Kernel-Based Learning & Multivariate Modeling

MIRI Master - DMKM Master

Lluís A. Belanche

`belanche@cs.upc.edu`

Soft Computing Research Group

Universitat Politècnica de Catalunya

2016-2017

Kernel-Based Learning & Multivariate Modeling

Contents by lecture

Support Vector Machines

Preliminaries

- Criterion for building a two-class classifier:

  Maximize the **width of the margin** between the classes

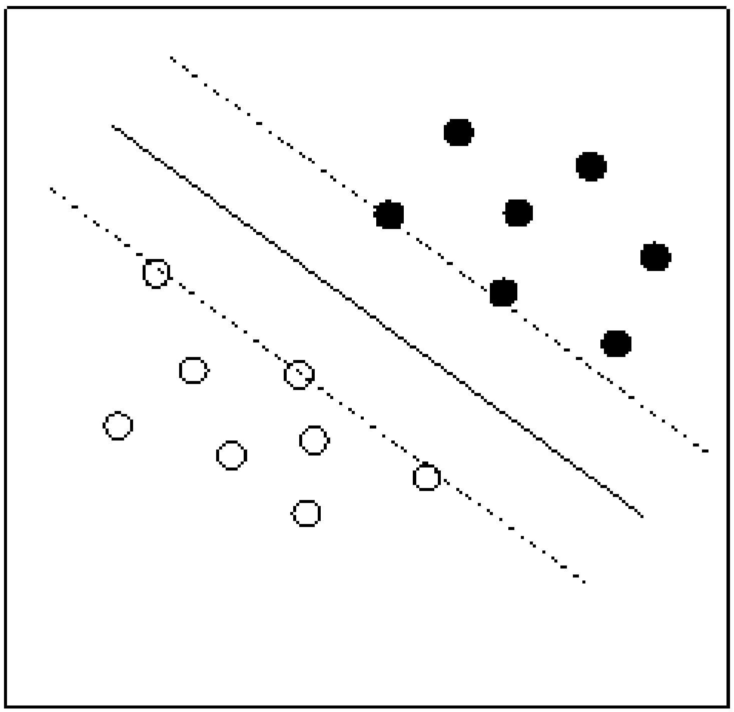
- **margin** = empty area around the decision boundary, defined by the distance to the nearest training examples

  These examples will be called the **support vectors**

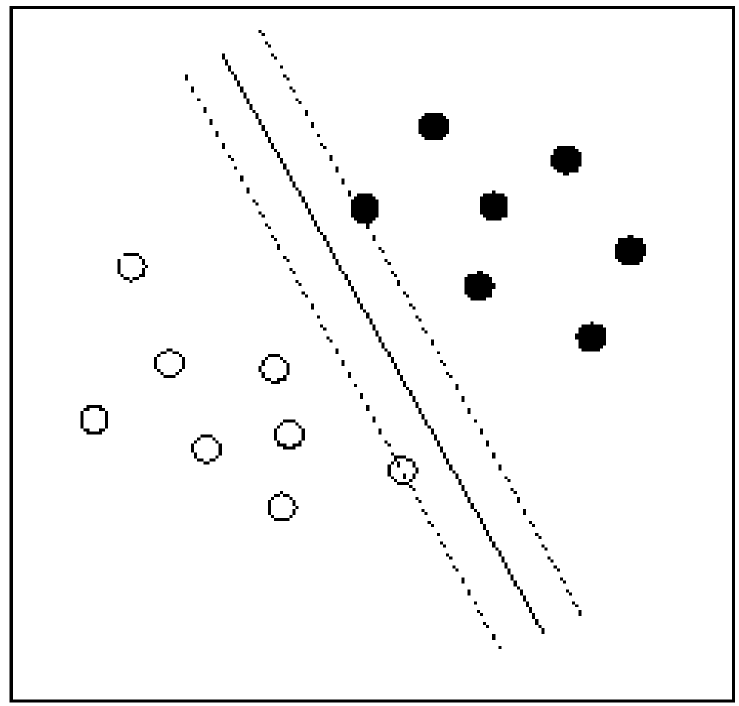
- **Goal:** find the **linear** decision boundary that maximizes this margin

Support Vector Machines

Preliminaries



(a) Larger margin



(b) Smaller margin

Which solution is more likely to lead to better **generalization**?

Support Vector Machines

Preliminaries

Working Hypotheses:

1. The data are linearly separable ("linsep") –very unlikely, but see later

2. The larger the margin, the better the generalization (an intuition)

Goal: find the separating hyperplane with the **largest margin**

Support Vector Machines

Preliminaries

For a two-class classifier, the **VC dimension** is the maximum number ϑ of points that can be separated in all possible 2^{ϑ} ways (**shattered**) by using functions representable by the classifier.

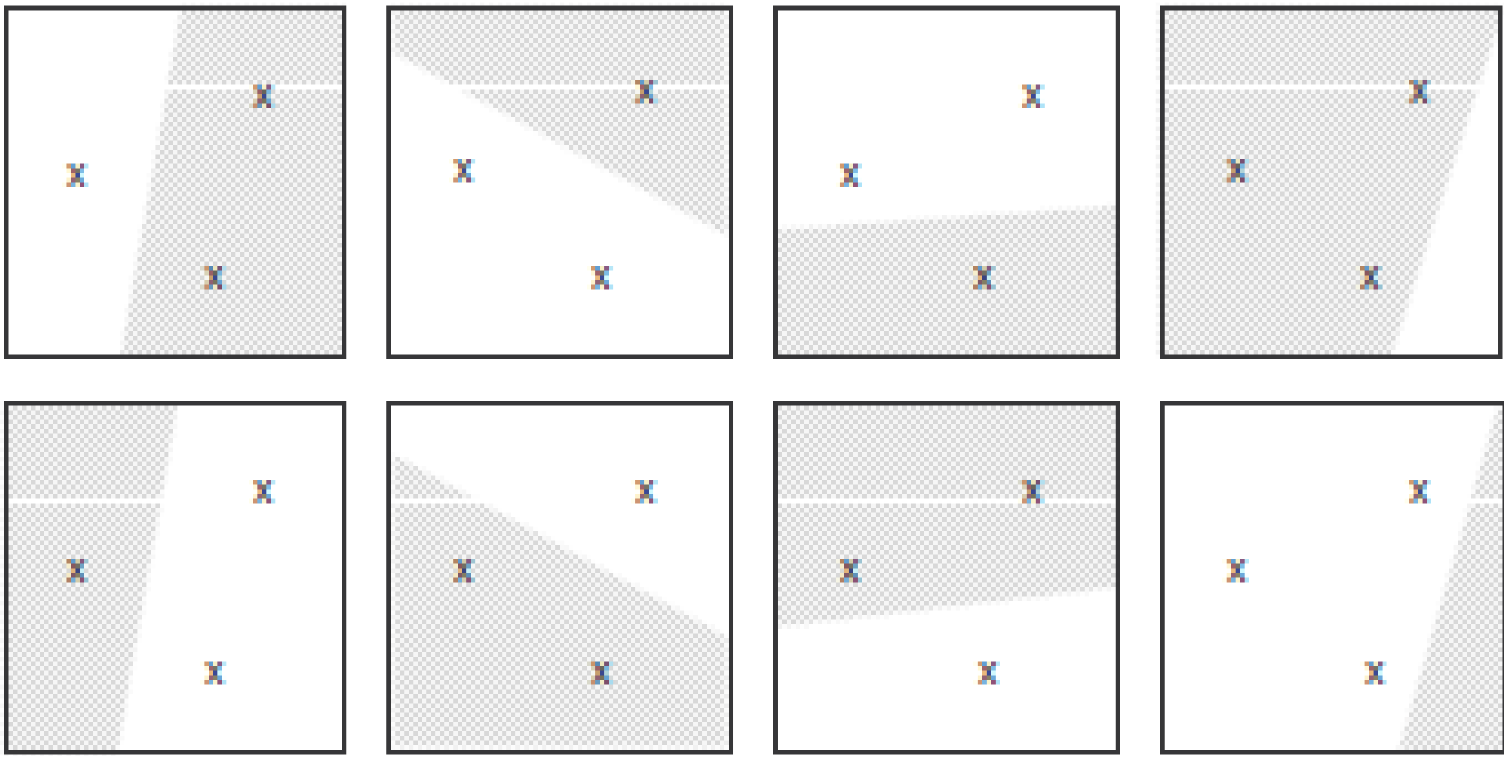
- Note it is *sufficient* that one set of ϑ points exists that can be shattered for the VC dimension to be at least ϑ

- If the VC dimension of a class is ϑ, this means there is at least one set of ϑ points that can be shattered by members of the class. It does not mean that every set of ϑ points can be shattered

- If no set of $\vartheta + 1$ points can be shattered by members of the class, then the VC dimension of the class is at most ϑ

Support Vector Machines

An example



- In $\mathbb{R}^2$ we can shatter these three points (VC dim is ≥ 3)
- No set of four or more points can be shattered (VC dim is < 4)

Support Vector Machines

Why is the VC dimension relevant?

Theorem (Vapnik and Chervonenkis, 1974). Let D be an i.i.d data sample of size N and $\mathcal{Y}$ a class of parametric binary classifiers. Let ϑ denote the VC dimension of $\mathcal{Y}$. Take $y \in \mathcal{Y}$ with empirical error $R_N(y)$ on D. For all $\eta > 0$ it holds true that, with probability at least $1 - \eta$, the true error of y is bounded by:

$$R(y) \leq R_N(y) + H(N, \vartheta, \eta)$$

where

$$H(N, \vartheta, \eta) = \sqrt{\frac{\vartheta(\ln(2N/\vartheta) + 1) - \ln(\eta/4)}{N}}$$

Support Vector Machines

Formalisation

We have a data set $D = \{(\boldsymbol{x}_1, t_1), \ldots, (\boldsymbol{x}_N, t_N)\}$, with $\boldsymbol{x}_n \in \mathbb{R}^d$ and $t_n \in \{-1, +1\}$, describing a two-class problem.

We wish to find a linear function y which best models D:

- Set up an **affine function** $g(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b$

- Obtain a **linear discriminant** as $y(\boldsymbol{x}) = \mathsf{sgn}(g(\boldsymbol{x}))$

- We would like to find $\boldsymbol{w}, b$ such that:

  $\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b > 0$, when $t_n = +1$
  $\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b < 0$, when $t_n = -1$

  that is $t_n(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b) > 0$ or simply $t_n\, g(\boldsymbol{x}_n) > 0$ $\qquad (1 \leq n \leq N)$

Support Vector Machines

Formalisation

- The quantity $t_n\, g(\boldsymbol{x}_n)$ is called the **functional margin** of $\boldsymbol{x}_n$ (there will be an "error" whenever $t_n\, g(\boldsymbol{x}_n) < 0$)

- Define the **loss** $L(t_n, \langle \boldsymbol{w}, \boldsymbol{x}_n \rangle) = \max(1 - t_n\, g(\boldsymbol{x}_n), 1)$

- Given the plane $\pi : g(\boldsymbol{x}) = 0$ ($\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b = 0$), the distance $d(\boldsymbol{x}, \pi) = \frac{|g(\boldsymbol{x})|}{\|\boldsymbol{w}\|}$ is called the **geometrical margin** of $\boldsymbol{x}$.

- The **optimal separating hyperplane** (OSH) is the one that maximizes the geometrical margin for linsep data:

$$\max_{\boldsymbol{w},b} \left\{ \min_{1 \le n \le N} d(\boldsymbol{x}_n, \pi) \right\} \qquad \text{subject to } t_n\,(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b) > 0 \ (1 \le n \le N)$$

Support Vector Machines

Formalisation

- Rescaling $\boldsymbol{w}, b$ such that $|\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b| = 1$ for the points closest to the hyperplane (the SVs), we obtain $|\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b| \geq 1$. The **support vectors** (SVs) are those $\{\boldsymbol{x}_n \ / \ |\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b| = 1\}$.

- The new loss is $\max(1 - t_n\, g(\boldsymbol{x}_n), 0) =: (1 - t_n\, g(\boldsymbol{x}_n))_+$ (**hinge loss**)

- The **margin** is twice the distance of any SV to the plane π:

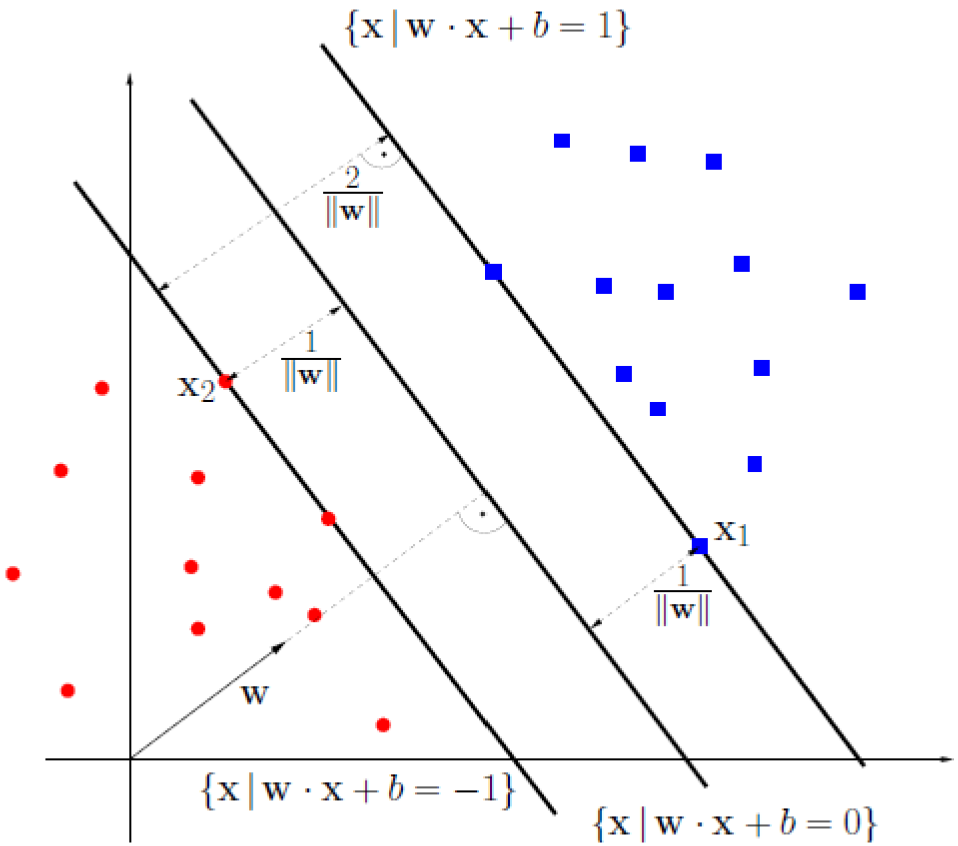
  $m = 2\, d(\boldsymbol{x}_{\mathsf{SV}}, \pi) = 2/\|\boldsymbol{w}\|$, since $|g(\boldsymbol{x}_{\mathsf{SV}})| = 1$

- Therefore we find the **canonical** OSH by solving

$$\max_{\boldsymbol{w}, b} \left\{ \frac{2}{\|\boldsymbol{w}\|} \ \ / \ \ t_n \left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b \right) \geq 1, \qquad 1 \leq n \leq N \right\}$$

Support Vector Machines

Geometrical view of the OSH



Support Vector Machines

A look on what's to come

1. The solution for $\boldsymbol{w}$ can be expressed as $\boldsymbol{w} = \sum_{n=1}^{N} t_n \alpha_n \boldsymbol{x}_n, \alpha_n \geq 0$.

   This is the **dual** form (consequence of the **representer theorem**)

2. A fraction of the training data vectors will have $\alpha_n = 0$ (**sparsity**, as a consequence of the error function chosen)

3. The $\boldsymbol{x}_n$ for which $\alpha_n > 0$ will coincide with the **support vectors**

4. The **discriminant function** is written

$$y_{\mathsf{SVM}}(\boldsymbol{x}) = \mathrm{sgn}(\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b) = \mathrm{sgn}\left(\sum_{n=1}^{N} t_n \alpha_n \langle \boldsymbol{x}, \boldsymbol{x}_n \rangle + b \right)$$

Support Vector Machines

More than an intuition

- Separating hyperplanes in $\mathbb{R}^d$ have VC dimension $d + 1$

- When we use a feature map into a very high dimension $D \in (\mathbb{N} \cup \{\infty\})$, VC dimension will grow accordingly

- If we bound the margin of the hyperplanes, we limit VC dimension (therefore, an explicit control on complexity)

Support Vector Machines

More than an intuition

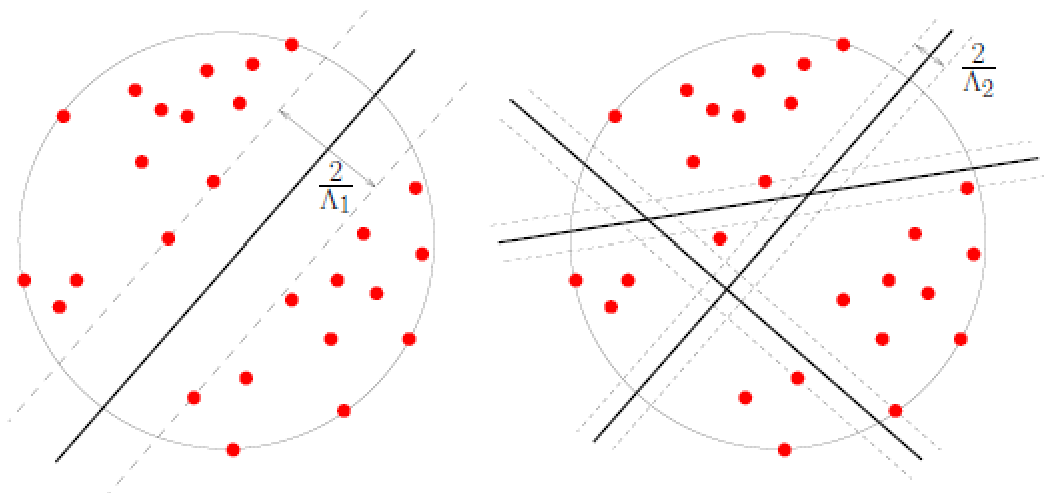
Theorem. Consider canonical hyperplanes $y(\boldsymbol{x}) = \mathrm{sgn}(\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b)$ and a data set $D = \{(\boldsymbol{x}_1, t_1), \ldots, (\boldsymbol{x}_N, t_N)\}$, with $\boldsymbol{x}_n \in \mathbb{R}^d$ and $t_n \in \{-1, +1\}$. The **subclass** of linear classifiers with margin $m \geq m_0$ has VC dimension ϑ bounded by

$$\vartheta \leq \min\left(\left\lceil \frac{R^2}{m_0^2} \right\rceil, d \right) + 1$$

where R is the radius of the smallest sphere centered at the origin containing the $\boldsymbol{x}_n$.

Support Vector Machines

More than an intuition



- Left: hyperplanes with a large margin have reduced chances to separate the data (the VC dimension is small)

- Right: smaller margins allow more separating hyperplanes (the VC dimension is large)

Support Vector Machines

Formulation

$$\textbf{minimize } (\boldsymbol{w}, b) \qquad \frac{1}{2}\|\boldsymbol{w}\|^2$$

$$\textbf{subject to } \; t_n \left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b\right) \geq 1, \qquad 1 \leq n \leq N$$

This is solved (numerically) by QP techniques:

- Quadratic (therefore convex) function subject to linear constraints

- Unique solution (or set of equivalent ones)

- Therefore, NO LOCAL MINIMA

Support Vector Machines

Formulation

For the set of constraints to be satisfied, the data set must be linsep; this is a very unrealistic requirement in practice

- We could aim at minimizing the **number of** violated constraints $|\{n \;\;/\;\; t_n\,(\langle \boldsymbol{w}, \boldsymbol{x}_n\rangle + b) < 1\}|$, but this turns out to be NP-hard ...

- Instead, we can minimize a convex function of $\boldsymbol{w}$:

$$\textbf{minimize}\ (\boldsymbol{w}, b) \qquad \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{n=1}^{N}(1 - t_n\, g(\boldsymbol{x}_n))_+$$

- Yes, the total **hinge loss**!

Support Vector Machines

Margin violations

- This problem can be rewritten as another QP, which allows for small margin violations ε_n called **slack** variables, for each $\boldsymbol{x}_n$:

$$\textbf{minimize } (\boldsymbol{w}, b, \{\varepsilon_n\}) \qquad \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{n=1}^{N} \varepsilon_n$$

$$\textbf{subject to } \; t_n\left(\langle \boldsymbol{w}, \boldsymbol{x}_n\rangle + b\right) \geq 1 - \varepsilon_n \textbf{ and } \varepsilon_n \geq 0 \; (1 \leq n \leq N)$$

- This is a **soft** margin ($\varepsilon_n > 0$ implying a functional margin $t_n g(\boldsymbol{x}_n) < 1$)
- The optimal slacks satisfy $\varepsilon_n = (1 - t_n\, g(\boldsymbol{x}_n))_+$
- For an error to occur, $\varepsilon_n > 1$ ($t_n g(\boldsymbol{x}_n) < 0$), and so $\sum_{n=1}^{N} \varepsilon_n$ is an upper bound on the number of training errors

Support Vector Machines

Excursion: Lagrange multipliers

The famous method of Lagrange multipliers allows the optimization of smooth functions subject to **equality constraints**.

The Karush, Kuhn and Tucker (KKT) method extends Lagrange's to include **inequality constraints**.

Consider the problem of minimizing $f(\boldsymbol{x})$ in a convex $\Omega \subset \mathbb{R}^d$, subject to:

- $g_j(\boldsymbol{x}) \leq 0, 1 \leq j \leq k$
- $h_j(\boldsymbol{x}) = 0, 1 \leq j \leq l$

Support Vector Machines

Excursion: Lagrange multipliers

Define the **Lagrangian** as:

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = f(\boldsymbol{x}) + \sum_{j=1}^{k} \alpha_j g_j(\boldsymbol{x}) + \sum_{j=1}^{l} \beta_j h_j(\boldsymbol{x})$$

where f, g_j, h_j are continuously differentiable functions.

Support Vector Machines

Excursion: Lagrange multipliers

Theorem. Necessary and sufficient conditions for a point $\boldsymbol{x}^*$ to be an optimum are the existence of $\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*$ such that:

1. $\frac{\partial \mathcal{L}(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)}{\partial \boldsymbol{x}} = 0$

2. $\frac{\partial \mathcal{L}(\boldsymbol{x}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)}{\partial \boldsymbol{\beta}} = 0$

3. $\alpha_j^* g_j(\boldsymbol{x}^*) = 0, 1 \leq j \leq k$ (KKT complementarity conditions)

4. $g_j(\boldsymbol{x}^*) \leq 0, 1 \leq j \leq k$

5. $\alpha_j^* \geq 0, 1 \leq j \leq k$

Support Vector Machines

SVM Lagrangian (primal)

We construct the **Lagrangian**:

$$\mathcal{L} = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{n=1}^{N} \alpha_n \Big\{ t_n \left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b \right) - 1 + \varepsilon_n \Big\} + C \sum_{n=1}^{N} \varepsilon_n - \sum_{n=1}^{N} \mu_n \varepsilon_n$$

- The $\alpha_n, \mu_n \geq 0$ are the **Lagrange multipliers**; the μ_n ensure that $\varepsilon_n \geq 0$
- The solution is a **saddle point** of $\mathcal{L}$: minimum w.r.t. $\boldsymbol{w}, b$ and the ε_n and maximum w.r.t. the α_n and μ_n

Support Vector Machines

Lagrangian form

The gradient of $\mathcal{L}$ with respect to $\boldsymbol{w}, b$ and ε_n must vanish:

$$\frac{\partial \mathcal{L}}{\partial b} = \sum_{n=1}^{N} \alpha_n t_n = 0, \qquad \frac{\partial \mathcal{L}}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum_{n=1}^{N} \alpha_n t_n \, \boldsymbol{x}_n = 0, \qquad \frac{\partial \mathcal{L}}{\partial \varepsilon_n} = C - \alpha_n - \mu_n = 0$$

In addition, the KKT complementarity conditions must hold:

$$\alpha_n \Big(t_n \left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b \right) - 1 + \varepsilon_n \Big) = 0$$

Support Vector Machines

Dual formulation

The Lagrangian $\mathcal{L}$ is convex; its optimization is equivalent to the maximization of its concave **dual problem** $\mathcal{L}_D$:

$$\textbf{maximize}\ (\{\alpha_n\}) \qquad \mathcal{L}_D = \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m t_n t_m \, \langle \boldsymbol{x}_n, \boldsymbol{x}_m \rangle$$

$$\textbf{subject to}\ \ 0 \leq \alpha_n \leq C \ (1 \leq n \leq N), \qquad \textbf{and} \quad \sum_{n=1}^{N} \alpha_n t_n = 0$$

- Neither $\mu_n, \varepsilon_n, \boldsymbol{w}, b$ appear in the dual form; maximization is only wrt the α_n
- This optimization problem is expressed *only* in terms of inner products of the data points: the dual lends itself to kernelisation
- How many free parameters? N (independent of data dimension)

Support Vector Machines

Dual formulation

A closer look at the KKT complementarity conditions:

- $\alpha_n = 0$ implies $t_n\, g(\boldsymbol{x}_n) > 1$ and $\varepsilon_n = 0$ ($\boldsymbol{x}_n$ is **not a SV**)

- $\alpha_n \in (0, C)$ implies $t_n\, g(\boldsymbol{x}_n) = 1$ and $\varepsilon_n = 0$ ($\boldsymbol{x}_n$ is a **non-bound SV**)

- $\alpha_n = C$ implies $t_n\ g(\boldsymbol{x}_n) < 1$ and $\varepsilon_n > 0$ ($\boldsymbol{x}_n$ is a **bound SV**)

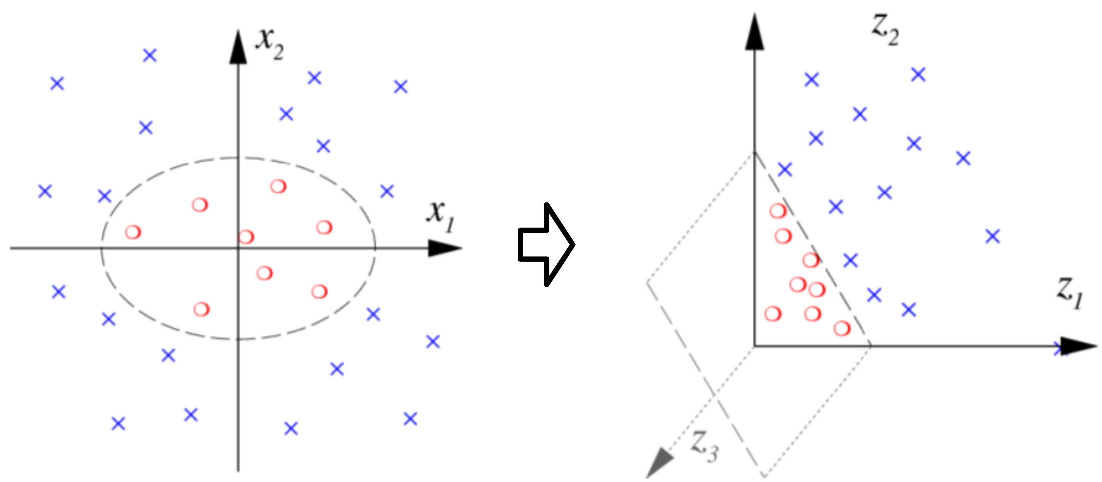
  (in particular, $\varepsilon_n > 1$ implies $\boldsymbol{x}_n$ is a **training error**)

Support Vector Machines

The SVM goes non-linear

Recall the idea of mapping input data into some Hilbert space (called the **feature space**) via a non-linear mapping $\phi : \mathcal{X} \to \mathcal{H}$

The associated kernel function is $k(\boldsymbol{u}, \boldsymbol{v}) = \langle \phi(\boldsymbol{u}), \phi(\boldsymbol{v}) \rangle_{\mathcal{H}}\,,\ \ \boldsymbol{u}, \boldsymbol{v} \in \mathcal{X}$



Support Vector Machines

SVM kernelization

- We now substitute $\boldsymbol{x}_n$ by $\phi(\boldsymbol{x}_n)$, then build the OSH in $\mathcal{H}$

- The dual of the new QP problem is formulated exactly as before, replacing $\langle \boldsymbol{x}_n, \boldsymbol{x}_m \rangle$ with $\langle \phi(\boldsymbol{x}_n), \phi(\boldsymbol{x}_m) \rangle_{\mathcal{H}} = k(\boldsymbol{x}_n, \boldsymbol{x}_m)$

- The discriminant function becomes:

$$y_{\mathsf{SVM}}(\boldsymbol{x}) = \mathsf{sgn}\left(\sum_{n=1}^{N} \alpha_n t_n k(\boldsymbol{x}, \boldsymbol{x}_n) + b \right)$$

Support Vector Machines

LOOCV bounds (I)

A rough but simple bound on LOOCV (leave-one-out CV) error can be computed as:

$$\text{LOOCV}(N) \leq \frac{1}{N}\mathbb{E}(N_{\text{SV}})$$

N_{SV} is the number of SVs for a given sample of size N
The $\mathbb{E}()$ is taken over all such samples

Support Vector Machines

LOO bounds (II)

Theorem. The LOOCV error of a stable SVM$^{(*)}$ on a set of training patterns X is bounded by $|\{n \ / \ 2\alpha_n R^2 + \varepsilon_n \geq 1\}|$, where R is an upper bound on $k(\boldsymbol{x}_n, \boldsymbol{x}_n)$.

- This quantity can be extracted easily from the solution

- This LOOCV error is an unbiased estimate of true error

$^{(*)}$ A SVM is stable if there is at least one non-bound SV (see T. Joachims; In ICML, 2000)

Support Vector Machines

Final remarks (I)

- The fact that the **OSH** is determined only by the support vectors is most remarkable, since usually this number will be small

- The **support vectors** (SVs) are:

  1. the only training examples that define the solution

  2. the most difficult examples to classify

- This means all the **relevant information** in the data set is summarized by the SVs: we would have obtained the same result by using *only* the SVs from the outset

Support Vector Machines

Final remarks (II)

- The SVM is specially well suited for "large d, low N" problems, because:

  1. complexity grows with N (non-parametric model)

  2. space requirements (the kernel matrix) also grows with N

  3. generalization error does not depend on d

- The "architecture" is determined automatically by the method (not by experimentation, as in neural networks)

Support Vector Machines

Hot topics

- Choice of **best kernel** is an open issue; **kernel design** is an active area of research

- More efficient algorithms for solving **big QP** problems are being developed

- Sometimes the **fraction of SVs** is very high (indicating a poor fit); it is possible to control this fraction directly (ν-SVMs)

- Performance usually depends on a careful choice of the external parameters: C and those of the kernel function; we need principled ways for **hyper-parameter** selection

Support Vector Machines

Where to look for more ...

- *An Introduction to Kernel-based Learning Algorithms*. K.-R. Mueller, S. Mika, G. Raetsch, K. Tsuda, and B. Schoelkopf, IEEE Neural Networks, 12(2):181-201, 2001.

- *A Tutorial on Support Vector Machines for Pattern Recognition*. Christopher Burges.
`https://research.microsoft.com/en-us/um/people/cburges/papers/svmtutorial.pdf`

- *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Bernhard Schoelkopf and Alexander J. Smola, MIT Press, 2001.

- *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Nello Cristianini and John Shawe-Taylor, Cambridge University Press, 2000.

- *Kernel Methods for Pattern Analysis*. John Shawe-Taylor and Nello Cristianini, Cambridge University Press, 2004.

- *The Nature of Statistical Learning Theory*. V. Vapnik, Springer, 2nd ed., 1999