Support vector networks

Séance « svn »

de l'UE « apprentissage automatique »

Bruno Bouzy

bruno.bouzy@parisdescartes.fr

www.mi.parisdescartes.fr/~bouzy

Reference

- These slides present the following paper:
  - C.Cortes, V.Vapnik, « support vector networks », Machine Learning (1995)
- They are commented with my personal view to teach the key ideas of SVN.
- The outline mostly follows the outline of the paper.

Outline

- Abstract
- Preamble
- « Introduction »
- Optimal hyperplane
- Soft margin hyperplane
- Dot-product in feature space
- General features of SVN
- Experimental analysis
- Conclusion

Abstract

- New learning method for 2-group classification
- Input vectors non-linearly mapped to a high dimension space (the feature space)
- In feature space: linear decision surface
- High generalisation ability
- Experiments: good performances in OCR

Preamble

- How to *separate* 2 *separable* classes ?

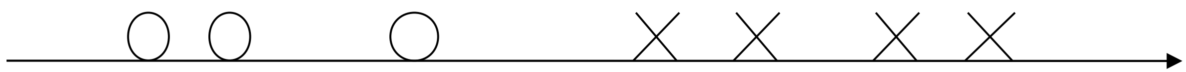


Figure 0a

Preamble

- Separating 2 separable classes: easy!

Simply choose a point somewhere in between two opposite examples

No error!

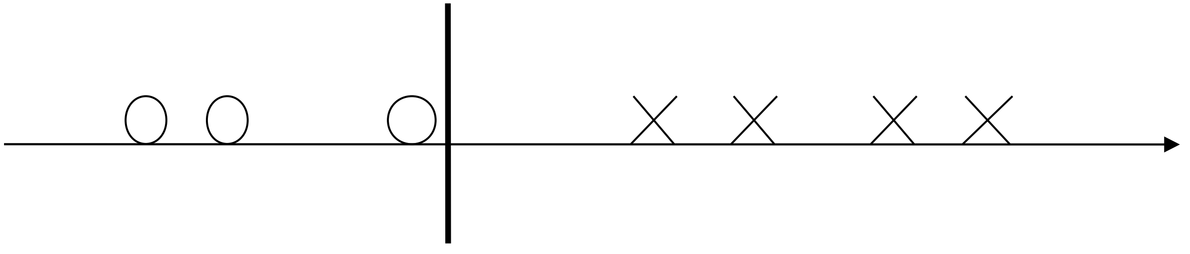


Figure 0b

Preamble

- What may happen when a new example comes ?

One error... (bouh... generalisation is poor)

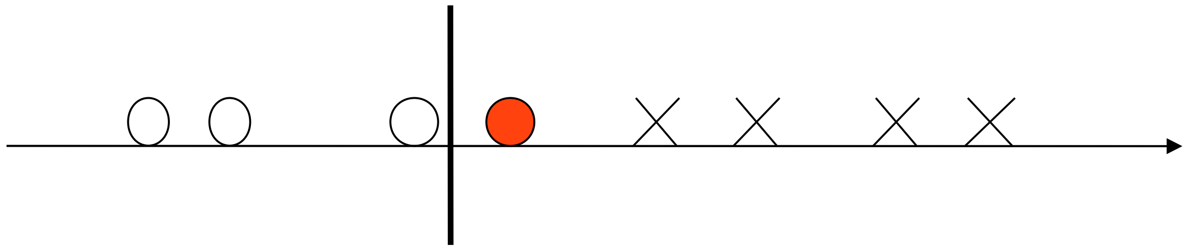


Figure 0c

Preamble

- How to *optimally* separate 2 separable classes ?

Figure 0d

Preamble

- *Optimally* separating 2 separable classes: not difficult!

Choose the point at the middle!

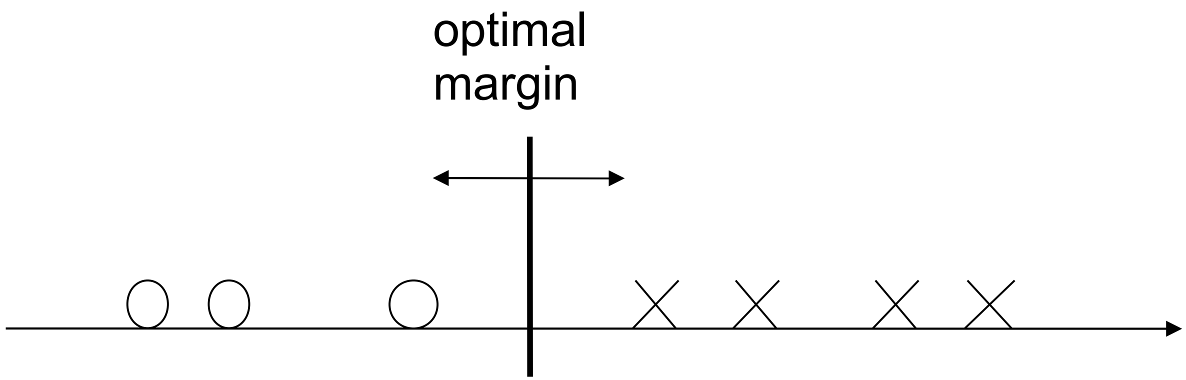


Figure 0e

Preamble

What may happen when a new example comes ?

better chance that no error this time

(whew... generalisation is better)

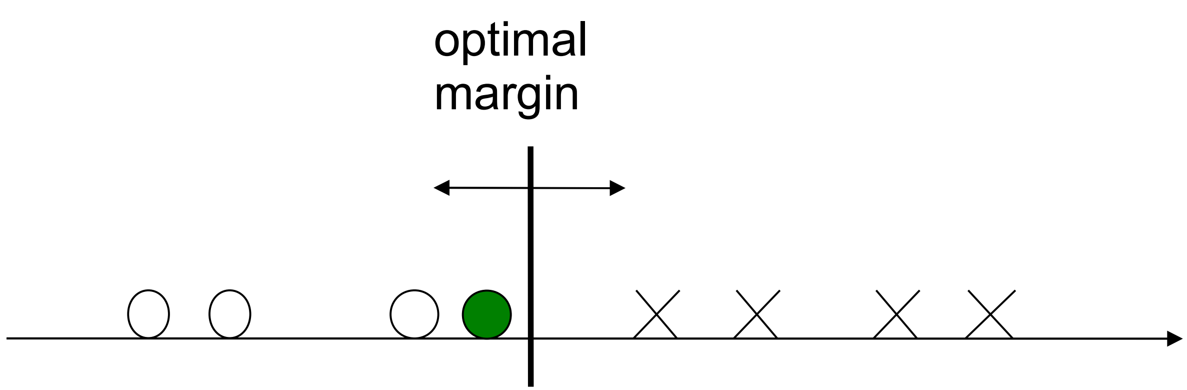


Figure 0f

Preamble

- *support vectors*

the optimal point depends only on some examples: the support vectors, and not on the other.

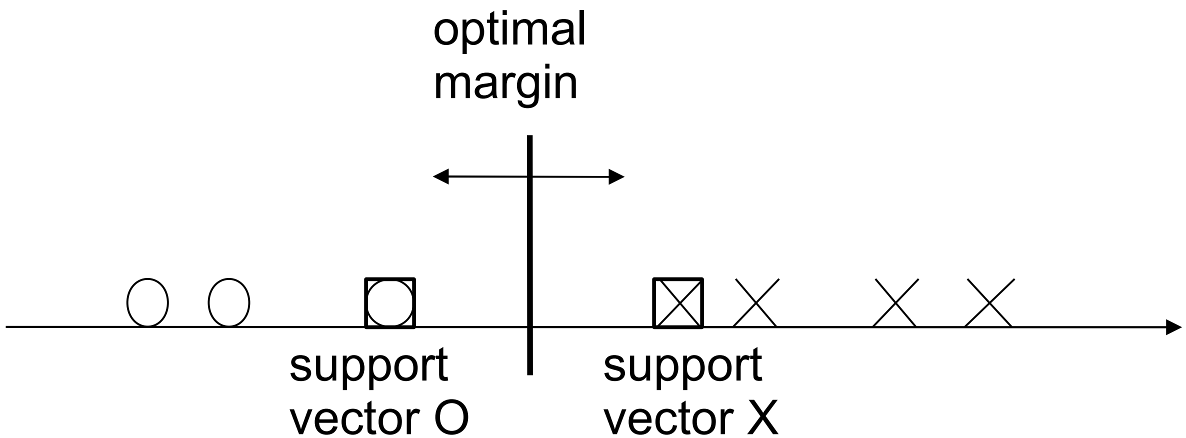


Figure 0g

Preamble

- How to separate 2 non-separable classes ?

huhu...

Figure 0h

Preamble

- minimizing the number of training errors...

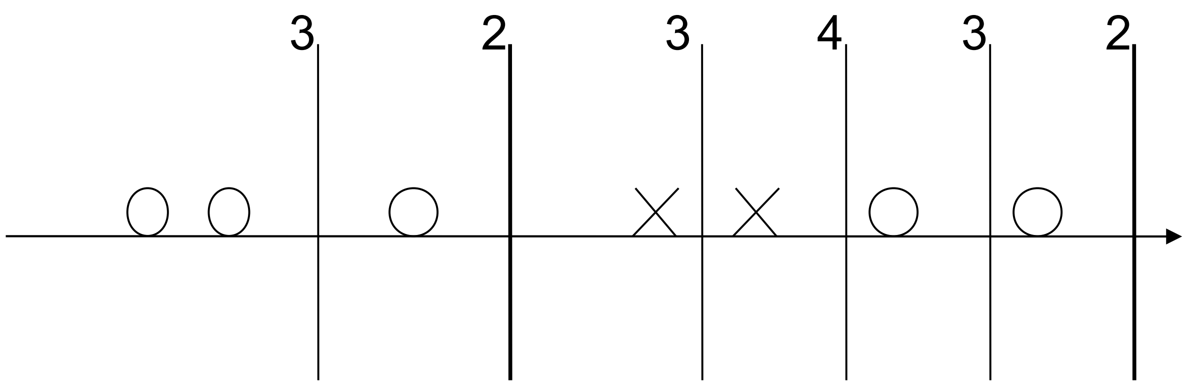


Figure 0i

Preamble

- *and* minimizing the number of test errors...

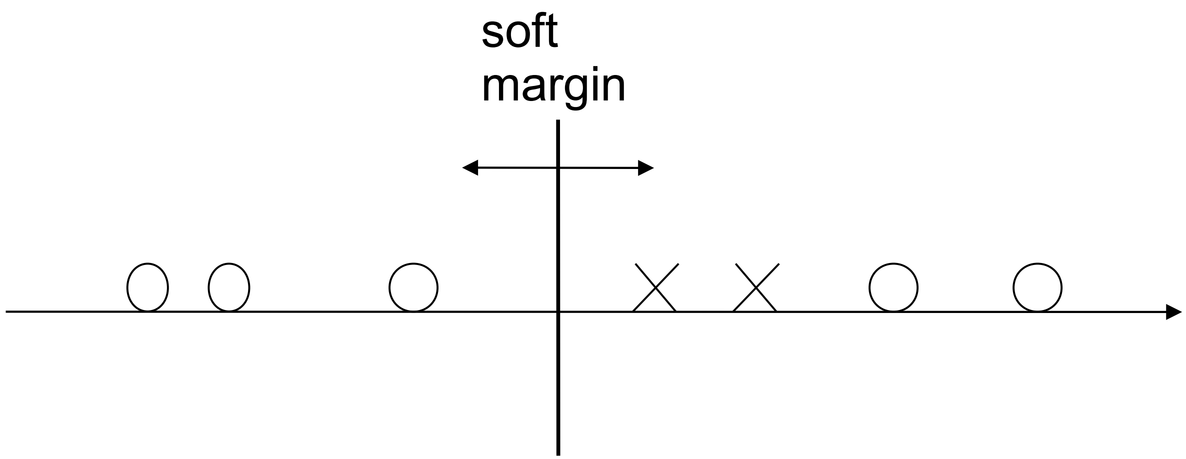


Figure 0j

Preamble

- Transforming the problem (dim=1) into a higher dimensional one? hoho?

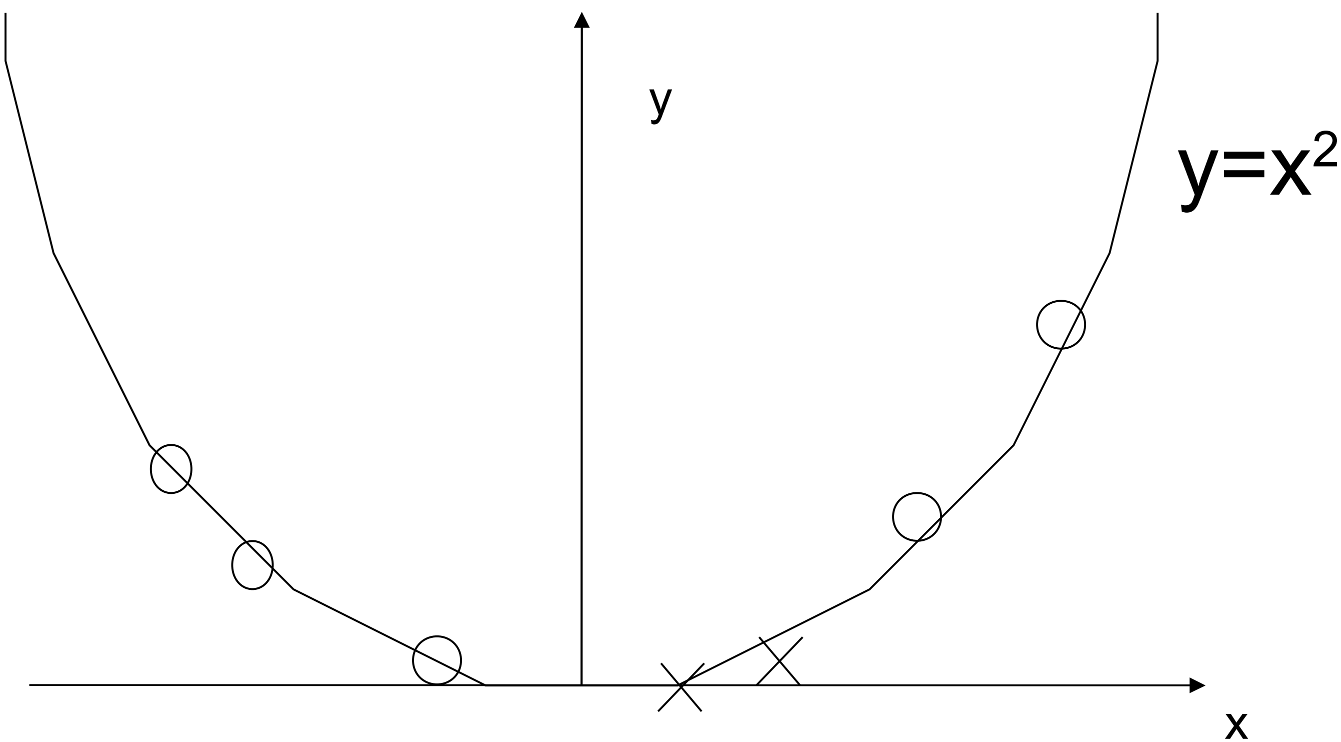


Figure 0k
Support-vector networks

Preamble

then separate...   huhu? Smart!

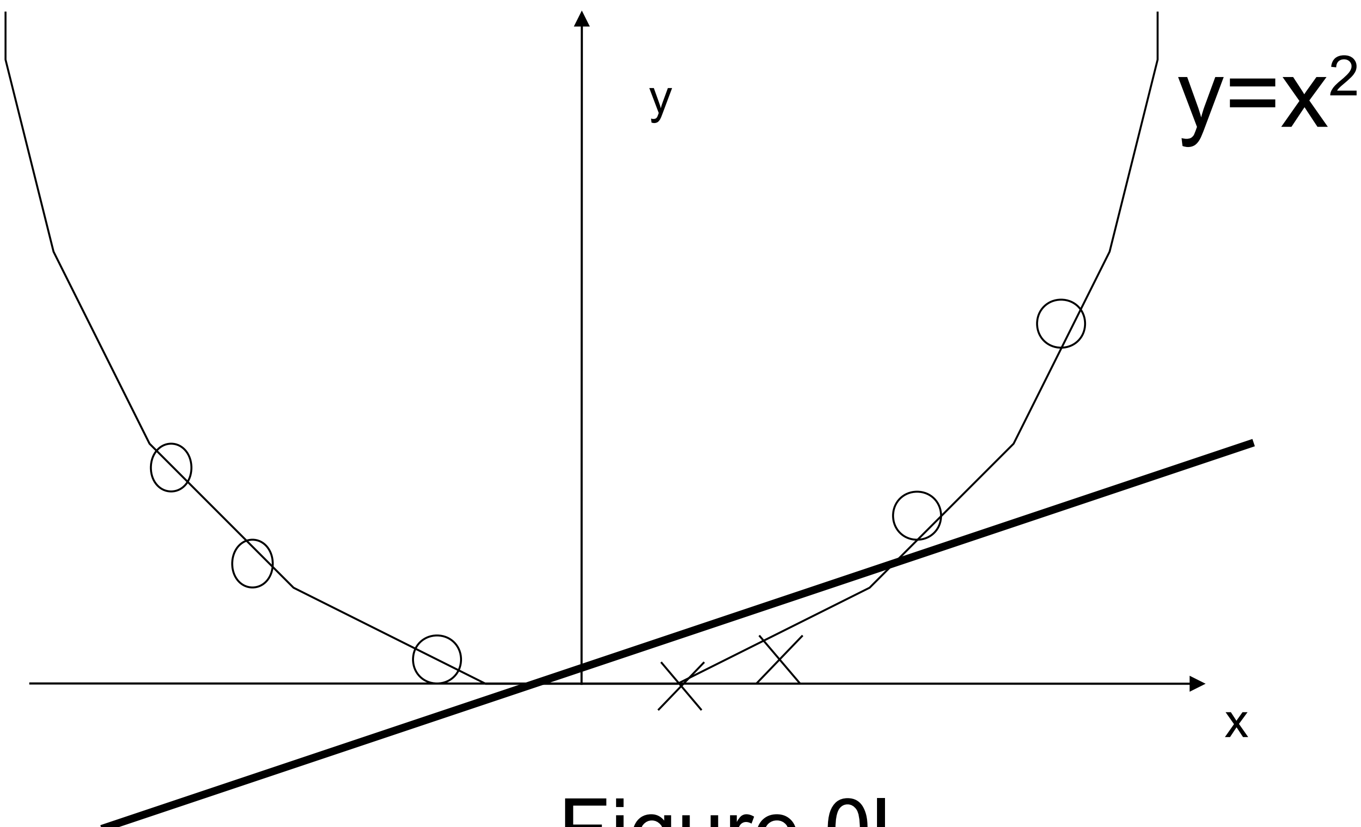


Figure 0I

Preamble summary

- Separable case:
  - **optimal margin** for good generalisation
  - **support vectors**
- Non separable case:
  - soft margin, minimizing the errors
- Key ideas:
  - Transform the problem **into a higher dimensional** separable problem
  - **Linear** separation

« Introduction »

- To obtain a decision surface to a polynomial of degree 2, create a feature space with $N=n(n+3)/2$ coordinates:
  - $z_1 = x_1, \quad z_2 = x_2, \quad \ldots, \quad z_n = x_n$
  - $z_{n+1} = x_1^2, \quad z_{n+2} = x_2^2, \quad \ldots, \quad z_{2n} = z_n^2$
  - $z_{2n+1} = x_1x_2, \quad z_{2n+2} = x_1x_3, \quad \ldots, \quad z_N=x_{n-1}x_n$

« Introduction »

- Conceptual problem:
  - How to find an hyperplane that generalizes well ?
- Technical problem:
  - How to computationally treate high dimensional space ?
  - (to construct a polynomial of degree 4 or 5 in a dimension 200 space, the high dimension can be 10^6)

« Introduction »

- The support vectors ⊡ , ⊠ determine the optimal margin (greatest separation):

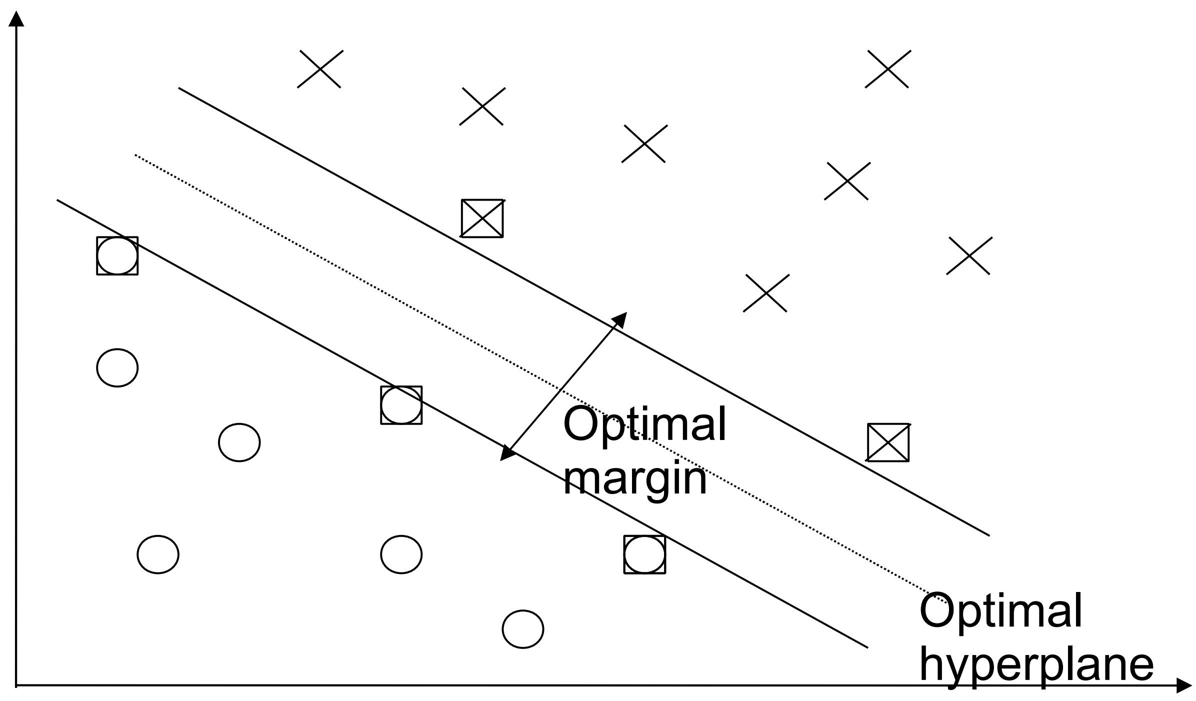


Figure 2

Support-vector networks

« Introduction »

- The bound on the error depends on the number of support vector (5):

  - E(pr(err))<= E(#suppVectors)/E(#trainVectors)

« Introduction »

- Optimal separating plane equation:

  $w_0.z+b_0 = 0$

- Weights of the vector hyperplane (6):

  $w_0 = \Sigma_i \, \alpha_i z_i$

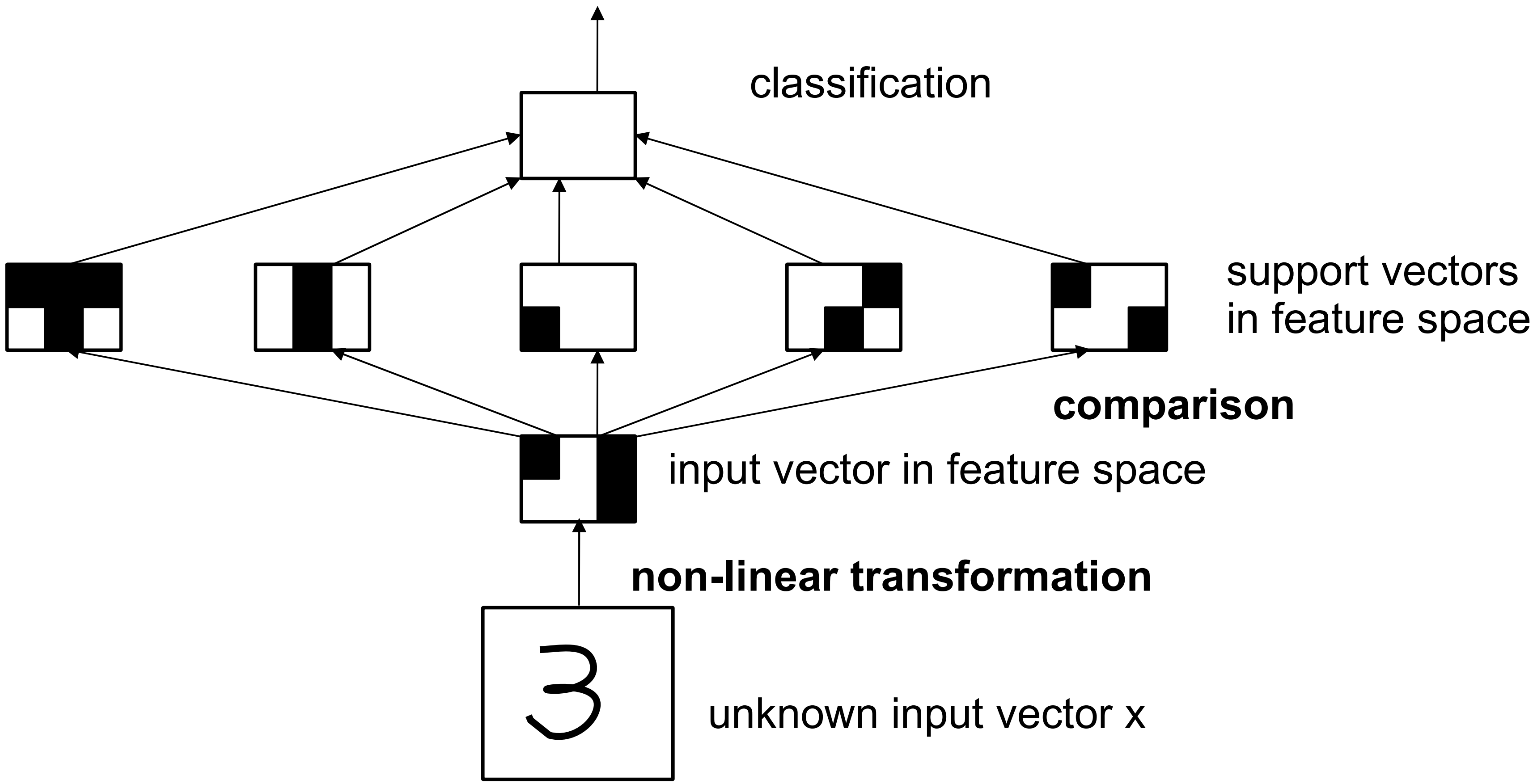
- Linear decision function (7):

  $I(z) = \mathrm{sign}(\Sigma_i \, \alpha_i z_i.z+b_0)$

« Introduction »

Figure 3

- classification by a SVN of an unknown pattern

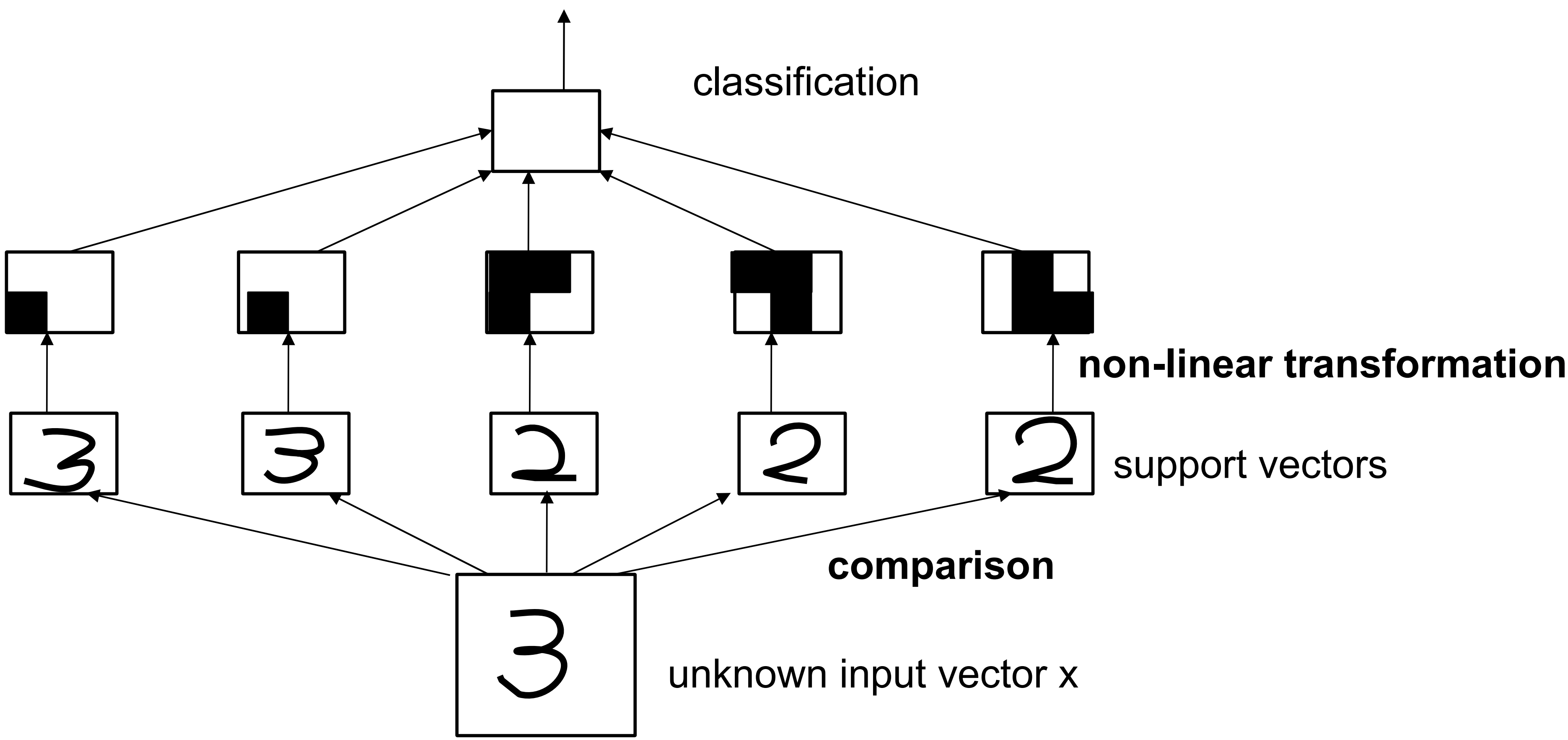


Support-vector networks

« Introduction »

Figure 4:

- 2^{nd} kind of classification by a SVN:



Support-vector networks

Optimal hyperplanes

- Set of labeled patterns (8):
  - $(y_1, x_1), \ldots, (y_l, x_l)$, with $y_i = \pm 1$
- Linearly separable (9):
  - if there exist vector w and scalar b such that:
  - $w.x_i+b >= 1$ if $y_i=1$ $w.x_i+b <= -1$ if $y_i=-1$
- Linearly separable (10):
  - if there exist vector w and scalar b such that:
  - $y_i.(w.x_i+b) >= 1$ for $i=1,\ldots,l$

Optimal hyperplanes

- Optimal hyperplane (11):

  $$w_0.x+b_0 = 0$$

- Separates the training data with maximal margin:
  - Determines the direction $w/|w|$ where the distance $\rho(w,b)$ (12) between the projections of the training vectors of the two classes is maximal (13):
  - $\rho(w,b) = \min_{x:y=1} x.w/|w| - \max_{x:y=-1} x.w/|w|$
  - $\rho(w_0,b_0) = 2/|w_0|$

Optimal hyperplanes

- The optimal hyperplane minimizes w.w under the constraints (10).

- Constructing an optimal hyperplane is a **quadratic problem**.

- Vectors such that (10) is an egality are termed **support vectors**.

Optimal hyperplanes

- The optimal hyperplane can be written as a linear combination of training vectors (14):
  - $W_0 = \Sigma_{i=1,l}\ y_i\ \alpha_i^0\ x_i$
  - with $\alpha_i^0 >= 0$ for all training vectors
  - and $\alpha_i^0 > 0$ for support vectors only
  - $\Lambda_0^T = (\alpha_1^0, \alpha_2^0, ..., \alpha_l^0)$

Optimal hyperplanes

- The quadratic programming problem :
  - $W(\Lambda) = \Lambda^T 1 - ½\, \Lambda^T D\, \Lambda$ (15)
  - with: $\Lambda >= 0$ (16)

  $\Lambda^T Y = 0$ (17)

  $D_{ij} = y_i y_j\, x_i . x_j$ for $i,j = 1, ..., l$ (18)

Optimal hyperplanes

- When the data of (8) can be separated by an hyperplane:
  - $W(\Lambda_0) = 2/\rho_0^2$
- When the data of (8) cannot be separated by an hyperplane:
  - For any large constant C, find Λ such that $W(\Lambda)>C$

Optimal hyperplanes

- Solving scheme:
  - Divide the training data into portions
  - Start out by solving the first portion
  - If first portion cannot be separated then end & failure
  - If first portion is separated,
  - Make a new training set with support vectors of the first portion and the training vectors of the second portion that do not satisfy (10)
  - Continue by solving this new portion
  - etc.

Soft margin hyperplane

- Case where the training data cannot be separated without error:
  - One may want to separate with the minimal numbers of errors
  - Non negative variables (23): $\xi_i >= 0 \qquad i=1,...,l$
  - Minimize (21): $\Phi(\xi) = \Sigma_i \xi_i$
  - Subject to (22): $y_i.(w.x_i+b) >= 1-\xi_i \qquad i=1,...,l$

Soft margin hyperplane

- Minimizing (21), one finds a minimal subset of training errors:
  - $(y_{i1}, x_{i1}), \ldots, (y_{ik}, x_{ik})$
- Minimize (25):
  - $\frac{1}{2} w^2 + CF(\sum_i \xi_i)$
  - subject to (22) and (23)
  - where F is monotonic convex, $F(0)=0$ $F(u)=u^2$
  - C constant

Soft margin hyperplane

- The programming problem :
  - $W(\Lambda) = \Lambda^T 1 - \frac{1}{2}[\Lambda^T D \Lambda + \delta^2/C]$ (26)
  - with: $\Lambda^T Y = 0$ (27)

    $\delta >= 0$ (28)

    $0 <= \Lambda <= \delta 1$ (29)
  - note that: $\delta = \max(\alpha_1, \alpha_2, \ldots, \alpha_l)$

Soft margin hyperplane

- The solution exists and is unique for any data set

- Not a quadratic problem but a convex programming problem

Dot-product in feature space

- Transform:
  - the n-dimensional input vector x into
  - an N-dimensional feature vector
  - through a function Φ :
    - $\Phi : R^n \rightarrow R^N$
    - $\Phi(x_i) = \Phi_1(x_i), \Phi_2(x_i), ..., \Phi_N(x_i)$
- Construct the N dimensional linear separator w and bias b

Dot-product in feature space

- To classify an unknown vector x:
  - Transform it into $\Phi(x)$ vector
  - Take the sign of (31): $f(x) = w \cdot \Phi(x) + b$
- w is linear combination of support vectors:

$$w = \Sigma_{i=1,l}\, y_i\, \alpha_i\, \Phi(x_i) \qquad (32)$$

  - $f(x) = \Sigma_{i=1,l}\, y_i\, \alpha_i\, \Phi(x).\Phi(x_i) + b \qquad (33)$

Dot-product in feature space

- Dot-product in Hilbert space
  - $\Phi(u) \,.\, \Phi(v) = K(u, v)$ (34)
- Any function K, symmetric can be expanded:
  - $K(u, v) = \Sigma_{i=1,\infty}\, \lambda_i\, \Phi_i(u).\Phi_i(v)$ (35)
  - λ_i eigenvalues and Φ_i eigenfunctions
  - $\int K(u,v)\Phi_i(u)du = \lambda_i\, \Phi_i(v)$
- To ensure (34) defines a dot-product:
  - $\lambda_i >= 0$

Dot-product in feature space

- Merser theorem:

  $\int\int K(u,v)g(u)g(v)dudv > 0$

  for g such that: $\int g(u)2du < \infty$

  iff

  $\lambda_i >= 0$

Dot-product in feature space

- Functions satisfying the Merser theorem:
  - $K(u, v) = \exp(-|u-v|/\sigma)$ (36)
  - $K(u, v) = (u.v + 1)^{d}$ (37)
- Decision surface:
  - $f(x) = \Sigma_{i=1,l}\ y_i\ \alpha_i\ K(x, x_i) + b$
- To find the α_i and x_i same solution scheme with:

$D_{ij} = y_i\, y_j\, K(x_i, x_j)$ $i,j = 1, \ldots, l$

General features of SVN

- Constructing the decision rules by SVN is efficient
  - Follow the scheme of soft margin
  - One unique solution
- SVN is a universal machine
  - By changing K, one obtains various machines
- SVN controls the generalization ability

Experimental results

- blablabla

Conclusion

- blablabla

Example of XOR

- blablabla