  $w \in \mathcal{W}_n(G_1 \times G_2)$ .

## Corollary

$$\begin{aligned} K_{\text{walk}}(G_1, G_2) &= \sum_{s \in \mathcal{S}} \Phi_s(G_1) \Phi_s(G_2) \\ &= \sum_{(w_1, w_2) \in \mathcal{W}(G_1) \times \mathcal{W}(G_2)} \lambda_{G_1}(w_1) \lambda_{G_2}(w_2) \mathbf{1}(l(w_1) = l(w_2)) \\ &= \sum_{w \in \mathcal{W}(G_1 \times G_2)} \lambda_{G_1 \times G_2}(w). \end{aligned}$$

## Computation of the $n$ th-order walk kernel

- For the  $n$ th-order walk kernel we have  $\lambda_{G_1 \times G_2}(w) = 1$  if the length of  $w$  is  $n$ , 0 otherwise.
- Therefore:

$$K_{nth\text{-}order}(G_1, G_2) = \sum_{w \in \mathcal{W}_n(G_1 \times G_2)} 1.$$

- Let  $A$  be the adjacency matrix of  $G_1 \times G_2$ . Then we get:

$$K_{nth\text{-}order}(G_1, G_2) = \sum_{i,j} [A^n]_{i,j} = \mathbf{1}^\top A^n \mathbf{1}.$$

- Computation in  $O(n|G_1||G_2|d_1 d_2)$ , where  $d_i$  is the maximum degree of  $G_i$ .

# Computation of random and geometric walk kernels

- In both cases  $\lambda_G(w)$  for a walk  $w = v_1 \dots v_n$  can be decomposed as:

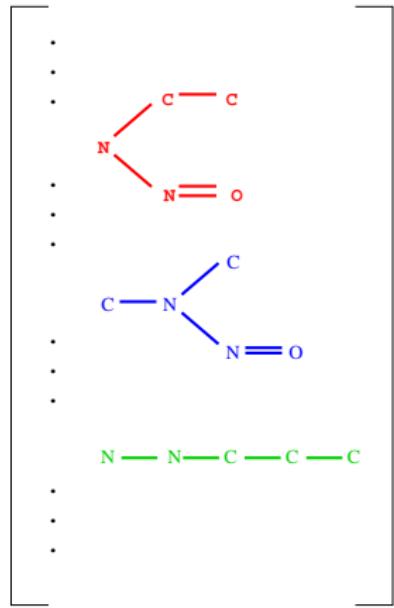
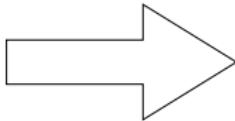
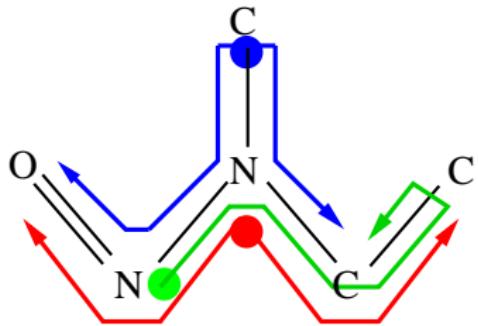
$$\lambda_G(v_1 \dots v_n) = \lambda^i(v_1) \prod_{i=2}^n \lambda^t(v_{i-1}, v_i).$$

- Let  $\Lambda_i$  be the vector of  $\lambda^i(v)$  and  $\Lambda_t$  be the matrix of  $\lambda^t(v, v')$ :

$$\begin{aligned} K_{\text{walk}}(G_1, G_2) &= \sum_{n=1}^{\infty} \sum_{w \in \mathcal{W}_n(G_1 \times G_2)} \lambda^i(v_1) \prod_{i=2}^n \lambda^t(v_{i-1}, v_i) \\ &= \sum_{n=0}^{\infty} \Lambda_i \Lambda_t^n \mathbf{1} \\ &= \color{red}{\Lambda_i (I - \Lambda_t)^{-1} \mathbf{1}} \end{aligned}$$

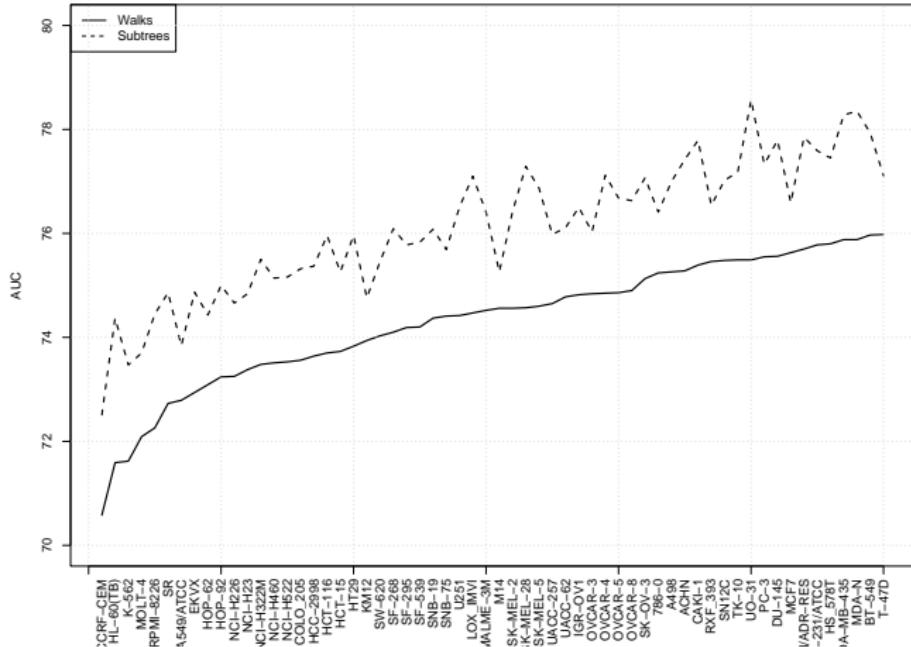
- Computation in  $O(|G_1|^3 |G_2|^3)$

Extension: branching walks (Ramon and Gärtner, 2003;  
Mahé and Vert, 2009)



$$\mathcal{T}(v, n+1) = \sum_{R \subset \mathcal{N}(v)} \prod_{v' \in R} \lambda_t(v, v') \mathcal{T}(v', n),$$

# 2D Subtree vs walk kernels

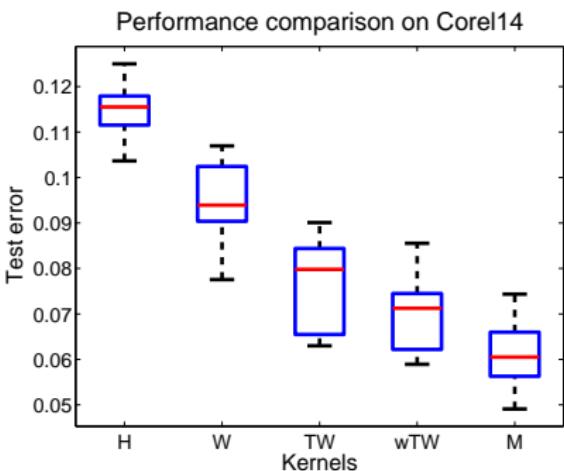


Screening of inhibitors for 60 cancer cell lines.

# Image classification (Harchaoui and Bach, 2007)

## COREL14 dataset

- 1400 natural images in 14 classes
- Compare kernel between histograms (H), walk kernel (W), subtree kernel (TW), weighted subtree kernel (wTW), and a combination (M).



# Outline

- 1 Learning in high dimension
- 2 Learning with  $\ell_2$  regularization

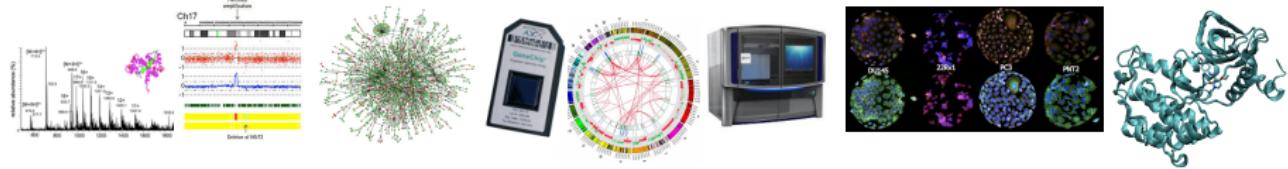
- Ridge regression
- Ridge logistic regression
- Linear hard-margin SVM
- Interlude: quick notes on constrained optimization
- Back to hard-margin SVM
- Soft-margin SVM
- Large-margin classifiers

- 3 Learning with kernels

- Kernel methods
- Positive definite kernels and RKHS
- Kernel examples
- Multiple Kernel Learning (MKL)

- 4 Conclusion

# Motivation



- We have seen how to make learning algorithms given a kernel  $K$  on some data space  $\mathcal{X}$
- Often we may have **several possible kernels**:
  - by **varying the kernel type or parameters** on a given description of the data (eg, linear, polynomial, Gaussian kernels with different bandwidths...)
  - because we have **different views of the same data**, eg, a protein can be characterized by its sequence, its structure, its mass spectrometry profile...
- How to **choose or integrate** different kernels in a learning task?

## Setting: learning with one kernel

- For any  $f : \mathcal{X} \rightarrow \mathbb{R}$ , let  $f^n = (f(x_1), \dots, f(x_n)) \in \mathbb{R}^n$
- Given a p.d. kernel  $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ , we learn with  $K$  by solving:

$$\min_{f \in \mathcal{H}} R(f^n) + \lambda \|f\|_{\mathcal{H}}^2, \quad (2)$$

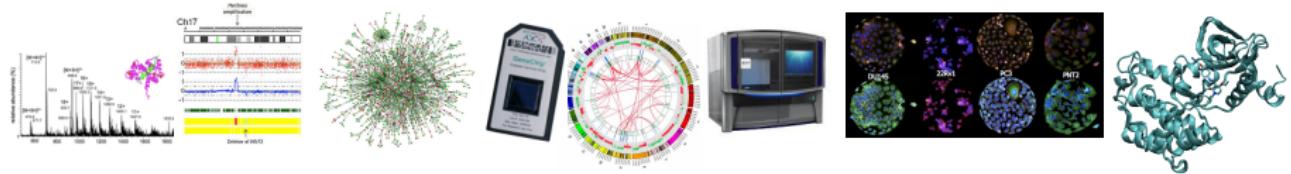
where  $\lambda > 0$  and  $R : \mathbb{R}^n \rightarrow \mathbb{R}$  is an **closed**<sup>1</sup> and **convex** empirical risk:

- $R(u) = \frac{1}{n} \sum_{i=1}^n (u_i - y_i)^2$  for kernel ridge regression
- $R(u) = \frac{1}{n} \sum_{i=1}^n \max(1 - y_i u_i, 0)$  for SVM
- $R(u) = \frac{1}{n} \sum_{i=1}^n \log(1 + \exp(-y_i u_i))$  for kernel logistic regression

---

<sup>1</sup> $R$  is closed if, for each  $A \in \mathbb{R}$ , the sublevel set  $\{u \in \mathbb{R}^n : R(u) \leq A\}$  is closed. For example, if  $R$  is continuous then it is closed.

# Sum kernel



## Definition

Let  $K_1, \dots, K_M$  be  $M$  kernels on  $\mathcal{X}$ . The sum kernel  $K_S$  is the kernel on  $\mathcal{X}$  defined as

$$\forall x, x' \in \mathcal{X}, \quad K_S(x, x') = \sum_{i=1}^M K_i(x, x').$$

## Sum kernel and vector concatenation

### Theorem

For  $i = 1, \dots, M$ , let  $\Phi_i : \mathcal{X} \rightarrow \mathcal{H}_i$  be a feature map such that

$$K_i(x, x') = \langle \Phi_i(x), \Phi_i(x') \rangle_{\mathcal{H}_i}.$$

Then  $K_S = \sum_{i=1}^M K_i$  can be written as:

$$K_S(x, x') = \langle \Phi_S(x), \Phi_S(x') \rangle_{\mathcal{H}_S},$$

where  $\Phi_S : \mathcal{X} \rightarrow \mathcal{H}_S = \mathcal{H}_1 \oplus \dots \oplus \mathcal{H}_M$  is the **concatenation** of the feature maps  $\Phi_i$ :

$$\Phi_S(x) = (\Phi_1(x), \dots, \Phi_M(x))^\top.$$

Therefore, summing kernels amounts to concatenating their feature space representations, which is a quite natural way to integrate different features.

## Proof

For  $\Phi_S(x) = (\Phi_1(x), \dots, \Phi_M(x))^\top$ , we easily compute:

$$\begin{aligned}\langle \Phi_S(x), \Phi_S(x') \rangle_{\mathcal{H}_S} &= \sum_{i=1}^M \langle \Phi_i(x), \Phi_i(x') \rangle_{\mathcal{H}_i} \\ &= \sum_{i=1}^M K_i(x, x') \\ &= K_S(x, x').\end{aligned}$$

# Example: data integration with the sum kernel

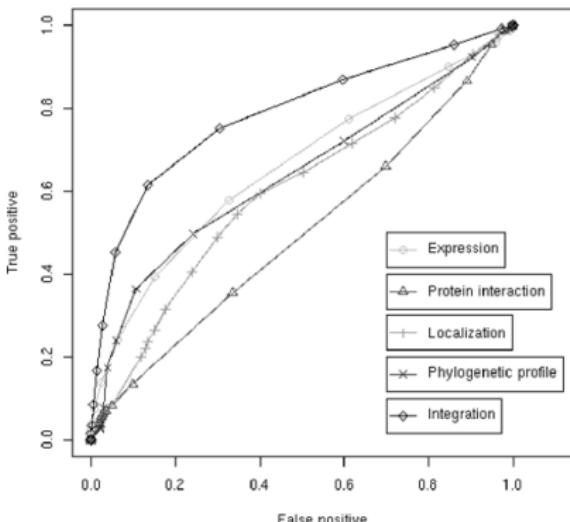


## Protein network inference from multiple genomic data: a supervised approach

Y. Yamanishi<sup>1,\*</sup>, J.-P. Vert<sup>2</sup> and M. Kanehisa<sup>1</sup>

<sup>1</sup>Bioinformatics Center, Institute for Chemical Research, Kyoto University, Gokasho, Uji, Kyoto 611-0011, Japan and <sup>2</sup>Computational Biology group, Ecole des Mines de Paris, 35 rue Saint-Honoré, 77305 Fontainebleau cedex, France

- $K_{\text{exp}}$  (Expression)
- $K_{\text{ppi}}$  (Protein interaction)
- $K_{\text{loc}}$  (Localization)
- $K_{\text{phy}}$  (Phylogenetic profile)
- $K_{\text{exp}} + K_{\text{ppi}} + K_{\text{loc}} + K_{\text{phy}}$   
(Integration)



# The sum kernel: functional point of view

## Theorem

The solution  $f^* \in \mathcal{H}_{K_S}$  when we learn with  $K_S = \sum_{i=1}^M K_i$  is equal to:

$$f^* = \sum_{i=1}^M f_i^*,$$

where  $(f_1^*, \dots, f_M^*) \in \mathcal{H}_{K_1} \times \dots \times \mathcal{H}_{K_M}$  is the solution of:

$$\min_{f_1, \dots, f_M} R \left( \sum_{i=1}^M f_i^n \right) + \lambda \sum_{i=1}^M \| f_i \|_{\mathcal{H}_{K_i}}^2.$$

## Generalization: The weighted sum kernel

### Theorem

The solution  $f^*$  when we learn with  $K_\eta = \sum_{i=1}^M \eta_i K_i$ , with  $\eta_1, \dots, \eta_M \geq 0$ , is equal to:

$$f^* = \sum_{i=1}^M f_i^*,$$

where  $(f_1^*, \dots, f_M^*) \in \mathcal{H}_{K_1} \times \dots \times \mathcal{H}_{K_M}$  is the solution of:

$$\min_{f_1, \dots, f_M} R \left( \sum_{i=1}^M f_i^n \right) + \lambda \sum_{i=1}^M \frac{\| f_i \|_{\mathcal{H}_{K_i}}^2}{\eta_i}.$$

## Proof (1/4)

$$\min_{f_1, \dots, f_M} R \left( \sum_{i=1}^M f_i^n \right) + \lambda \sum_{i=1}^M \frac{\| f_i \|_{\mathcal{H}_{K_i}}^2}{\eta_i}.$$

- $R$  being convex, the problem is strictly convex and has a **unique solution**  $(f_1^*, \dots, f_M^*) \in \mathcal{H}_{K_1} \times \dots \times \mathcal{H}_{K_M}$ .
- By the representer theorem, there exists  $\alpha_1^*, \dots, \alpha_M^* \in \mathbb{R}^n$  such that

$$f_i^*(x) = \sum_{j=1}^n \alpha_{ij}^* K_i(x_j, x).$$

- $(\alpha_1^*, \dots, \alpha_M^*)$  is the solution of

$$\min_{\alpha_1, \dots, \alpha_M \in \mathbb{R}^n} R \left( \sum_{i=1}^M K_i \alpha_i \right) + \lambda \sum_{i=1}^M \frac{\alpha_i^\top K_i \alpha_i}{\eta_i}.$$

## Proof (2/4)

- This is equivalent to

$$\min_{u, \alpha_1, \dots, \alpha_M \in \mathbb{R}^n} R(u) + \lambda \sum_{i=1}^M \frac{\alpha_i^\top K_i \alpha_i}{\eta_i} \quad \text{s.t.} \quad u = \sum_{i=1}^M K_i \alpha_i.$$

- This is equivalent to the saddle point problem:

$$\min_{u, \alpha_1, \dots, \alpha_M \in \mathbb{R}^n} \max_{\gamma \in \mathbb{R}^n} R(u) + \lambda \sum_{i=1}^M \frac{\alpha_i^\top K_i \alpha_i}{\eta_i} + 2\lambda \gamma^\top (u - \sum_{i=1}^M K_i \alpha_i).$$

- By Slater's condition, strong duality holds, meaning we can invert min and max:

$$\max_{\gamma \in \mathbb{R}^n} \min_{u, \alpha_1, \dots, \alpha_M \in \mathbb{R}^n} R(u) + \lambda \sum_{i=1}^M \frac{\alpha_i^\top K_i \alpha_i}{\eta_i} + 2\lambda \gamma^\top (u - \sum_{i=1}^M K_i \alpha_i).$$

## Proof (3/4)

- Minimization in  $u$ :

$$\min_u R(u) + 2\lambda\gamma^\top u = -\max_u \left\{ -2\lambda\gamma^\top u - R(u) \right\} = -R^*(-2\lambda\gamma),$$

where  $R^*$  is the Fenchel dual of  $R$ :

$$\forall v \in \mathbb{R}^n \quad R^*(v) = \sup_{u \in \mathbb{R}^n} u^\top v - R(u).$$

- Minimization in  $\alpha_i$  for  $i = 1, \dots, M$ :

$$\min_{\alpha_i} \left\{ \lambda \frac{\alpha_i^\top K_i \alpha_i}{\eta_i} - 2\lambda\gamma^\top K_i \alpha_i \right\} = -\lambda\eta_i \gamma^\top K_i \gamma,$$

where the minimum in  $\alpha_i$  is reached for  $\alpha_i^* = \eta_i \gamma$ .

## Proof (4/4)

- The dual problem is therefore

$$\max_{\gamma \in \mathbb{R}^n} \left\{ -R^*(-2\lambda\gamma) - \lambda\gamma^\top \left( \sum_{i=1}^M \eta_i K_i \right) \gamma \right\}.$$

- Note that if learn from a single kernel  $K_\eta$ , we get the same dual problem

$$\max_{\gamma \in \mathbb{R}^n} \left\{ -R^*(-2\lambda\gamma) - \lambda\gamma^\top K_\eta \gamma \right\}.$$

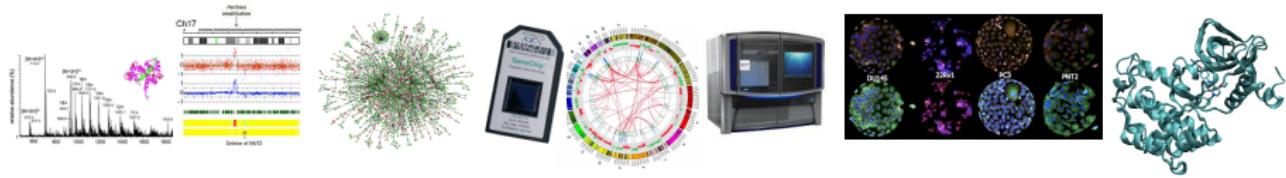
- If  $\gamma^*$  is a solution of the dual problem, then  $\alpha_i^* = \eta_i \gamma^*$  leading to:

$$\forall x \in \mathcal{X}, \quad f_i^*(x) = \sum_{j=1}^n \alpha_{ij}^* K_i(x_j, x) = \sum_{j=1}^n \eta_i \gamma_j^* K_i(x_j, x)$$

- Therefore,  $f^* = \sum_{i=1}^M f_i^*$  satisfies

$$f^*(x) = \sum_{i=1}^M \sum_{j=1}^n \eta_i \gamma_j^* K_i(x_j, x) = \sum_{j=1}^n \gamma_j^* K_\eta(x_j, x). \quad \square$$

# Learning the kernel



## Motivation

- If we know how to weight each kernel, then we can learn with the weighted kernel

$$K_{\eta} = \sum_{i=1}^M \eta_i K_i$$

- However, usually we don't know...
- Perhaps we can optimize the weights  $\eta_i$  during learning?

# An objective function for $K$

## Theorem

For any p.d. kernel  $K$  on  $\mathcal{X}$ , let

$$J(K) = \min_{f \in \mathcal{H}} \{ R(f^n) + \lambda \| f \|_{\mathcal{H}}^2 \} .$$

The function  $K \mapsto J(K)$  is **convex**.

This suggests a principled way to "learn" a kernel: define a convex set of candidate kernels, and minimize  $J(K)$  by convex optimization.

## Proof

- We have shown by strong duality that

$$J(K) = \max_{\gamma \in \mathbb{R}^n} \left\{ -R^*(-2\lambda\gamma) - \lambda\gamma^\top K\gamma \right\}.$$

- For each  $\gamma$  fixed, this is an affine function of  $K$ , hence convex
- A supremum of convex functions is convex. □

## MKL (Lanckriet et al., 2004)

- We consider the set of **convex combinations**

$$K_\eta = \sum_{i=1}^M \eta_i K_i \quad \text{with} \quad \eta \in \Sigma_M = \left\{ \eta_i \geq 0, \sum_{i=1}^M \eta_i = 1 \right\}$$

- We optimize both  $\eta$  and  $f^*$  by solving:

$$\min_{\eta \in \Sigma_M} J(K_\eta) = \min_{\eta \in \Sigma_M} \min_{f \in \mathcal{H}_{K_\eta}} \left\{ R(f^n) + \lambda \| f \|_{\mathcal{H}_{K_\eta}}^2 \right\}$$

- The problem is **jointly convex** in  $(\eta, \alpha)$  and can be solved efficiently.
- The output is both a set of weights  $\eta$ , and a predictor corresponding to the kernel method trained with kernel  $K_\eta$ .
- This method is usually called **Multiple Kernel Learning (MKL)**.

# Example: protein annotation



## A statistical framework for genomic data fusion

Gert R. G. Lanckriet<sup>1</sup>, Tijl De Bie<sup>3</sup>, Nello Cristianini<sup>4</sup>,  
Michael I. Jordan<sup>2</sup> and William Stafford Noble<sup>5,\*</sup>

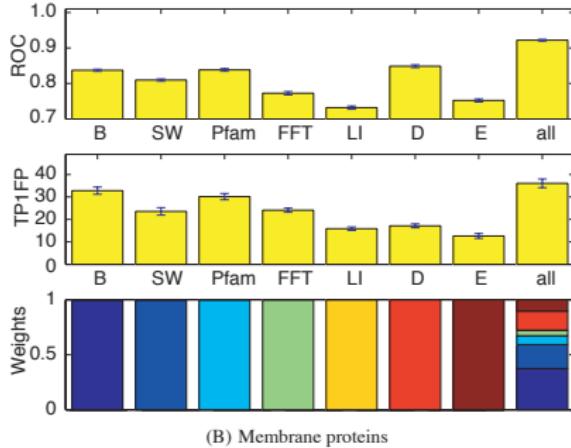
<sup>1</sup>Department of Electrical Engineering and Computer Science, <sup>2</sup>Division of Computer Science, Department of Statistics, University of California, Berkeley 94720, USA,

<sup>3</sup>Department of Electrical Engineering, ESAT-SCD, Katholieke Universiteit Leuven 3001, Belgium,

<sup>4</sup>Department of Statistics, University of California, Davis 95618, USA and

<sup>5</sup>Department of Genome Sciences, University of Washington, Seattle 98195, USA

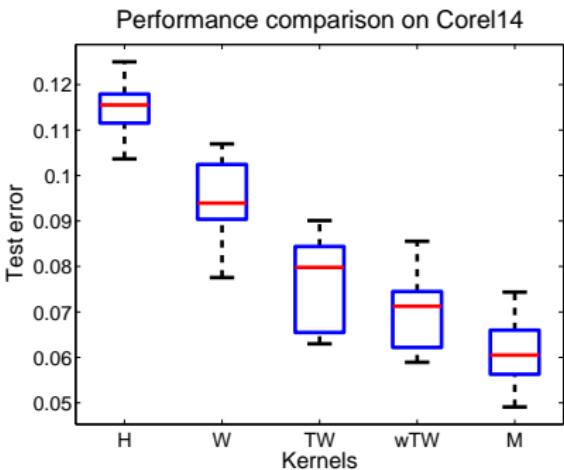
Kernel	Data	Similarity measure
$K_{SW}$	protein sequences	Smith-Waterman
$K_B$	protein sequences	BLAST
$K_{Pfam}$	protein sequences	Pfam HMM
$K_{FFT}$	hydropathy profile	FFT
$K_{LI}$	protein interactions	linear kernel
$K_D$	protein interactions	diffusion kernel
$K_E$	gene expression	radial basis kernel
$K_{RND}$	random numbers	linear kernel



# Example: Image classification (Harchaoui and Bach, 2007)

## COREL14 dataset

- 1400 natural images in 14 classes
- Compare kernel between histograms (H), walk kernel (W), subtree kernel (TW), weighted subtree kernel (wTW), and a combination by MKL (M).



## MKL revisited (Bach et al., 2004)

$$K_\eta = \sum_{i=1}^M \eta_i K_i \quad \text{with} \quad \eta \in \Sigma_M = \left\{ \eta_i \geq 0, \sum_{i=1}^M \eta_i = 1 \right\}$$

### Theorem

The solution  $f^*$  of

$$\min_{\eta \in \Sigma_M} \min_{f \in \mathcal{H}_{K_\eta}} \left\{ R(f^n) + \lambda \| f \|_{\mathcal{H}_{K_\eta}}^2 \right\}$$

is  $f^* = \sum_{i=1}^M f_i^*$ , where  $(f_1^*, \dots, f_M^*) \in \mathcal{H}_{K_1} \times \dots \times \mathcal{H}_{K_M}$  is the solution of:

$$\min_{f_1, \dots, f_M} \left\{ R \left( \sum_{i=1}^M f_i^n \right) + \lambda \left( \sum_{i=1}^M \| f_i \|_{\mathcal{H}_{K_i}} \right)^2 \right\}.$$

## Proof (1/2)

$$\begin{aligned} & \min_{\eta \in \Sigma_M} \min_{f \in \mathcal{H}_{K_\eta}} \left\{ R(f^n) + \lambda \| f \|_{\mathcal{H}_{K_\eta}}^2 \right\} \\ &= \min_{\eta \in \Sigma_M} \min_{f_1, \dots, f_M} \left\{ R \left( \sum_{i=1}^M f_i^n \right) + \lambda \sum_{i=1}^M \frac{\| f_i \|_{\mathcal{H}_{K_i}}^2}{\eta_i} \right\} \\ &= \min_{f_1, \dots, f_M} \left\{ R \left( \sum_{i=1}^M f_i^n \right) + \lambda \min_{\eta \in \Sigma_M} \left\{ \sum_{i=1}^M \frac{\| f_i \|_{\mathcal{H}_{K_i}}^2}{\eta_i} \right\} \right\} \\ &= \min_{f_1, \dots, f_M} \left\{ R \left( \sum_{i=1}^M f_i^n \right) + \lambda \left( \sum_{i=1}^M \| f_i \|_{\mathcal{H}_{K_i}} \right)^2 \right\}, \end{aligned}$$

## Proof (2/2)

where the last equality results from:

$$\forall \mathbf{a} \in \mathbb{R}_+^M, \quad \left( \sum_{i=1}^M a_i \right)^2 = \inf_{\eta \in \Sigma_M} \sum_{i=1}^M \frac{a_i^2}{\eta_i},$$

which is a direct consequence of the Cauchy-Schwarz inequality:

$$\sum_{i=1}^M a_i = \sum_{i=1}^M \frac{a_i}{\sqrt{\eta_i}} \times \sqrt{\eta_i} \leq \left( \sum_{i=1}^M \frac{a_i^2}{\eta_i} \right)^{\frac{1}{2}} \left( \sum_{i=1}^M \eta_i \right)^{\frac{1}{2}}.$$

## Algorithm: simpleMKL (Rakotomamonjy et al., 2008)

- We want to minimize in  $\eta \in \Sigma_M$ :

$$\min_{\eta \in \Sigma_M} J(K_\eta) = \min_{\eta \in \Sigma_M} \max_{\gamma \in \mathbb{R}^n} \left\{ -R^*(-2\lambda\gamma) - \lambda\gamma^\top K_\eta \gamma \right\}.$$

- For a **fixed**  $\eta \in \Sigma_M$ , we can compute  $f(\eta) = J(K_\eta)$  by using a **standard solver** for a single kernel to find  $\gamma^*$ :

$$J(K_\eta) = -R^*(-2\lambda\gamma^*) - \lambda\gamma^{*\top} K_\eta \gamma^*.$$

- From  $\gamma^*$  we can also **compute the gradient** of  $J(K_\eta)$  with respect to  $\eta$ :

$$\frac{\partial J(K_\eta)}{\partial \eta_i} = -\lambda\gamma^{*\top} K_i \gamma^*.$$

- $J(K_\eta)$  can then be minimized on  $\Sigma_M$  by a projected gradient or reduced gradient algorithm.

## Sum kernel vs MKL

- Learning with the sum kernel (uniform combination) solves

$$\min_{f_1, \dots, f_M} \left\{ R \left( \sum_{i=1}^M f_i^n \right) + \lambda \sum_{i=1}^M \|f_i\|_{\mathcal{H}_{K_i}}^2 \right\}.$$

- Learning with MKL (best convex combination) solves

$$\min_{f_1, \dots, f_M} \left\{ R \left( \sum_{i=1}^M f_i^n \right) + \lambda \left( \sum_{i=1}^M \|f_i\|_{\mathcal{H}_{K_i}} \right)^2 \right\}.$$

- Although MKL can be thought of as optimizing a convex combination of kernels, it is more correct to think of it as a penalized risk minimization estimator with the **group lasso** penalty:

$$\Omega(f) = \min_{f_1 + \dots + f_M = f} \sum_{i=1}^M \|f_i\|_{\mathcal{H}_{K_i}}.$$

## Example: ridge vs LASSO regression

- Take  $\mathcal{X} = \mathbb{R}^d$ , and for  $x = (x_1, \dots, x_d)^\top$  consider the **rank-1 kernels**:

$$\forall i = 1, \dots, d, \quad K_i(x, x') = x_i x'_i.$$

- A function  $f_i \in \mathcal{H}_{K_i}$  has the form  $f_i(x) = \beta_i x_i$ , with  $\|f_i\|_{\mathcal{H}_{K_i}} = |\beta_i|$
- The sum kernel is  $K_S(x, x') = \sum_{i=1}^d x_i x'_i = x^\top x$ , a function  $\mathcal{H}_{K_S}$  is of the form  $f(x) = \boldsymbol{\beta}^\top x$ , with norm  $\|f\|_{\mathcal{H}_{K_S}} = \|\boldsymbol{\beta}\|_{\mathbb{R}^d}$ .
- Learning with the **sum kernel** solves a **ridge regression** problem:

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^d} \left\{ R(X\boldsymbol{\beta}) + \lambda \sum_{i=1}^d \beta_i^2 \right\}.$$

- Learning with **MKL** solves a **LASSO regression** problem:

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^d} \left\{ R(X\boldsymbol{\beta}) + \lambda \left( \sum_{i=1}^d |\beta_i| \right)^2 \right\}.$$

## Extensions (Micchelli et al., 2005)

For  $r > 0$ ,  $K_\eta = \sum_{i=1}^M \eta_i K_i$  with  $\eta \in \Sigma_M^r = \left\{ \eta_i \geq 0, \sum_{i=1}^M \eta_i^r = 1 \right\}$

### Theorem

The solution  $f^*$  of

$$\min_{\eta \in \Sigma_M^r} \min_{f \in \mathcal{H}_{K_\eta}} \left\{ R(f^n) + \lambda \|f\|_{\mathcal{H}_{K_\eta}}^2 \right\}$$

is  $f^* = \sum_{i=1}^M f_i^*$ , where  $(f_1^*, \dots, f_M^*) \in \mathcal{H}_{K_1} \times \dots \times \mathcal{H}_{K_M}$  is the solution of:

$$\min_{f_1, \dots, f_M} \left\{ R \left( \sum_{i=1}^M f_i^n \right) + \lambda \left( \sum_{i=1}^M \|f_i\|_{\mathcal{H}_{K_i}}^{\frac{2r}{r+1}} \right)^{\frac{r+1}{r}} \right\}.$$

# Outline

- 1 Learning in high dimension
- 2 Learning with  $\ell_2$  regularization
  - Ridge regression
  - Ridge logistic regression
  - Linear hard-margin SVM
  - Interlude: quick notes on constrained optimization
  - Back to hard-margin SVM
  - Soft-margin SVM
  - Large-margin classifiers
- 3 Learning with kernels
  - Kernel methods
  - Positive definite kernels and RKHS
  - Kernel examples
  - Multiple Kernel Learning (MKL)
- 4 Conclusion

## In one slide...

- Learning in high dimension requires regularization, e.g., by  $\ell_2$  penalty for linear methods
- Kernels allow to transform any  $\ell_2$ -regularized linear models into a nonlinear model, thanks to the kernel trick
- There exists many kernels, which correspond to different feature spaces (of finite or infinite dimensions)
- We can combine and learn kernels, e.g., for integration of heterogeneous data
- Hot research topics
  - Large-scale ML with kernels
  - Deep kernel methods

## In one slide...

- Learning in high dimension requires regularization, e.g., by  $\ell_2$  penalty for linear methods
- Kernels allow to transform any  $\ell_2$ -regularized linear models into a nonlinear model, thanks to the kernel trick
- There exists many kernels, which correspond to different feature spaces (of finite or infinite dimensions)
- We can combine and learn kernels, e.g., for integration of heterogeneous data
- Hot research topics
  - Large-scale ML with kernels
  - Deep kernel methods

MURAKOZE

## References