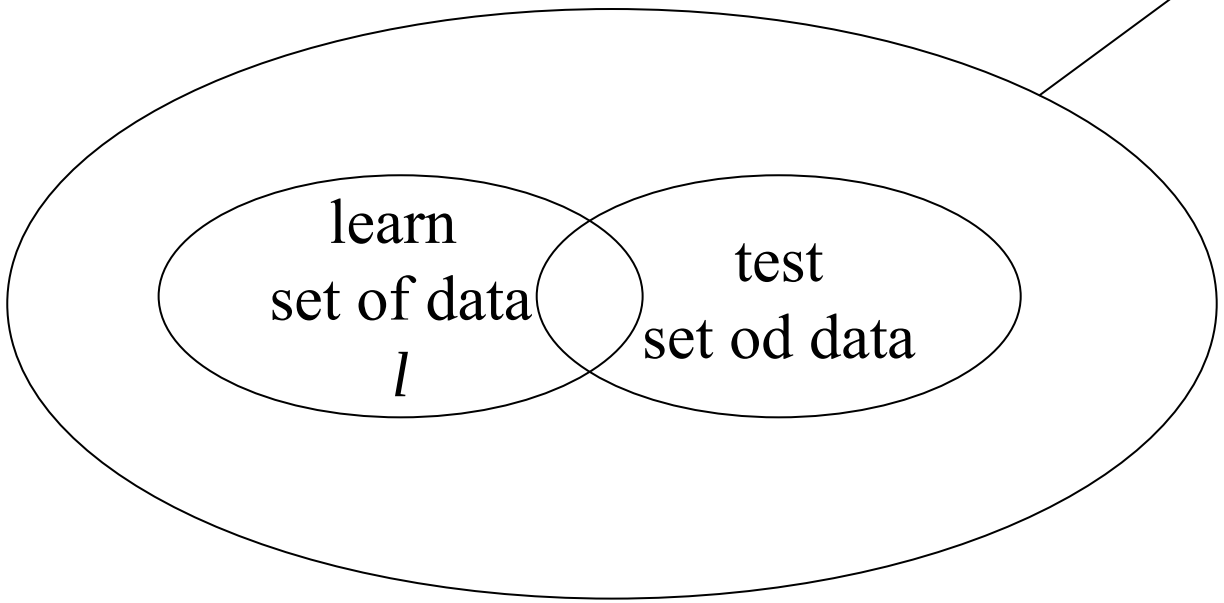
Chapter 10
The Support-Vector-Machine (SVM)

A statistical approach of learning theory for designing an optimal classifier

Content:

1. Problem
2. VC-Dimension and minimization of overall error
3. Linear SVM
   - Separable classes (example)
   - Non-separable classes (example)
4. Non-linear SVM
   - Trick with kernel function (example)
   - Mercer`s Theorem
5. Properties and complexity
6. Presentation of research projects:
   - Claus Bahlmann: recognition of handwriting
   - Olaf Ronneberger: Autom. recognition of pollen
   - Bernard Haasdonk: Tangential distance and SVM

Pattern recognition experiment



Stochastic process with *unknown* joint distribution: $P(\mathbf{x},y)$

Sought is a mapping function
$f(\mathbf{x},\alpha)$: $\mathbf{x}_i \rightarrow y_i$
that is charachterized by the unknown parameters of the vector α, which minimizes the expected value of the mapping error:

$$R(\alpha)=E\{R_{test}(\alpha)\}$$

empirical error $R_{emp}(\boldsymbol{\alpha})$

test error (generalization ability) $R_{test}(\boldsymbol{\alpha})$

Learning theoretical approach

Supervised learning:

- Given: *l* observations (Learning sample) from the pattern recognition experiment with joint distribution $P(\mathbf{x},y)$ with corresponding class maps (labels) (initially restraint to two-class-problem); besides that *no previous knowledge* exists:

$$\{(\mathbf{x}_i, y_i) \in P(\mathbf{x}, y)\} \quad i = 1,\ldots,l \quad \text{with: } \mathbf{x}_i \in \mathbb{R}^N, \quad y_i \in \{+1,-1\}$$

Sought:

- deterministic mapping function, $f(\mathbf{x},\boldsymbol{\alpha})$: $\mathbf{x} \rightarrow y$ based on a learning sample, which minimizes the *expected value of the mapping error for the test set of data (expected risk)* :

$$R(\boldsymbol{\alpha}) = E\{R_{test}(\boldsymbol{\alpha})\} = E\left\{\tfrac{1}{2}\left|y - f(\mathbf{x},\boldsymbol{\alpha})\right|\right\} = \int \tfrac{1}{2}\left|y - f(\mathbf{x},\boldsymbol{\alpha})\right| dP(\mathbf{x},y)$$

$$= \int \tfrac{1}{2}\left|y - f(\mathbf{x},\boldsymbol{\alpha})\right| p(\mathbf{x},y)\,d\mathbf{x}dy \quad \text{if joint distribution known}$$

Problem: this expression cannot be evaluated, since $P(\mathbf{x},y)$ is not available and thus not very useful!

Central problem of the statistical learn theory: When does a small learning error lead to a small real error?

- The *empirical risk*, i.e. the error rate for a given training set of data can be calculated easily according to:

$$R_{emp}(\boldsymbol{\alpha}) = \frac{1}{2l}\sum_{i=1}^{l}\left|y_i - f(\mathbf{x}_i, \boldsymbol{\alpha})\right|$$

- The approach with a neural net and e.g. backpropagation learning is satisfied with an **E**mpirical **R**isk **M**inimization (ERM)
- The question: How close are we to the *real* error after *l* training examples? And: How good can we estimate the real risk from the empirical risk (**S**tructural **R**isc **M**inimization (SRM) instead of Empirical Risk Minimization (ERM)) ? This contains generalization ability!
- The learning theory of Vapnik-Chervonenkis tries to answer that queation!

An upper limit of generalization ability of learning with the VC-theory

(Vapnik/Chervonenkis)

With probability (1-η) the following estimate for the effective error applies (i.e. e.g. with η = 0,05 and thus with a probability of 95%):

$$R(\boldsymbol{\alpha}) \le R_{emp}(\boldsymbol{\alpha}) + \underbrace{\Phi(h,l,\eta)}_{\text{VC-Konfidenz}}$$

mit: $$\Phi(h,l,\eta) = \sqrt{\frac{h\left(\log\frac{2l}{h}+1\right)-\log\left(\frac{\eta}{4}\right)}{l}}$$

Design target:
keep VC-confidence as little as possible and thus the estimate as exact as possible!

- l number of trainig examples
- h VC-dimension of the used hypothesis space

Remarkably this expression is independent on the underlying joint distribution P(x,y) ! (provided that the training and testing sets of data are considered statistically independent), i.e. in case there are a couple of different learning machines to choose from (accompanied by special families of mapping functions f(**x**,α)), the machine, that yields the lowest value for confidence Φ for given empirical error, is to be chosen.

The VC-dimension is a property for given function classes $\{f(\alpha)\}$

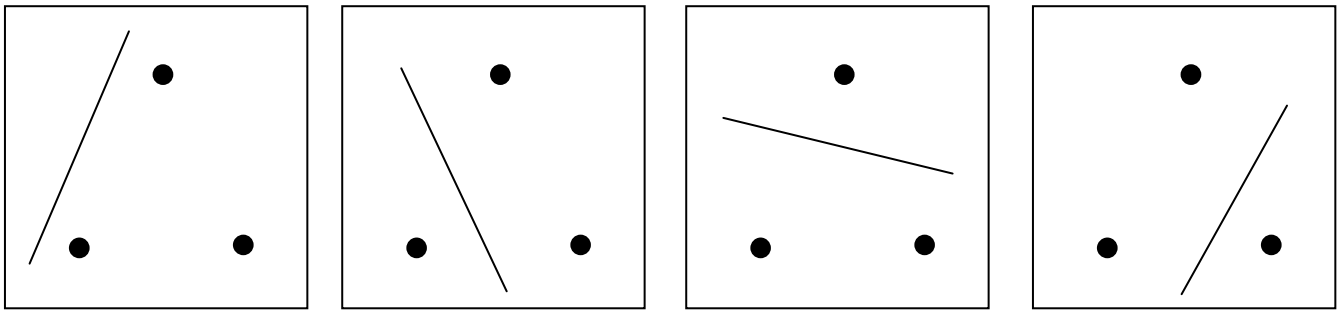
- A given set of l points can be divided into 2^l possible classes for the two-class. The VC-dimension of a set of functions $\{f(\alpha)\}$, is defined as the maximal number of training points, that can be separated by this class of functions in all possible constellations.

- provides measures for „capacity“ of function classes
  - function classes e.g.
    - set of linear separation functions
    - set of artificial NN (MLP)
    - set of polynomial functions
    - set of radial basis functions

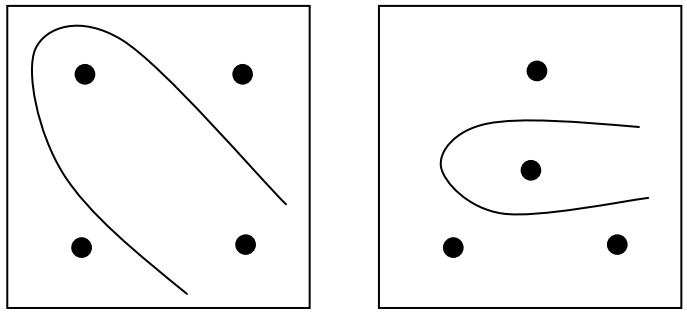
- The VC-confidence grows monotonically with h: According to that, when choosing a learning machine whose empirical risk is 0, the one with minimal VC-dimension of associated set of mapping functions is to be chosen.

VC-dimension *h* of hyperplanes in $\mathbb{R}^2$

- Three points in $\mathbb{R}^2$ can be separated in all conformations with *hyperplanes* (lines)



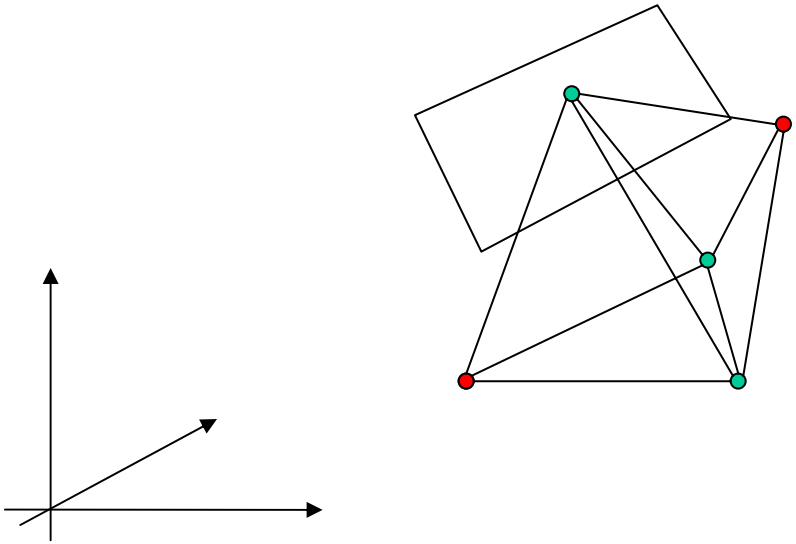
- Four points in $\mathbb{R}^2$ cannot be separated with *hyperplanes* in all conformations anymore; but e.g. with *polynomials*!



  - => hyperplanes in $\mathbb{R}^2$: $h = 3$
  - Generally: hyperplanes in $\mathbb{R}^N$: $h = N+1$
  - the VC-dimension of polymonials increases with their degree!

VC-dimension h of hyperplanes in $\mathbb{R}^3$

- Four points in $\mathbb{R}^3$ can be separated with *hyperplanes* (planes) in all conformations
- Five points in $\mathbb{R}^3$ cannot be separated anymore with *hyperplanes* in all conformations (e.g. the red points in one class); but e.g. *polynomials*!



- => Hyperplanes in $\mathbb{R}^3$: $h = 4$
- Generally: Hyperplanes in $\mathbb{R}^N$: $h = N+1$

VC-dimension of function classes

– Statements about VC-dimension of function classes possible!

- VC-dimension of set of *hyperplanes* in $\mathbb{R}^N$

$$h \leq \min(\frac{R^2}{\delta^2}, N) + 1$$

R: radius, in which all data points lie
δ: Margin between the hyperplanes

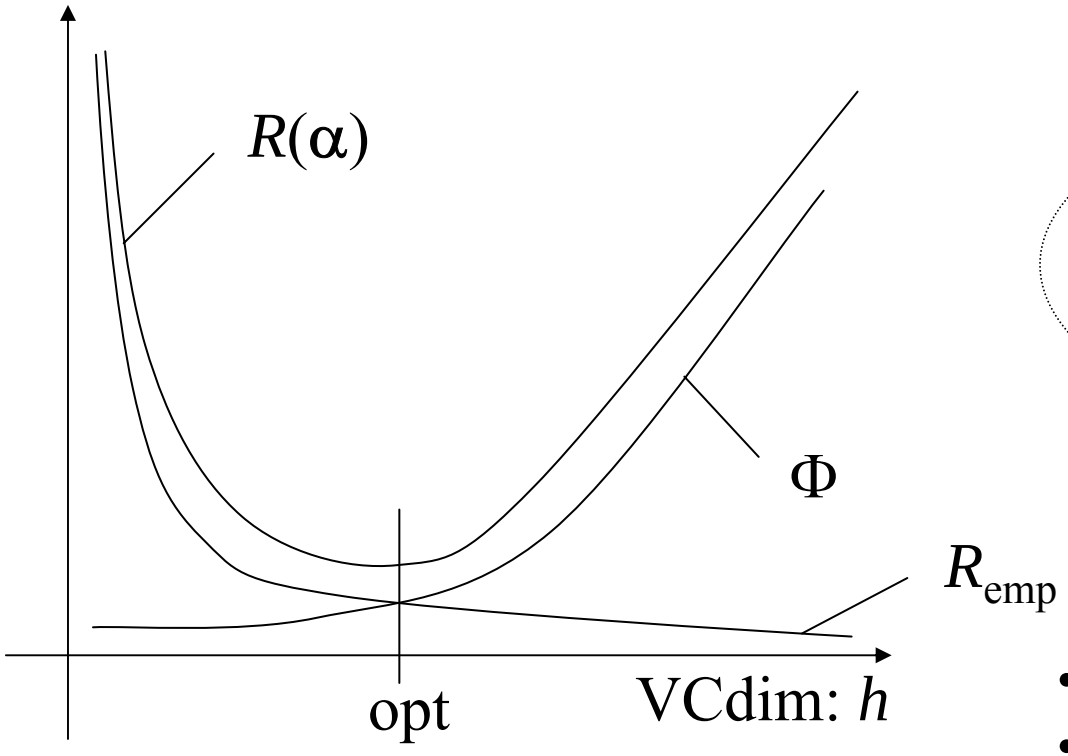
- VC-dimension of the set of *polynom kernels* of degree p: $K(x_i,x_j)=(x_i \cdot x_j)^p$

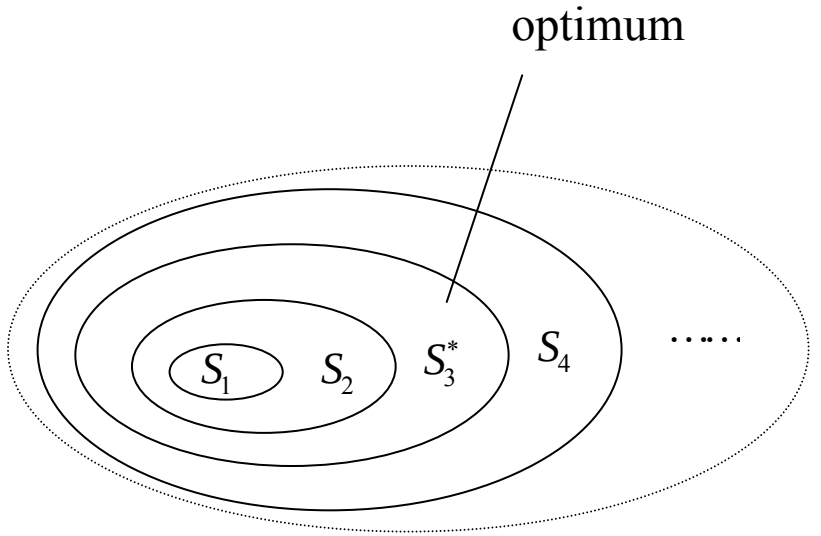
$$h = \begin{pmatrix} N + p - 1 \\ p \end{pmatrix} + 1$$

– Thus “Structural Risk Minimization“ possible

Structural Risk Minimization (SRM)

- SRM means minimizing the estimate for $R(\alpha)$ over cumulative function classes S_i . These form the hypothesis spaces with cumulative VC-dimension VCdim=h_i ($h_1<h_2<h_3<...$).
- Compromise between empirical risk and generalization ability





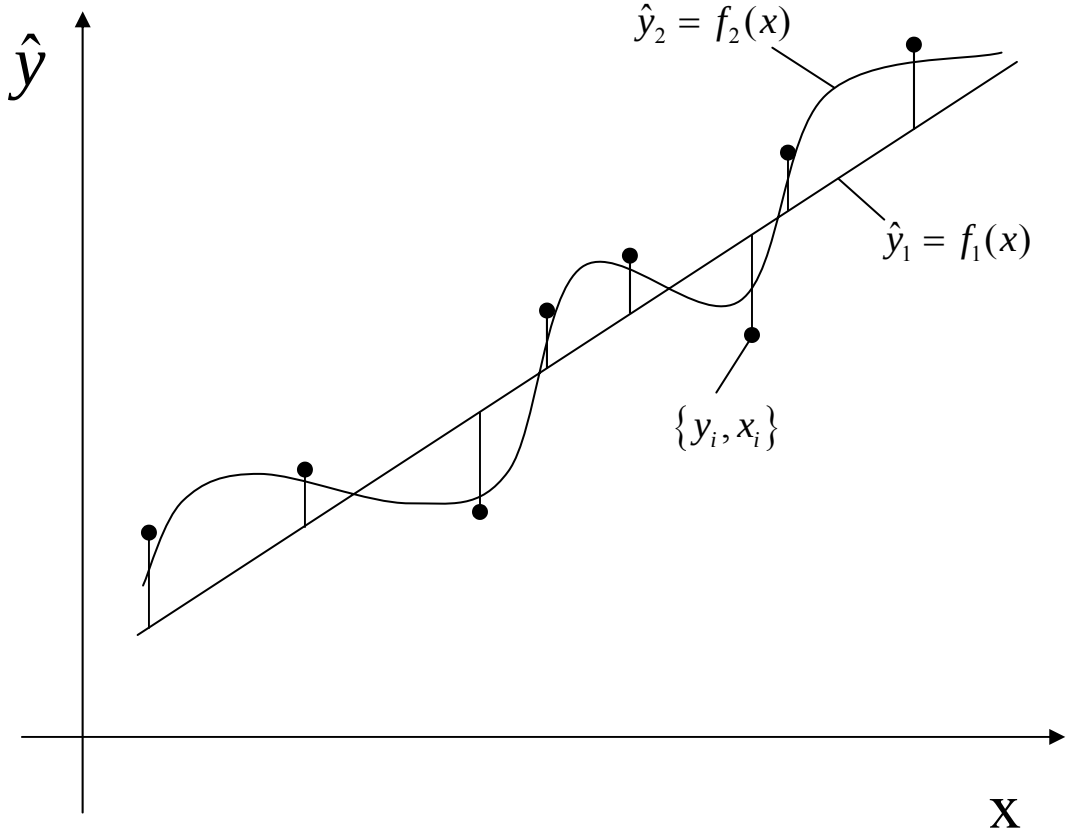
- Φ increases with h
- R_{emp} decreases with h (overfitting)

Notes for designing

There exist several expressions for the same phenomenon:

- Bias/variance-compromise
- Generalization-/Overfitting-compromise
- Empirical error/VC-dimension-(capacity control) -compromise

Bias-variance-tradeoff



f_1 :

- Bias $\quad \frac{1}{n}\sum_{i=1}^{n}(y_i - f_1(x_i))$ little
- variance $\quad \frac{1}{n}\sum_{i=1}^{n}(y_i - f_1(x_i))^2$ big

good generalization (simple model)!

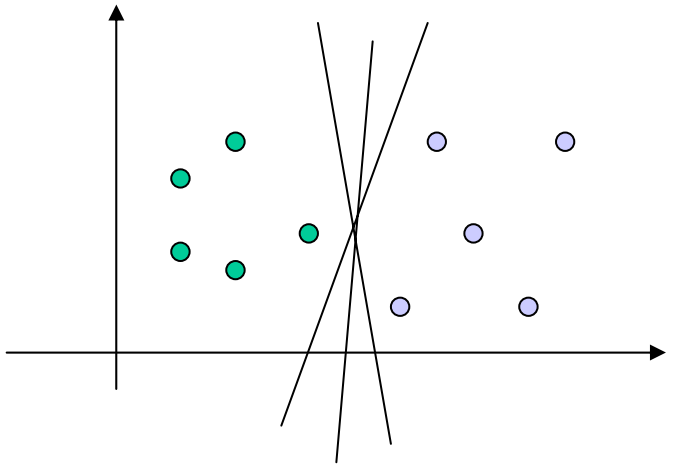
f_2 :

- Bias $\quad \frac{1}{n}\sum_{i=1}^{n}(y_i - f_2(x_i))$ big
- variance $\quad \frac{1}{n}\sum_{i=1}^{n}(y_i - f_2(x_i))^2$ little

bad generalization (overfitting)!

Linearly separable classes

- Data in $\mathbb{R}^2$



  - General hyperplane $\langle \mathbf{w}, \mathbf{x} \rangle + b = 0$
  - classification via $f(x) = \mathrm{sgn}(\langle \mathbf{w}, \mathbf{x} \rangle + b)$
  - e.g. Rosenblatt´s Perceptron (1956)

    Iterative learning, correction after every false classification

    -> solution not unique

Optimal hyperplane: maximizing the margin

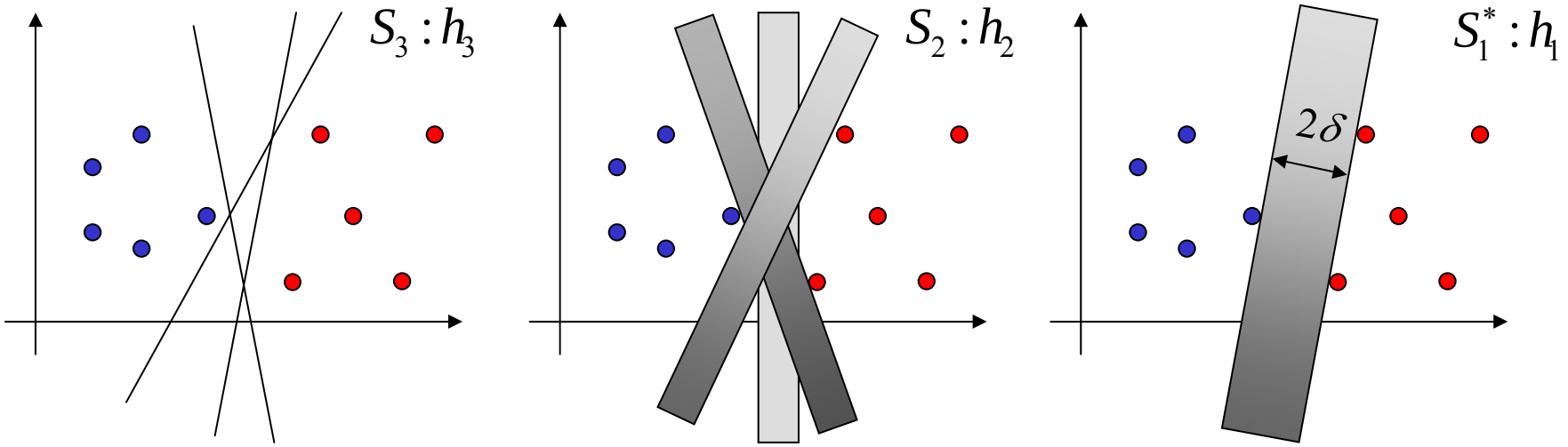
- Considering a linearly separable two-class-problem all lines, that separate both classes, can reach an empirical error zero
- The confidence is minimized by a polynom of minimal VC-dimension, that is a hyperplane
- The VC-dimension can be decreased further by large margin hyperplanes

• Separating hyperplane with maximal margin		• Separating hyperplane with minimal VC-dimension

- This result is plausible. With constant intra-class dispersion the classification certainty increases with increasing inter-class distance.
- or: with constant inter-class distance the maximal intra-class dispersion must be maximally feasible.

“Large-margin“-classifier

– Hyperplane with largest margin [Vap63]



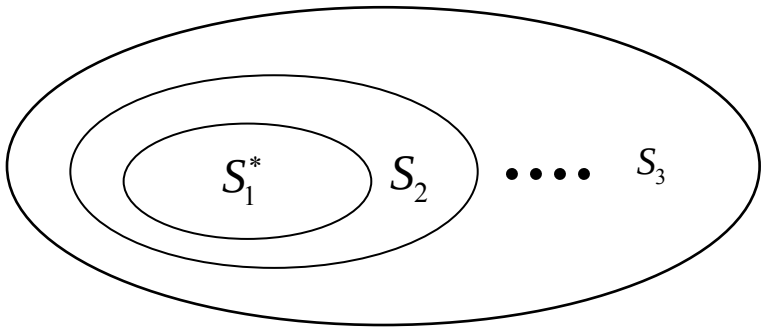
high VC-dimension

in $\mathbb{R}^N : N+1$

medium VC-dimension
variability gets smaller!

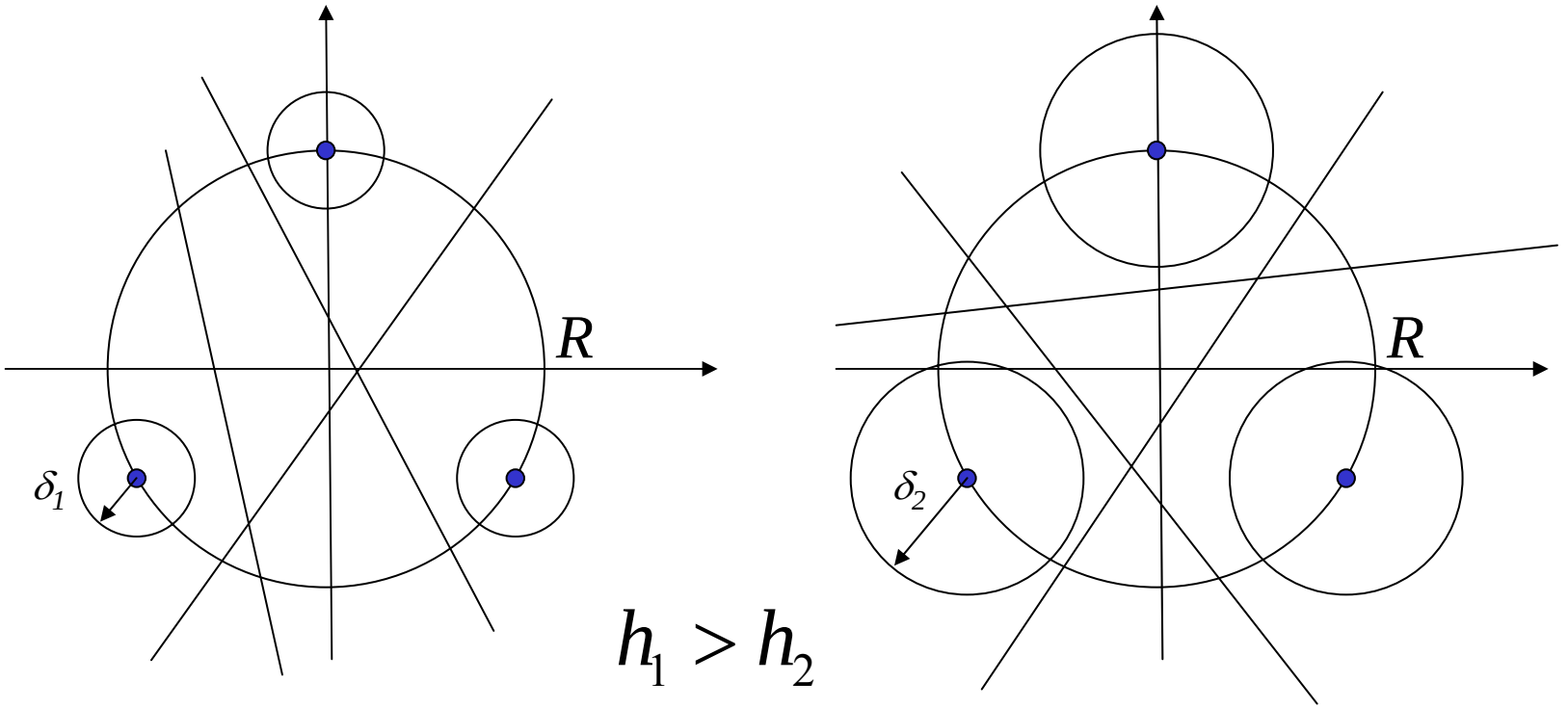
smallest VC-dimension
mit maximal breadth
variability equals zero

- Clearly reasonable
- theoretically founded
- solution depends on few data: =>“support-vectors“



$h_1 < h_2 < \ldots < h_3$

VC-dimension of „broad“ hyperplanes

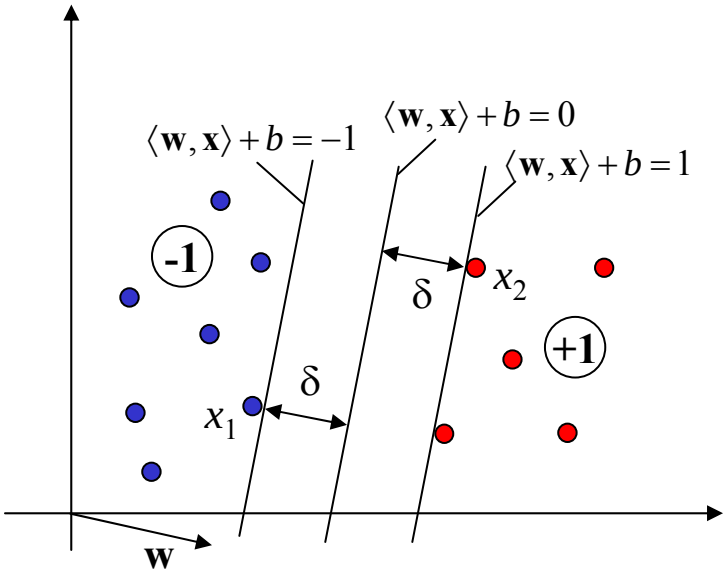


$$h_1 > h_2$$

The VC-dimension h of hyperplanes with minimal distance δ of the points to be separated is limited by:

$$h \leq \frac{R^2}{\delta^2} + 1 \qquad \Rightarrow \text{big margin } \delta\text{, small VCdim } h$$

Formalization: sought is separation plane with maximum margin δ



distance of a point to the separating plane:

$$d(\mathbf{w}, b; \mathbf{x}) = \frac{|\langle \mathbf{w}, \mathbf{x} \rangle + b|}{\|\mathbf{w}\|}$$

The distance between the canonical hyperplanes results from projection of $\mathbf{x}_1$-$\mathbf{x}_2$ to the normal vector $\mathbf{w}/\|\mathbf{w}\|$ under the constraint:

The data is classified correctly if:

$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) > 0$$

Note: Introducing *canonical hyperplanes* by points of both classes that lie closest to the separation plane. The expression is invariant to a positive re-scaling:

$$y_i(\langle a\mathbf{w}, \mathbf{x}_i \rangle + ab) > 0$$

A is chosen, so that the following applies:

$\langle \mathbf{w}, \mathbf{x}_1 \rangle + b = -1$ margin for blue class

$\langle \mathbf{w}, \mathbf{x}_2 \rangle + b = +1$ margin for red class

$\{\mathbf{x}_1, \mathbf{x}_2\}$ support vectors

$$2\delta = \left| \langle \frac{\mathbf{w}}{\|\mathbf{w}\|}, (\mathbf{x}_2 - \mathbf{x}_1) \rangle \right| = \frac{1}{\|\mathbf{w}\|} \left| \underbrace{\langle \mathbf{w}, \mathbf{x}_2 \rangle}_{(1-b)} - \underbrace{\langle \mathbf{w}, \mathbf{x}_1 \rangle}_{-(1+b)} \right| = \frac{2}{\|\mathbf{w}\|}$$

$$\Rightarrow \quad \boxed{\delta = 1/\|\mathbf{w}\|}$$

Maximization of δ is equal to minimization of $\|\mathbf{w}\|^2$ =>

Optimization problem

Primal OP:

$$\text{minimize} \quad J(\mathbf{w},b) = \tfrac{1}{2}\|\mathbf{w}\|^2 = \tfrac{1}{2}\langle \mathbf{w},\mathbf{w}\rangle$$

$$\text{under side cond.:} \quad \forall i \left[y_i(\langle \mathbf{w},\mathbf{x}_i\rangle + b) \geq 1 \right]$$

Introducing a Lagrange function:

$$L(\mathbf{w},b,\alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{l} \alpha_i [y_i(\langle \mathbf{w},\mathbf{x}_i\rangle + b) - 1]$$

with: $\alpha_i \geq 0$

The partial derivates after w_i, b and the α_i lead after inserting in the primal OP to the equivalent:

Wolf-duale OP:

$$\text{maximize} \quad L'(\mathbf{w},b,\boldsymbol{\alpha}) = \sum_{i=1}^{l} \alpha_i - \tfrac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i,\mathbf{x}_j\rangle$$

$$\text{under the constraint:} \quad 0 \leq \alpha_i \leq C \ (\text{C}=\infty \text{ in case of no "slack"})$$

$$\text{and} \quad \sum_{i=1}^{l} y_i \alpha_i = 0$$

This is a positive semi-definite problem, that can be solved numeric iteratively using convex quadratic programming!

Solution:

Solution of the dual problem yields uniquely the desired hyper plane (equations also apply for the case with slack variables!)

$$\mathbf{w}^* = \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i = \sum_{\mathbf{x}_i \in SV} \alpha_i y_i \mathbf{x}_i$$

$$b^* = -\tfrac{1}{2}\left(\max_{i, y_i = -1} (\langle \mathbf{w}^*, \mathbf{x}_i \rangle) + \min_{i, y_i = +1} (\langle \mathbf{w}^*, \mathbf{x}_i \rangle) \right) = -\tfrac{1}{2}\left(\max_{\mathbf{x}_i \in SV, y_i = -1} (\langle \mathbf{w}^*, \mathbf{x}_i \rangle) + \min_{\mathbf{x}_i \in SV, y_i = +1} (\langle \mathbf{w}^*, \mathbf{x}_i \rangle) \right)$$

classification: $f(\mathbf{x}) = \operatorname{sgn}(\langle \mathbf{w}^*, \mathbf{x} \rangle + b^*) = \operatorname{sgn}(\sum_{\mathbf{x}_i \in SV} \alpha_i y_i \langle \mathbf{x}_i, \mathbf{x} \rangle + b^*)$

Solution only dependent on support vectors!!

Observations:

- For support vectors applies: $0 < \alpha_i < \infty$
- For all non-support-vectors applies: $\alpha_i = 0$
  -> support vectors, “sparse“-representation of solution
- Uniqueness of plane, *global optimum*!!

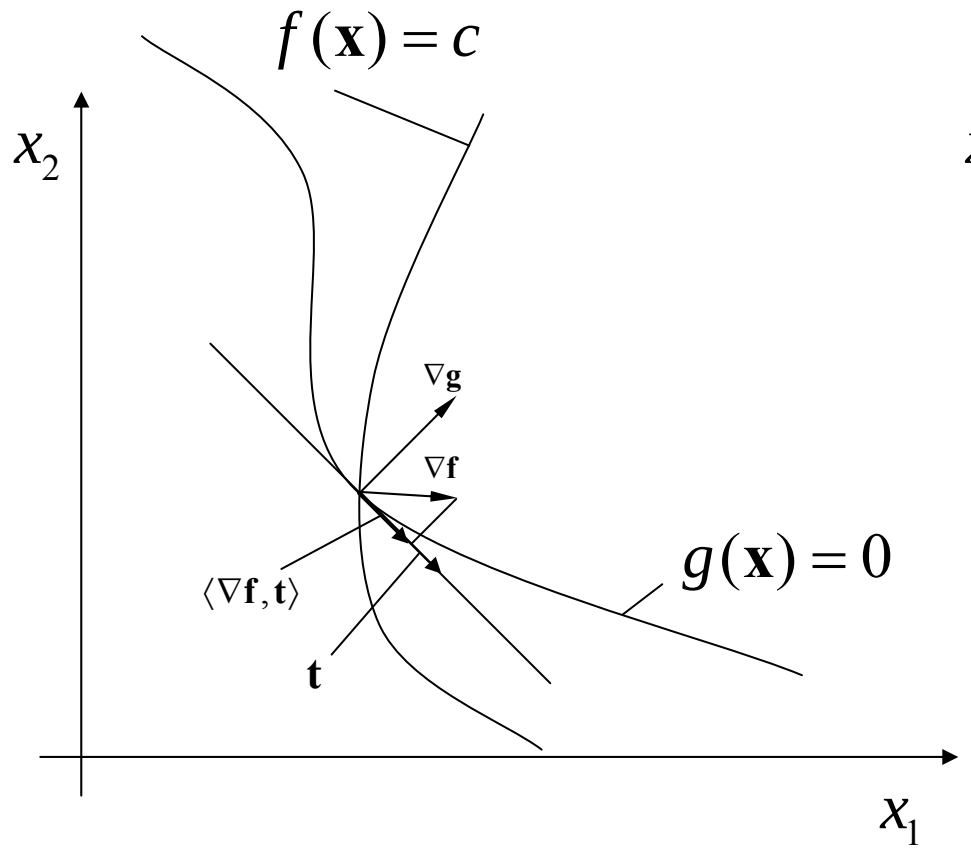
The gradient is a normal vector to the curve $g(\mathbf{x})=0$

Considering the Taylor expansion of g regarding a small vectorial noise $\boldsymbol{\varepsilon}$ at x on the curve $g(\mathbf{x})=0$ results in:

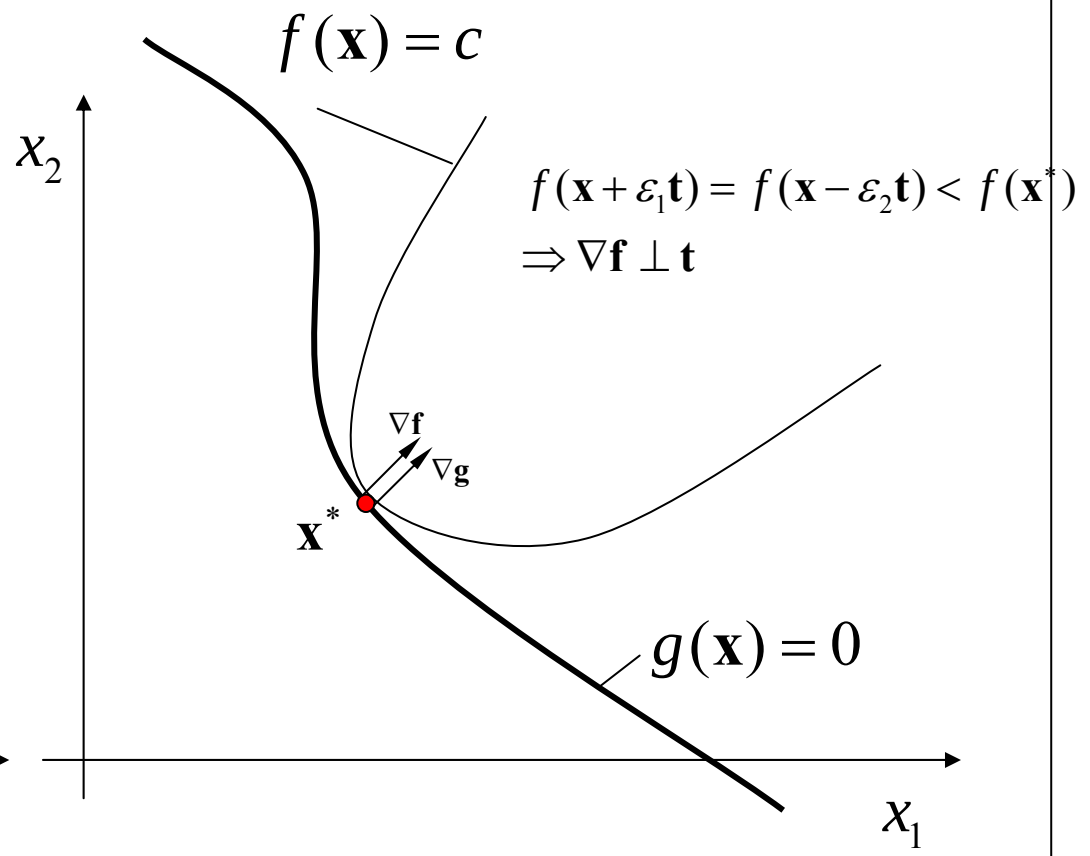
$$g(\mathbf{x}+\boldsymbol{\varepsilon}) = g(\mathbf{x}) + \boldsymbol{\varepsilon}^T \nabla \mathbf{g} + \ldots$$

If the noise $\boldsymbol{\varepsilon}$ moves along the curve, $g(\mathbf{x}+\boldsymbol{\varepsilon}) = g(\mathbf{x})$ applies and thus $\boldsymbol{\varepsilon}^T \nabla \mathbf{g}(\mathbf{x}) = 0$. From that we can see, that the gradient is orthogonal to the surface $g(\mathbf{x})=0$ (normal vector).

Constraint for a stationary point **x***



a) relations at a non-static point



b) relations at a static point

a) If the projection of $\nabla \mathbf{f}$ to the tangential direction $\mathbf{t}$ contains a contribution other than 0, the criterion for optimization can be improved by motion along the curve of the side condition in this direction! => no stationary point!

b) The quality function f(x) is worsened at a static point in *both* tangential directions. $\nabla \mathbf{f}$ is orthogonal to $\mathbf{t}$.

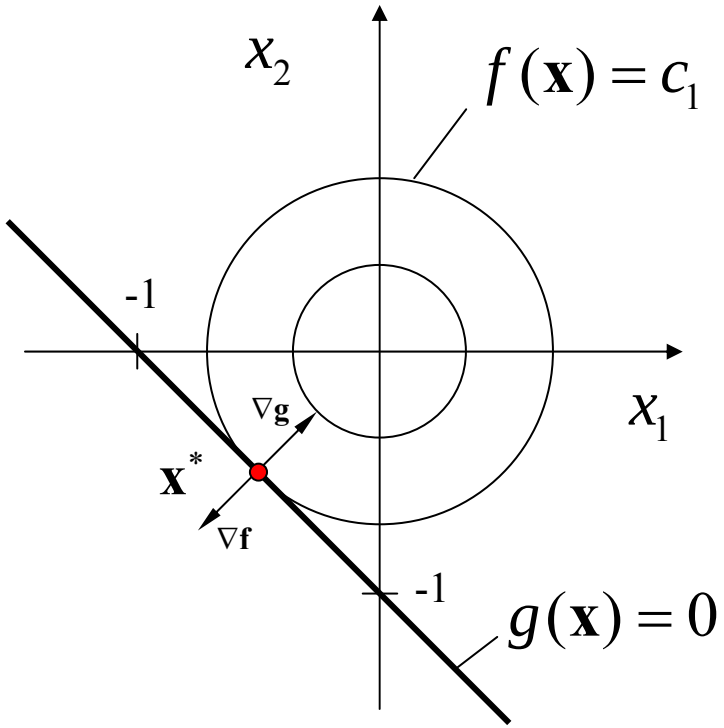
Example:

1) $f(\mathbf{x}) = x_1^2 + x_2^2$

2) NB: $g(\mathbf{x}) = x_1 + x_2 + 1 = 0$

$$\nabla \mathbf{f} = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \end{bmatrix} = 2 \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\nabla \mathbf{g} = \begin{bmatrix} \frac{\partial g}{\partial x_1} \\ \frac{\partial g}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$



$$\boxed{\nabla \mathbf{f}^* \parallel \nabla \mathbf{g}^*}$$

$L(\mathbf{x}, \lambda) = f(\mathbf{x}) + \lambda g(\mathbf{x})$

necc. constr. for an optimum:

1) $\nabla \mathbf{L} = \frac{\partial L}{\partial \mathbf{x}} = \nabla \mathbf{f} + \lambda \nabla \mathbf{g} = \mathbf{0}$

2) $\frac{\partial L}{\partial \lambda} = g(\mathbf{x}) = 0$

$\Rightarrow$

1) $2x_1 + \lambda = 0$

$2x_2 + \lambda = 0$

2) $x_1 + x_2 + 1 = 0$

$$\Rightarrow \lambda^* = 1 \text{ and } \mathbf{x}^* = -\begin{bmatrix} 1/2 \\ 1/2 \end{bmatrix}$$

$$\Rightarrow \nabla \mathbf{f}(\mathbf{x}^*) = \begin{bmatrix} -1 \\ -1 \end{bmatrix} = -\lambda^* \nabla \mathbf{g}(\mathbf{x}^*) = -1 \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

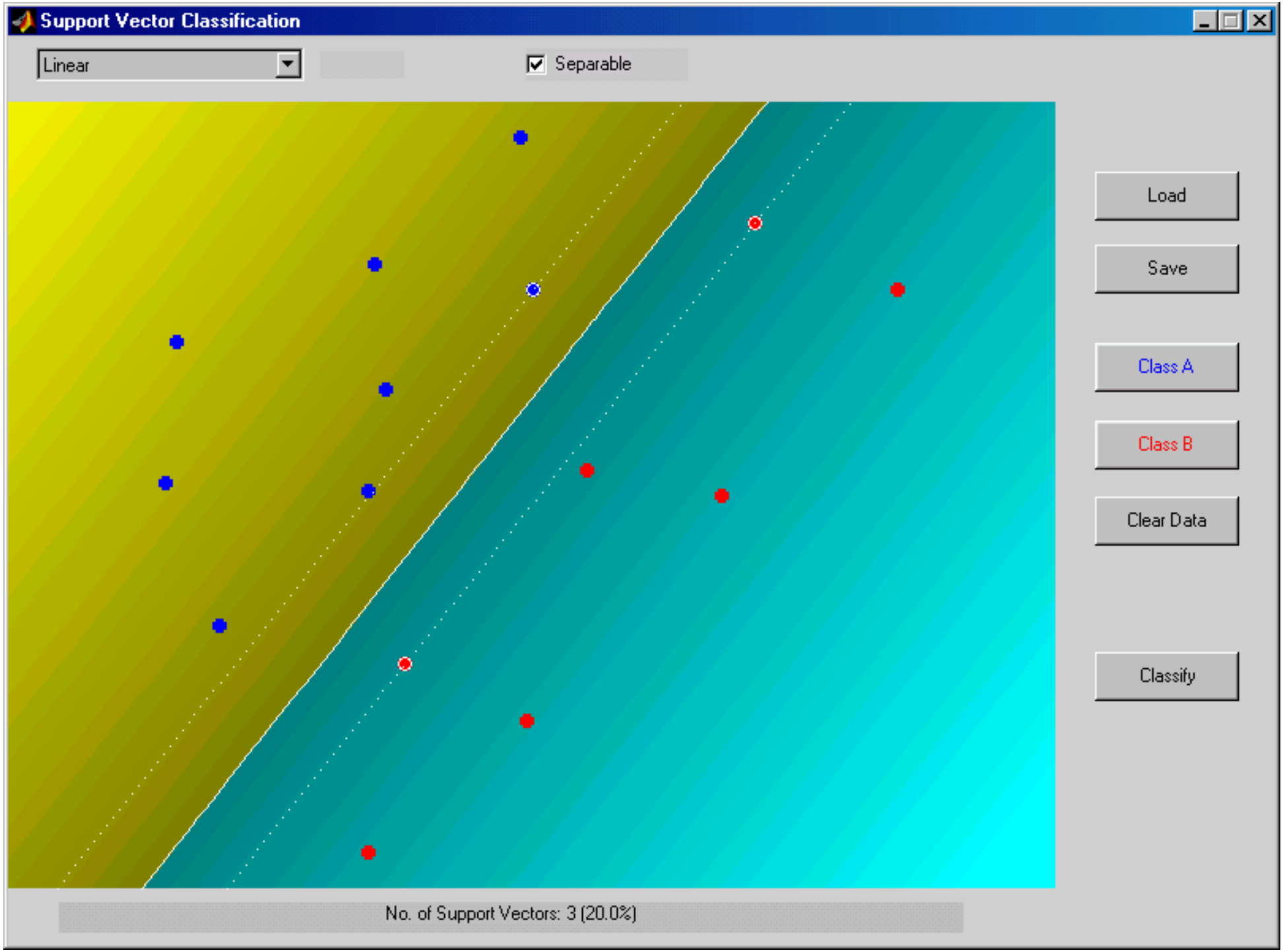
Demos with MATLAB

(svmmatlab, uiclass.m)

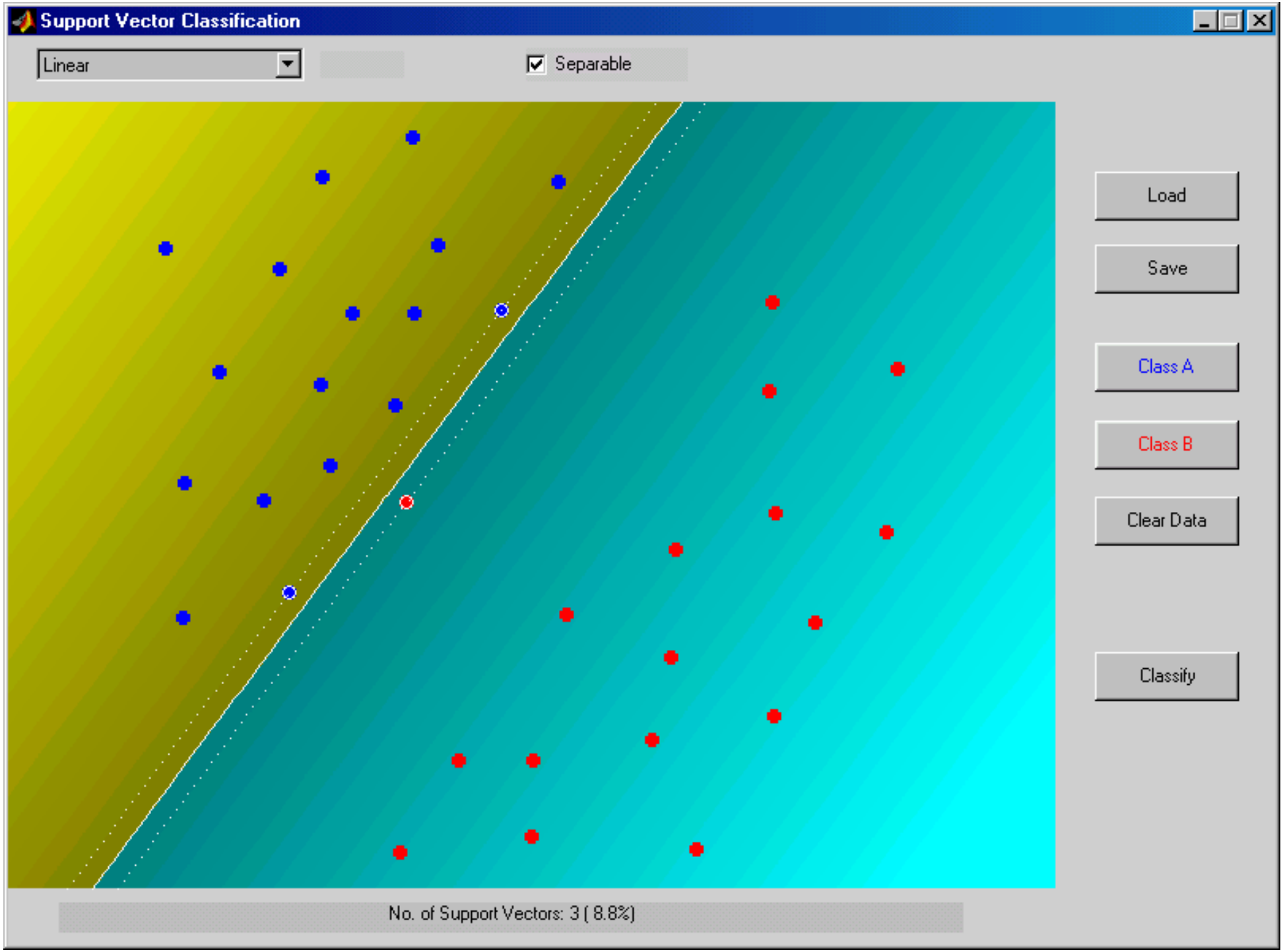
- Linear classifier
  - hard margin, problem separable
  - hard, small margin due to outlier, problem separable, but bad generalization => increasing empirical errors with profits at generlization => introduce soft margin
- Non-linear classifier
  - For non-linear problems the VC-dimension of the separating hyperplanes and thus their capacity have to be increased!
  - Linear separation of a quadratic problem with a soft margin
  - Polynomial separation (p=2), hard margin
  - Polynomial separation (p=4), hard margin
  - Banana shape with polynomial kernels
  - Banana shape with Gaussian radial basis functions

Starting Matlab-Demo
matlab-SVM.bat

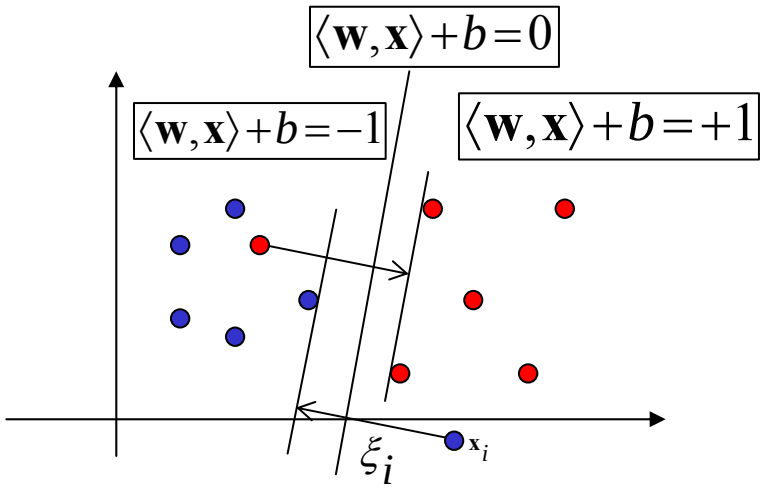
Linearly separable classes; hard margin



Linearly separable classes; hard margin, small margin, bad generalization



Non-separable case



Penalizing margin violations
with "slack"-variables
[Smith68] ->"Soft-Margin" SVM

minimizing: $\|\mathbf{w}\|^2 + C\sum_{i=1}^{l} \xi_i$

with: $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i$ and $\xi_i \geq 0$

Reasons for this kind of generalization:

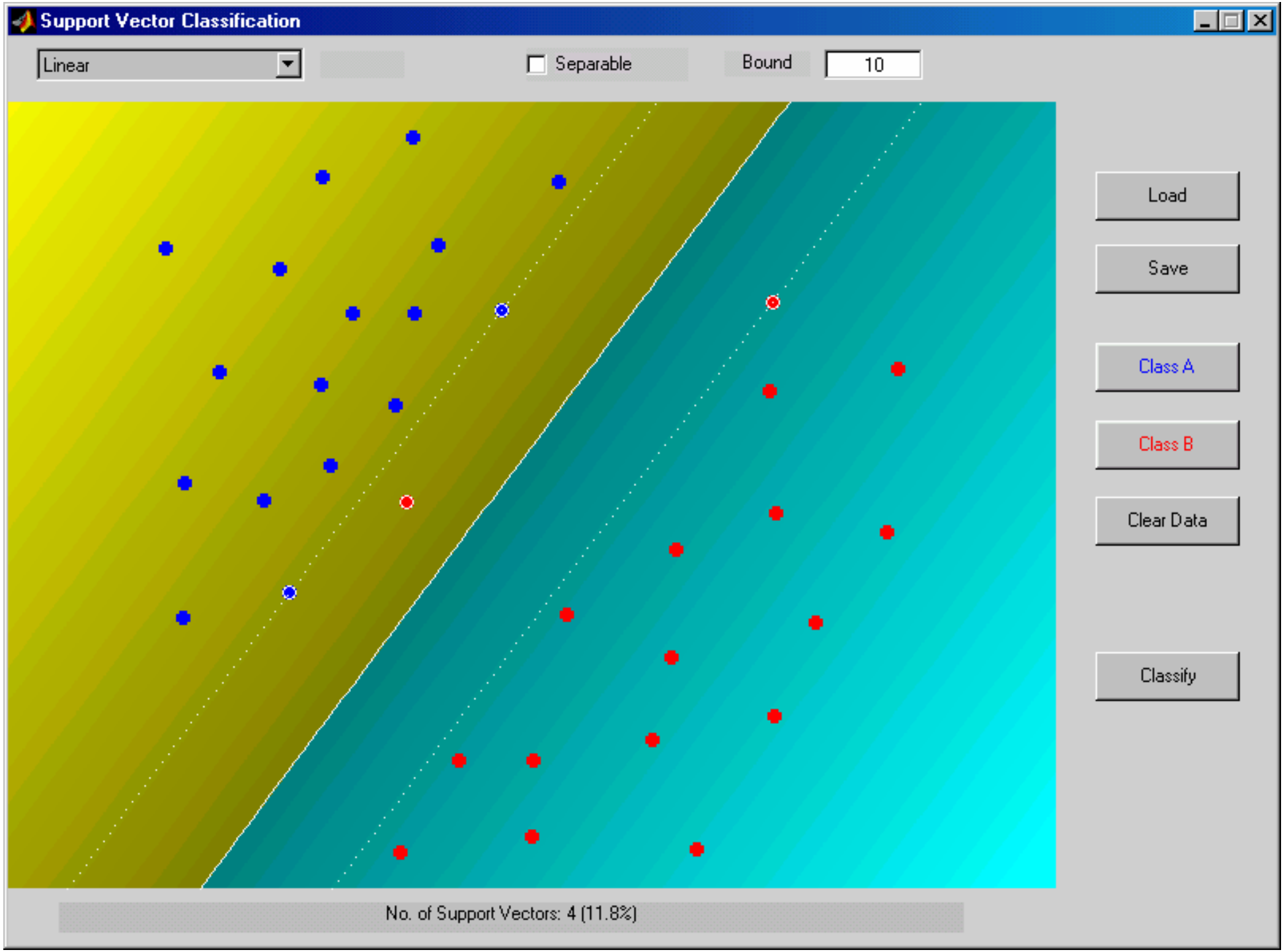
- there are no solutions with previous approach of hard margin
- generalization of outliers in the margin can be improved at the cost of a slightly increased empirical error

Now we have to distinguish three cases:

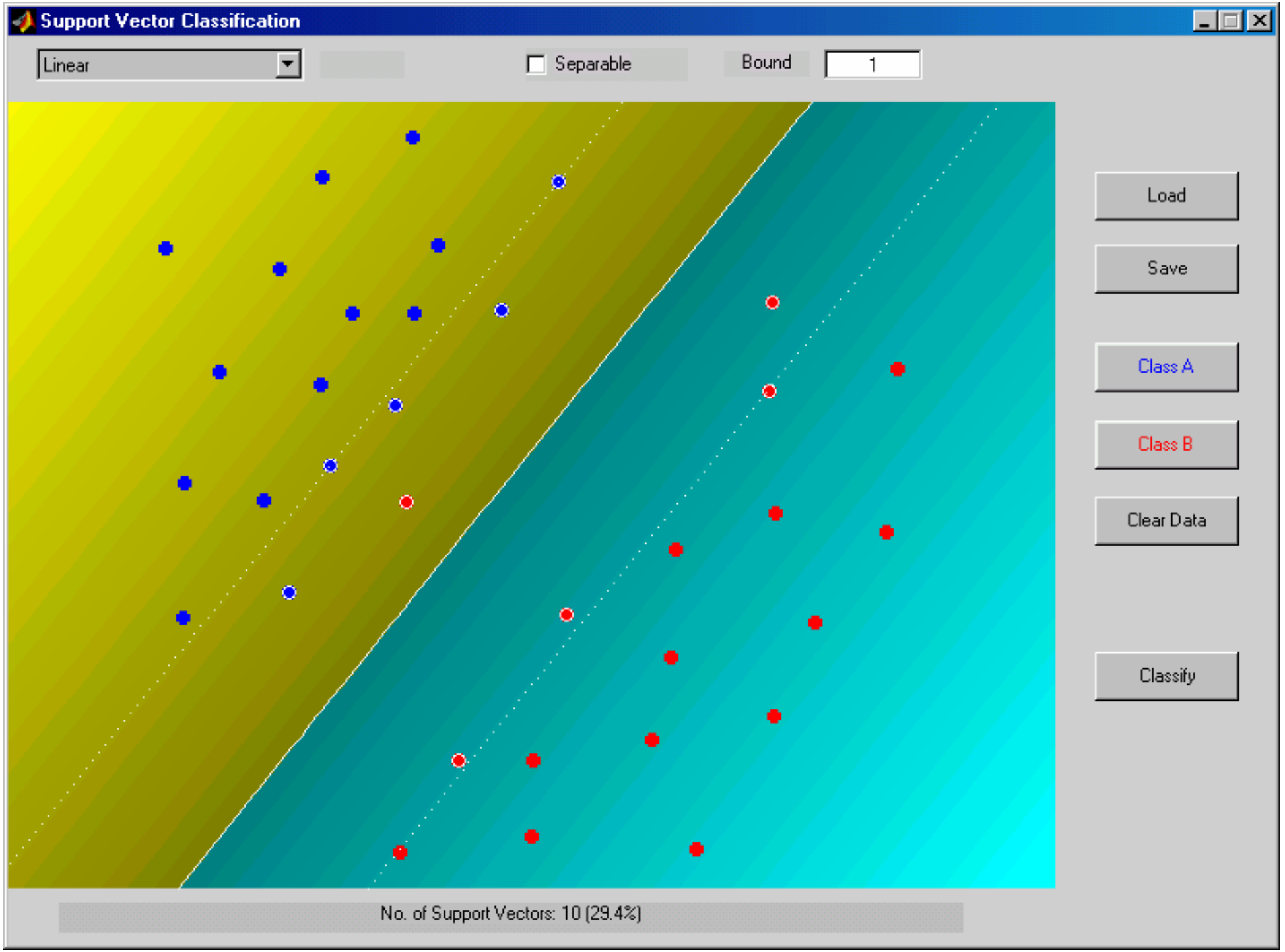
- $0<\alpha_i<C$ ⇔ SV with $\xi_i = 0$ (lie on canon. hyperplanes)
- $\alpha_i=C$ ⇔ more SV with $\xi_i > 0$ (falsely assigned)
- $\alpha_i=0$ ⇔ for the remaining vectors $\mathbf{x}_i$

$C \to \infty$ corresponds to separable approach with hard margin!

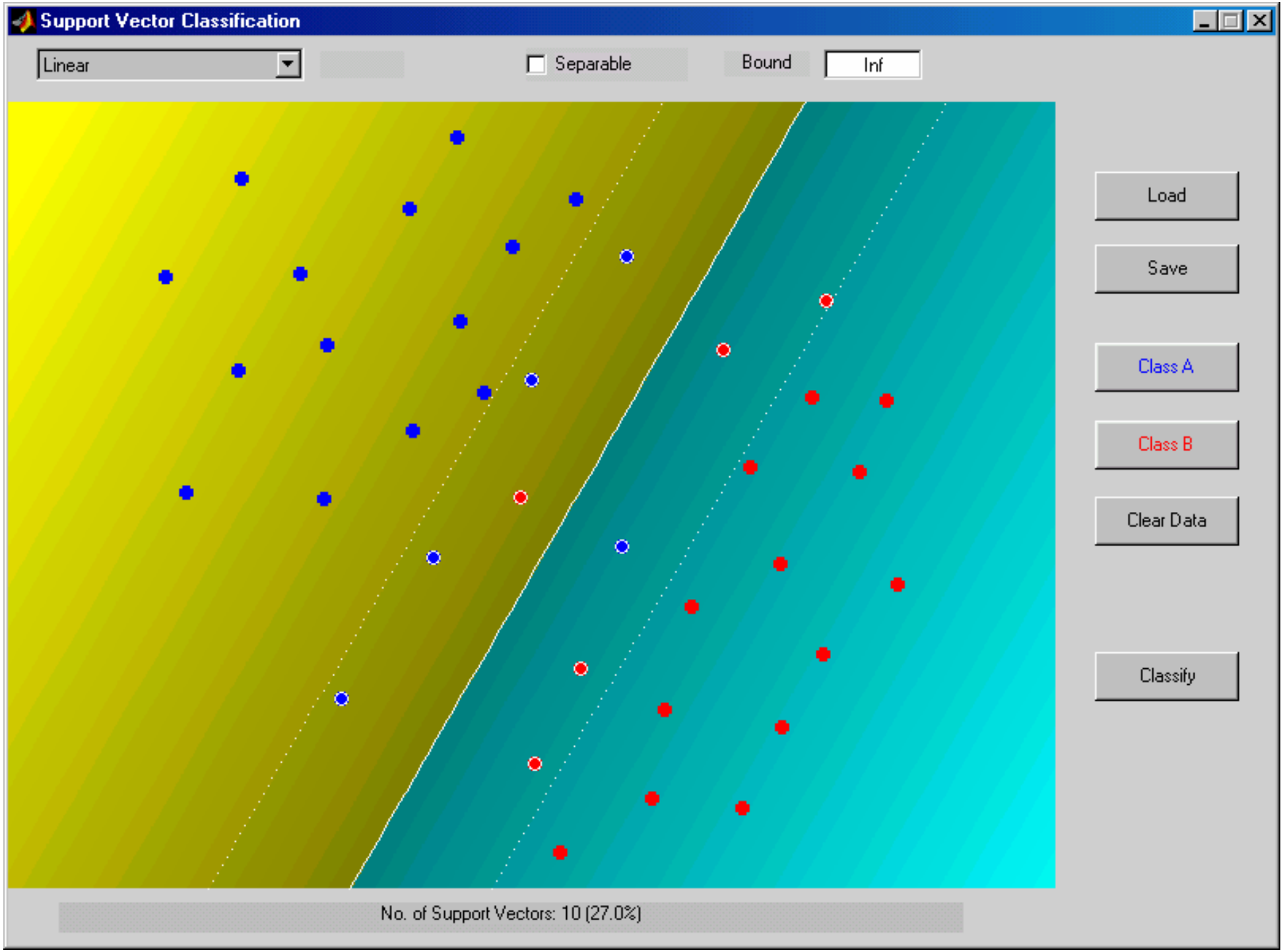
Linearly sep. classes; soft, broad margin => R_{emp} greater, good generalization



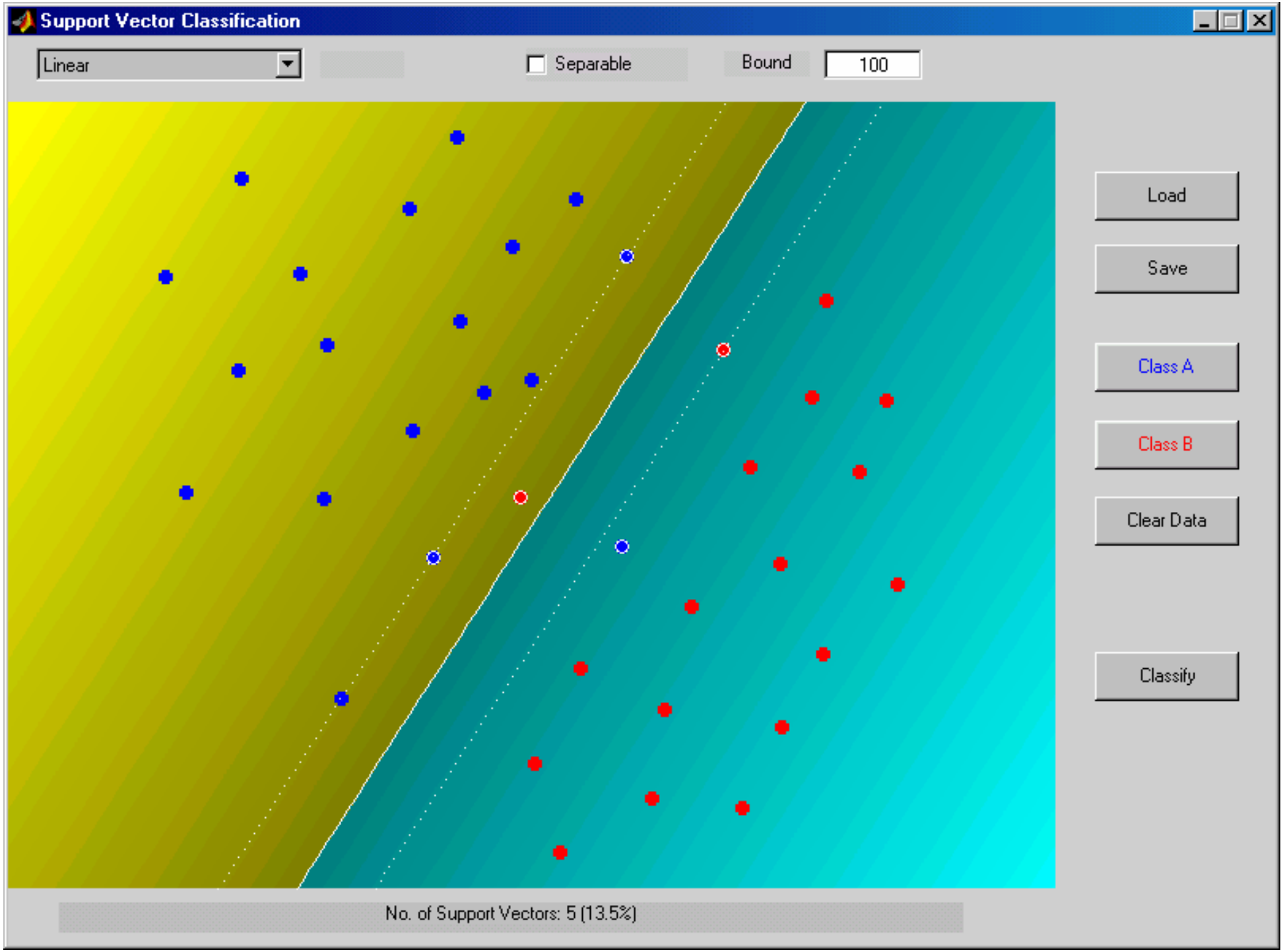
Linearly sep. classes; soft, broad margin => R_{emp} greater, good generalization



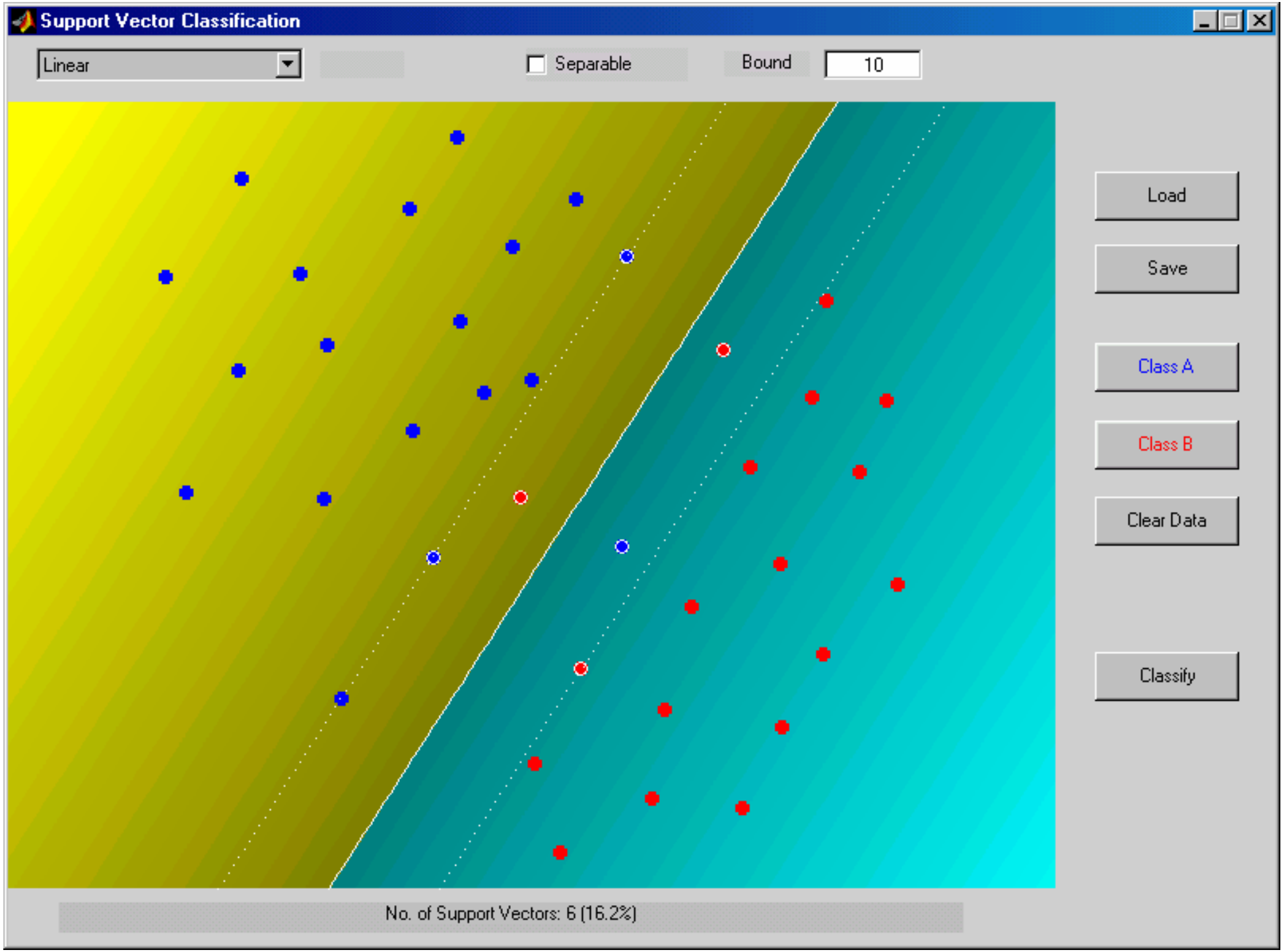
Linearly separable classes; hard margin



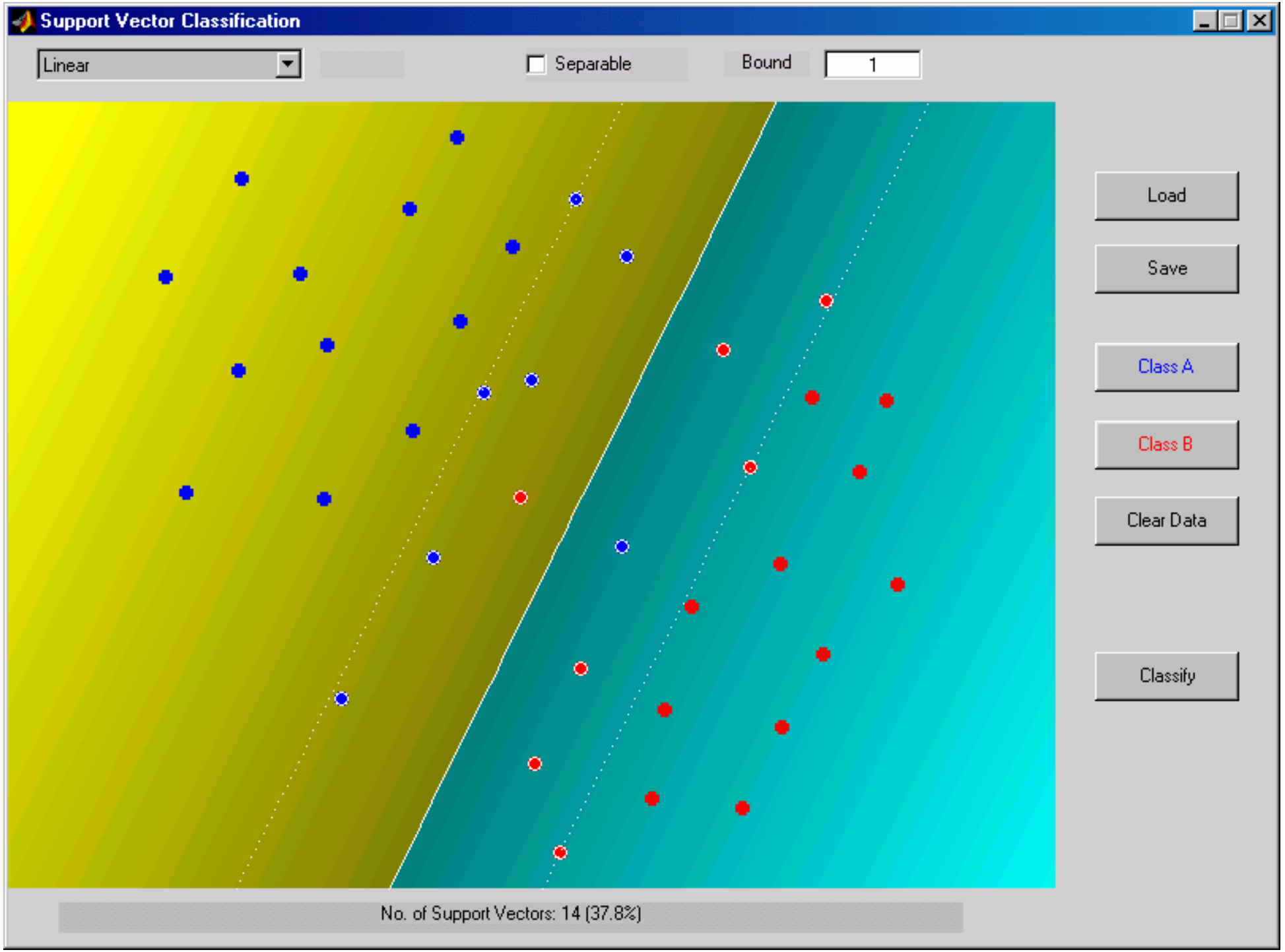
Linearly separable classes; hard margin



Linearly separable classes; hard margin

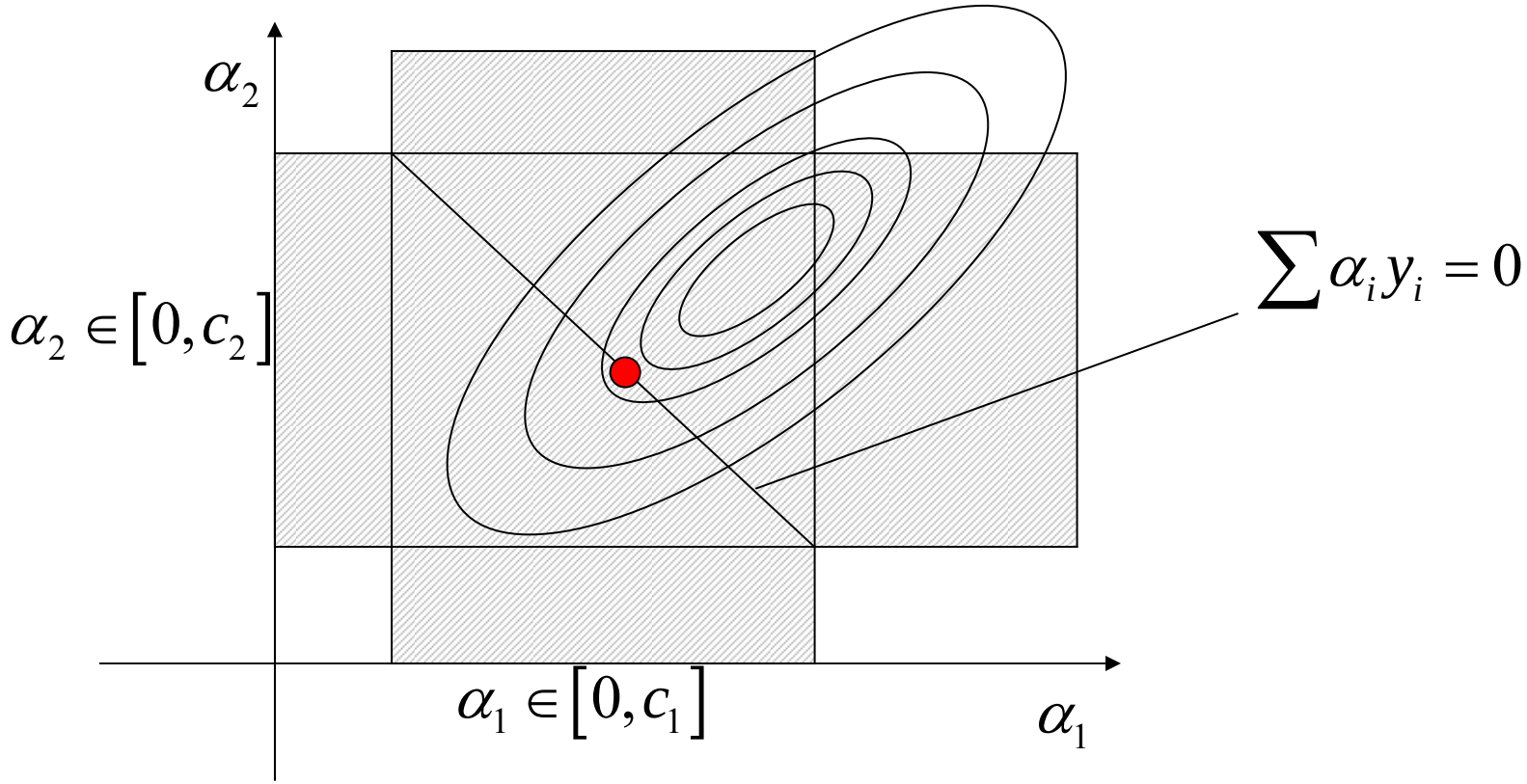


Linearly separable classes; hard margin

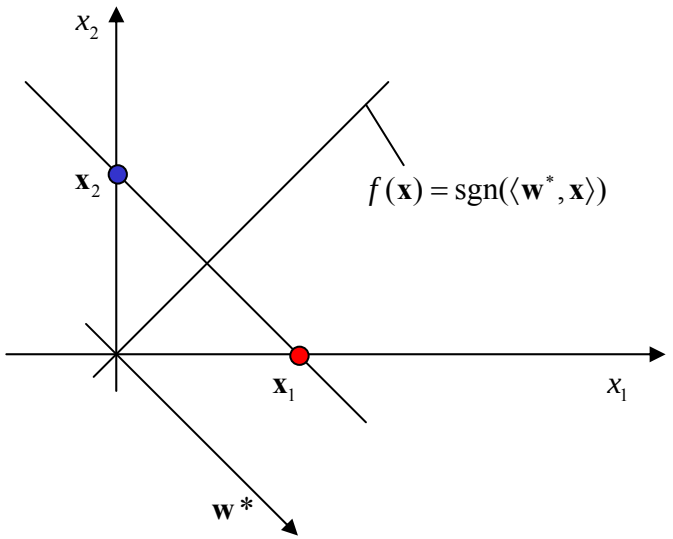


Global, unique optimum

- Optimization of a quadratic problem in **w** (convex)
- under linear constraints a global optimum results
- Admitted solution areas are convex sets; more intersected constraints also result in convex sets => when intersecting these sets with the original problem the properties of a global optimum are obtained, even if this optimum lies on the margin!



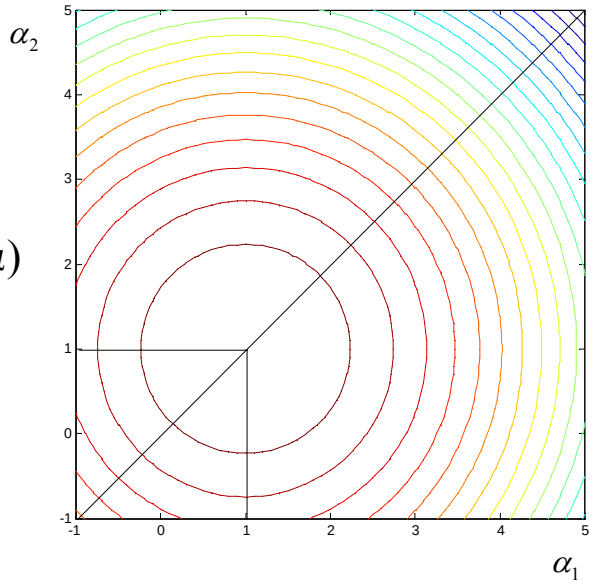
Example: Designing a SVM for 2 points



$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \mathbf{x}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad y_1 = +1 \quad y_2 = -1$$

Both vectors are support vectors!

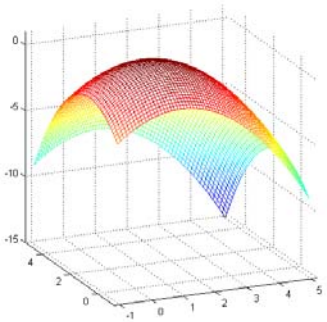
$L'(\boldsymbol{\alpha})$



1) maximize $L'(\boldsymbol{\alpha}) = \sum_{i=1}^{2} \alpha_i - \frac{1}{2} \sum_{i=1}^{2} \sum_{j=1}^{2} y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle = (\alpha_1 + \alpha_2) - \frac{1}{2}(\alpha_1^2 + \alpha_2^2)$

2) under side cond.: $0 \le \alpha_i < \infty$

3) and $\sum_{i=1}^{2} y_i \alpha_i = (\alpha_1 - \alpha_2) = 0$



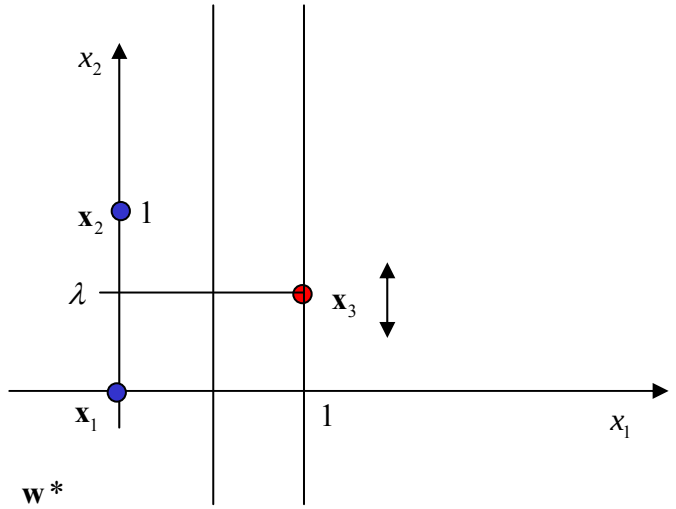
Solution: $\alpha_1 = \alpha_2 = 1$

$$\mathbf{w}^* = \sum_{\mathbf{x}_i \in SV} \alpha_i y_i \mathbf{x}_i = \mathbf{x}_1 - \mathbf{x}_2 = \begin{bmatrix} +1 \\ -1 \end{bmatrix}$$

$$b^* = -\frac{1}{2}\left(\max_{\mathbf{x}_i \in SV, y_i = -1} (\langle \mathbf{w}^*, \mathbf{x}_i \rangle) + \min_{\mathbf{x}_i \in SV, y_i = +1} (\langle \mathbf{w}^*, \mathbf{x}_i \rangle) \right) = -\frac{1}{2}(1-1) = 0$$

classification: $f(\mathbf{x}) = \text{sgn}(\langle \mathbf{w}^*, \mathbf{x} \rangle)$

Example: Designing a SVM for 3 points



$$\mathbf{x}_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \mathbf{x}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad \mathbf{x}_3 = \begin{bmatrix} 1 \\ \lambda \end{bmatrix} \qquad y_1 = y_2 = +1 \qquad y_3 = -1$$

case I: for $0 \le \lambda \le 1$ three support vectors
case II: for other values: two support vect.

1) maximize $L'(\boldsymbol{\alpha}) = \sum_{i=1}^{2} \alpha_i - \frac{1}{2} \sum_{i=1}^{2} \sum_{j=1}^{2} y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle = (\alpha_1 + \alpha_2 + \alpha_3) - \frac{1}{2}(\alpha_2^2 - 2\lambda\alpha_2\alpha_3 + (1+\lambda^2)\alpha_3^2)$

2) under the side cond.: $0 \le \alpha_i < \infty$

3) and $\sum_{i=1}^{3} y_i \alpha_i = (\alpha_1 + \alpha_2 - \alpha_3) = 0 \quad \Rightarrow \quad \alpha_1 = \alpha_3 - \alpha_2$

3) in 1) inserted results in: $\boxed{L'(\boldsymbol{\alpha}) = 2\alpha_3 - \frac{1}{2}(\alpha_2^2 - 2\lambda\alpha_2\alpha_3 + (1+\lambda^2)\alpha_3^2)}$

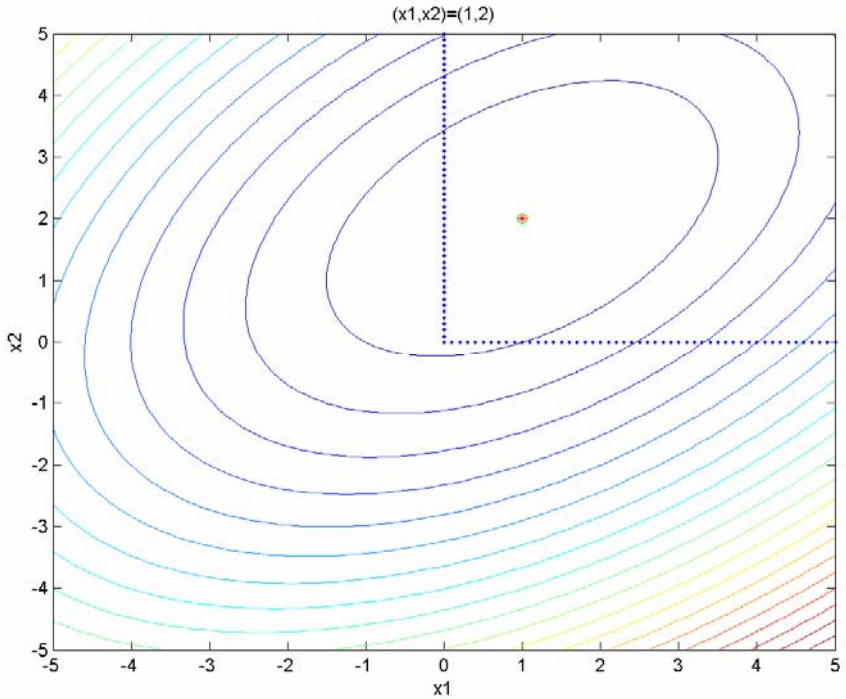
Solution: $\boxed{\alpha_1 = 2(1-\lambda) \quad \alpha_2 = 2\lambda \quad \alpha_3 = 2\lambda}$ for $0 \le \lambda \le 1$ are all $\alpha_i \ge 0$ (case I)

thus applies:

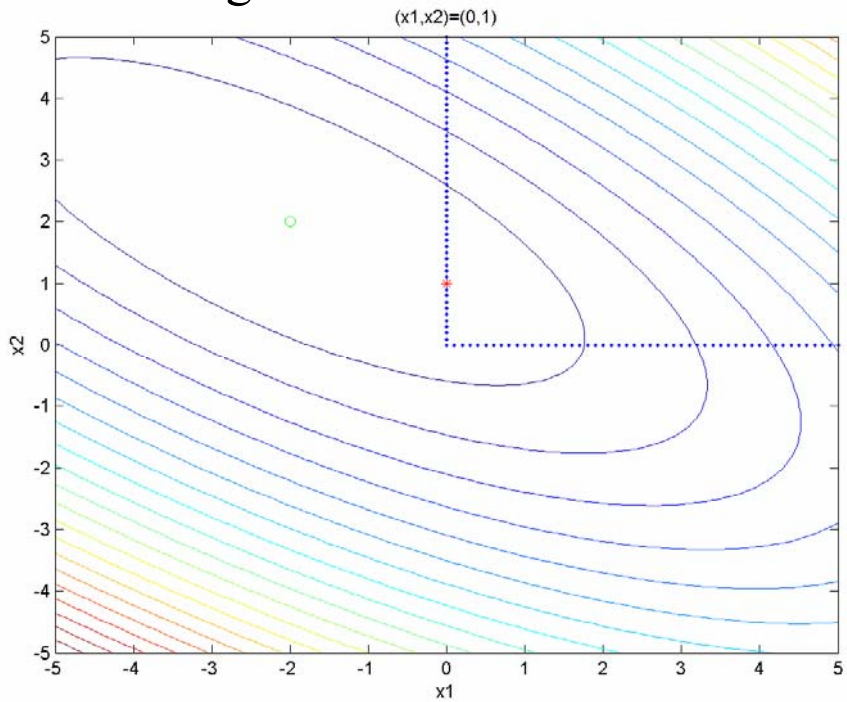
$$\mathbf{w}^* = \sum_{\mathbf{x}_i \in SV} \alpha_i y_i \mathbf{x}_i = 2\lambda \begin{bmatrix} 0 \\ 1 \end{bmatrix} - 2 \begin{bmatrix} 1 \\ \lambda \end{bmatrix} = \begin{bmatrix} -2 \\ 0 \end{bmatrix}$$

$$b^* = -\frac{1}{2}\left(\langle \mathbf{w}^*, \mathbf{x}_3 \rangle + \min_{\mathbf{x}_i \in SV, y_i = +1} (0, \langle \mathbf{w}^*, \mathbf{x}_2 \rangle) \right) = -\frac{1}{2}(-2+0) = 1$$

case I: λ=0,5 absolute maximum within the area $\alpha_i \geq 0$

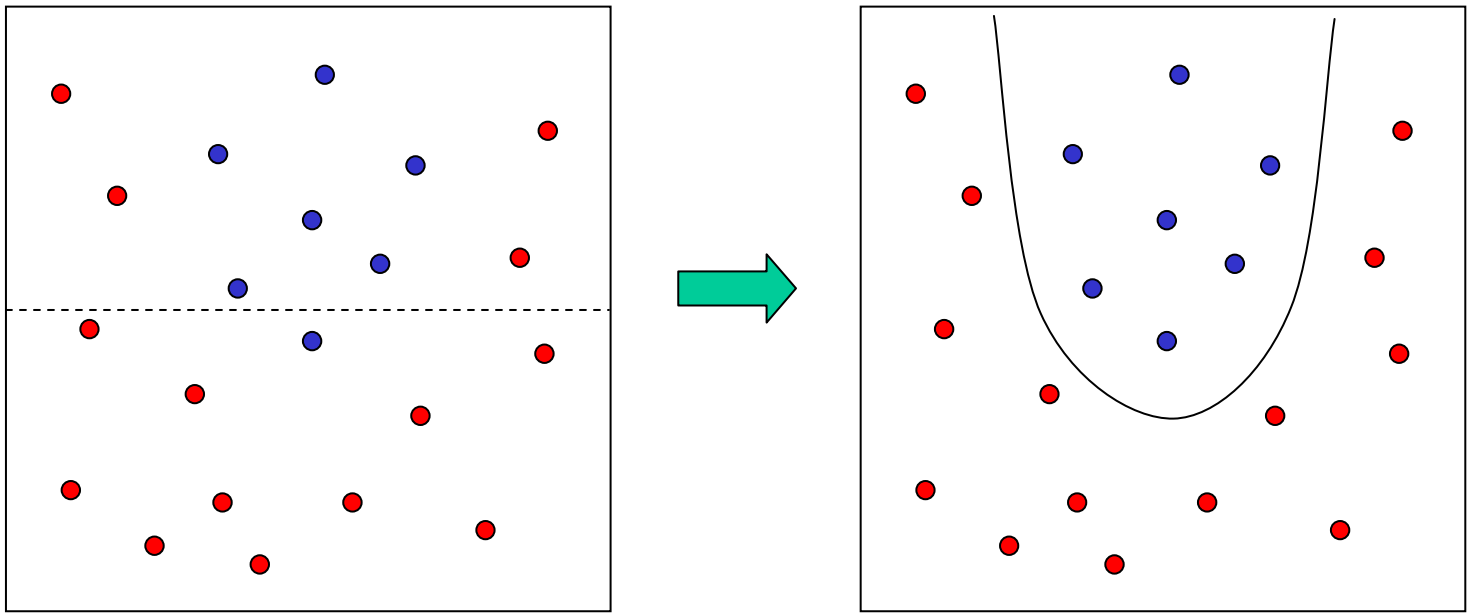


case II: λ = -1 absolute maximum outside of the area $\alpha_i \geq 0$, solution at the margin



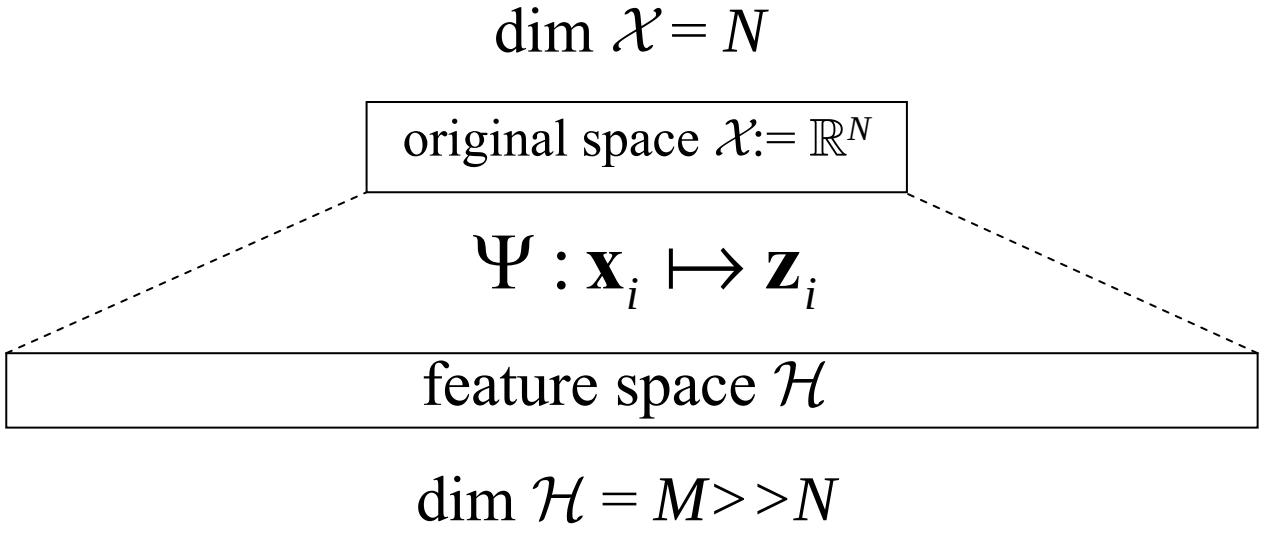
Non-linear problems

- some problems have non-linear class boundaries
- hyperplanes do not reach satisfying accuracy



Expanding the hypothesis space

Idea: Find hyperplane in higher dimensional feature space

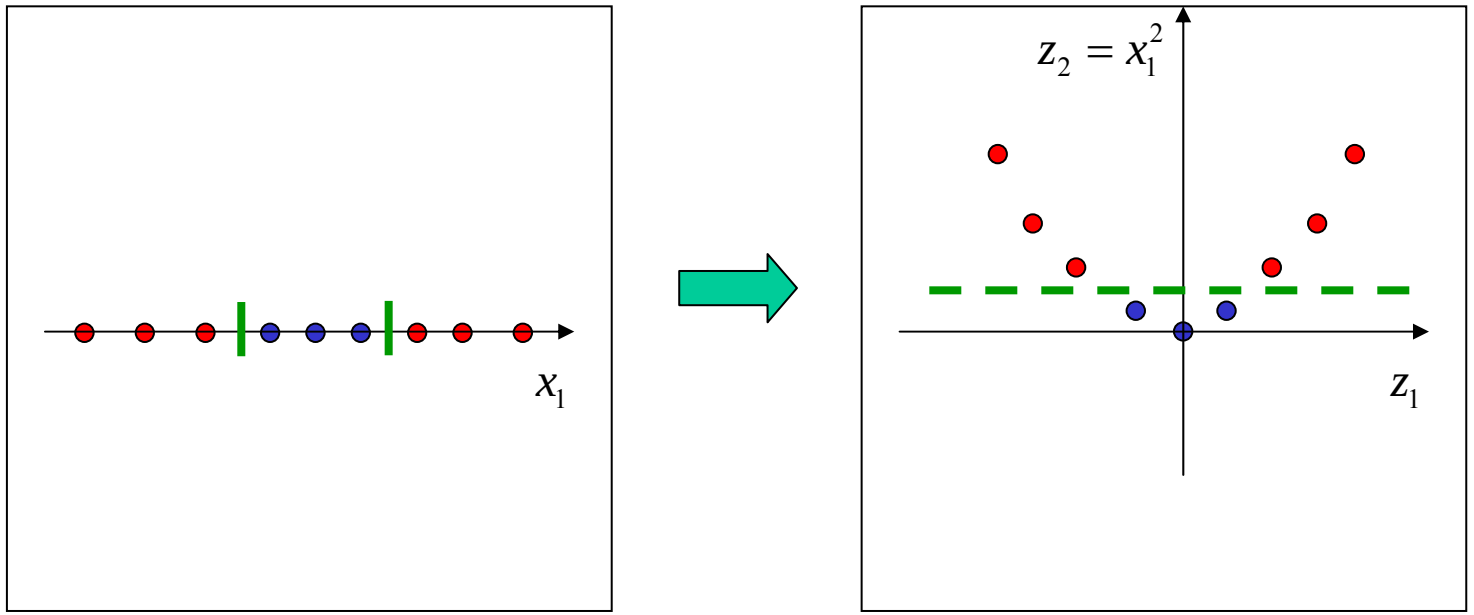


The separating hyperplane in the feature space is a non-linear separating area in the original space (see XOR-problem with polynom classifier)

Non-linear problems

- One-dimensional original space: x_1
- Two-dimensional feature space:

$$\Psi(x_1) = \mathbf{z}^T = \left[z_1 = x_1, z_2 = x_1^2 \right]^T$$



linearly not separable

linear separation

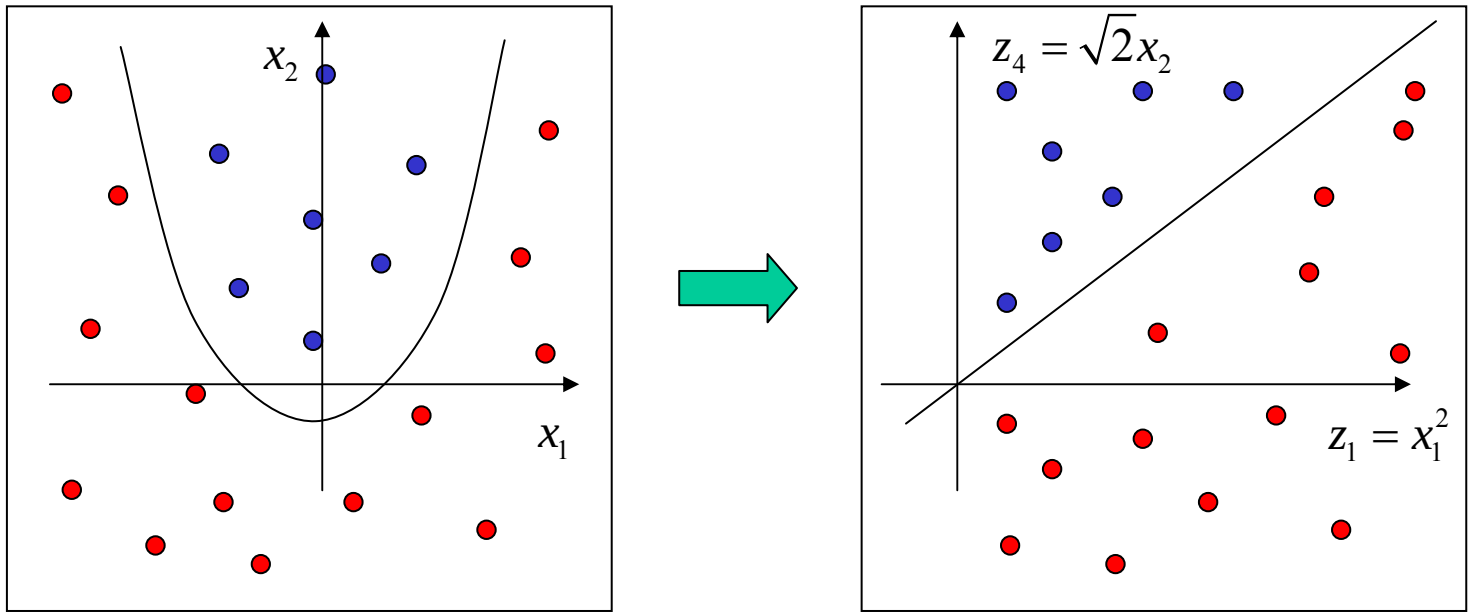
Non-linear problems

- original space:

$$\mathbf{x}=(x_1,x_2) \quad \text{(two-dimensional)}$$

- feature space:

$$\Psi(\mathbf{x}) = \mathbf{z}^T = \left[z_1 = x_1^2, z_2 = x_2^2, z_3 = \sqrt{2}x_1, z_4 = \sqrt{2}x_2, z_5 = \sqrt{2}x_1x_2, z_6 = 1 \right]^T$$



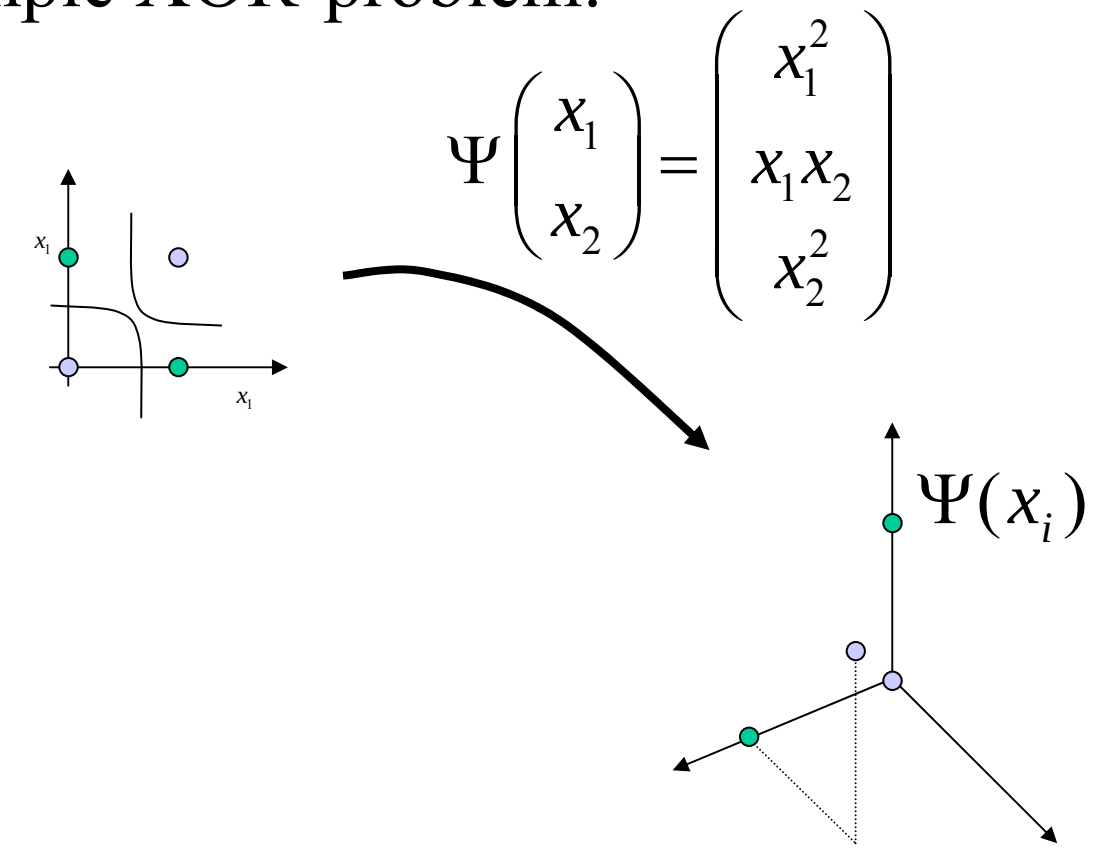
- $x_2 > x_1^2$ non-linear
- $x_2 < x_1^2$ separation

- $z_4 > \sqrt{2}z_1$ linear
- $z_4 < \sqrt{2}z_1$ separation

Non-linear expansion

Pre-compute non-linear map $\Psi(\mathbf{x}) : \mathbb{R}^N \rightarrow \mathcal{H}$

- example XOR-problem:



Consequences

- Effect:
  - increasing separability
  - separation area in the original space is non-linear
- Questions:
  1. Is the hyperplane optimal?
  2. Is the complexity in high-dimensional spaces high?
- ad 1: optimality is sustained, again positive semi-definite, since the same scalar products occur in the function to be optimized, only in a new space $\mathcal{H}$
- ad 2: the high complexity in the high-dimensional feature space $\mathcal{H}$ can be reduced using the kernel trick. The inner product in H has a equivalent formulation with kernel function in the original space $\mathcal{X}$

The kernel trick

Problem: Very high dimension of the feature space! Polynoms of p-th degree above dimension N of the original space lead to $O(M=N^p)$ dimensions of the feature space!

Solution: In the dual OP only scalar products $<\mathbf{x}_i,\mathbf{x}_j>$ occur. In the corresponding problem in the feature space also only scalar products in $< \Psi(\mathbf{x}_i), \Psi(\mathbf{x}_j) >$ occur. They do not need to be calculated explicitly, but can be expressed with reduced complexity using kernel functions in the original space $\mathcal{X}$:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \langle \Psi(\mathbf{x}_i), \Psi(\mathbf{x}_j) \rangle = \langle \mathbf{z}_i, \mathbf{z}_j \rangle$$

Example:

For $\Psi(\mathbf{x}) = \mathbf{z}^T = \left[z_1 = x_1^2, z_2 = x_2^2, z_3 = \sqrt{2}x_1, z_4 = \sqrt{2}x_2, z_5 = \sqrt{2}x_1x_2, z_6 = 1 \right]^T$

calculates $K(\mathbf{x}_i, \mathbf{x}_j) = (\langle \mathbf{x}_i, \mathbf{x}_j \rangle + 1)^2 = \langle \Psi(\mathbf{x}_i), \Psi(\mathbf{x}_j) \rangle$

the scalar product in the feature space.

Kernel functions often used

Polynom kernel $K(\mathbf{x}_i, \mathbf{x}_j) = (\langle \mathbf{x}_i, \mathbf{x}_j \rangle + 1)^2$

Gaussian kernel $K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\left\|\mathbf{x}_i - \mathbf{x}_j\right\|^2 / (2\sigma^2)\right)$

Sigmoid kernel $K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\kappa \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \theta)$

The resulting classifiers are comparable to polynom classifiers, radial basis functions and neural networks (however they are motivated differently).

General requirement: Mercer´s constraint guarantees, that a certain kernel function is in fact a scalar product in any space, but does not explain how the corresponding map Ψ and the space $\mathcal{H}$ look .

Also: Linear combinations of valid kernels obtain new kernels (i.e. the sum of two positive definite functions is again positive definite).

Mercer‘s theorem

There exists a map Ψ and developement .

$$K(\mathbf{x}_i, \mathbf{x}_j) = \langle \Psi(\mathbf{x}_i), \Psi(\mathbf{x}_j) \rangle$$

exactly when for an *arbirtrary* $g(\mathbf{x})$ with

$$\int g(\mathbf{x})^2 d\mathbf{x} < \infty$$

applies, that K is a symmetrical, positive semidefinte function in x :

$$\int K(\mathbf{x}_i, \mathbf{x}_j) g(\mathbf{x}_i) g(\mathbf{x}_j) d\mathbf{x}_i d\mathbf{x}_j \geq 0$$

Still, in some cases the kernel functions do not meet the Mercer constraint, but lead to a positive semi-definite Hessian matrix for a *certain* set of training data and thus converge to a global optimum.

Final formulation

– Training:

$$\text{maximize: } L(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j)$$

$$\text{with: } \sum_{i=1}^{l} y_i \alpha_i = 0 \quad \text{and} \quad 0 \le \alpha_i \le C \quad \text{(hard margin: } C = \infty)$$

– Classifying an unknown object $\mathbf{x}$ ($\alpha_i \neq 0$ for all SV):

$$\mathbf{w}^* = \sum_{i=1}^{l} \alpha_i y_i \Psi(\mathbf{x}_i) = \sum_{\mathbf{x}_i \in SV} \alpha_i y_i \Psi(\mathbf{x}_i)$$

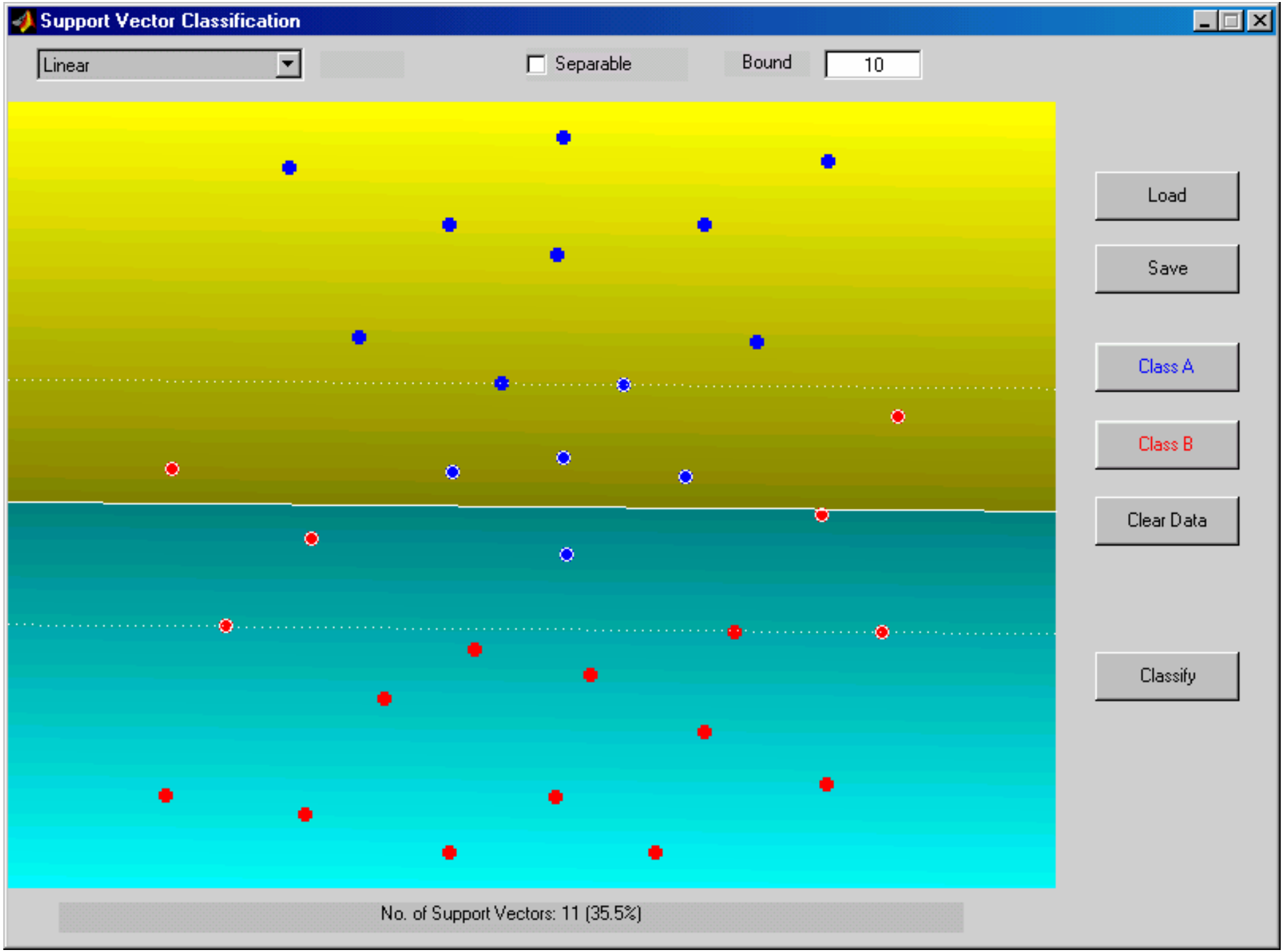
$$b^* = -\tfrac{1}{2} \left(\max_{i, y_i = -1} \left(\sum_{j=1}^{l} y_j \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \right) + \min_{i, y_i = +1} \left(\sum_{j=1}^{l} y_j \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \right) \right)$$

$$= -\tfrac{1}{2} \left(\max_{\mathbf{x}_i \in SV, y_i = -1} \left(\sum_{\mathbf{x}_j \in SV} y_j \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \right) + \max_{\mathbf{x}_i \in SV, y_i = -1} \left(\sum_{\mathbf{x}_j \in SV} y_j \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \right) \right)$$

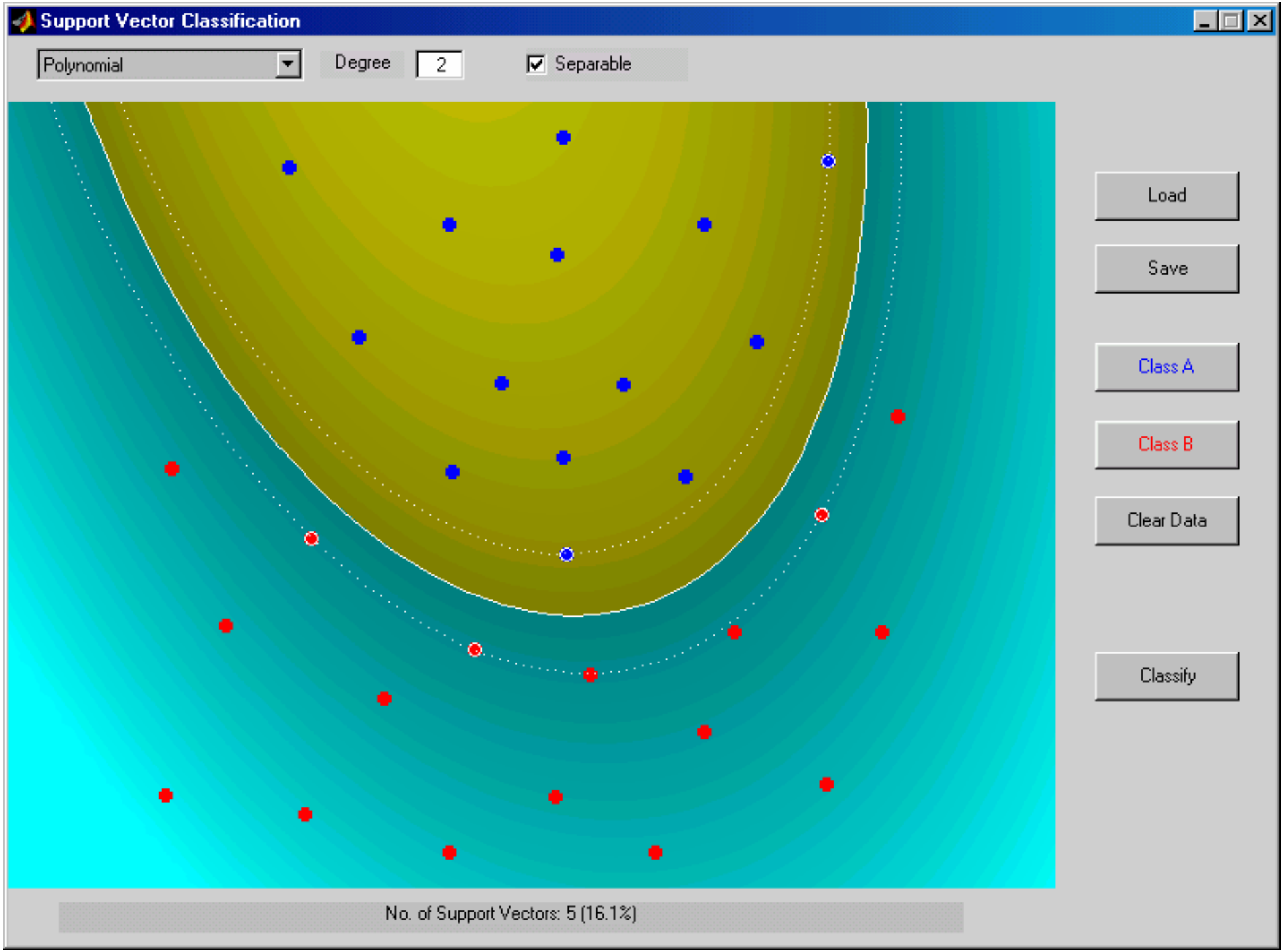
$$f(\mathbf{x}) = \operatorname{sgn}(\langle \mathbf{w}^*, \Psi(\mathbf{x}) \rangle + b^*) = \operatorname{sgn}(\sum_{\mathbf{x}_i \in SV} \alpha_i y_i \langle \Psi(\mathbf{x}_i), \Psi(\mathbf{x}) \rangle + b^*) \qquad \text{(not calculated that way!)}$$

$$= \operatorname{sgn}\left(\sum_{i=1}^{l} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b^* \right) = \operatorname{sgn}\left(\sum_{\mathbf{x}_i \in SV} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b^* \right)$$

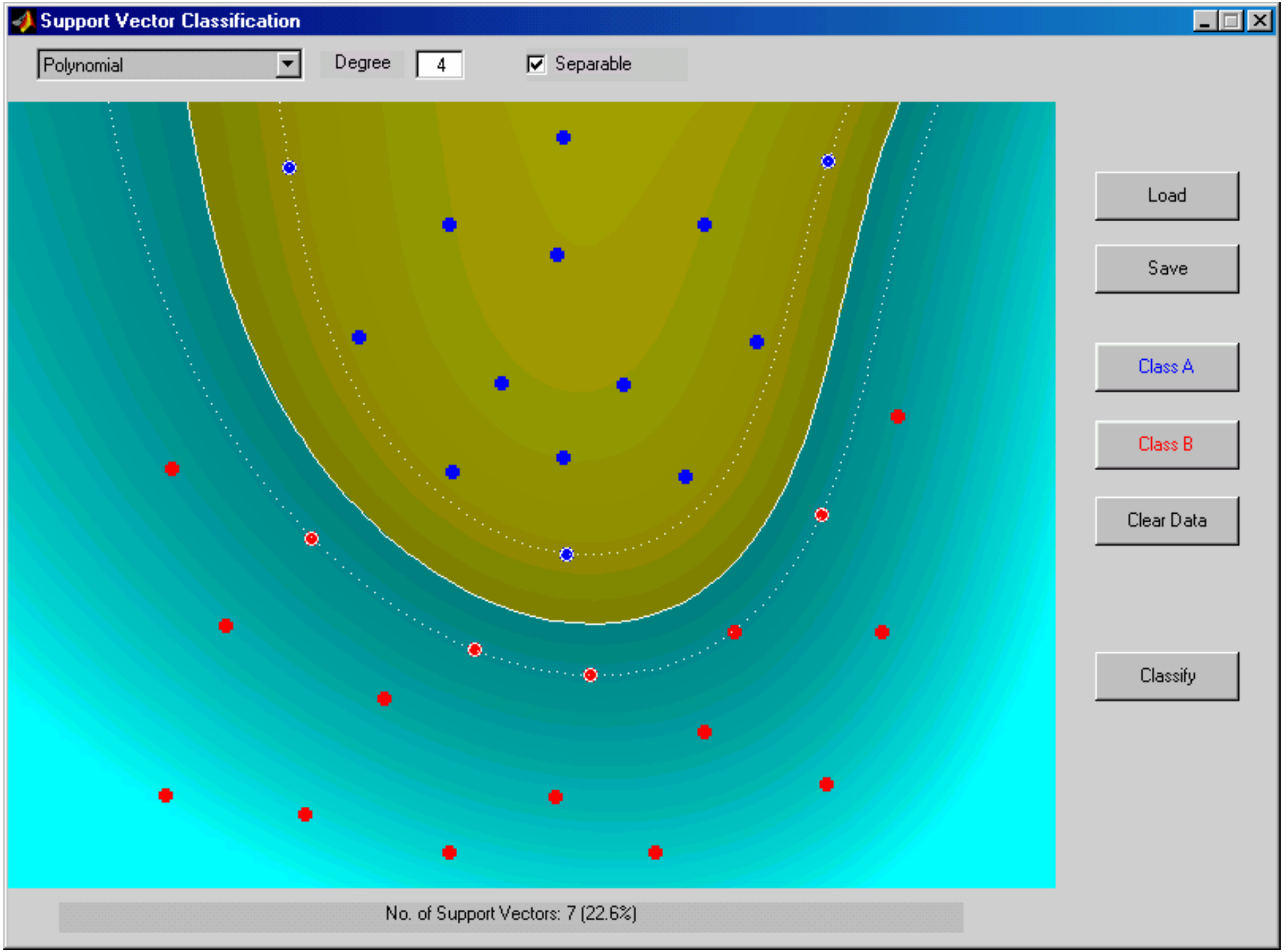
Linear separation of a quadratic problem; soft margin



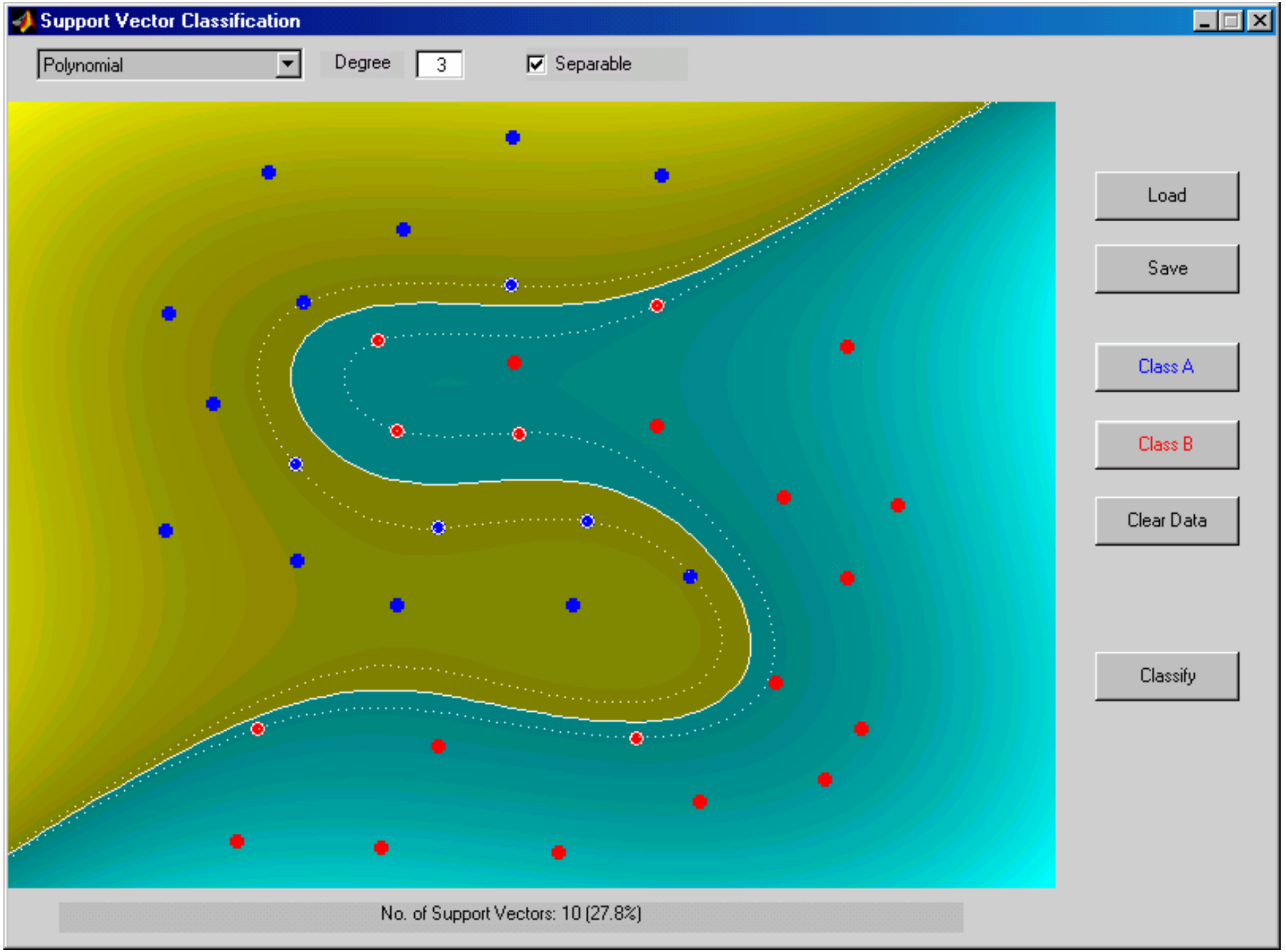
Polynomial separation of a quadratic problem; hard margin



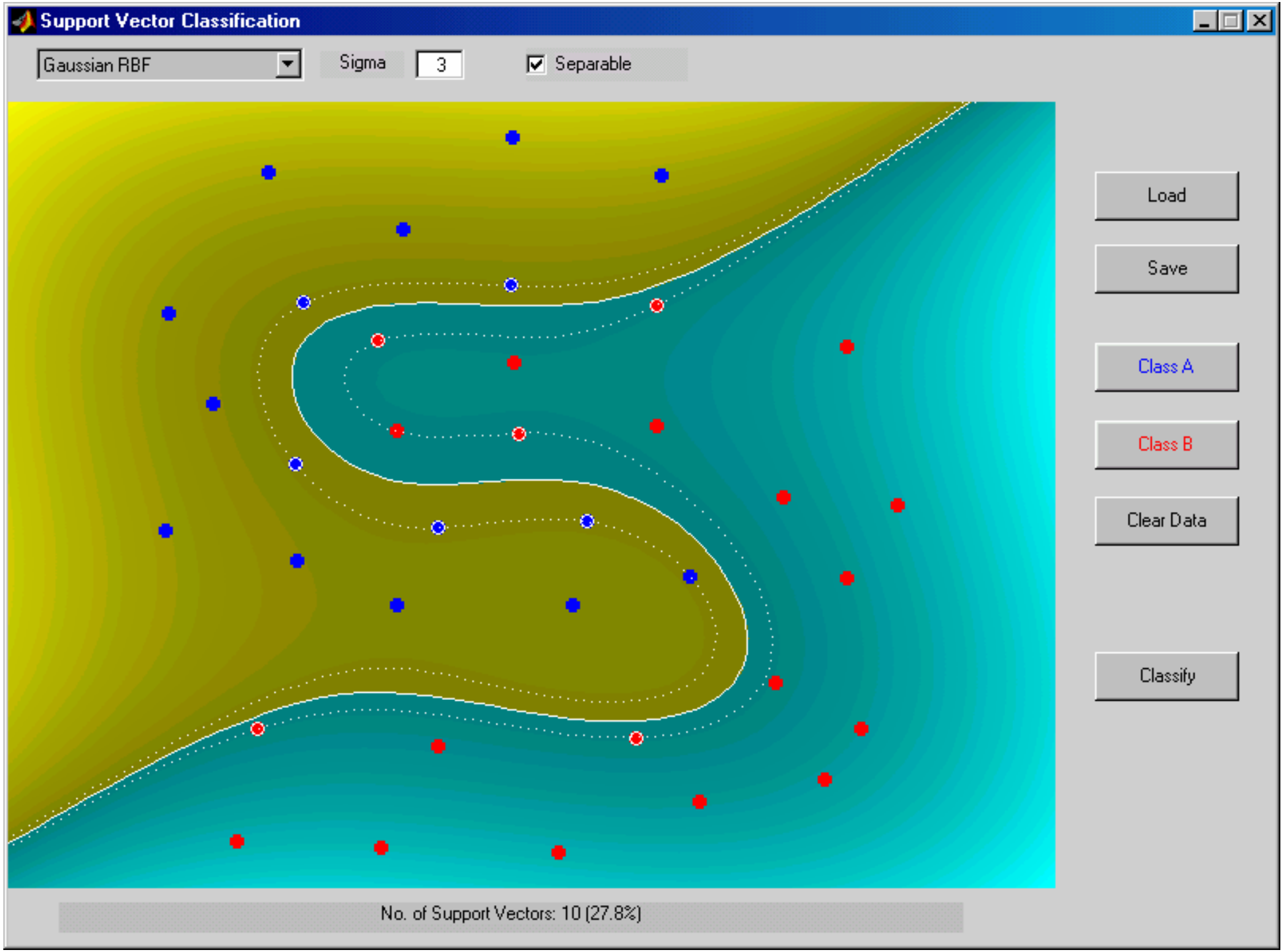
Polynomial separation of a quadratic problem; hard margin



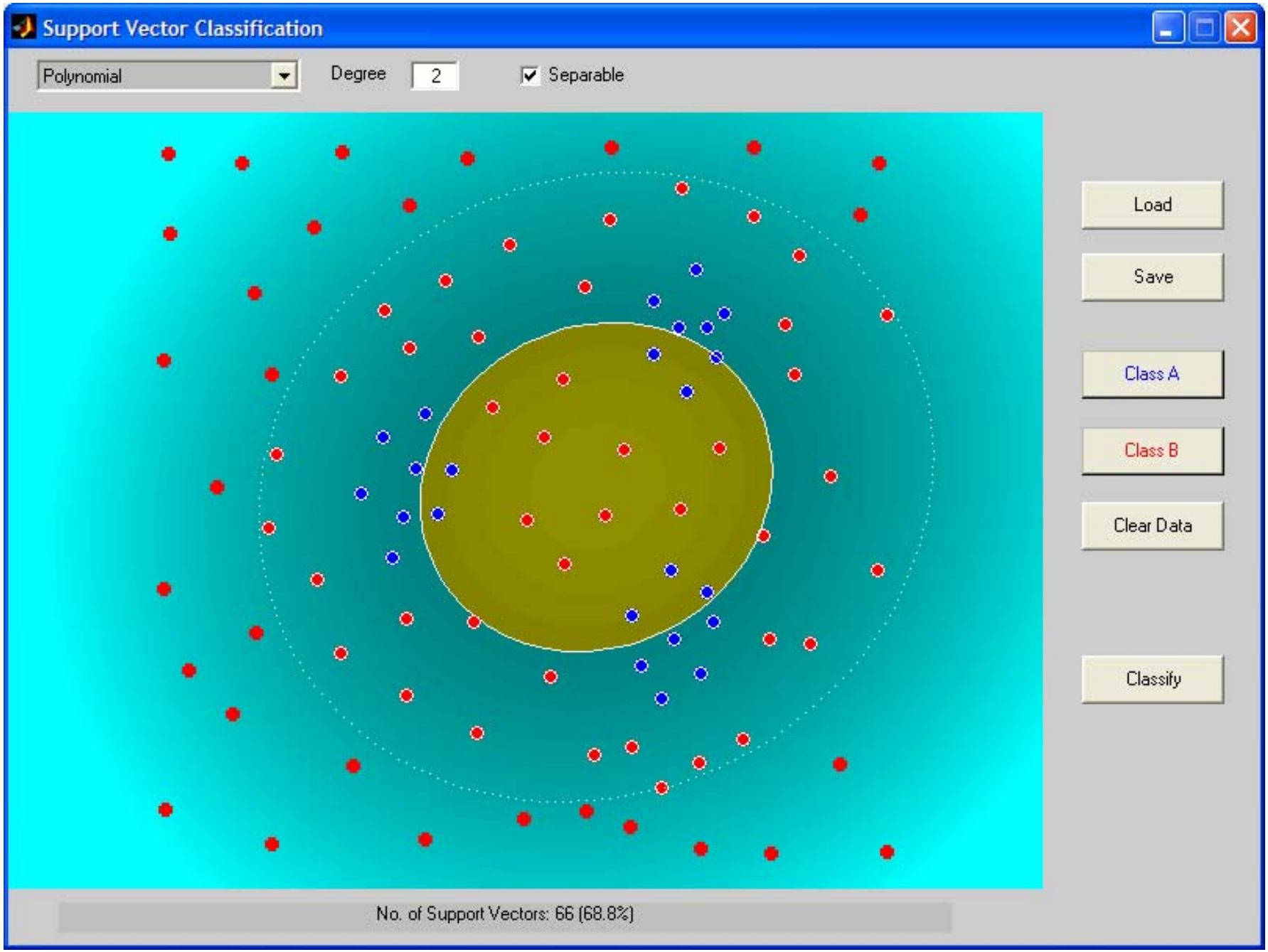
Non-linearly separable classes; hard margin



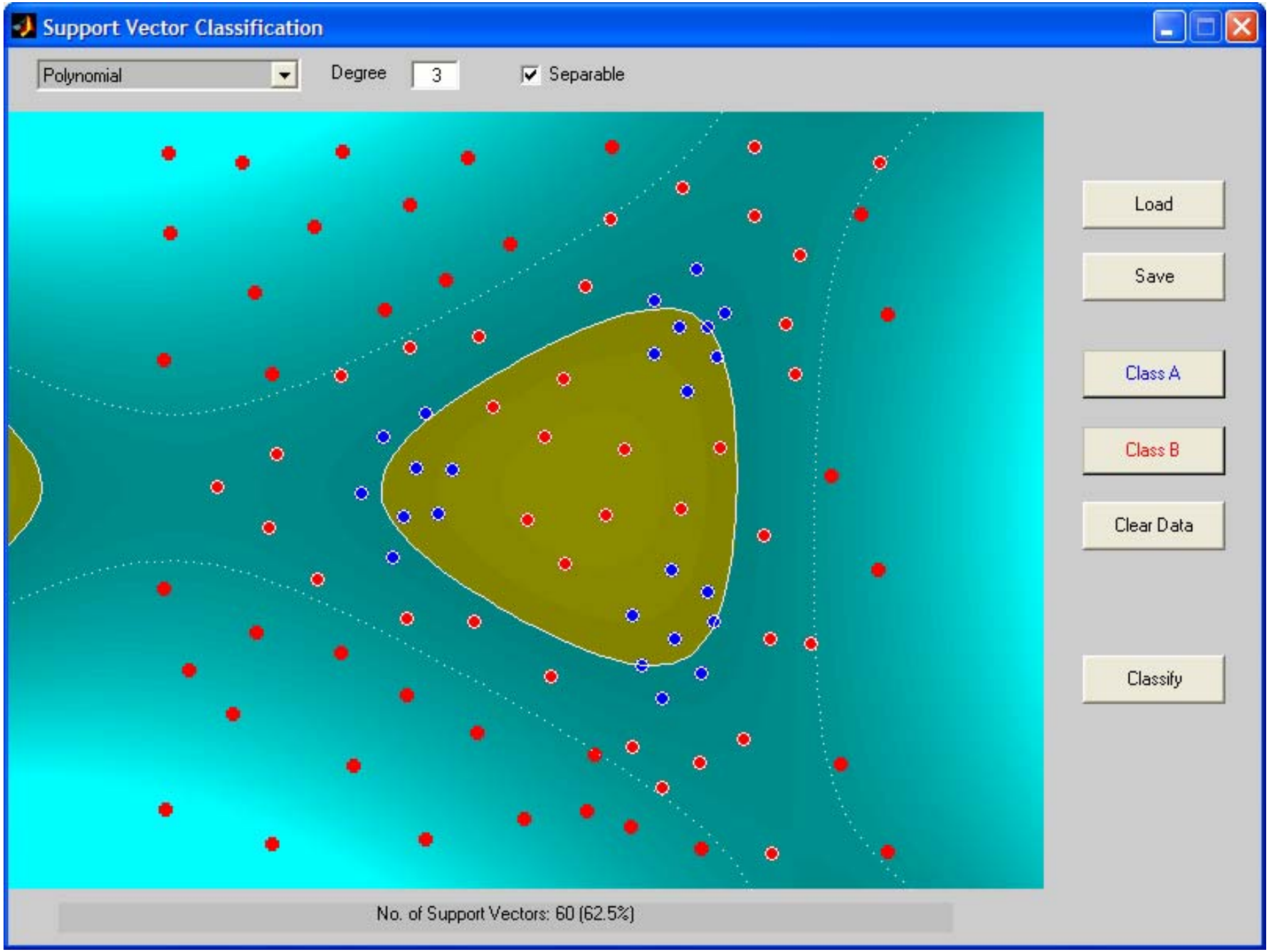
Non-linearly separable classes; hard margin



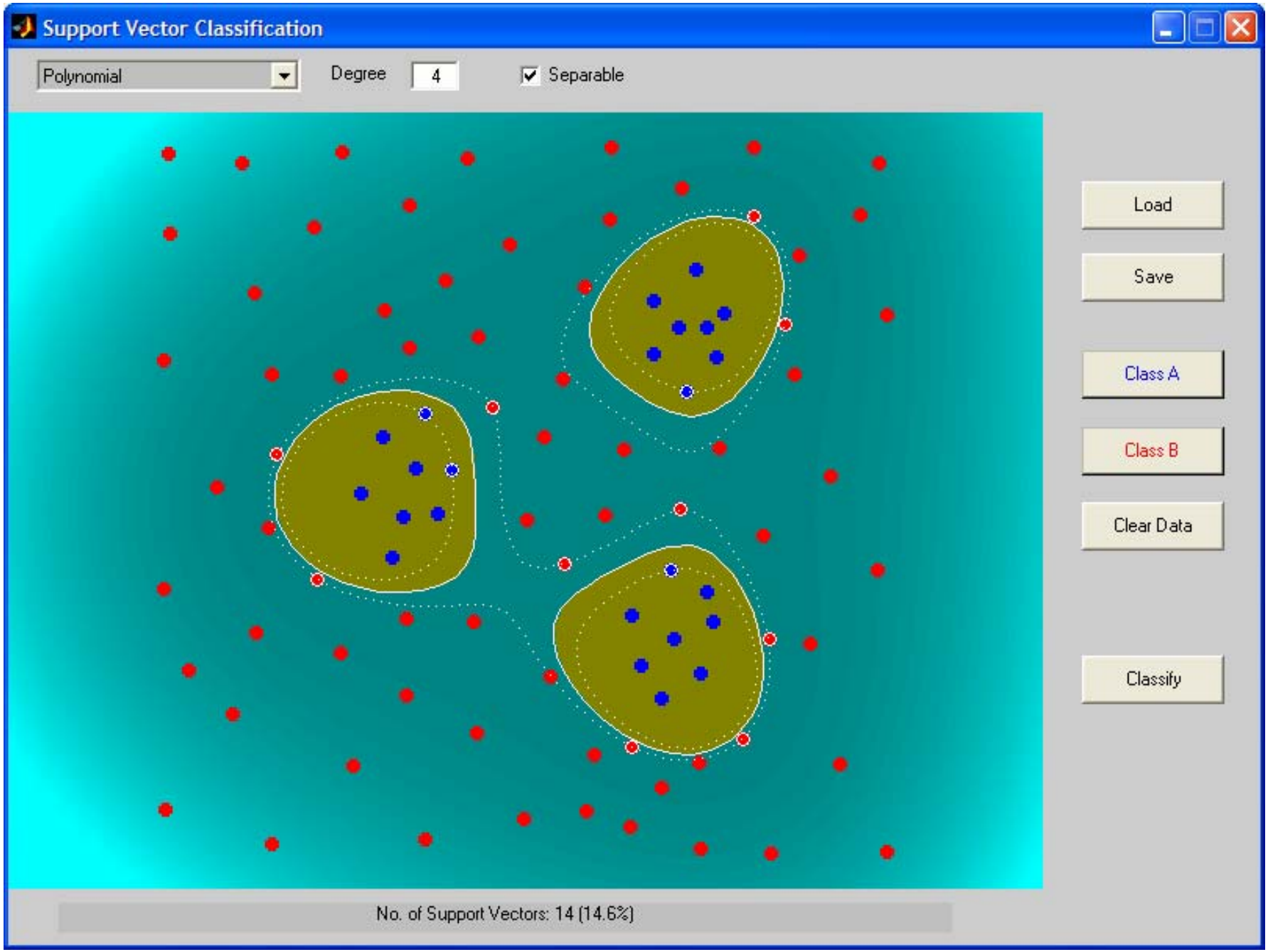
Example: „isles“, polynomal with $p = 2$



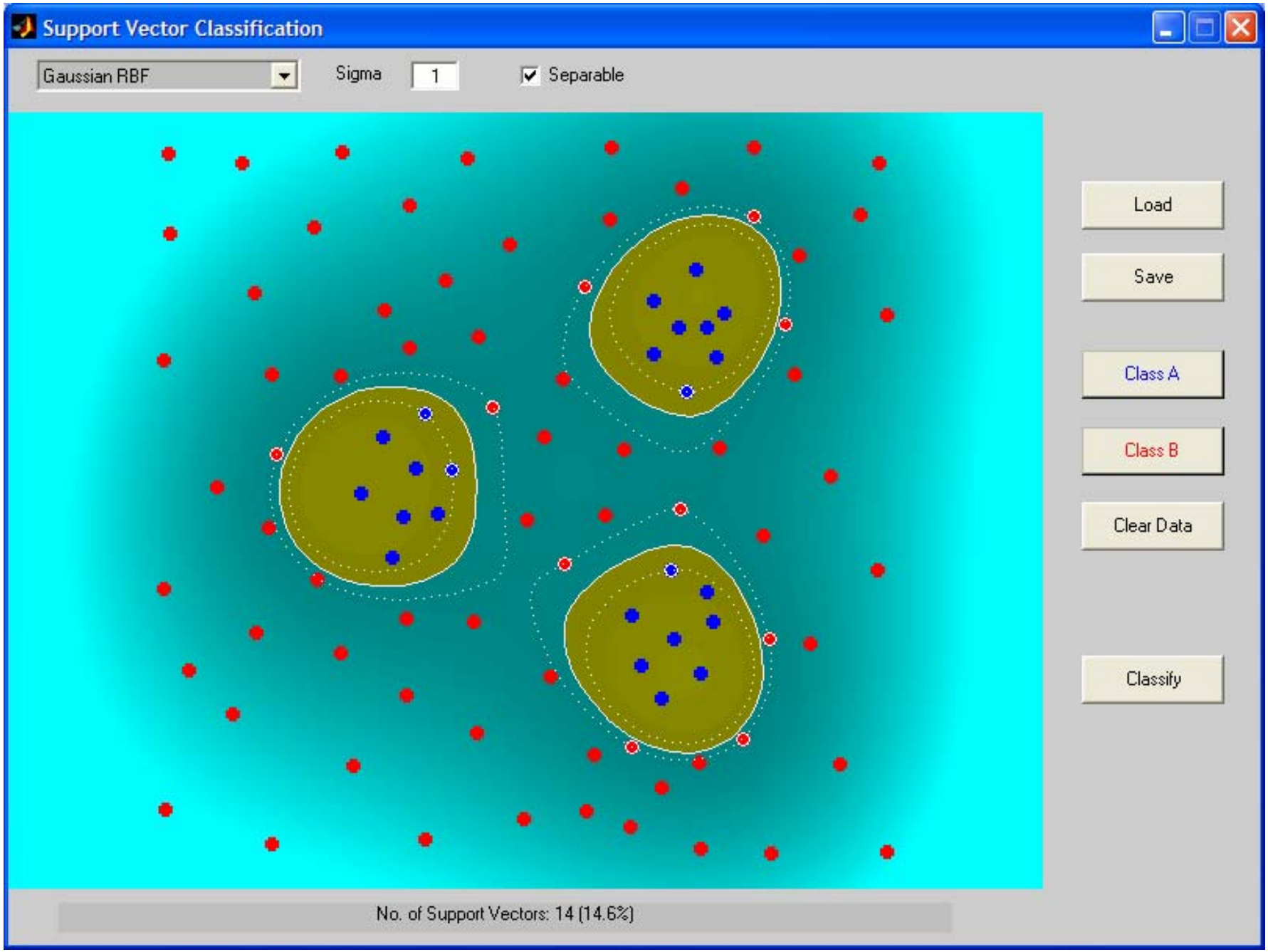
Example: „isles“, polynomal with $p = 3$

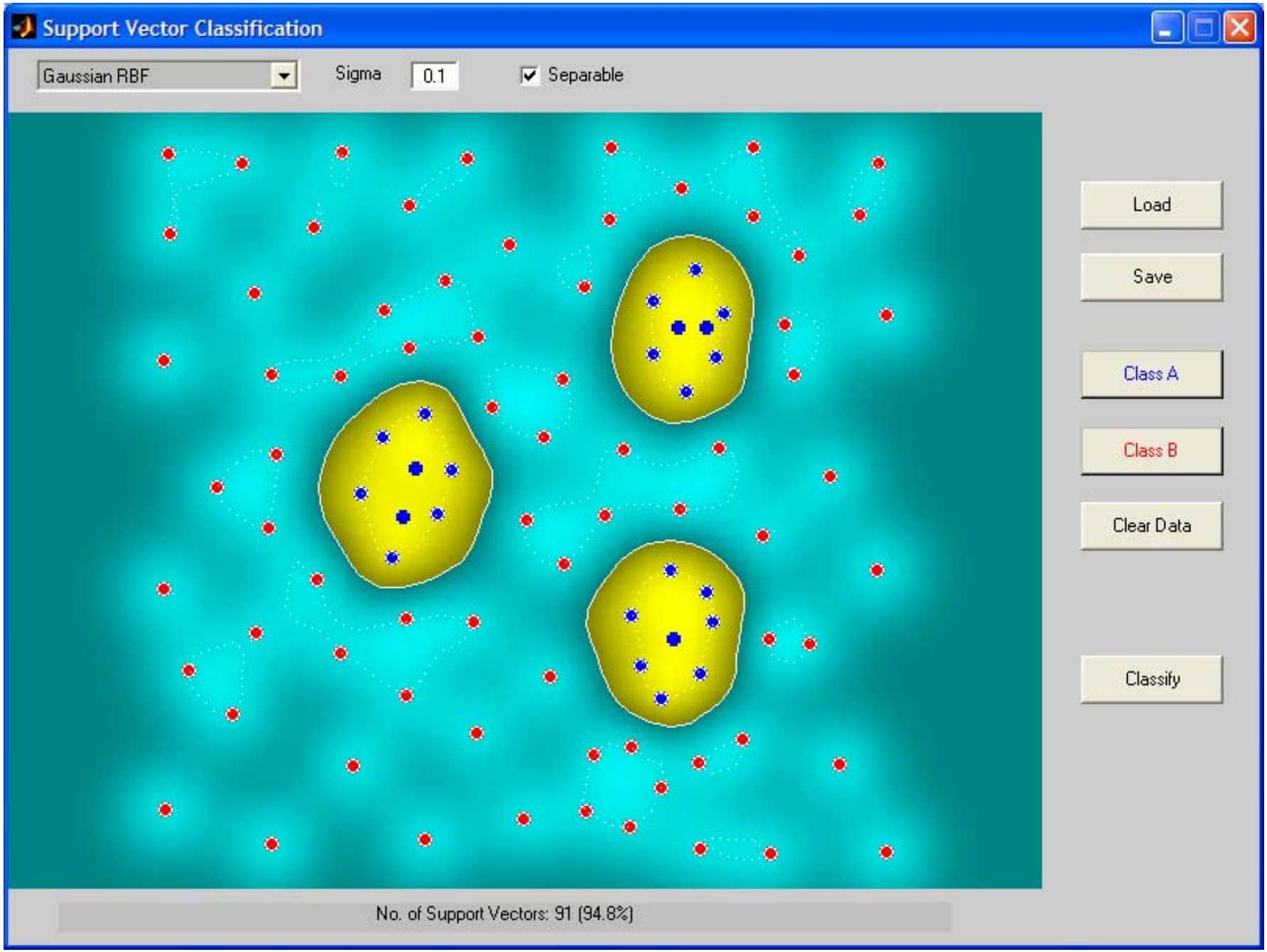


Example: „isles“, polynomal with $p = 4$



Example: „isles“, RBFs with σ=1




Support Vector Classification
Gaussian RBF
Sigma
0.1
Separable
Load
Save
Class A
Class B
Clear Data
Classify
No. of Support Vectors: 91 (94.8%)

Application example: sign recognition

Example: US Postal Service Digits [Schö95/96]

16x16 grey scale images

7291 training, 2007 testing samples

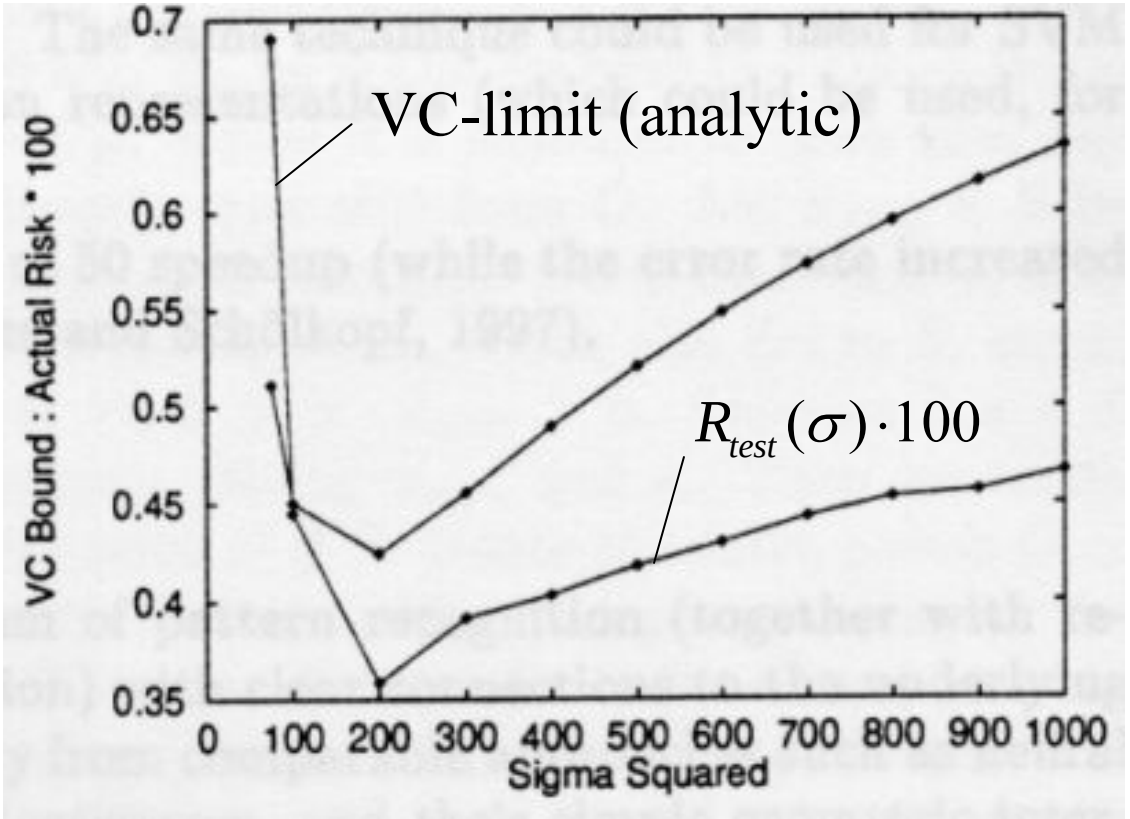
Results:

classifier	**error rate**
human	2.5%
2-layer NN	5.9%
5-layer NN	5.1%
SVM (polynomal degree 3)	4.0%
SVM + invariance	3.2%

SVM+invariance: quintuple original set of data by shifting the grey scale patterns in the 4 main directions (N,S,O,W) about one pixel.

SVM with RBF, VC-limit compared to actual test error rate

(only one parameter: σ)



- Unfortunately rough approximates, but qualitatively significant (minimum at the same location/place).
- **Alternative:** seek σ by minimizing test error through cross validation. Forming the expected value over all leave-one-out experiments results in an estimate of the real risk
- The cost of a leave-one-out experiment can be reduced to checking the SV, since only the SV has influence on the measured error rate.

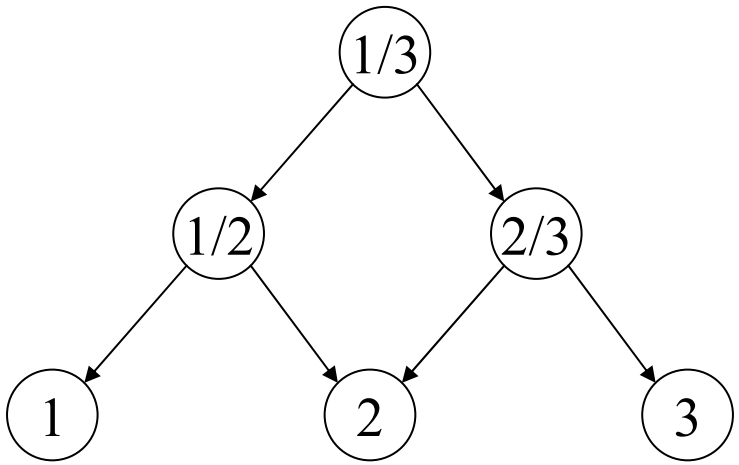
SVM: Properties and complexity

Advantages:

- According to the current understanding, the SVM provides very good results and finds (certain conditions given) a global minimum (NN normally finds only suboptimal solutions)
- Sparse-description of the solution over N_s support vectors
- Easily applicable (few parameters (C,σ), no a-priori-knowledge needed, no design); however the exact choice of (C,σ) is one of the biggest disadvantages in the design
- Geometrically visualizable functionality
- Theoretical statements about result: global optimum, generalization ability
- SRM possible, but difficult to manage
- also problems with big feature spaces can be solved (100.000 training and testing patterns)
- If problem is semidefinite: the dual problem has a lot of solutions, but all lead to the same hyperplane!

Disadvantages:

- VC-estimate very imprecise and not very utilisable in practice
- multi-class approach still matter of research
  - approach: one SVM per class, i.e. cost and space complexity increases with number of classes



Reduction of a multi-class problem to a series of binary decision problems

Disadvantages:

- No quantitative quality-statement about classification
- Slow, space-intensive learning:

  Runtime: $O(N_s^3)$ (inversion of Hessian-matrix/Newton-alg.)

  (however in practice less than quadratic)

  Space: $O(N_s^2)$

- Approach for improvement: divide into partial problems
- Slow classification with $O(MN_s)$, with $M = \dim(\mathcal{H})$ (i.e. in the case without kernel functions $M=N=\dim(\mathbf{x}_i)$), but for appropiately chosen kernel functions also applies: $M=N$.

Literature:

(1) C.J.C. Burges, „A tutorial on support vector machines for pattern recognition“, Knowledge Discovery and Data Mining, 2(2), 1998. (http://www.kernel-machines.org/tutorial.html)

(2) V. Vapnik, „Statistical Learning Theory“, Wiley, New York, 1998.

(3) N. Cristianini, J. Shawe-Taylor, „An introduction to support vector machines and other kernel-based learning methods“, Cambridge Univ. Press, Cambridge 2000.

(4) B. Schölkopf, A. J. Smola, „Learning with kernels“, MIT Press, Boston, 2002.

(5) S.R. Gunn, „Support Vector Machines for Classification and Regression“, Technical Report, Department of Electronics and Computer Science, University of Southampton, 1998.