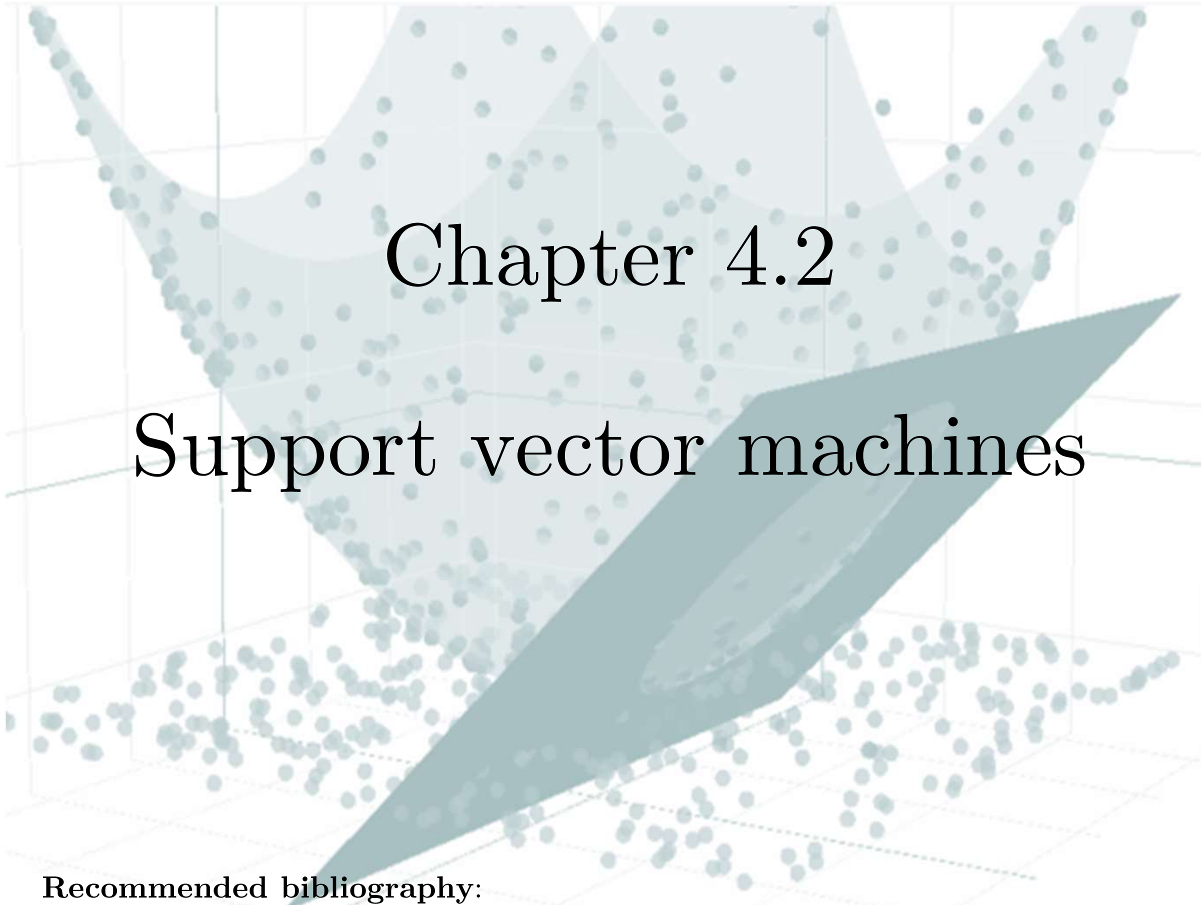


Chapter 4.2

Support vector machines

Recommended bibliography:
C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer (2006)
C. Cortes, V. Vapnik, "Support-Vector Networks", *Machine Learning*, 20, 273-297 (1995)

Credits: Some figures are taken from *Pattern Classification (2nd ed) by R. O. Duda, P. E. Hart and D. G. Stork, John Wiley & Sons, 2000* with the permission of the authors

Course overview

Pre-processing → Feature selection → Classification

Techniques
PCA (lecture 3) MDA (lecture 3)

	Density function for classes	Classifier
Supervised	Parametrized Gaussian	Linear discriminant Quadratic discriminant[1] (lecture 2)
Supervised	Non-parametric	K-nearest neighbours[2] Fuzzy logic (lecture 4)
Supervised	Unknown	SVM[3] Neural networks[3] Decision trees[4] (lecture 4)
Unsupervised	Parametrized Gaussian	K-means (lecture 6)
Unsupervised	Unknown	Clustering (lecture 6)

1. Useful only if covariance matrices are not rank deficient.
2. Useful with the number of features is very large, even larger that the number of training vectors.
3. Imposes a structure to the classifier irrespective of the training data base.
4. Useful when non-numeric features are present.

CONTENTS

1. INTRODUCTION

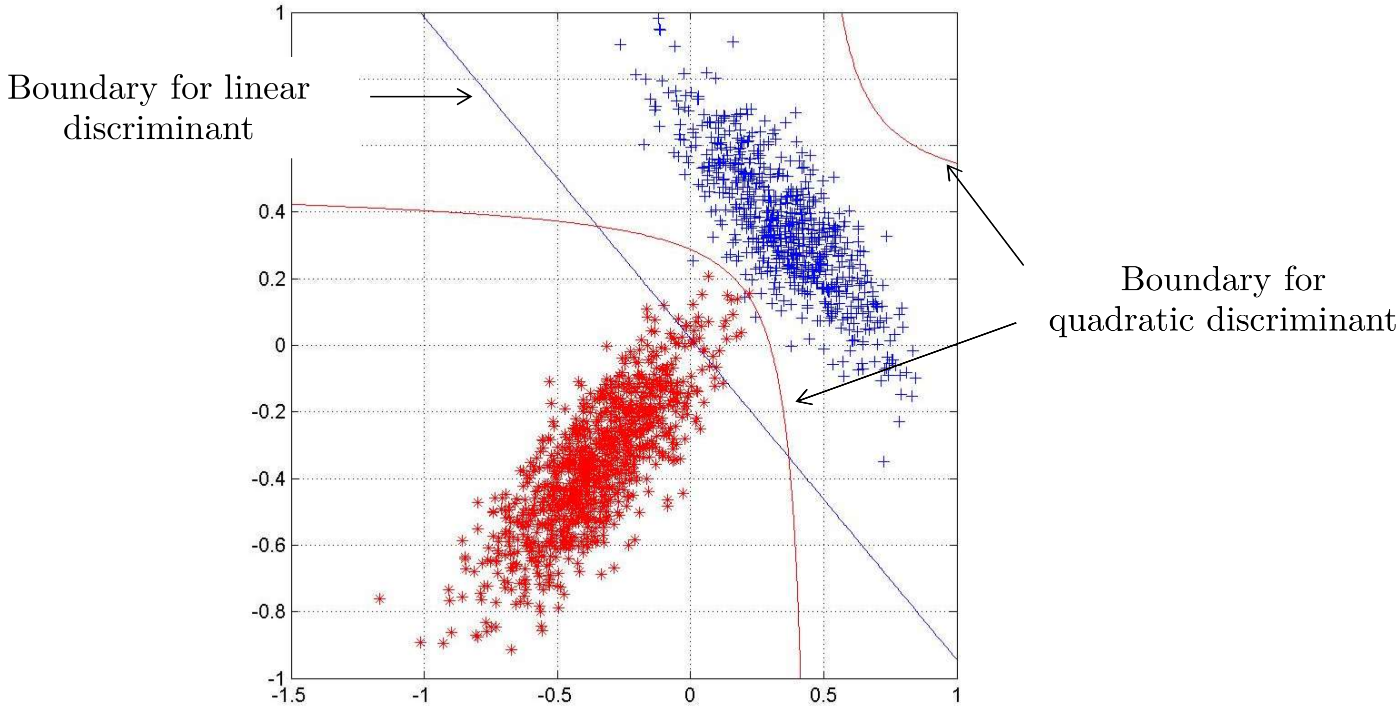
It is assumed:

- The pdf of classes are unknown
- Decision boundaries are:
  - Linear
  - Non-linear functions, using a non-linear transformation on feature vectors plus a linear discriminant

Interesting properties:

- The discriminant is a simple function of the training database: unlike KNN, we do not need all the database to classify new vectors
- The parameters of the discriminant are efficiently estimated

For Gaussian classes with different covariance matrices, linear discriminants can provide undesired boundaries...



It is apparent that a better hyperplane can be defined. Let us drop the Gaussian assumption and develop an agnostic classifier.

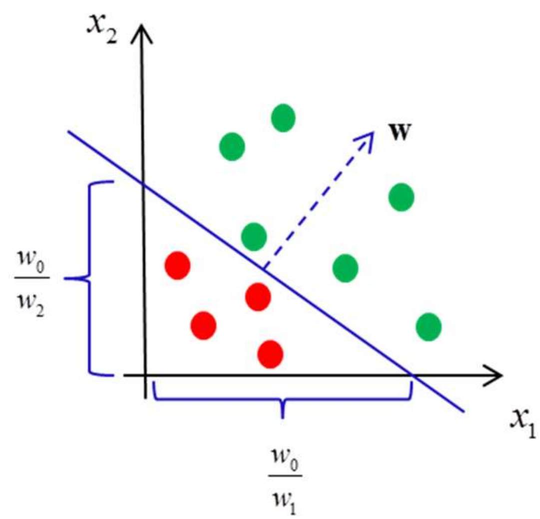
Let us face the following problems...

- Non-overlapping classes
- Overlapping classes
- Classes non-separable with a hyperplane
- Distributed databases (in a separate seminar)

2. LINEAR DISCRIMINANTS AND BOUNDARIES

Definition of a linear discriminant: $g(\mathbf{x}_i) = \mathbf{w}^T \mathbf{x}_i + w_0$

- Feature vector: $\mathbf{x}_i$
- Weighting vectors: $\mathbf{w} = [w_1 \ w_2]$
- Offset: w_0
- Class of $\mathbf{x}_i$: $y_i \in \{-1, 1\}$ if $c = 2$
- Parameters of the classifer $\begin{pmatrix} \mathbf{w} \\ w_0 \end{pmatrix}$



- c classes, c discriminant functions (for each class 1-vs-all, but it is not the only way, see section 6)

$$g_j(\mathbf{x}) = \mathbf{w}_j{}^T \mathbf{x} + w_{0,j} \qquad j = 1, \ldots, c$$

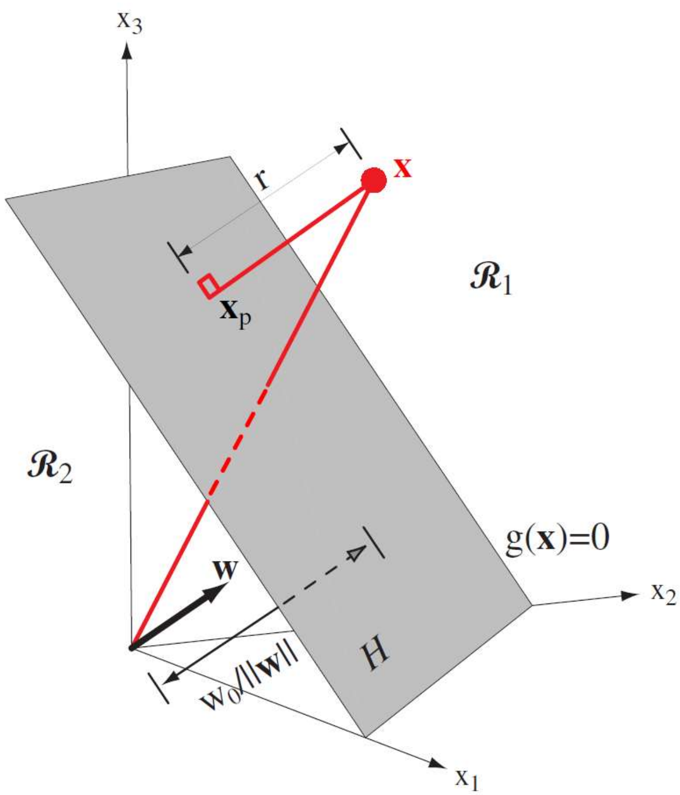
Geometric interpretation…

- Decision boundaries are hyperplanes defined by $\mathbf{w}$ and w_0
  - $\mathbf{w}$ is orthogonal to the hyperplane
  - w_0 positions the surfare on a given point
- $g(\mathbf{x})$ provides a measure of the distance of a point $\mathbf{x}$ to the hyperplane. Let us prove it…

Hyperplane H: $\mathbf{x}_0 \in \mathrm{H} \Rightarrow g(\mathbf{x}_0) = 0$

Projection of $\mathbf{x}$ on H: $\mathbf{x}_p$

Any vector $\mathbf{x}$ can be written as $\mathbf{x} = \mathbf{x}_p + r\frac{\mathbf{w}}{\|\mathbf{w}\|}$



$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0 = \mathbf{w}^T\left(\mathbf{x}_p + r\frac{\mathbf{w}}{\|\mathbf{w}\|}\right) + w_0 =$$

$$= \mathbf{w}^T\mathbf{x}_p + w_0 + \mathbf{w}^T r\frac{\mathbf{w}}{\|\mathbf{w}\|} = g(\mathbf{x}_p) + r\frac{\|\mathbf{w}\|^2}{\|\mathbf{w}\|} =$$

$$= 0 + r\|\mathbf{w}\| = \pm d(\mathbf{x}, \mathrm{H}_s)\|\mathbf{w}\|$$

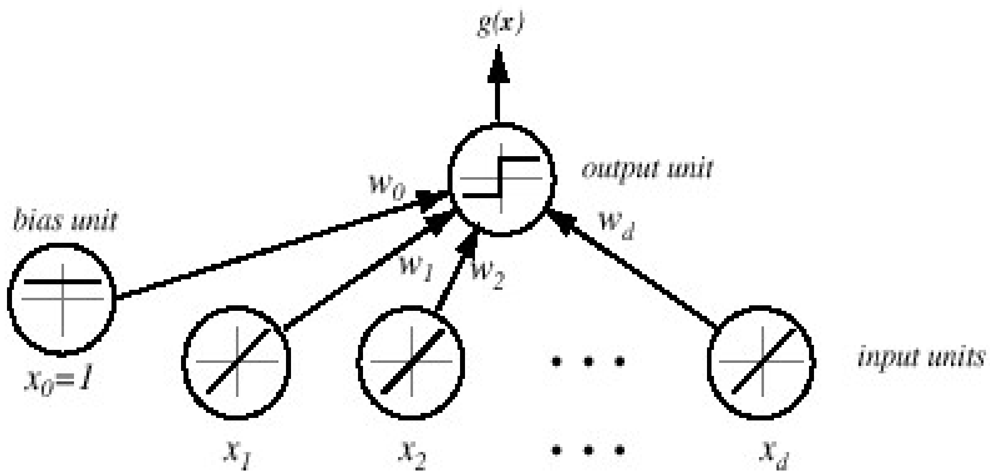
The discriminant is positive
if $\mathbf{x}$ is above the hyperplane,
and viceversa.

Distance of $\mathbf{x}$ to H: $\boxed{r = \pm d(\mathbf{x}, \mathrm{H})}$

$$\boxed{g(\mathbf{x}) = \pm d(\mathbf{x}, \mathrm{H})\|\mathbf{w}\|}$$

If generating a 2 categories classifier...

$$g(\mathbf{x}) \underset{\omega_2}{\overset{\omega_1}{\gtrless}} 0 \qquad \Rightarrow \qquad \hat{y} = sign\left(g(\mathbf{x})\right)$$



3. THE ROSENBLAT'S PERCEPTRON

Having in mind that for y_i = -1 the discriminant is $g(\mathbf{x}_i) < 0$, select $\mathbf{w}$ and w_0 such that the following function is minimized:

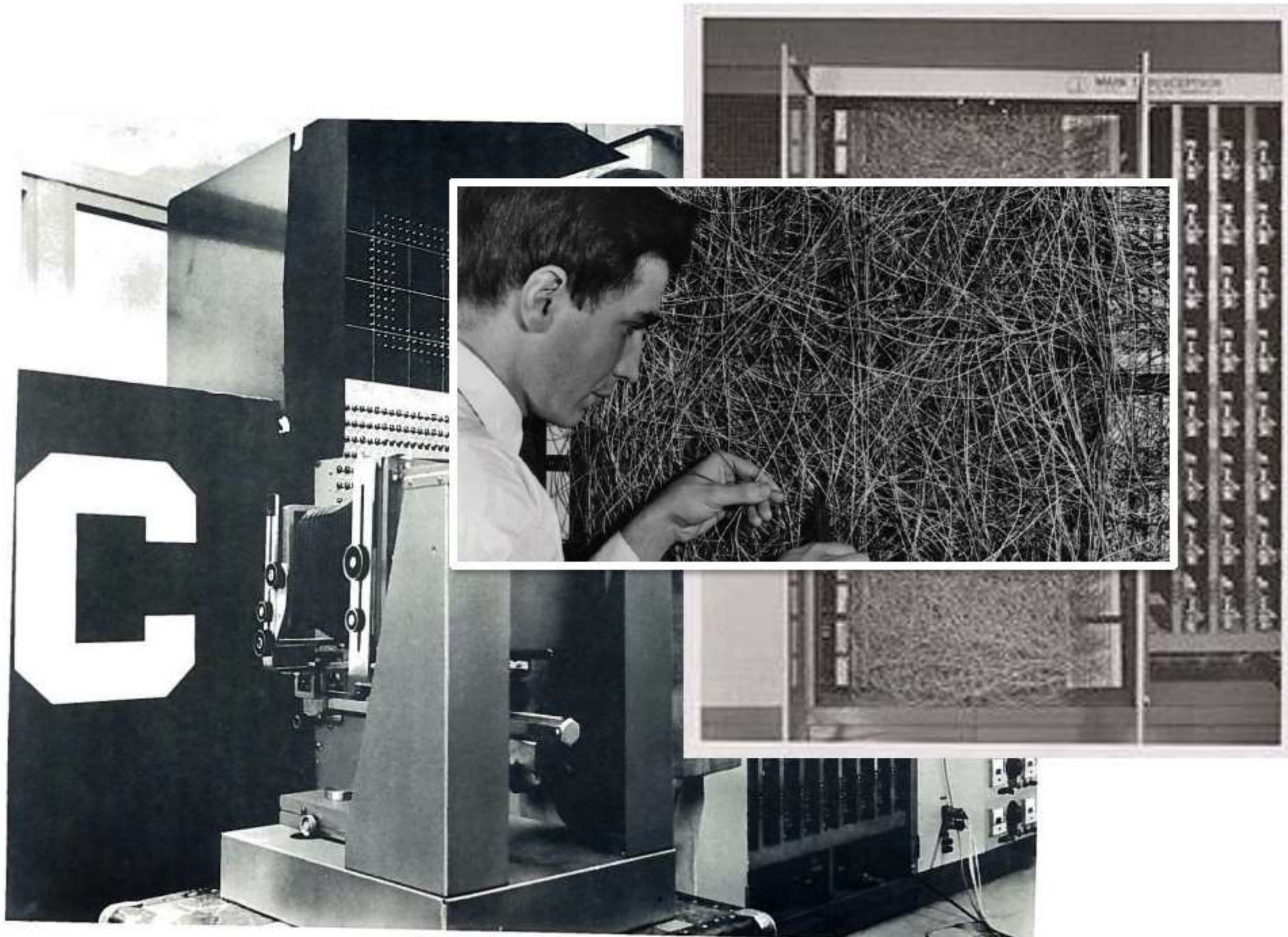
$$D(\mathbf{w}, w_0) = -\sum_{i \in \mathcal{M}} y_i g(\mathbf{x}_i) \quad \text{subject to} \quad \|\mathbf{w}\| = 1$$

where $\mathcal{M}$ is the set of all incorrectly classified vectors. We could use a gradient algorithm in the resolution:

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \mu \nabla_{\mathbf{w}} D = \mathbf{w}(k) + \mu \sum_{i \in \mathcal{M}} y_i \mathbf{x}_i$$

$$w_o(k+1) = w_o(k) - \mu \nabla_{w_o} D = w_o(k) + \mu \sum_{i \in \mathcal{M}} y_i$$

$$\mathbf{w}(k+1) \leftarrow \mathbf{w}(k+1) / \|\mathbf{w}(k+1)\|$$



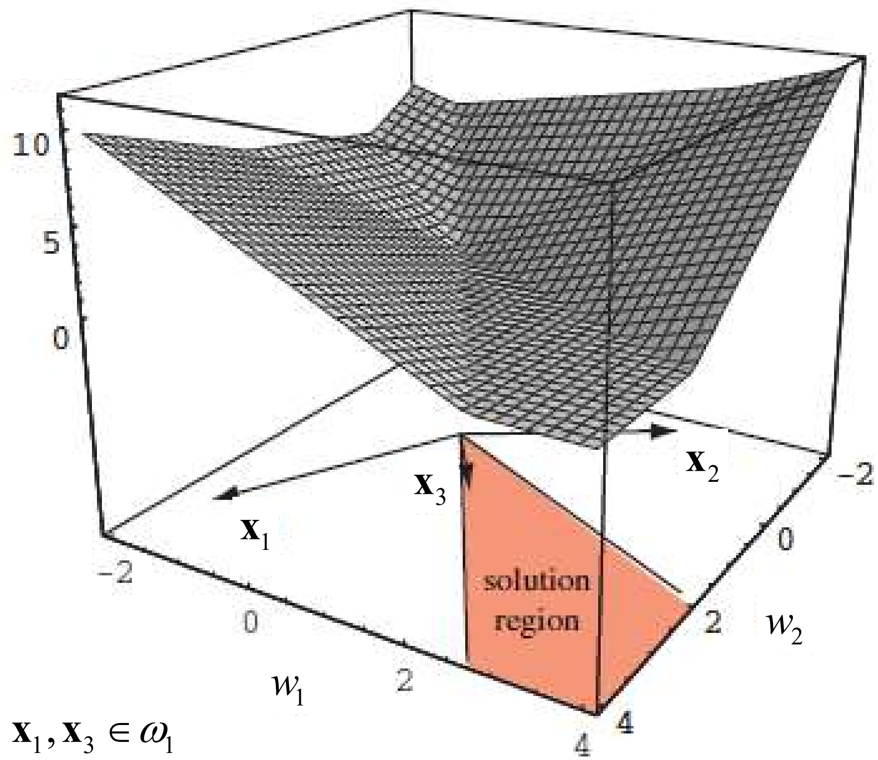
THE MARK I PERCEPTRON

(1957)

Problems of the perceptron:

1. If data are separable, there exists infinite solutions.
2. The number of iterations to convergence can be very large, especially if the distance between classes is small.
3. If classes are not separable, the algorithm does not converge.

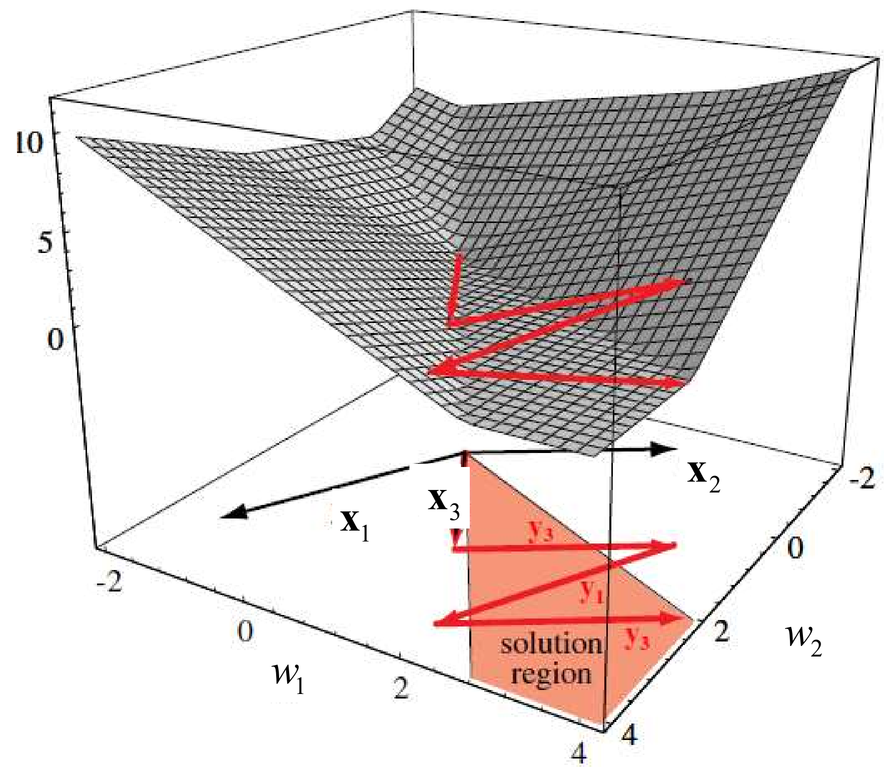
Number of solutions...



$\mathbf{x}_1, \mathbf{x}_3 \in \omega_1$

$\mathbf{x}_2 \in \omega_2$

Convergence...

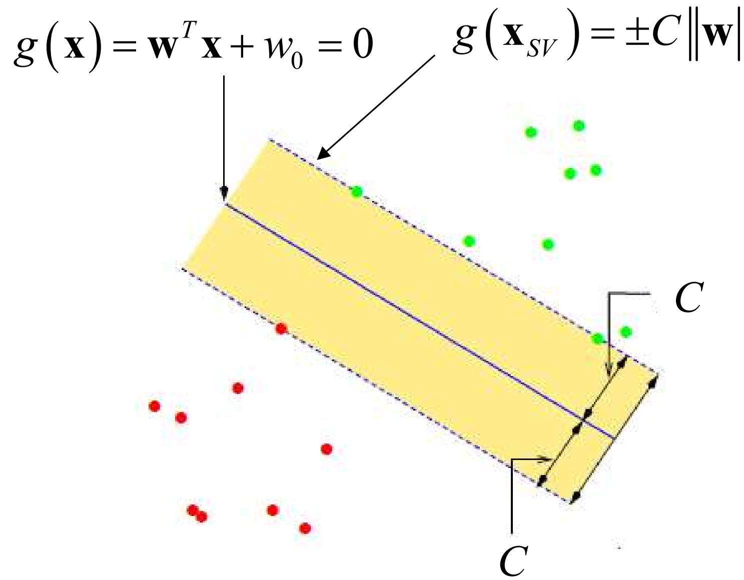


4. CLASSIFICATION BASED ON SUPPORT VECTORS

Let us face the unicity of solution:

- Compute the hyperplane that maximizes the distance to the data.
- The distance C is defined as the width of the "tube" that does not contain vectors.

- The parameters of the hyperplane are defined by those vectors on the surface of the "tube" (the *support vectors*). These are the closest vectors to the hyperplane.



4.1 LINEAR BOUNDARY WITH SEPARABLE CLASSES

Let us define the problem as:

{-1,1}, class of $\mathbf{x}_i$

$$\underset{\mathbf{w},w_o}{\text{maximize}}\ C \qquad \text{subject to} \quad \tfrac{1}{\|\mathbf{w}\|} y_i\left(\mathbf{w}^T\mathbf{x}_i + w_o\right) \geq C \qquad i = 1,\ldots,N$$

so that all vectors are at a distance larger than C of the hyperplane boundary. In this way, we guarantee:

1. A single solution for the hyperplane
2. Better performance in terms of error rate on the test data base

Using the change of variable $\|\mathbf{w}\| = 1/C$ ...

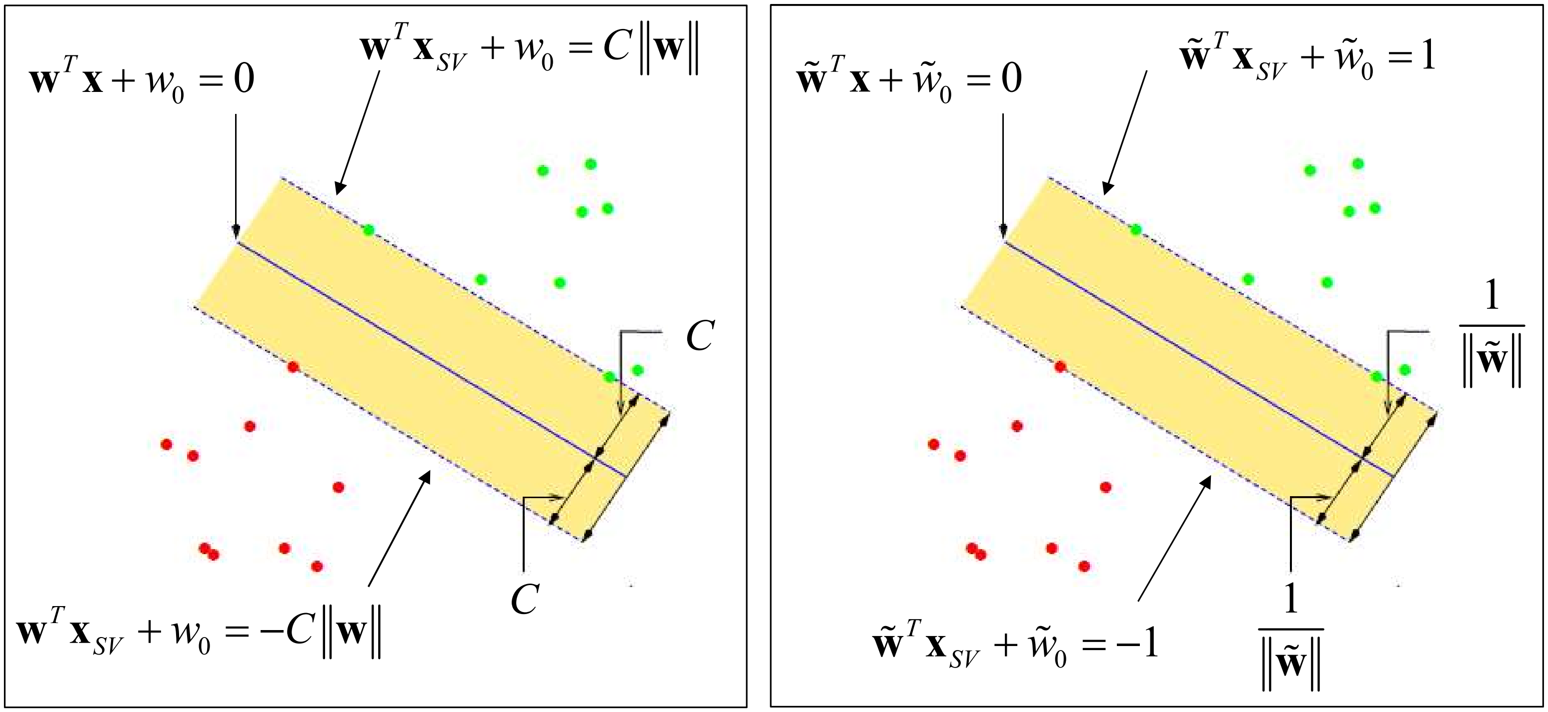
$$\underset{\mathbf{w},w_o}{\text{minimize}}\ \tfrac{1}{2}\|\mathbf{w}\|^2 \qquad \text{subject to} \quad y_i\left(\mathbf{w}^T\mathbf{x}_i + w_o\right) \geq 1 \qquad i = 1,\ldots,N$$

the Lagrangian is formulated:

Lagrange multipliers

$$\mathcal{L}\left(\mathbf{w}, w_0, \alpha_1,\ldots,\alpha_N\right) = \tfrac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N}\alpha_i\left(y_i\left(\mathbf{w}^T\mathbf{x}_i + w_0\right) - 1\right)$$

Enforcing $C\|\mathbf{w}\|=1$ is equivalent to $\tilde{\mathbf{w}} \leftarrow \mathbf{w}/\left(C\|\mathbf{w}\|\right)$ $\quad \tilde{w}_0 \leftarrow w_0/\left(C\|\mathbf{w}\|\right)$
and the geometry of the problem is not modified...

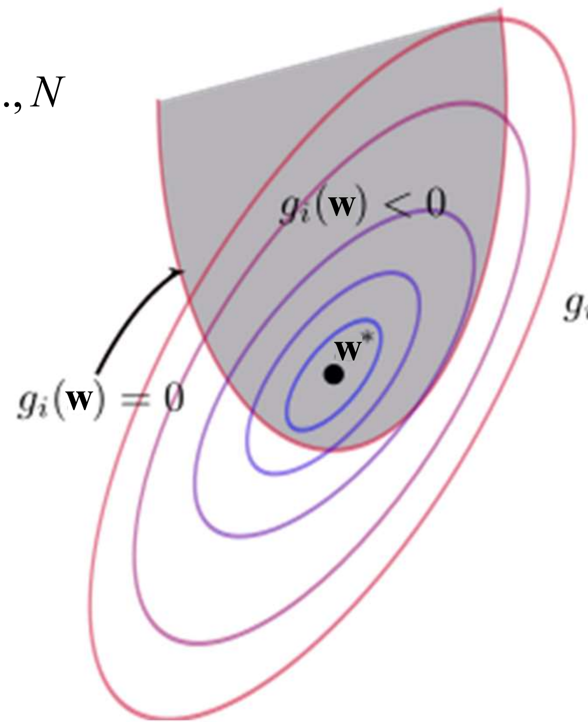


Maximizing C is equivalent to maximize $\dfrac{1}{\|\tilde{\mathbf{w}}\|}$, or minimize $\|\tilde{\mathbf{w}}\|^2$

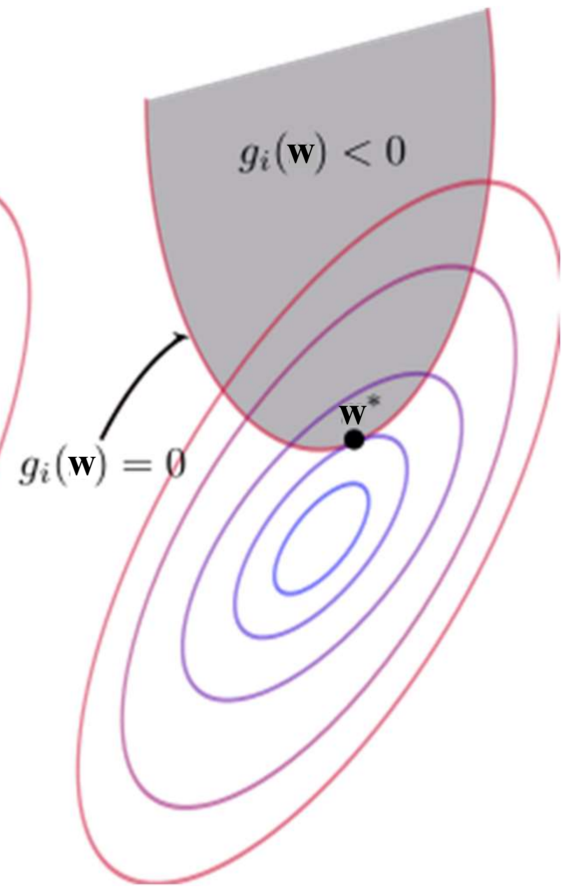
Optimization of **convex problems** has been well studied:

$$\underset{\mathbf{w}}{\text{minimize}}\; f(\mathbf{w})$$

$$\text{subject to}\;\; g_i(\mathbf{w}) \le 0 \qquad i = 1, \ldots, N$$



Solution is inside the constraints region



Solution is on the border of the constraints region

The following problem is convex:

$$\begin{aligned} &\underset{\mathbf{w}}{\text{minimize}}\ f(\mathbf{w}) \\ &\text{subject to } g_i(\mathbf{w}) \le 0 \qquad i = 1,\dots,N \end{aligned}$$

if $f(\mathbf{w})$ is convex and the region defined by $g_i(\mathbf{w})$ is convex.

Define the Lagrangian function:

Multipliers

$$\mathcal{L}(\mathbf{w},\alpha_1,\dots,\alpha_N) = f(\mathbf{w}) + \sum_{i=1}^{N} \alpha_i g_i(\mathbf{w})$$

The Karush-Kuhn-Tucker (KKT) conditions are necessary and sufficient for convex problems:

$$\begin{aligned} &\nabla_{\mathbf{w}} \mathcal{L} = \mathbf{0} \\ &\alpha_i \ge 0 \qquad i = 1,\dots,N \\ &\alpha_i g_i(\mathbf{w}) = 0 \quad i = 1,\dots,N \end{aligned}$$

SVM problema is convex, so we apply the KKT conditions:

$$\nabla_{\mathbf{w}} \mathcal{L} = \mathbf{0} \longrightarrow \mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$$

$$\nabla_{w_o} \mathcal{L} = 0 \longrightarrow 0 = \sum_{i=1}^{N} \alpha_i y_i$$

$$\alpha_i \geq 0 \quad \forall i$$

Complementarity slackness

$$\alpha_i \left(y_i \left(\mathbf{w}^T \mathbf{x}_i + w_0 \right) - 1 \right) = 0 \begin{cases} \text{for } \mathbf{x}_i \text{ on the tube surface, } y_i \left(\mathbf{w}^T \mathbf{x}_i + w_0 \right) - 1 = 0 \\ \text{for all other } \mathbf{x}_i \quad \Rightarrow \quad \alpha_i = 0 \end{cases}$$

Plug the optimum values in $\mathcal{L}\left(\mathbf{w}, w_0, \alpha_1, \ldots, \alpha_N\right)$ to get the dual problem (where variables are now the multipliers) which is always concave. Now, the problem is quadratic and is maximized on α_i (standard optimization software can be used):

$$\mathcal{L} = \sum_{i=1}^{N} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k \mathbf{x}_i^T \mathbf{x}_k \qquad \text{subject to} \quad \begin{cases} \alpha_i \geq 0 \\ \sum_{i=1}^{N} \alpha_i y_i = 0 \end{cases}$$

Using the optimum values of α, we get a solution for $\mathbf{w}$ and for w_0 using:

$$\mathbf{w}=\sum_{j=1}^{N}\alpha_j y_j \mathbf{x}_j=\sum_{j=1}^{N_{SV}}\alpha_j y_j \mathbf{x}_j$$

$$w_0=-\mathbf{w}^T\mathbf{x}_i+y_i \quad \text{or better, averaging} \quad w_0=\frac{1}{N_{SV}}\sum_{i=1}^{N_{SV}}\left(-\mathbf{w}^T\mathbf{x}_i+y_i\right)$$

where N_{SV} are the *support vectors* (a small fraction of all) for which $\alpha > 0$, that is, those on the surfare of the tube which fit this equality:

$$y_i\left(\mathbf{w}^T\mathbf{x}_i+w_0\right)=1$$

Note that for the rest of vectors:

$$y_i\left(\mathbf{w}^T\mathbf{x}_i+w_0\right)>1 \qquad \text{and} \qquad \alpha_i=0$$

The classification of $\mathbf{x}$ reduces to compute the sign of the discrimimant:

$$\hat{y}=sign\left(g\left(\mathbf{x}\right)\right)=sign\left(\mathbf{w}^T\mathbf{x}+w_0\right)=sign\left(\sum_{i=1}^{N_{SV}}\alpha_i y_i \mathbf{x}_i^T\mathbf{x}+w_0\right)$$

We can write all in a convenient matrix form...

$$\mathcal{L} = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} - \mathbf{w}^T\boldsymbol{\Phi}\boldsymbol{\alpha}_c - w_0\boldsymbol{\alpha}_c^T\mathbf{1} + \boldsymbol{\alpha}^T\mathbf{1} \qquad \boldsymbol{\alpha} = \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix}$$

$$\boldsymbol{\alpha}_c = \begin{pmatrix} \alpha_1 y_1 \\ \vdots \\ \alpha_N y_N \end{pmatrix} = \mathbf{Y}\boldsymbol{\alpha} \qquad \boldsymbol{\Phi} = \begin{pmatrix} \mathbf{x}_1 & \cdots & \mathbf{x}_N \end{pmatrix} \qquad \mathbf{Y} = diag(y_1, \ldots, y_N)$$

The gradient w.r.t. the parameters is:

$$\nabla\mathcal{L}_{\mathbf{w}} = \mathbf{w} - \boldsymbol{\Phi}\boldsymbol{\alpha}_c = 0 \quad \Rightarrow \quad \mathbf{w} = \boldsymbol{\Phi}\boldsymbol{\alpha}_c$$

$$\nabla\mathcal{L}_{w_0} = -\boldsymbol{\alpha}_c^T\mathbf{1} = 0$$

Replacing on the Lagrangian we obtain the dual form of the problem:

$$\mathcal{L} = \boldsymbol{\alpha}^T\mathbf{1} - \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{Y}^T\boldsymbol{\Phi}^T\boldsymbol{\Phi}\mathbf{Y}\boldsymbol{\alpha}$$

$$\text{subject to} \quad \begin{cases} \boldsymbol{\alpha} \geq 0 \\ \mathbf{1}^T\mathbf{Y}\boldsymbol{\alpha} = 0 \end{cases}$$

Quadratic programming problem

Steps to follow in the resolution…

1. Optimize $\mathcal{L}$ w.r.t. $\boldsymbol{\alpha}$ using numerical techniques (e.g. `fmincon` in MATLAB)

2. Obtain the weighting vector from the optimum $\boldsymbol{\alpha}$: $\mathbf{w} = \boldsymbol{\Phi}\mathbf{Y}\boldsymbol{\alpha}$

3. Obtain the offset w_o by enforcing all active restrictions, using a single vector or averaging all active restrictions equations (that is, with the support vectors):

$$y_k\left(\mathbf{w}^T\mathbf{x}_k + w_0\right) = 1 \quad \Rightarrow$$

$$w_0 = \frac{1}{N_{SV}}\sum_{i=1}^{N_{SV}}\left(y_i - \mathbf{w}^T\mathbf{x}_i\right) = \frac{1}{N_{SV}}\left(\mathbf{1}^T\mathbf{y} - \sum_{i=1}^{N_{SV}}\boldsymbol{\alpha}^T\mathbf{Y}^T\boldsymbol{\Phi}^T\mathbf{x}_i\right) =$$

$$= \frac{1}{N_{SV}}\left(\mathbf{1}^T\mathbf{y} - \boldsymbol{\alpha}^T\mathbf{Y}^T\boldsymbol{\Phi}^T\sum_{i=1}^{N_{SV}}\mathbf{x}_i\right) = \frac{1}{N_{SV}}\left(\mathbf{1}^T\mathbf{y} - \boldsymbol{\alpha}^T\mathbf{Y}^T\boldsymbol{\Phi}^T\boldsymbol{\Phi}\mathbf{1}\right)$$

This is a pre-computed scalar that does not depend on $\mathbf{x}$

4. Classification rule $\hat{y} = sign\left(\mathbf{w}^T\mathbf{x} + w_0\right) = sign\left(\underbrace{\boldsymbol{\alpha}^T\mathbf{Y}^T\boldsymbol{\Phi}^T}_{\text{pre-computed row vector, using the support vectors}}\mathbf{x} + w_0\right)$

4.2 LINEAR BOUNDARY WITH NON-SEPARABLE CLASSES

No hyperplane can separate the two classes, but we can try anyways to derive a hyperplane assuming that some training vectors can be wrongly classified:

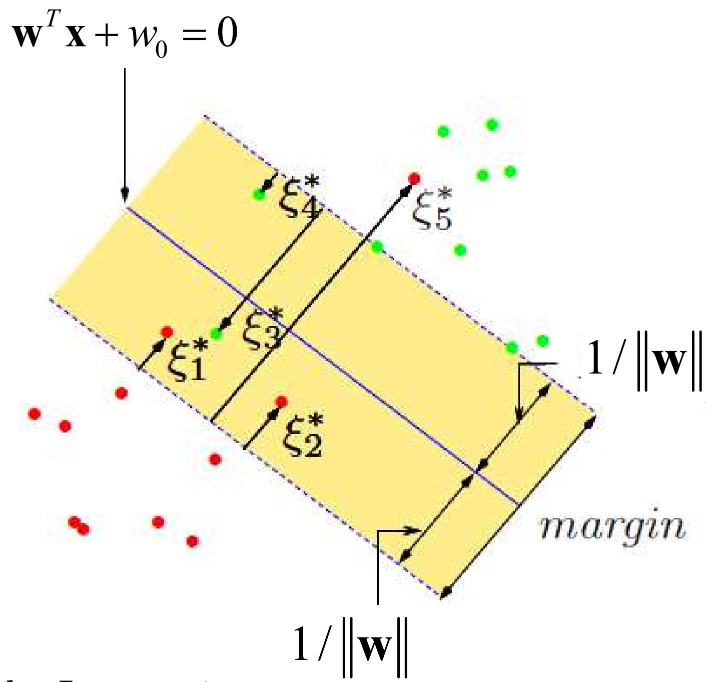
$$y_i\left(\mathbf{w}^T\mathbf{x}_i + w_o\right) \geq 1 - \xi_i \qquad i = 1,\ldots,N$$

and introducing a penalization for the non-null values of ξ:

$$\underset{\xi,\mathbf{w},w_o}{\text{minimize}} \ \tfrac{1}{2}\left\|\mathbf{w}\right\|^2 + P\sum_{i=1}^{N}\xi_i \quad \text{subject to} \begin{cases} y_i\left(\mathbf{w}^T\mathbf{x}_i + w_o\right) \geq 1 - \xi_i \\ \xi_i \geq 0 \end{cases} \qquad i = 1,\ldots,N$$

$card(\xi_i > 0)$ is the number of vectors inside the tube margins. Vectors outside the tube are associated to $\xi_i = 0$ (they are correctly classified so the restriction is fit with inequality, and minimizing the cost function implies $\xi_i = 0$).

Vectors lying on the decision boundary have $\xi_i = 1$. Can you check that?



Let us optimize the Lagrangian:

$$\mathcal{L} = \tfrac{1}{2}\|\mathbf{w}\|^2 + P\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i\left(y_i\left(\mathbf{w}^T\mathbf{x}_i + w_0\right) - \left(1 - \xi_i\right)\right) - \sum_{i=1}^{N}\beta_i\xi_i$$

We could also use non-linear functions of ξ_i here, but the solution would not be so simple...

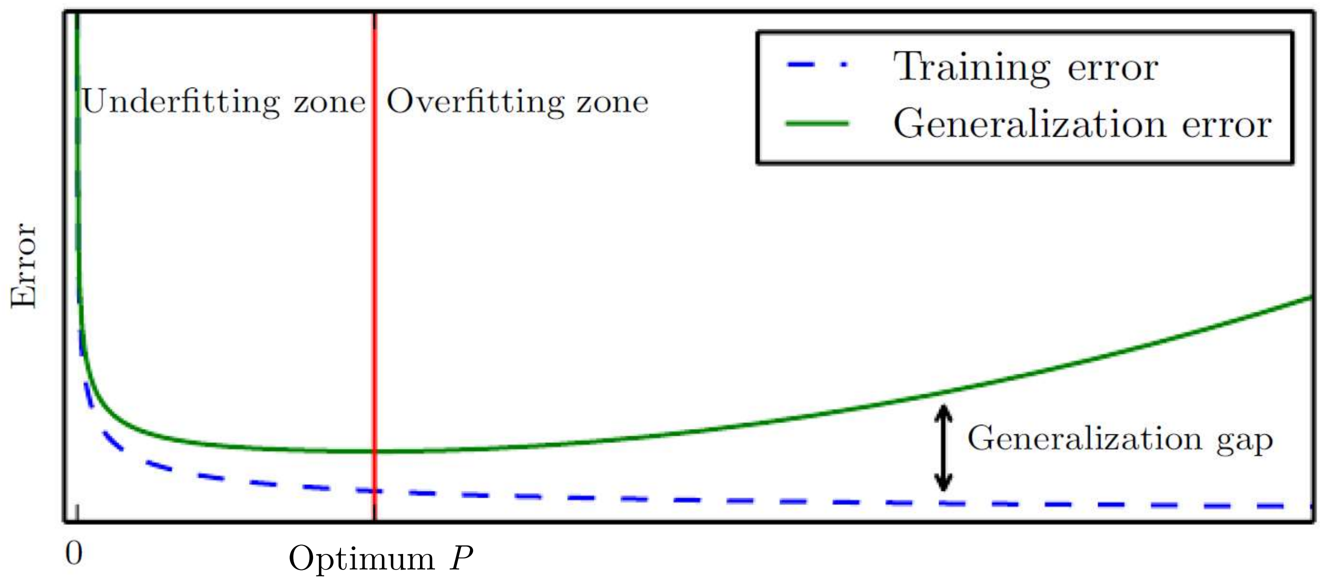
The convex problem is the same, except that the restriction on α_i has changed. The expressions for the optimal hyperplane are the same as before, though the optimum α_i are different:

$$\mathcal{L} = \sum_{i=1}^{N} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k \mathbf{x}_i^T \mathbf{x}_k \qquad \text{subject to} \qquad \begin{cases} 0 \le \alpha_i \le P \\ \sum_{i=1}^{N} \alpha_i y_i = 0 \end{cases}$$

Note that if $P \to \infty$ we obtain the solution for the separable case. If the problem is separable, some values of α_i will be limited by P.

- How are support vectors defined now? Those for which $\alpha_{\mathrm{i}} \neq 0$ (which include those vectors having $\xi_{\mathrm{i}} > 0$)

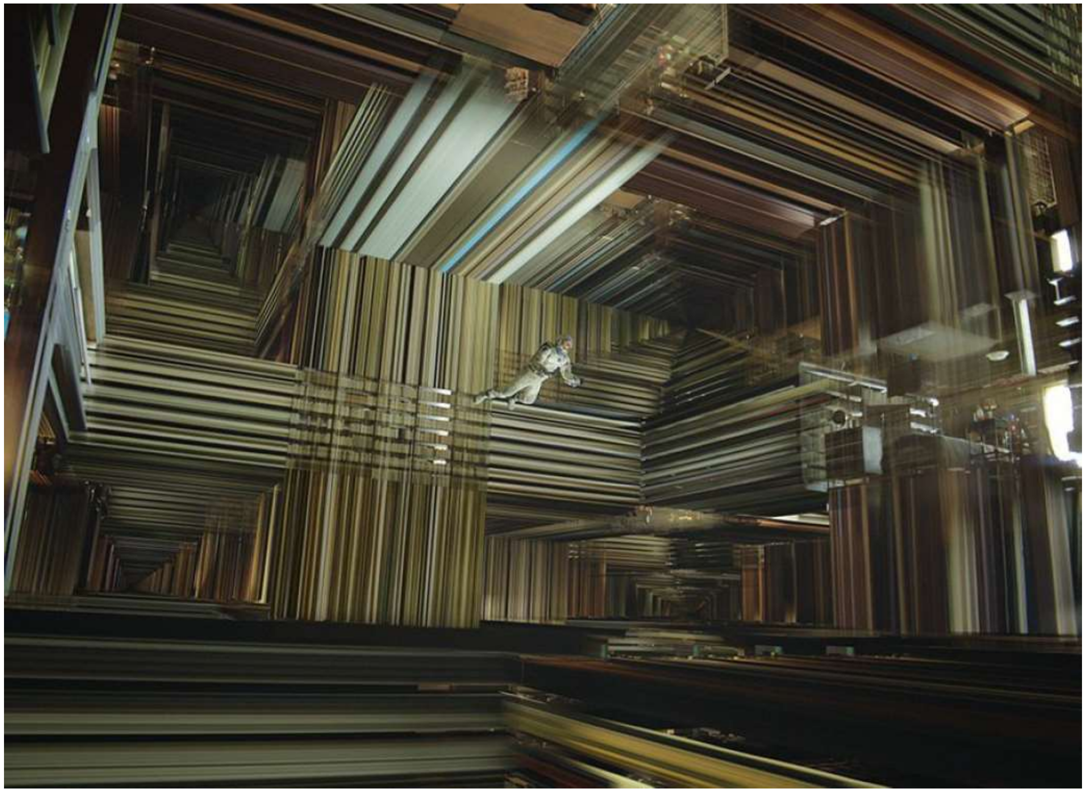
- Large values of P imply **overtraining**. Why? Many ξ_i are zero, hence only a few support vectors. Tube is narrow. Since $card(\xi_i > 0)/N$ gives an upper bound on the fraction of misclassified vectors, the error in training is low.



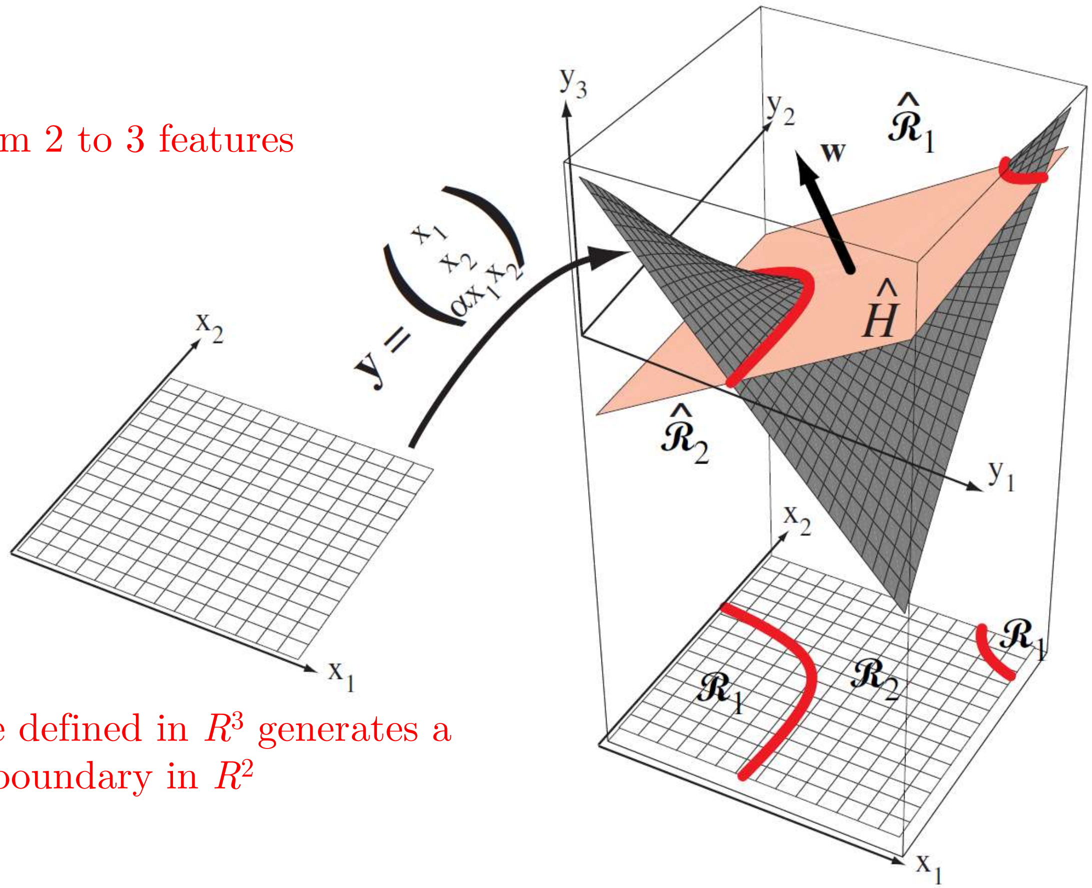
- Another interpretation: large P entail narrower tube, lower number of support vectors, more variability when randomly changing the training data base $\rightarrow$ higher variance.

5. SVM FOR NON-LINEAR DECISION BOUNDARIES

In those problems for which classes are not linearly separable, we will travel to a higher dimensional space and return with a solution...

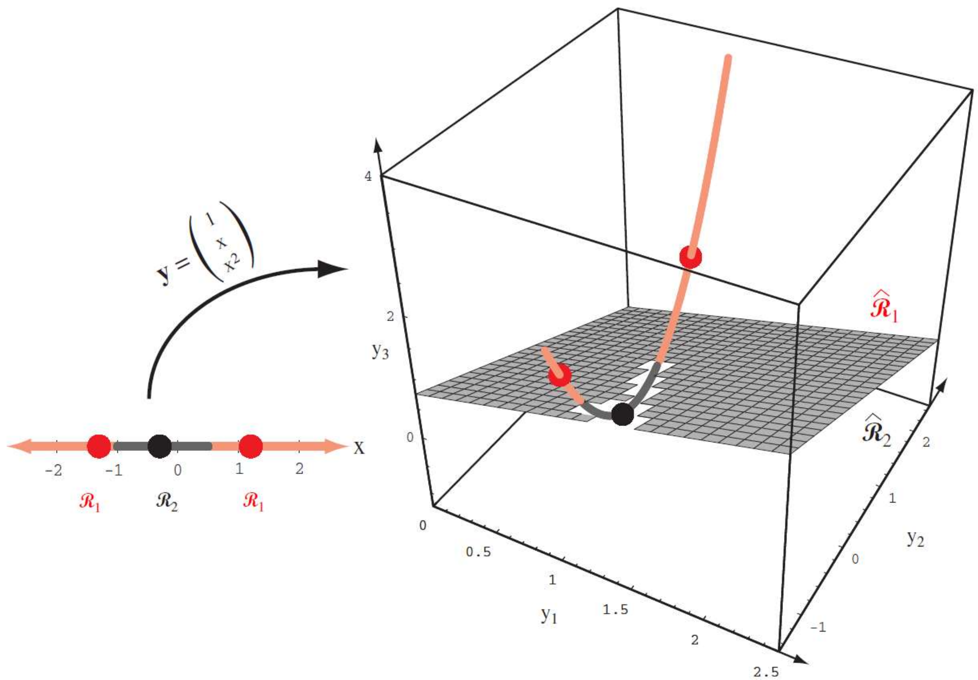


Example 1: from 2 to 3 features

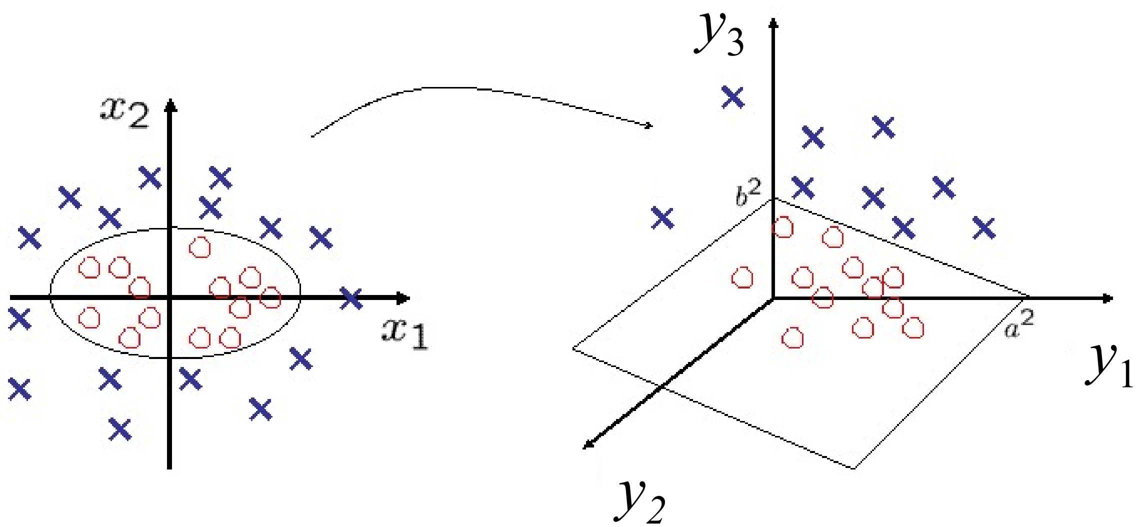


... a hyperplane defined in R^3 generates a non-linear boundary in R^2

Example 2: from 1 to 3 features



Example 3: from 2 to 3 features



$$\mathbf{x} = (x_1, x_2) \rightarrow \mathbf{y} = \left(x_1^2, \sqrt{2}x_1x_2, x_2^2\right)$$

KERNEL FUNCTION AND MATRIX

Let us define $\boldsymbol{\varphi}$ as a non-linear transform on data that converts our vector space to a Hilbert space of a larger dimension $d' \geq d$.

$$\boldsymbol{\varphi}:\mathbb{R}^{d}\rightarrow\mathbb{R}^{d'} \qquad \boldsymbol{\varphi}(\mathbf{x}_n)=\begin{pmatrix}\phi_1(\mathbf{x}_n)\\ \vdots \\ \phi_{d'}(\mathbf{x}_n)\end{pmatrix} \quad \Rightarrow \quad \boldsymbol{\Phi}=\begin{pmatrix}\boldsymbol{\varphi}(\mathbf{x}_1) & \cdots & \boldsymbol{\varphi}(\mathbf{x}_N)\end{pmatrix}$$

- **Kernel function:** it is the scalar product of two transformed vectors:

$$K:\mathbb{R}^{d\times d}\rightarrow\mathbb{R} \qquad K(\mathbf{x}_k,\mathbf{x}_n)=\boldsymbol{\varphi}(\mathbf{x}_k)^T\,\boldsymbol{\varphi}(\mathbf{x}_n)$$

- **Kernel matrix** ($N \times N$) contains all scalar products between transformed vectors:

$$\mathbf{K}=\begin{pmatrix}K(\mathbf{x}_1,\mathbf{x}_1) & \cdots & K(\mathbf{x}_1,\mathbf{x}_N)\\ \vdots & \ddots & \vdots\\ K(\mathbf{x}_N,\mathbf{x}_1) & \cdots & K(\mathbf{x}_N,\mathbf{x}_N)\end{pmatrix}=\boldsymbol{\Phi}^T\boldsymbol{\Phi} \qquad \text{(see slides 23 and 24)}$$

About the kernel function...

- Allows finding non-linear boundaries without knowing nor explicitly applying the transformation $\boldsymbol{\varphi}$
- The Mercer's theorem guaranties that, given a positive definite matrix $\mathbf{K}$ there exists a $\boldsymbol{\varphi}(\cdot)$ such that $K(\mathbf{x},\mathbf{z}) = \boldsymbol{\varphi}(\mathbf{x})^T \boldsymbol{\varphi}(\mathbf{z})$

Typical kernel functions...

- Linear kernel $K(\mathbf{x},\mathbf{z}) = \mathbf{x}^T\mathbf{z} + c$
- Polynomial kernel $K(\mathbf{x},\mathbf{z}) = \left(\alpha\mathbf{x}^T\mathbf{z} + c\right)^p$
- Gaussian kernel (radial basis function, RBF) $K(\mathbf{x},\mathbf{z}) = \exp\left(-\frac{1}{\sigma^2}\|\mathbf{x}-\mathbf{z}\|^2\right)$
- $\mathbf{A}$ positive semi-definite matrix $K(\mathbf{x},\mathbf{z}) = \mathbf{x}^T\mathbf{A}\mathbf{z}$

Possibilities in the construction of kernels, out of a primitive kernel…

- $f(\cdot)$ any scalar function used as $K(\mathbf{x},\mathbf{z}) = f(\mathbf{x}) K_1(\mathbf{x},\mathbf{z}) f(\mathbf{z})$

- $q(\cdot)$ polynomial of positive coefficients $K(\mathbf{x},\mathbf{z}) = q(K_1(\mathbf{x},\mathbf{z}))$

- $K(\mathbf{x},\mathbf{z}) = K_1(\mathbf{x},\mathbf{z}) + K_2(\mathbf{x},\mathbf{z})$

- $K(\mathbf{x},\mathbf{z}) = K_1(\mathbf{x},\mathbf{z}) K_2(\mathbf{x},\mathbf{z})$

- $K(\mathbf{x},\mathbf{z}) = \exp(K_1(\mathbf{x},\mathbf{z}))$

When applied to SVM it suffices to modify the Lagrangian:

$$\mathcal{L} = \sum_{i=1}^{N} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k K(\mathbf{x}_i, \mathbf{x}_k) \qquad \text{subject to} \quad \begin{cases} 0 \le \alpha_i \le P \\ \sum_{i=1}^{N} \alpha_i y_i = 0 \end{cases}$$

or in matrix form, we can replicate the previous expressions...

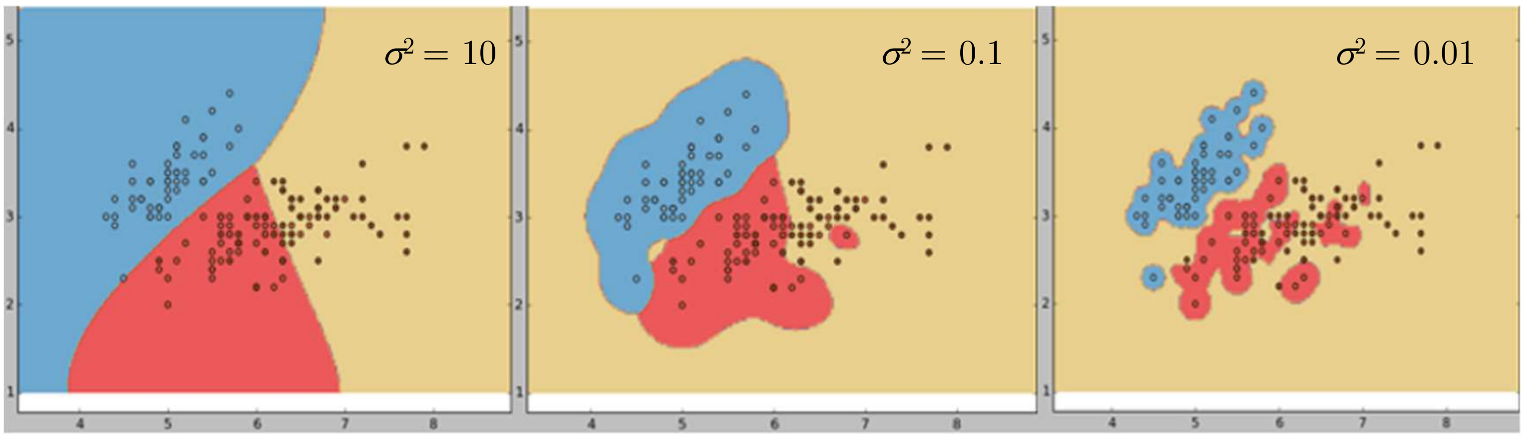
$$\mathcal{L} = \boldsymbol{\alpha}^T \mathbf{1} - \tfrac{1}{2} \boldsymbol{\alpha}^T \mathbf{Y}^T \boldsymbol{\Phi}^T \boldsymbol{\Phi} \mathbf{Y} \boldsymbol{\alpha} \qquad \text{subject to} \quad \begin{cases} 0 \le \alpha_i \le P \\ \mathbf{1}^T \mathbf{Y} \boldsymbol{\alpha} = 0 \end{cases}$$

Maximize $\mathcal{L}$ with restrictions using numerical techniques, and follow the steps in slide 24, using kernels.

For instance, let us adopt a Gaussian kernel. The optimum values of σ^2 and P can be obtained by exhaustive search on a set of values:

$$P \in \left\{2^{-5}, 2^{-3}, \ldots, 2^{15}\right\} \qquad \sigma^2 \in \left\{2^{-15}, 2^{-13}, \ldots, 2^{3}\right\}$$

using a validation data base.



The hyperparameters of the kernel must be validated to avoid overfitting, as in the two right plots.

The clasification of vector $\mathbf{x}$ is done as:

$$\hat{y} = sign\left(\boldsymbol{\alpha}^T \mathbf{Y}^T \mathbf{s} + w_0\right) = sign\left(\sum_{i=1}^{N_{SV}} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + w_0\right)$$

whereby the vector $\mathbf{x}$ is “compared” through the kernel with all the support vectors of the training data base: those N_{SV} for which $\alpha_i \neq 0$ (with KNN we had to carry all the training data base!).

Let us use expressions in slide 23 for the computation of w_0:

$$w_0 = \frac{1}{N_{\mathrm{SV}}}\left(\mathbf{1}^T\mathbf{y} - \boldsymbol{\alpha}^T\mathbf{Y}^T\boldsymbol{\Phi}^T\boldsymbol{\Phi}\mathbf{1}\right) = \frac{1}{N_{\mathrm{SV}}}\left(\mathbf{1}^T\mathbf{y} - \boldsymbol{\alpha}^T\mathbf{Y}^T\mathbf{K}\mathbf{1}\right)$$

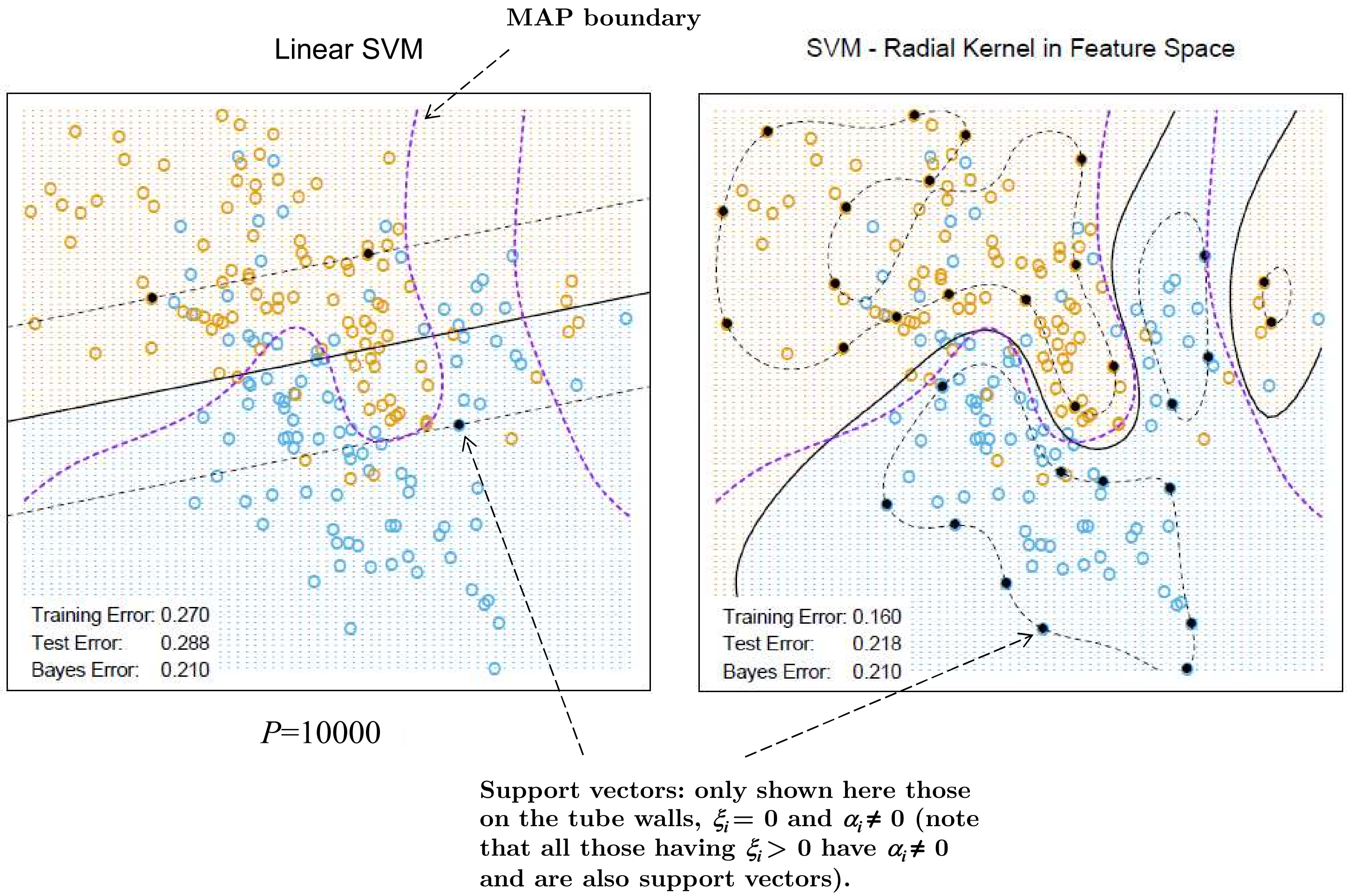
where $\mathbf{K}$ is the kernel matrix for the support vectors and it is precomputed:

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

$$\mathbf{Y} = diag(y_1,\dots,y_N)$$

$$\mathbf{1} = [1,\dots,1]$$

Example: Two classes, two-features vectors

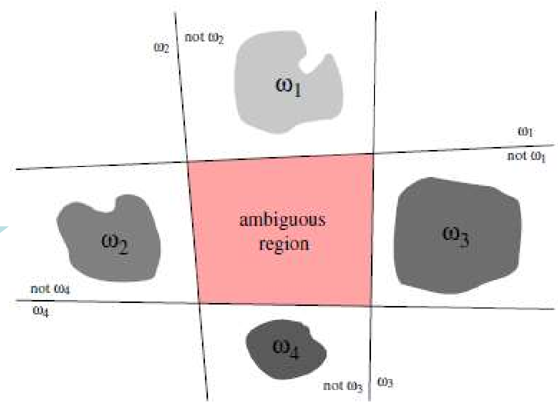


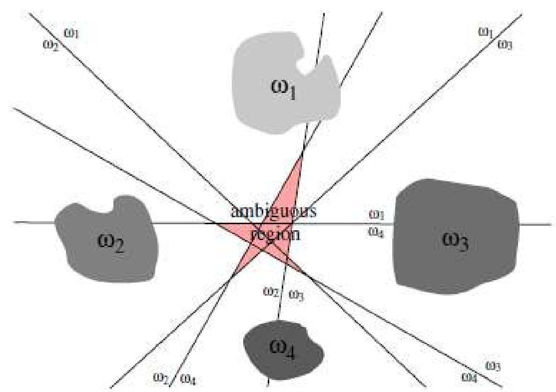
6. MULTIPLE CATEGORIES

Ways to extend the problem to $c > 2$ clases

- c two-class problems (one-vs-all), if there are many classes each one of the c problems will be unbalanced

- $c(c\text{-}1)/2$ two-class problems (one-vs-one)





7. CONCLUSIONS

In general, linear discriminants…

- Measure the distance of vectors to the separating hyperplanes
- We can adapt them for more than two categories

SVM…

- Is very useful for non-linearly separable problems
- Resilience to over-fitting: we estimate a small number of parameters.
- Dificulties:
  - Find the suitable kernel function.
  - For each new vector to classify $\mathbf{x}$, we have to evaluate $K(\mathbf{x}_i, \mathbf{x})$ with the support vectors $\rightarrow$ computational cost?
- The concept of kernel applies also to other classifiers.