SD-TSIA204 - Statistics: linear models SVMs and kernels

Ekhiñe Irurozki
Télécom Paris
Slides by Florence d'Alché-Buc

Table of contents

1. Introduction
2. Linear SVM
3. Nonlinear SVM and Kernels
4. Support Vector Regression
5. Conclusion et References
6. References

Outline

Introduction

Linear SVM

Nonlinear SVM and Kernels

Support Vector Regression

Conclusion et References

References

Statistical learning: a methodology

- The main problems to be solved :
  - **Representation problem**: determine in which representation space the data will be encoded and determine which family of mathematical functions will be used
  - **Optimization problem** : formulate the learning problem as an optimization problem (with statistical criteria), develop an optimization algorithm
  - **Model Selection**: determine the complexity of the model or any other hyperparameter prior to test
  - **Evaluation problem**: fix evaluation metrics, provide a performance estimate on test set

Machine Learning: two tasks

Let $\mathcal{S}_n = \{(x_i, y_i)\}_{i=1}^n$, a i.i.d. sample drawn from μ a joint probability distribution defined on (X, Y): X takes its values in $\mathbb{R}^d$ and Y is real-valued.

- **Learning**: get $h_n = \mathcal{A}(\mathcal{S}_n, \mathcal{H}, \ell, \lambda, \Omega)$ with
  - $\mathcal{S}_n$: training data
  - $\mathcal{H}$: class of functions
  - λ: some hyperparameter
  - ℓ: Local loss function
  - Ω: regularizing function
  - $\mathcal{A}$: learning algorithm
- **Prediction**: given x, and compute $h_n(x)$

Today's lesson (1) - starting point

Let $\mathcal{S}_n = \{(x_i, y_i)\}_{i=1}^n$, a i.i.d. sample drawn from μ a joint probability distribution defined on (X, Y): X takes its values in $\mathbb{R}^d$ and Y is real-valued.

- **Learning**: get $h_n = \mathcal{A}(\mathcal{S}_n, \mathcal{H}, \ell, \lambda, \Omega)$ with
  - $\mathcal{S}_n$: training data
  - $\mathcal{H}$: class of functions $h(\mathbf{x}) = \text{sign}(\mathbf{w}^T\mathbf{x} + b)$
  - λ: some hyperparameter
  - ℓ: Local loss function HINGE loss / Margin loss
  - Ω: regularizing function ℓ_2 norm
  - $\mathcal{A}$: learning algorithm
- **Prediction**: given x, and compute $h_n(x)$

Outline

Introduction

Linear SVM

Nonlinear SVM and Kernels

Support Vector Regression

Conclusion et References

References

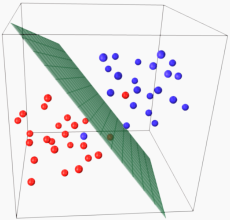
Linear Separator

Definition

Soit $\mathbf{x} \in \mathbb{R}^p$

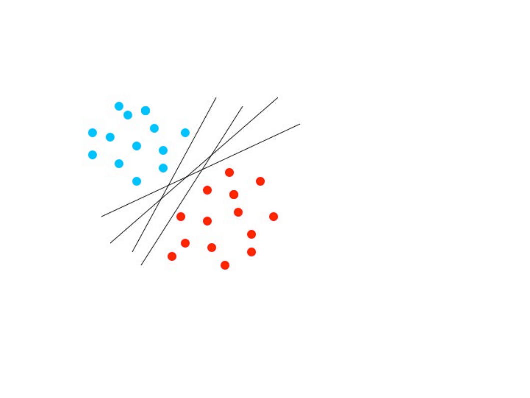
$h(\mathbf{x}) = \text{sign}(\mathbf{w}^* \phi(\mathbf{x}) + b)$

Equation $\mathbf{w}^T\mathbf{x} + b = 0$ defines an hyperplane in the euclidean space $\mathbb{R}^p$



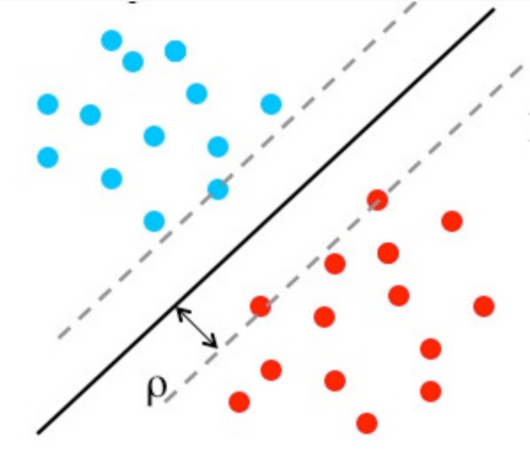
Example in 3D

Data linearly separable



What to choose ?

Margin criterion



Margin Criterion

Geometrical margin

- To separate data, let us consider a triplet of hyperplanes:
  - H: $\mathbf{w}^T\mathbf{x} + b = 0$, $H_1 : \mathbf{w}^T\mathbf{x} + b = 1$, $H_{-1} : \mathbf{w}^T\mathbf{x} + b = -1$
- We call *geometrical margin*, $\rho(\mathbf{w})$ the smallest distance between the data and Hyperplane H thus, here half of the distance between H_1 and H_{-1}
- A simple calculation gives : $\rho(\mathbf{w}) = \frac{1}{||\mathbf{w}||}$.

New objective function to optimize

How to find w and b ?

- Maximmize the margin $\rho(\mathbf{w})$ while separating the data using H_1 and H_{-1}
- Classify the blue data $(y_i = 1)$: $\mathbf{w}^T\mathbf{x}_i + b \geq 1$
- Classify the red data $(y_i = -1)$: $\mathbf{w}^T\mathbf{x}_i + b \leq -1$

Linear SVM: separable case

Optimization in the primal

$$\begin{aligned} &\underset{\mathbf{w},b}{\text{minimize}} && \frac{1}{2}\left\|\mathbf{w}\right\|^2 \\ &\text{under constraints} && y_i(\mathbf{w}^T\mathbf{x}_i + \mathrm{b}) \geq 1,\ i = 1,\dots,n. \end{aligned}$$

Référence

Boser, B. E.; Guyon, I. M.; Vapnik, V. N. (1992). "A training algorithm for optimal margin classifiers". Proceedings of the fifth annual workshop on Computational learning theory - COLT '92. p. 144.

Programming under inequality constraints

Problem of the following kind:
$\min_\theta f(\theta)$
s.c. $g(\theta) \leq 0$

- Here: $g(\theta)$: linear (afiine) constraints
- f is strictly convex

1. Lagrangian: $J(\theta, \lambda) = f(\theta) + \lambda g(\theta)$, where $\lambda \geq 0$ is a Lagrangian coefficient
2. The problem benefits from the saddle point theorem: there is a unique solution to $\min_\theta \max_\lambda J(\theta, \lambda)$.
3. We can start by solving the problem in λ or in θ, the two solutions are linked.

Programming under inequality constraints

$$\begin{aligned} \underset{\mathbf{w},b}{\text{minimize}} \quad & \frac{1}{2}\left\|\mathbf{w}\right\|^2 \\ \text{s. t.} \quad & 1 - y_i(\mathbf{w}^T\mathbf{x}_i + \mathrm{b}) \leq 0,\ i = 1, \ldots, n. \end{aligned}$$

Lagrangian

$$\mathcal{L}(\mathbf{w}, b, \alpha) = \frac{1}{2}||\mathbf{w}||^2 + \sum_i \alpha_i(1 - y_i(\mathbf{w}^T\mathbf{x}_i + \mathrm{b}))$$

$$\forall i, \alpha_i \geq 0$$

Karush-Kuhn-Tucker conditions

At the extremum, we have:

$$
\begin{aligned}
\nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w}) = \mathbf{w} - \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i = 0 \\
\nabla_b \mathcal{L}(b) = -\sum_{i=1}^{n} \alpha_i y_i = 0 \\
\forall i, \alpha_i \geq 0 \\
\forall i, \alpha_i [1 - y_i(\mathbf{w}^T \mathbf{x}_i + b)] = 0
\end{aligned}
$$

Obtaining the α_i 's : solution the dual

$$\mathcal{L}(\alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i^T \mathbf{x}_j)$$

- Maximize $\mathcal{L}$ under the constraints $\alpha_i \geq 0$ et $\sum_i \alpha_i y_i = 0, \forall i = 1, \ldots, n$
- Call for a quadratic solver

Optimal Margin Hyperplan (linear SVM)

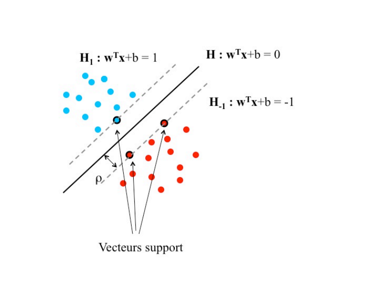
Assume the Lagrangian coefficients α_i have been found :

Linear SVM equation

$$f(\mathbf{x}) = \text{signe}(\sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b)$$

To classify a novel **x**, this classifier makes all the support data vote with an importance weight equal to $\alpha_i \mathbf{x}_i^T \mathbf{x}$ that measures how much **x** is close to the support data.

Support Vectors



Training data $\mathbf{x}_i$ such that $\alpha_i \neq 0$ belong to either H_1 or H_{-1}. Only those data, called **support vectors**, are taking into account in $\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$
NB : b is obtained by choosing one (or all) support data such that $(\alpha_i \neq 0)$

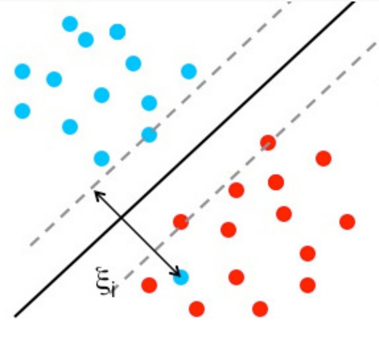
Realistic case: linear SVM in the case of nonlinearly separable data

For each training data, introduce a slack variableξ_i :

New problem in the primal space

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \frac{1}{2}\left\|\mathbf{w}\right\|^2 + C\sum_{i=1}^{n}\xi_i \\ \text{such that:} \quad & y_i(\mathbf{w}^T\mathbf{x}_i + \mathrm{b}) \geq 1 - \xi_i \; i = 1, \ldots, n. \\ & \xi_i \geq 0 \; i = 1, \ldots, n. \end{aligned}$$

Realistic case: linear SVM in the case of nonlinearly separable data



Notion of soft margin

Lagrangian

Let us introduce, Lagrangian coefficients: $\alpha_i, \xi_i, \mu_i, i = 1, \ldots, n$

$$\mathcal{L}(\mathbf{w}, b, \xi, \alpha, \xi, \mu) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_i \xi_i + \sum_i \alpha_i(1 - \xi_i - y_i(\mathbf{w}^T\mathbf{x}_i + \mathrm{b})) - \sum_i \xi_i\mu_i$$

Write the first optimality conditions ...

$$\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}) = \mathbf{w} - \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i = 0$$

$$\nabla_b\mathcal{L}(b) = -\sum_{i=1}^{n} \alpha_i y_i = 0$$

$$\forall i = 1, \ldots n, \frac{\partial \mathcal{L}(\xi)}{\partial \xi_i} = C - \alpha_i - \mu_i = 0$$

Realistic case: linear SVM in the case of nonlinearly separable data

Dual problem

$$\max_{\alpha} \quad \sum_i \alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{under the constraints} \quad 0 \leq \alpha_i \leq C \;\; i = 1, \ldots, n.$$

$$\sum_i \alpha_i y_i = 0$$

Karush-Kuhn-Tucker Conditions (KKT)

Let α^* be the solution of the dual problem:

$$\forall i, [y_i f_{w^*,b^*}(x_i) - 1 + \xi_i^*] \leq 0 \tag{1}$$

$$\forall i, \alpha_i^* \geq 0 \tag{2}$$

$$\forall i, \alpha_i^* [y_i f_{w^*,b^*}(x_i) - 1 + \xi_i^*] = 0 \tag{3}$$

$$\forall i, \mu_i^* \geq 0 \tag{4}$$

$$\forall i, \mu_i^* \xi_i^* = 0 \tag{5}$$

$$\forall i, \alpha_i^* + \mu_i^* = C \tag{6}$$

$$\forall i, \xi_i^* \geq 0 \tag{7}$$

$$\mathbf{w}^* = \sum_i \alpha_i^* y_i \mathbf{x}_i \tag{8}$$

$$\sum_i \alpha_i^* y_i = 0 \tag{9}$$

$$\tag{10}$$

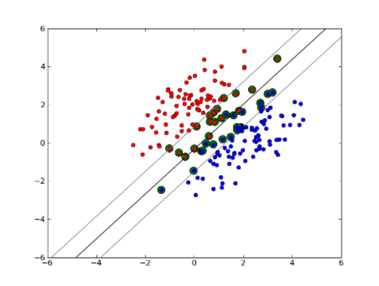
What are the support vectors?

A training datapoint x_i is called *a support vector* if $\alpha_i^* \neq 0$.

- if $\alpha_i^* = 0$ (x_i is not a support vector), then $\mu_i^* = C > 0$ and thus, $\xi_i^* = 0$: x_i is well classified
- if $0 < \alpha_i^* < C$ (x_i is a support vector) then $\mu_i^* > 0$ and thus, $\xi_i^* = 0$: x_i is such that: $y_i f(x_i) = 1$
- if $\alpha_i^* = C$ (x_i is a support vector), then $\mu_i^* = 0$, $\xi_i^* = 1 - y_i f_{w^*,b^*}(x_i)$

NB : we compute b^* by using i such that $0 < \alpha_i^* < C$

Realistic case: linear SVM with soft margin



- A training data is a support vector if either it lies on the hyperplanes H_1, H_{-1} or between H_1 and H_{-1}
- C is a hyperparameter that controls the compromise between the model complexity and the training classification error

SVM as a penalized regression problem

Optimization in the primal space

$$\min_{\mathbf{w},b} \quad \sum_{i=1}^{n}(1 - y_i(\mathbf{w}^T\mathbf{x}_i + b))_+ + \lambda\frac{1}{2}\left\|\mathbf{w}\right\|^2$$

With: $(z)_+ = max(0, z)$

$f(\mathbf{x}) = \text{signe}(h(\mathbf{x}))$

Loss function: $L(\mathbf{x}, y, h(\mathbf{x})) = (1 - yh(\mathbf{x}))_+$

$yh(\mathbf{x})$ is called the classifier margin

Optimization: subgradient or proximal approach

Outline

Introduction

Linear SVM

Nonlinear SVM and Kernels

- Kernels

Support Vector Regression

Conclusion et References

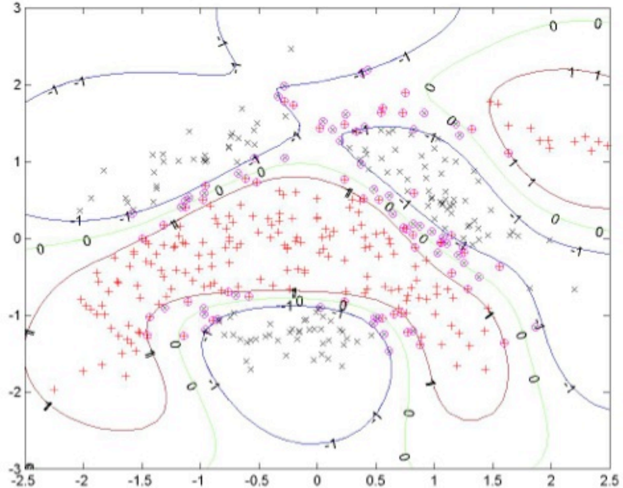
References

Today's lesson (2) - starting point

Let $\mathcal{S}_n = \{(x_i, y_i)\}_{i=1}^n$, a i.i.d. sample drawn from μ a joint probability distribution defined on (X, Y): X takes its values in $\mathbb{R}^d$ and Y is real-valued.

- **Learning**: get $h_n = \mathcal{A}(\mathcal{S}_n, \mathcal{H}, \ell, \lambda, \Omega)$ with
  - $\mathcal{S}_n$: training data
  - $\mathcal{H}$: class of functions $h(\mathbf{x}) = \text{sign}(< \mathbf{w}, \phi(\mathbf{x}) >_{\mathcal{F}} + b)$
  - λ: some hyperparameter
  - ℓ: Local loss function HINGE loss / Margin loss
  - Ω: regularizing function ℓ_2 norm
  - $\mathcal{A}$: learning algorithm
- **Prediction**: given x, and compute $h_n(x)$

Support Vector Machine : nonlinear frontiers in 2D space



Remark 1

Finding the Optimal Margin Hyperplane does involve only inner products between training data

$$
\begin{array}{ll}
\max\limits_{\alpha} & \sum\limits_{i} \alpha_i - \frac{1}{2} \sum\limits_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \\
\text{sous les contraintes} & 0 \leq \alpha_i \leq C \ i = 1, \ldots, n. \\
& \sum\limits_{i} \alpha_i y_i \ i = 1, \ldots, n.
\end{array}
$$

Let us use a feature map

If data are transformed according a nonlinear feature map $\phi : \mathcal{X} \to \mathcal{F}$, and if we know how to compute all the inner products $\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$, then we are able to learn a nonlinear decision frontier.

$$\begin{array}{ll} \max\limits_{\alpha} & \sum\limits_i \alpha_i - \frac{1}{2} \sum\limits_{i,j} \alpha_i \alpha_j y_i y_j < \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) > \\ \text{sous les contraintes} & 0 \leq \alpha_i \leq C \; i = 1, \ldots, n. \\ & \sum\limits_i \alpha_i y_i = 0. \end{array}$$

To classify a new datapoint $\mathbf{x}$, we only need to be able to calculate $\phi(\mathbf{x})^T \phi(\mathbf{x}_i)$.

Nonlinear SVM with a feature map

Choose $\phi : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}^p$ a nonlinear mapping.

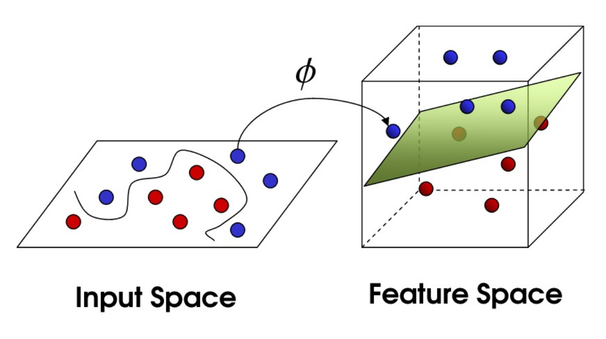
Linear SVM equation

$$f(\mathbf{x}) = \text{sign}(\sum_{i=1}^{n} \alpha_i y_i < \phi(\mathbf{x}_i), \phi(\mathbf{x}) > +b)$$

Kernel trick

If we substitute $\mathbf{x}_i^T \mathbf{x}_j$ by the image of a function k : $k(\mathbf{x}_i, \mathbf{x}_j)$ such that there exists a feature space $\mathcal{F}$ and feature map $\phi : \mathcal{X} \rightarrow \mathcal{F}$ et $\forall(\mathbf{x}, \mathbf{x}') \in \mathcal{X}, k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^T \phi(\mathbf{x}')$, then We are able to apply the same learning algorithm (Optimal Margin Hyperplane) and we get
$f(\mathbf{x}) = \text{signe}(\sum_{i=1}^{n} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) + b)$
Such functions do exist and they are called PDS kernels: positive definite symmetric kernels.

Kernel trick and feature space 1/2



Kernel trick and feature space 2/2

$$\begin{tikzcd}
\mathscr{F}_x \arrow[dr, "g"] & \\
\mathscr{X} \arrow[u, "\phi_x"] \arrow[r, "h"'] & \mathbb{R}
\end{tikzcd}$$

Kernel trick and feature space 2/2

$$\begin{tikzcd}
\mathscr{F}_{\mathrm{x}} \arrow[dr, "\mathbf{g}"] & \\
\mathscr{X} \arrow[u, "\phi_{\mathrm{x}}"] \arrow[r, "\mathrm{h}"'] & \mathbb{R}
\end{tikzcd}$$

$$h(\mathbf{x}) = \sum_{i=1}^{n} \beta_i \phi(x)^T \phi(x_i) = \sum_{i=1}^{n} \beta_i k(x, x_i),$$

with $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ a positive definite symmetric kernel.

Outline

Introduction

Linear SVM

Nonlinear SVM and Kernels

- Kernels

Support Vector Regression

Conclusion et References

References

Kernels

Definition
Let $\mathcal{X}$ be a non empty set. Let k:$\mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$, be a symmetric function. Function k is called a Positive Definite Symmetric kernel if and only if for any finite set of size m, $\{\mathbf{x}_1, \ldots, \mathbf{x}_m\} \subset \mathcal{X}$, and any column vector $\mathbf{c} \in \mathbb{R}^m$,
$\mathbf{c}^T K \mathbf{c} = \sum_{i,j=1}^{m} c_i c_j k(x_i, x_j) \geq 0$

NB: any finite Gram matrix built from k and a finite number of elements of $\mathcal{X}$ is semi-definite positive

Kernel properties

Moore-Aronzajn Theorem (1950)
Let K be PDS kernel. Then, there exists a Hilbert Space called *Feature Space* and a function called a *feature map* $\phi : \mathcal{X} \to \mathcal{F}$, such that $\forall (x, x') \in \mathcal{X}^2, \langle \phi(x), \phi(x') \rangle_{\mathcal{F}} = k(x, x')$.
Moreover, there exists a unique feature space $\phi(x) = k(\cdot, x) \in \mathcal{F}$ that satisfies the reproducing property, i.e.:

$$\forall f \in \mathcal{F}, \forall x \in \mathcal{X}, \langle k(\cdot, x), f \rangle_{\mathcal{F}} = f(x)$$

We refer to this kernel as the canonical one.
Please also notice that:

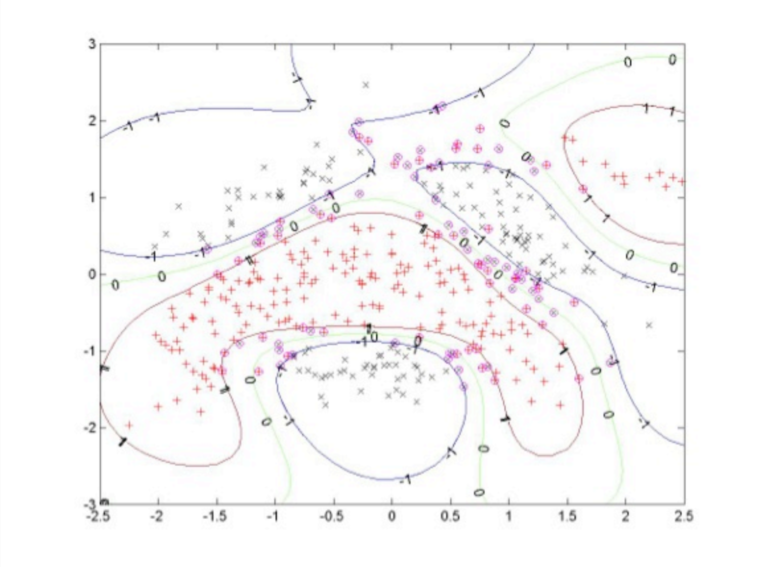
$$\forall (x, x') \in \mathcal{X}^2, \langle k(\cdot, x), k(\cdot, x') \rangle_{\mathcal{F}} = k(x, x')$$

Kernels

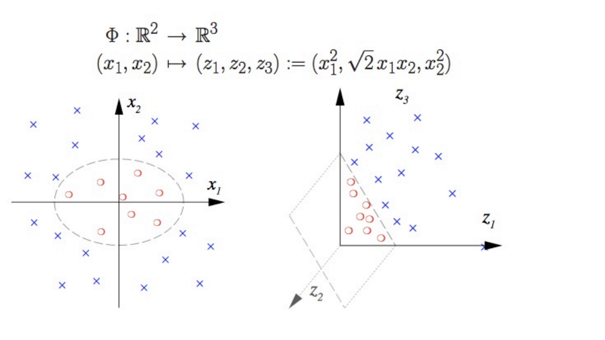
Kernel between vectors
$\forall \mathbf{x}, \mathbf{x}' \in \mathbb{R}^p$

- Trivial linear kernel : $k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'$
- Polynomial kernel : $k(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^T \mathbf{x}' + c)^d$
- Gaussian kernel : $k(\mathbf{x}, \mathbf{x}') = \exp(-\gamma ||\mathbf{x} - \mathbf{x}'||^2)$

Support Vector Machine



Example: polynomial kernel



$$\Phi : \mathbb{R}^2 \to \mathbb{R}^3$$
$$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)$$

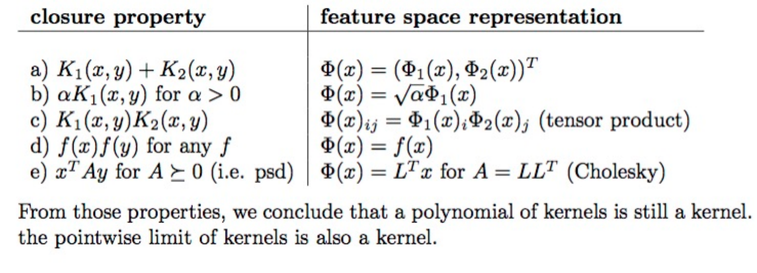
Example : polynomial kernel

Kernel trick

We notice that $\phi(\mathbf{x}_1)^T\phi(\mathbf{x}')$ can be computed without working in $\mathbb{R}^3$

We can define directly $k(\mathbf{x},\mathbf{x}') = \phi(\mathbf{x})^T\phi(\mathbf{x}') = (\mathbf{x}^T\mathbf{x}')^2$

Closure properties of kernels



closure property	feature space representation
a) $K_1(x,y) + K_2(x,y)$	$\Phi(x) = (\Phi_1(x), \Phi_2(x))^T$
b) $\alpha K_1(x,y)$ for $\alpha > 0$	$\Phi(x) = \sqrt{\alpha}\Phi_1(x)$
c) $K_1(x,y)K_2(x,y)$	$\Phi(x)_{ij} = \Phi_1(x)_i \Phi_2(x)_j$ (tensor product)
d) $f(x)f(y)$ for any f	$\Phi(x) = f(x)$
e) $x^T Ay$ for $A \succeq 0$ (i.e. psd)	$\Phi(x) = L^T x$ for $A = LL^T$ (Cholesky)

From those properties, we conclude that a polynomial of kernels is still a kernel. the pointwise limit of kernels is also a kernel.

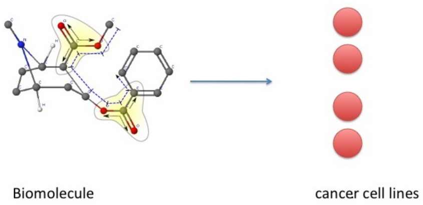
Much more interesting: kernels for complex objects

Kernels for

- **Complex (unstructured) objects**: texts, images, documents, signal, biological objects (gene, mRNA,protein, ...), functions, histograms
- **Structured objects**: sequences, trees, graphs, any composite objects

This made the success of kernels in computational biology, information retrieval (categorization for instance), but also in unexpected areas such as software metrics ....

Example: predict the property of a molecule



- **Inputs** : molecule (drug candidate)
- **Output** : activity on a cancer line (or several cancer lines)

A regression problem from structured data.

Kernel for labeled graphs

For a given length L, let us first enumerate all the paths of length $\ell \leq L$ in the training dataset (data are molecule = labeled graphs). Let m be the size of this (huge) set. For a graph, define $\phi(G) = (\phi_1(G), \ldots, \phi_m(G), \ldots, \phi_L(G))^T$ where $\phi_m(T)$ is 1 if the m^{th} path appears in the labeled graph G, and 0 otherwise.

Kernel for labeled graphs

Definition 1:

$$k_L(G, G') = < \phi(G), \phi(G') >$$

Tanimoto kernel

$$k_L^t(G, G') = \frac{k_L(G, G')}{k_L(G, G) + k_L(G', G') - k_L(G, G')}$$

idea: k_m^t calculates the ratio between the number of elements of the intersection of the two sets of paths (G and G' are seen as bags of paths) and the number of elements of the union of the two sets.
Reference: Ralaivola et al. 2005, Su et al. 2011

Convolution kernels

Definition:

Suppose that $x \in \mathcal{X}$ is a **composite structure** and $x_1, \ldots, x_D$ are its "parts" according a relation R such that $(R(x, x_1, x_2, \ldots, x_D)$ is true, with $x_d \in \mathcal{X}_d$ for each $1 \leq d \leq D$, D being a positive integer. k_d be a PDS kernel on a set $\mathcal{X} \times \mathcal{X}$, for all (x,x'), we define:

$$k_{conv}(x, x') = \sum_{(x_1,\ldots,x_d) \in R^{-1}(x), (x'_1,\ldots,x'_d) \in R^{-1}(x')} \prod_{d=1}^{D} k_d(x_d, x'_d)$$

$R^{-1}(x) =$ all decompositions $(x_1, \ldots, x_D)$ such that $(R(x, x_1, x_2, \ldots, x_D)$. k_{conv} is a PDS kernel as well. Intuitive kernel, used as a building principle for a lot of other kernels. Next, we will see two examples.

Fisher kernel

Combine the advantages of graphical models and discriminative methods
Let $x \in \mathbb{R}^p$ be the input vector of a classifier.

- Learn a generative model $p_\theta(x)$ from unlabeled data $x_1, \ldots, x_n$
- Define the Fisher vector as : $\mathbf{u}_\theta(x) = \nabla_\theta \log p_\theta(x)$
- Estimate the Fisher Information matrix of p_θ:
  $F_\theta = \mathbb{E}_{x \sim p_\theta}[\mathbf{u}_\theta(x)\mathbf{u}_\theta(x)^T]$
- **Definition**: $k_{Fisher}(x, x') = \mathbf{u}_\theta(x)^T F_\theta^{-1} \mathbf{u}_\theta(x)$

Applications
Classification of secondary structure of proteins, topic modeling in documents, image classification and object recognition, audio signal classification ... Ref: Haussler, 1998. Perronnin et al. 2013.

Kernel Design

- Use closure properties to build new kernels from existing ones
- Kernels can be defined for various objects:
    - **Structured objects**: (sets), graphs, trees, sequences, . . .
    - Unstructured data with underlying structure: texts, images, documents, signal, biological objects
- **Kernel learning**:
    - Hyperparameter learning: see Chapelle et al. 2002
    - Multiple Kernel Learning: given $k_1, \dots, k_m$, learn a convex combination $\sum_i \beta_i k_i$ of kernels (see SimpleMKL Rakotomamonjy et al. 2008, unifying view in Kloft et al. 2010)

Outline

Introduction

Linear SVM

Nonlinear SVM and Kernels

Support Vector Regression

Conclusion et References

References

Regression from ML point of view

Probabilistic and Statistical Framework 1/2

- Let X be a random vector $\mathcal{X} = \mathbb{R}^p$
- and Y be a continuous random variable$\mathcal{Y} = \mathbb{R}$
- Let $\mathbb{P}$ be the joint probability law of (X,Y)
- Let $\mathcal{S}_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, i.i.d. sample from $\mathbb{P}$.

Regression from ML point of view

Probabilistic and Statistical Framework 2/2

- Let $h : \mathbb{R}^p \to \mathbb{R} \in \mathcal{H}$, $\mathcal{H}$: some family of functions
- Let $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ be a local loss function
- Empirical risk : $R_n(h) = \frac{1}{n} \sum_i \ell(y_i, h(x_i))$, Regularizing term: $\Omega(h)$ which measures *complexity* de h.
- We search for : $\hat{h} = \arg\min_{h \in \mathcal{H}} R_n(h) + \lambda \Omega(h)$

Regression function

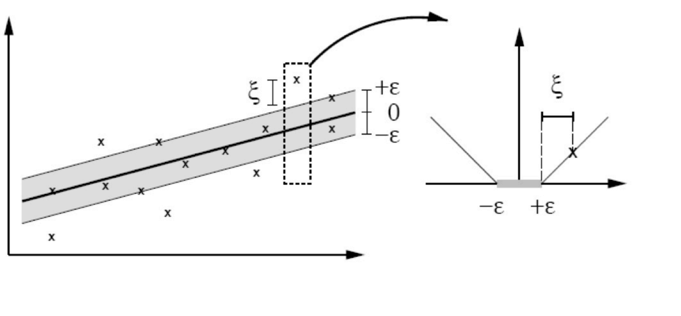
Theorem (Minimal Risk under Squared Error Loss (MSE)
When ℓ is the squared loss: $\ell(y, h(x)) = (y - h(x))^2$, the best solution for the regression problem is the so-called regression function $h^*(x) = \mathbb{E}[Y|x]$. h^* is the function that provides the minimal risk.

Proof:

Let h a predictive model.
Show that $R(h) = \mathbb{E}[(\mathbb{E}[Y|X] - h(X))^2 + R(h^*)$. Then, $R(h) \geq R(h^*)$ for any predictive model h, and therefore, $\min R(h) = R(h^*)$.

Support Vector Regression

- Extend the idea of maximal soft margin to regression
- Impose an ϵ-tube : ϵ-insensitive loss $|y' - y|_\epsilon = max(0, |y' - y| - \epsilon)$



Support Vector Regression

SVR in the primal space
Given C and ϵ

$\min_{w,b,\xi} \frac{1}{2}\left\|\mathbf{w}\right\|^2 + C\sum_i(\xi_i + \xi_i^*)$

s.c.

$\forall i = 1, \dots n, y_i - f(x_i) \leq \epsilon + \xi_i$

$\forall i = 1, \dots n, f(x_i) - y_i \leq \epsilon + \xi_i^*$

$\forall i = 1, \xi_i \geq 0, \xi_i^* \geq 0$

with $f(x) = \langle w, \phi(x)\rangle + b$

General case : ϕ is a feature map associated with a positive definite kernel k.

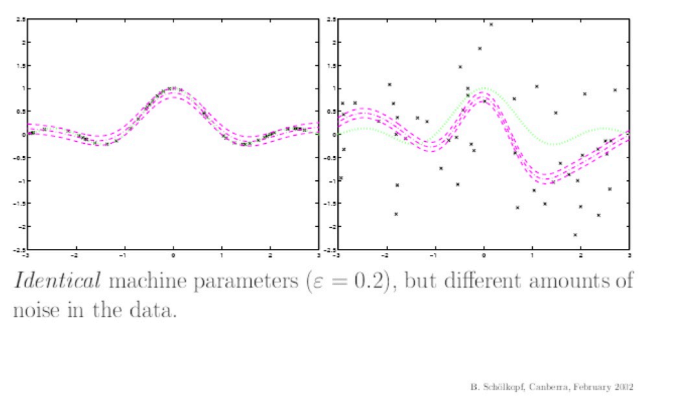
Solution in the dual

$\min_{\alpha,\alpha^*} \sum_{i,j}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)k(x_i, x_j) + \epsilon \sum_i(\alpha_i + \alpha_i^*) - \sum_i y_i(\alpha_i - \alpha_i^*)$
s.c. $\sum_i(\alpha_i - \alpha_i^*) = 0$ and $0 \leq \alpha_i \leq C$ and $0 \leq \alpha_i^* \leq C$
$w = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\phi(x_i)$

Solution

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

Support Vector Regression: example in 1D



Identical machine parameters ($\varepsilon = 0.2$), but different amounts of noise in the data.

B. Schölkopf, Canberra, February 2002

Outline

Introduction

Linear SVM

Nonlinear SVM and Kernels

Support Vector Regression

Conclusion et References

References

Supervised Classification by Support Vector Machine

Advantages

- A unique minimum, convex programming with linear constraints: exact solution !
- Some properties of the Gaussian kernel: kernel machine with a Gaussian kernel = universal approximator
- Flexibility: the kernel is chosen to be adapted to the nature of data, a systematic way to deal with **complex data**
- Multi-lass: M classifiers - One-versus-all
- Structural Risk driven : algorithm inspired from Vapnik and Chervonenkis 's works
- Can be composed with a preprocessing (first neural network layers) - kernel learning
- Kernels and their approximations are used to shed light on deep neural networks

Supervised Classification by Support Vector Machine

Drawbacks

- Kernel choice
- Solver expensive in time and memory - does not scale !
- In order to scale up kernel methods: go through approximations of Gram matrix or Random Fourier Features (approximation spectral approximation of the kernel function itself)

Conclusion

Kernel trick (working with canonical feature map: $\phi(x) = k(\cdot, x)$ and associated Hilbert Space (called reproducing Hilbert space)

- apply to many other algorithms !
- Easiest: Ridge Regression
- PCA → Kernel PCA
- CCA → Kernel CCA
- Kalman Filtering → Kernel Kalman Filter ....

Outline

Introduction

Linear SVM

Nonlinear SVM and Kernels

Support Vector Regression

Conclusion et References

References

References

- Article really cool (a bit of maths, preparation to M2) : A tutorial review of RKHS methods in Machine Learning, Hofman , Schoelkopf, Smola, 2005 (https://www.researchgate.net/publication/228827159_A_Tutorial_Review_of_RKHS_Methods_in_Machine_Learning)
- FOundations of Machine Learning, chapter about kernels, Morhi et al., MIT press (2012).
- BOSER, Bernhard E., Isabelle M. GUYON, and Vladimir N. VAPNIK, 1992. A training algorithm for optimal margin classifiers. In: COLT â92: Proceedings of the Fifth Annual Workshop on Computational Learning Theory. New York, NY, USA: ACM Press, pp. 144-152.
- CORTES, Corinna, and Vladimir VAPNIK, 1995. Support-vector networks. Machine Learning, 20(3), 273-297.