

# Kapitel 10

## Die Support-Vektor-Maschine (SVM)

Ein statistischer Ansatz der  
Lerntheorie zum Entwurf eines  
optimalen Klassifikators

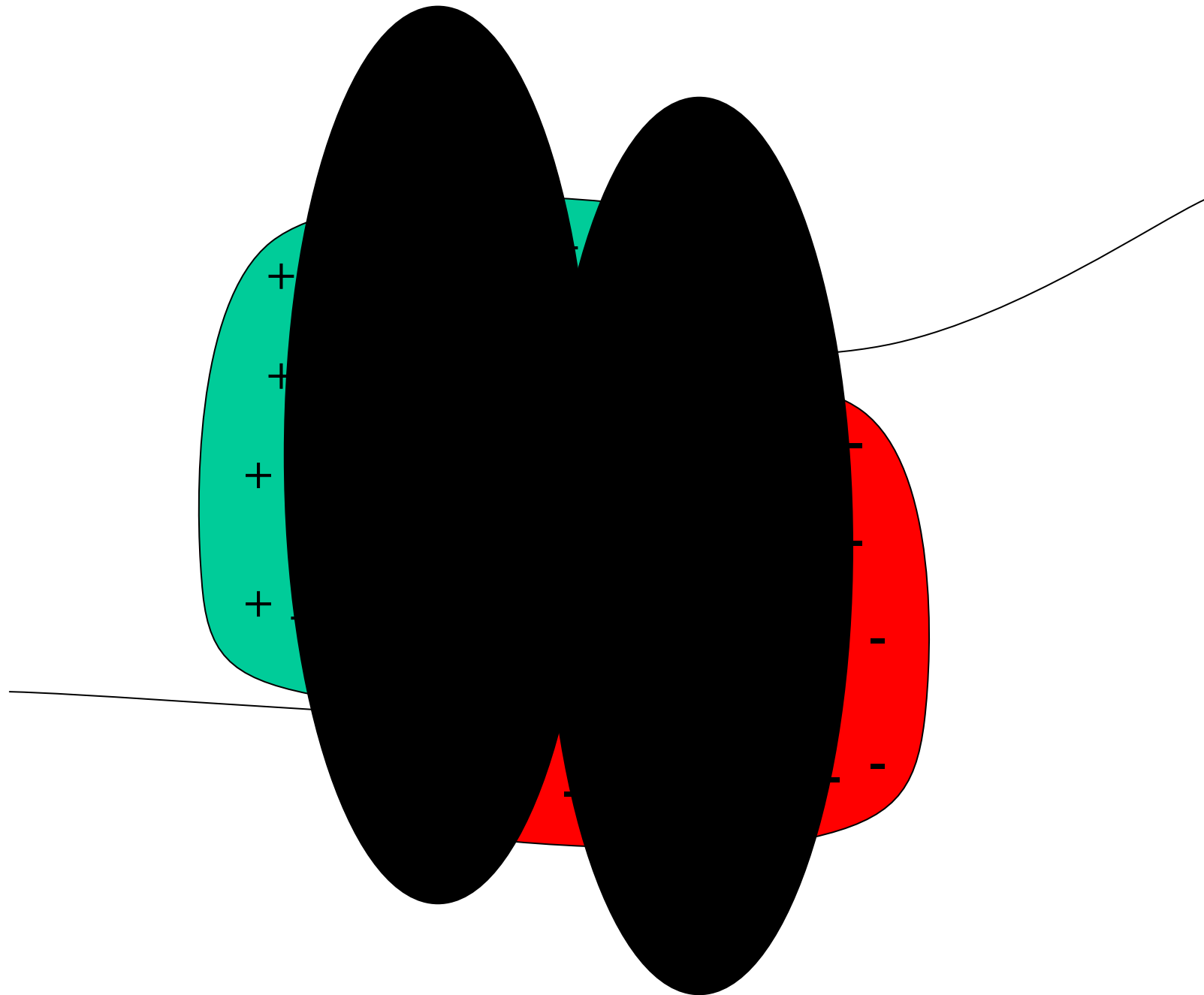
# Inhalt:

1. Problemstellung
2. VC-Dimension und Gesamtfehlerminimierung
3. Lineare SVM
  - Separierbare Klassen (Beispiel)
  - Nichtseparierbare Klassen (Beispiel)
4. Nichtlineare SVM
  - Trick mit Kernfunktion (Beispiel)
  - Mercer`s Theorem
5. Eigenschaften und Berechnungskomplexität
6. Beispiele und Demos mit MATLAB

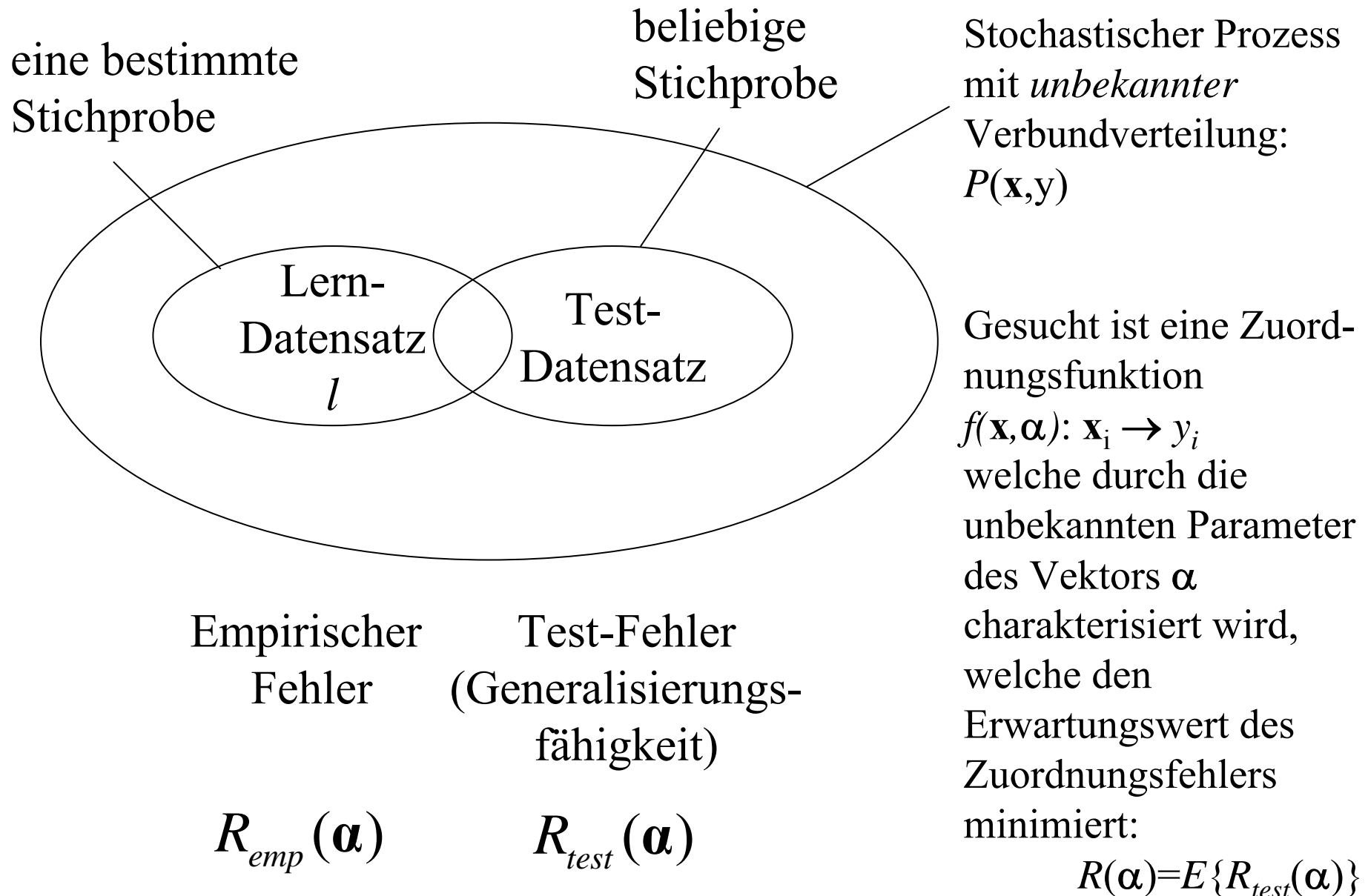
# Inhalt:

1. Problemstellung
2. VC-Dimension und Gesamtfehlerminimierung
3. Lineare SVM
  - Separierbare Klassen (Beispiel)
  - Nichtseparierbare Klassen (Beispiel)
4. Nichtlineare SVM
  - Trick mit Kernfunktion (Beispiel)
  - Mercer`s Theorem
5. Eigenschaften und Berechnungskomplexität
6. Vorstellung von Forschungsprojekten:
  - Claus Bahlmann: Handschrifterkennung
  - Olaf Ronneberger: Autom. Erkennung von Blütenpollen
  - Bernard Haasdonk: Tangentendistanz und SVM

# Durch Normalverteilungen schlecht darstellbare Klassen



# Mustererkennungsexperiment



# Lerntheoretischer Ansatz

## Überwachtes Lernen:

- Gegeben:  $l$  Beobachtungen (Lernstichprobe) aus dem Mustererkennungsexperiment mit der Verbundverteilung  $P(\mathbf{x}, y)$  mit den dazugehörigen Klassenzuordnungen (Labels) (Zunächst Beschränkung auf Zweiklassenproblem); ansonsten existiert *keinerlei Vorwissen*:

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

## Gesucht:

- deterministische Zuordnungsfunktion,  $f(\mathbf{x}, \boldsymbol{\alpha}): \mathbf{x} \rightarrow y$  aufbauend auf eine Lernstichprobe, welche den *Erwartungswert des Zuordnungsfehlers beim Testdatensatz (expected risk)* minimiert:

$$\begin{aligned} R(\boldsymbol{\alpha}) &= E \{ R_{test}(\boldsymbol{\alpha}) \} = E \left\{ \frac{1}{2} |y - f(\mathbf{x}, \boldsymbol{\alpha})| \right\} = \int \frac{1}{2} |y - f(\mathbf{x}, \boldsymbol{\alpha})| dP(\mathbf{x}, y) \\ &= \int \frac{1}{2} |y - f(\mathbf{x}, \boldsymbol{\alpha})| p(\mathbf{x}, y) d\mathbf{x} dy \quad \text{falls Verbundverteilungsdichte bekannt} \end{aligned}$$

**Problem:** dieser Ausdruck kann jedoch nicht ausgewertet werden, da  $P(\mathbf{x}, y)$  nicht zur Verfügung steht und ist somit wenig hilfreich!

# Zentrales Problem der statistischen Lerntheorie: Wann führt ein niedriger Trainingsfehler zu einem niedrigen echten Fehler?

- Das *empirische Risiko*, nämlich die Fehlerrate für einen gegebenen Trainingsdatensatz lässt sich leicht berechnen gemäß:

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

- Der Ansatz mit einem Neuronalen Netz und dem Backpropagation Learning z.Bsp. begnügt sich mit einer **Empirischen Risiko Minimierung (ERM)**
- Hier nun die Frage: Wie nah ist man nach  $l$  Trainingsbeispielen am *echten* Fehler? Und: Wie gut kann aus dem empirischen Risiko das echte Risiko abgeschätzt werden (**Structural Risk Minimization (SRM)** anstatt Empirical Risk Minimization (ERM)) ? Dies beinhaltet die Generalisierungsfähigkeit!
- Eine Antwort darauf versucht die Lerntheorie von Vapnik-Chervonenkis zu geben!

# Eine Obergrenze für die Generalisierungsfähigkeit des Lernens mit der VC-Theorie

(Vapnik/Chervonenkis)

Mit der Wahrscheinlichkeit  $(1-\eta)$  gilt die folgende obere Abschätzung für den tatsächlichen Fehler (d.h. z.Bsp. mit  $\eta = 0,05$  und damit mit einer Wahrscheinlichkeit von 95%):

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

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

- $l$  Anzahl der Trainingsbeispiele
- $h$  VC-Dimension des verwendeten Hypothesenraums

Bemerkenswert ist, dass dieser Ausdruck unabhängig ist von der zugrundeliegenden Verbundverteilung  $P(\mathbf{x}, y)$  ! (vorausgesetzt, die Trainings- und Testdatensätze werden statistisch unabhängig gezogen). D.h. falls man die Auswahl zwischen verschiedenen Lernmaschinen hat (einhergehend mit speziellen Familien von Abbildungsfunktionen  $f(\mathbf{x}, \alpha)$ ), entscheidet man sich für die Maschine, welche bei gegebenem empir. Fehler den geringsten Wert für die Konfidenz  $\Phi$  liefert.

Die Unsicherheit  $\Phi$  wächst mit  $h$  und erniedrigt sich bei wachsender Lernstichprobe  $l$

Entwurfsziel:  
VC-Konfidenz möglichst klein und damit die Abschätzung möglichst scharf halten!

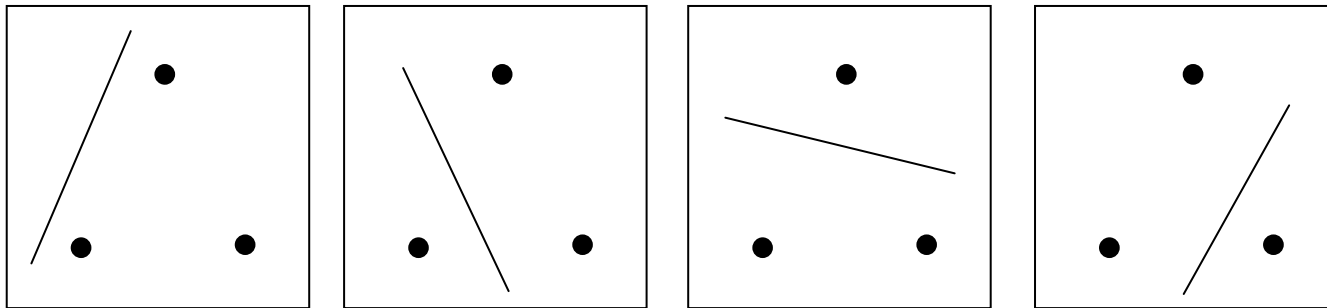


# Die VC-Dimension ist eine Eigenschaft für gegebene Funktionenklassen $\{f(\mathbf{x}, \alpha)\}$

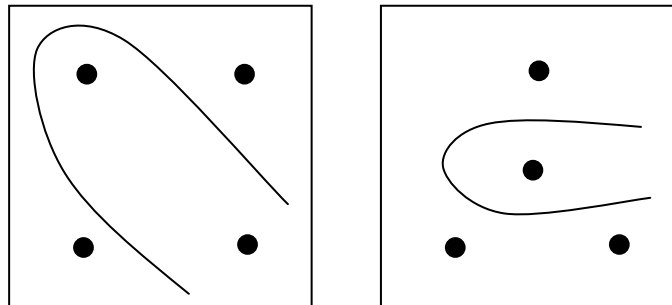
- Eine gegebene Menge von  $l$  Punkten kann für den Zweiklassen-Fall in  $2^l$  mögliche Klassen aufgeteilt werden. Die VC-Dimension einer Menge von Funktionen  $\{f(\mathbf{x}, \alpha)\}$ , ist definiert als die maximale Anzahl von Trainingspunkten, welche durch diese Klasse von Funktionen in allen möglichen Zuordnungen separiert werden können.
- Liefert Maße für „Kapazität“ oder Variabilität von Funktionenklassen
  - Funktionenklassen z.B.
    - Menge der linearen Trennfunktionen
    - Menge künstlichen NN (MLP)
    - Menge der Polynomfunktionen
    - Menge der radialen Basisfunktionen
- Die VC-Konfidenz wächst monoton mit  $h$ : Demnach würde man bei einer Auswahl von Lernmaschinen, deren empirisches Risiko 0 ist, sich für diejenige entscheiden, deren assoziierte Menge von Abbildungsfunktionen minimale VC-Dimension besitzt (Modell so einfach wie möglich!).

# VC-Dimension $h$ von Hyperebenen, hier: $\mathbb{R}^2$

- Drei Punkte in  $\mathbb{R}^2$  lassen sich mit *Hyperebenen* (Geraden) in allen Konstellationen trennen



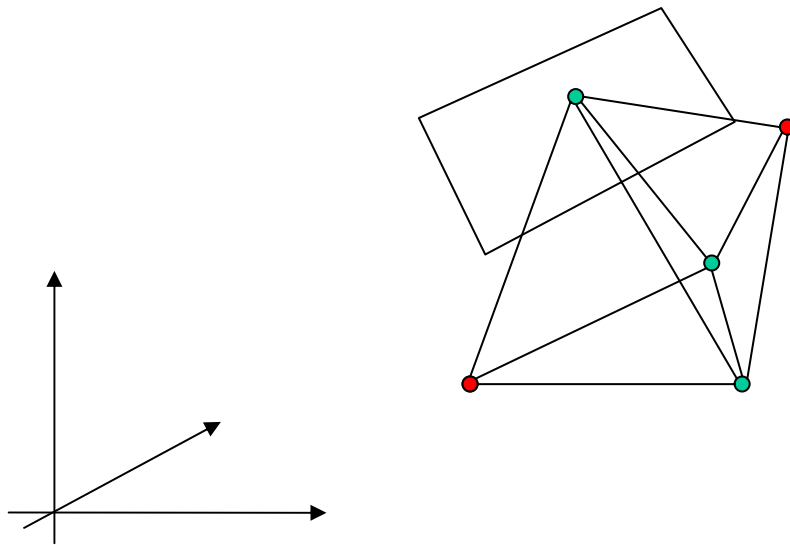
- Vier Punkte in  $\mathbb{R}^2$  lassen sich nicht mehr mit *Hyperebenen* in allen Konstellationen trennen; jedoch z.B. gelingt es mit *Polynomen*!



- $\Rightarrow$  Hyperebenen in  $\mathbb{R}^2$  :  $h = 3$
- die VC-Dimension bei Polynomen wächst mit ihrem Grad!

# VC-Dimension $h$ von Hyperebenen in $\mathbb{R}^3$

- Vier Punkte in  $\mathbb{R}^3$  lassen sich mit *Hyperebenen* (Ebenen) in allen Konstellationen trennen
- Fünf Punkte in  $\mathbb{R}^3$  lassen sich nicht mehr mit *Hyperebenen* in allen Konstellationen trennen (z.B. die roten Punkte in einer Klasse); jedoch z.B. gelingt es mit *Polynomen*!



- $\Rightarrow$  Hyperebenen in  $\mathbb{R}^3$  :  $h = 4$
- Allgemein: Hyperebenen in  $\mathbb{R}^N$  :  $h = N+1$

# VC-Dimension von Funktionenklassen

– Aussagen über VC-Dimension von Funktionenklassen möglich!

- VC-Dimension der Menge von *Hyperebenen* der Breite  $\delta$  in  $\mathbb{R}^N$

$$h \leq \min\left(\frac{R^2}{\delta^2}, N\right) + 1 \quad (\text{siehe dazu auch Folie 19})$$

$R$ : Radius, innerhalb dessen alle Datenpunkte liegen

$\delta$ : Margin zwischen den Hyperebenen

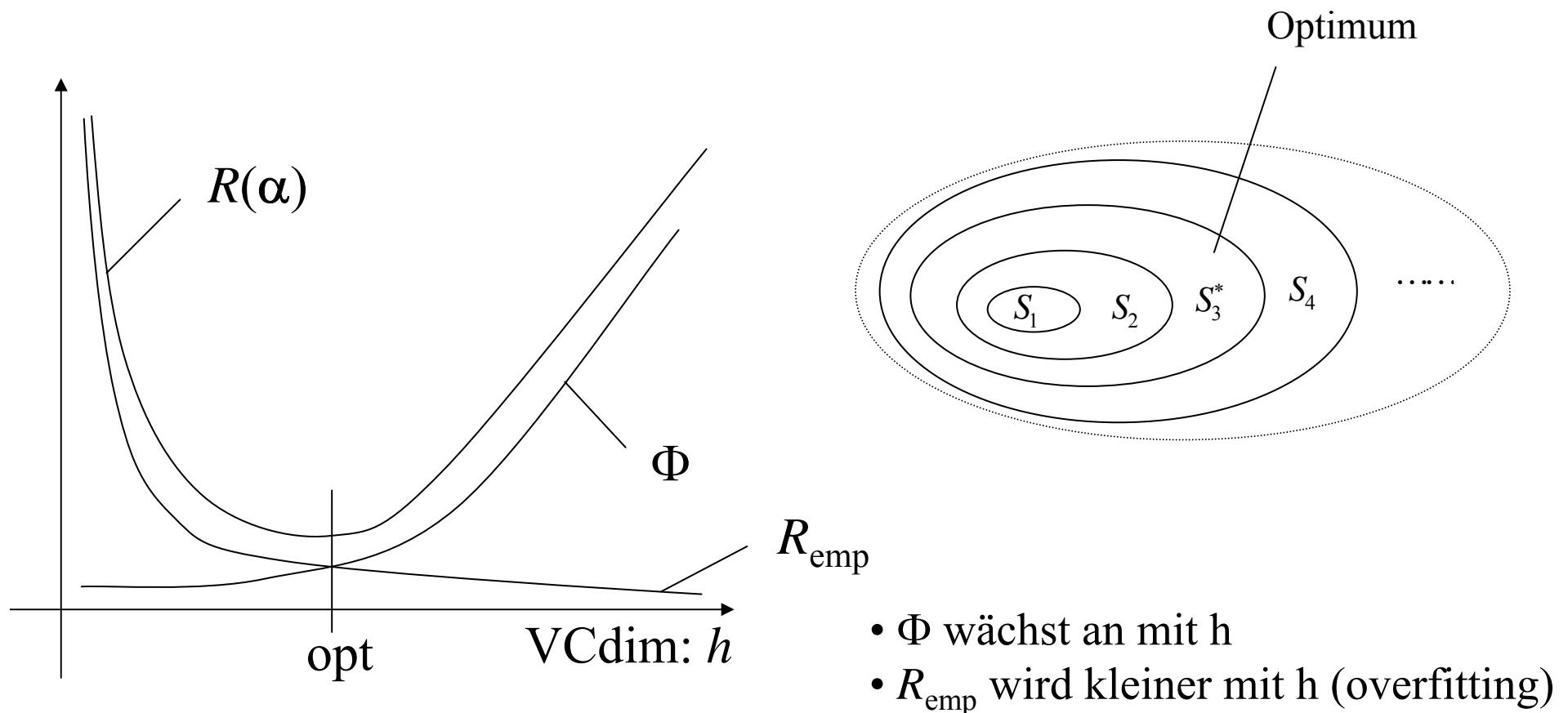
- VC-Dimension der Menge von *Polynomkernen* vom Grad  $p$ :  $K(x_i, x_j) = (x_i \cdot x_j)^p$

$$h = \binom{N + p - 1}{p} + 1$$

– Hiermit “Structural Risk Minimization“ möglich

# Structural Risk Minimization (SRM)

- SRM ist Minimieren der Abschätzung für  $R(\alpha)$  über anwachsende Funktionenklassen  $S_i$ . Diese bilden Hypothesenräume mit anwachsender VC-Dimension  $VCdim=h_i$  ( $h_1 < h_2 < h_3 < \dots$ ).
- Kompromiss zwischen empirischem Risiko und Generalisierungsfähigkeit

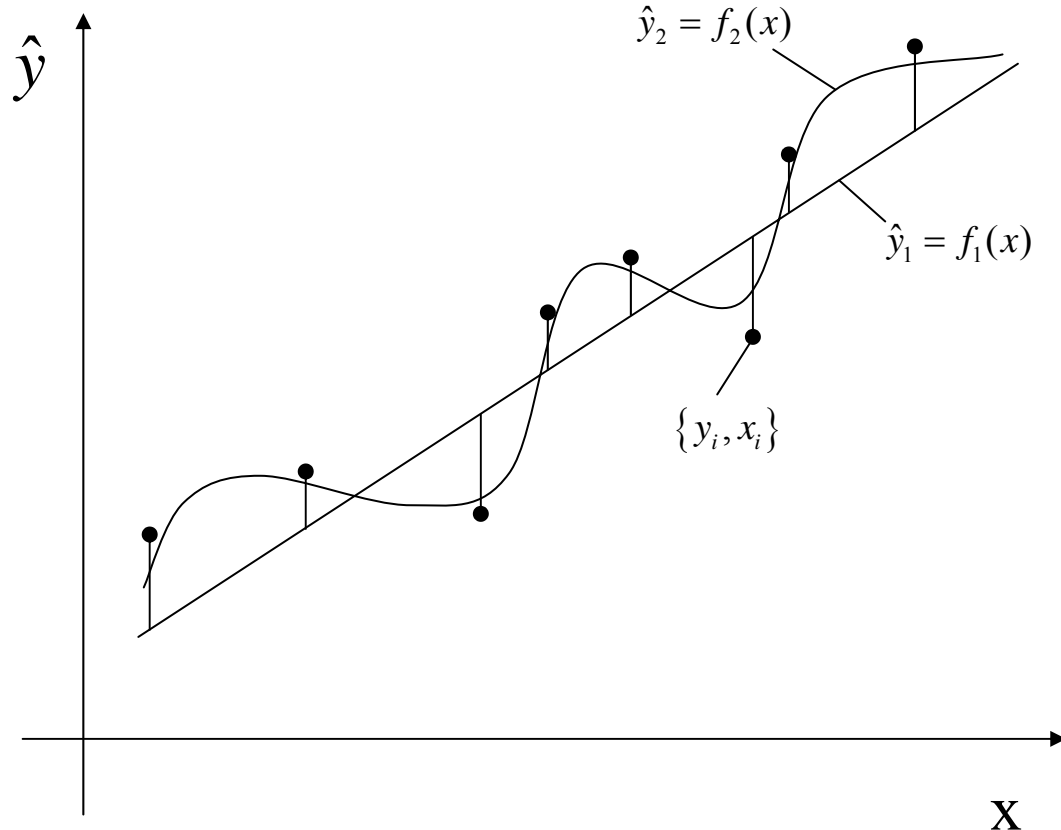


# Entwurfshinweise

Mehrere Ausdrücke für das gleiche Phänomen:

- Bias/Varianz-Kompromiss
- Generalsierungs/Overfitting-Kompromiss
- Empirischer Fehler/VC-Dimensions-(Kapazitätskontrolle)-Kompromiss

# Bias-Varianz-Tradeoff bei der Funktionsapproximation



$$f_1 : \begin{cases} \text{Bias} & \frac{1}{n} \sum_{i=1}^n (y_i - f_1(x_i)) \quad \text{klein} \\ \text{Varianz} & \frac{1}{n} \sum_{i=1}^n (y_i - f_1(x_i))^2 \quad \text{groß} \end{cases}$$

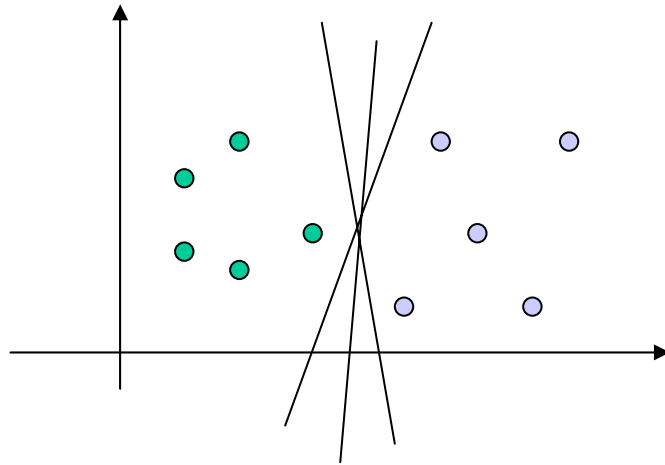
großer empirischer Fehler, aber  
gute Generalisierung (einfaches Modell)!

$$f_2 : \begin{cases} \text{Bias} & \frac{1}{n} \sum_{i=1}^n (y_i - f_2(x_i)) \quad \text{groß} \\ \text{Varianz} & \frac{1}{n} \sum_{i=1}^n (y_i - f_2(x_i))^2 \quad \text{klein} \end{cases}$$

kleiner empirischer Fehler, aber  
schlechte Generalisierung (overfitting)!

# Zunächst Überlegungen für linear separierbare Klassen

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



- Allgemeine Hyperebene  $\langle \mathbf{w}, \mathbf{x} \rangle + b = 0$
- Klassifikation via  $f(x) = \text{sgn}(\langle \mathbf{w}, \mathbf{x} \rangle + b)$
- z.B. Rosenblatt's Perceptron (1956)
  - Iteratives Lernen, Korrektur nach jeder Fehlklassifikation
  - >keine Eindeutigkeit der Lösung



# Optimale Hyperebene: Maximierung des Randes

- Geht man von einem linear separierbarem Zweiklassenproblem aus, so erreichen alle Geraden, welche beide Klassen trennen einen empirischen Fehler Null
- Die Konfidenz wird minimiert durch ein Polynom minimaler VC-Dimension, nämlich eine Hyperebene
- Die VC-Dimension kann weiter abgesenkt werden durch „breite Hyperebenen“ (large margin hyperplanes)

• Trennende Hyperebene mit maximalem Rand

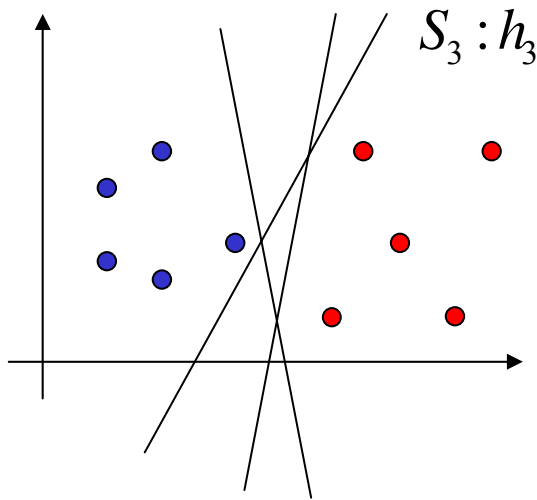


• Trennende Hyperebene mit minimaler VC-Dimension

- Dieses Ergebnis ist plausibel. Bei konstanter Intraklassenstreuung, wächst die Klassifikationssicherheit mit wachsendem Interklassenabstand.
- oder: bei konstant gehaltenem Interklassenabstand muss die Intraklassenstreuung maximal zulässig werden.

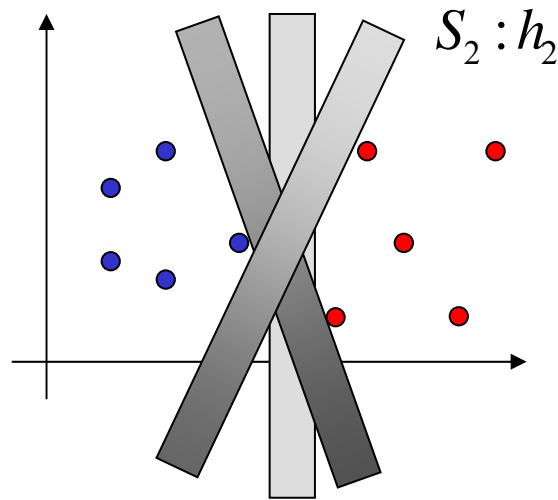
# “Large-margin“-Klassifikator

– Hyperebene mit größtem Rand [Vap63]



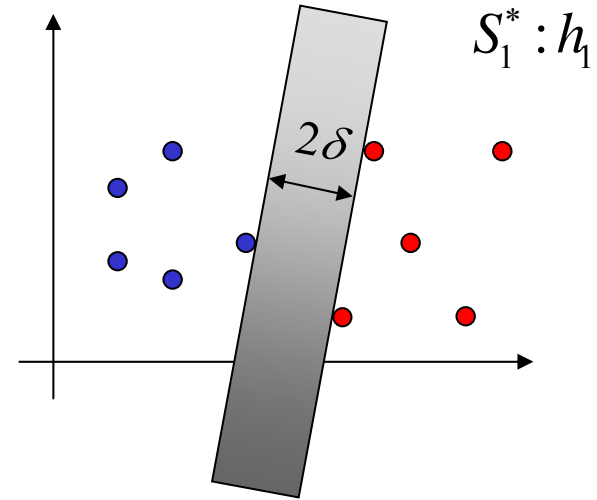
$S_3 : h_3$

hohe VC-Dimension  
in  $\mathbb{R}^N : N + 1$



$S_2 : h_2$

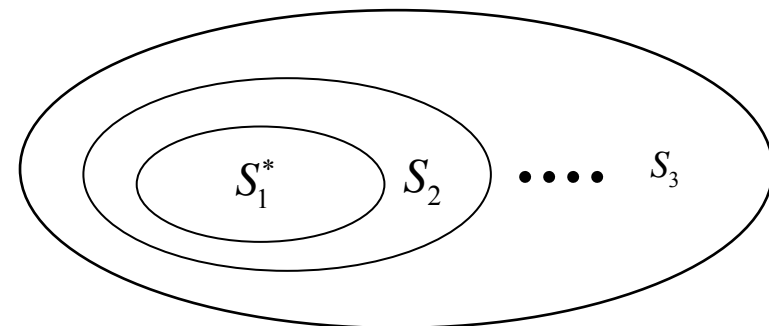
mittlere VC-Dimension  
Variabilität wird kleiner!



$S_1^* : h_1$

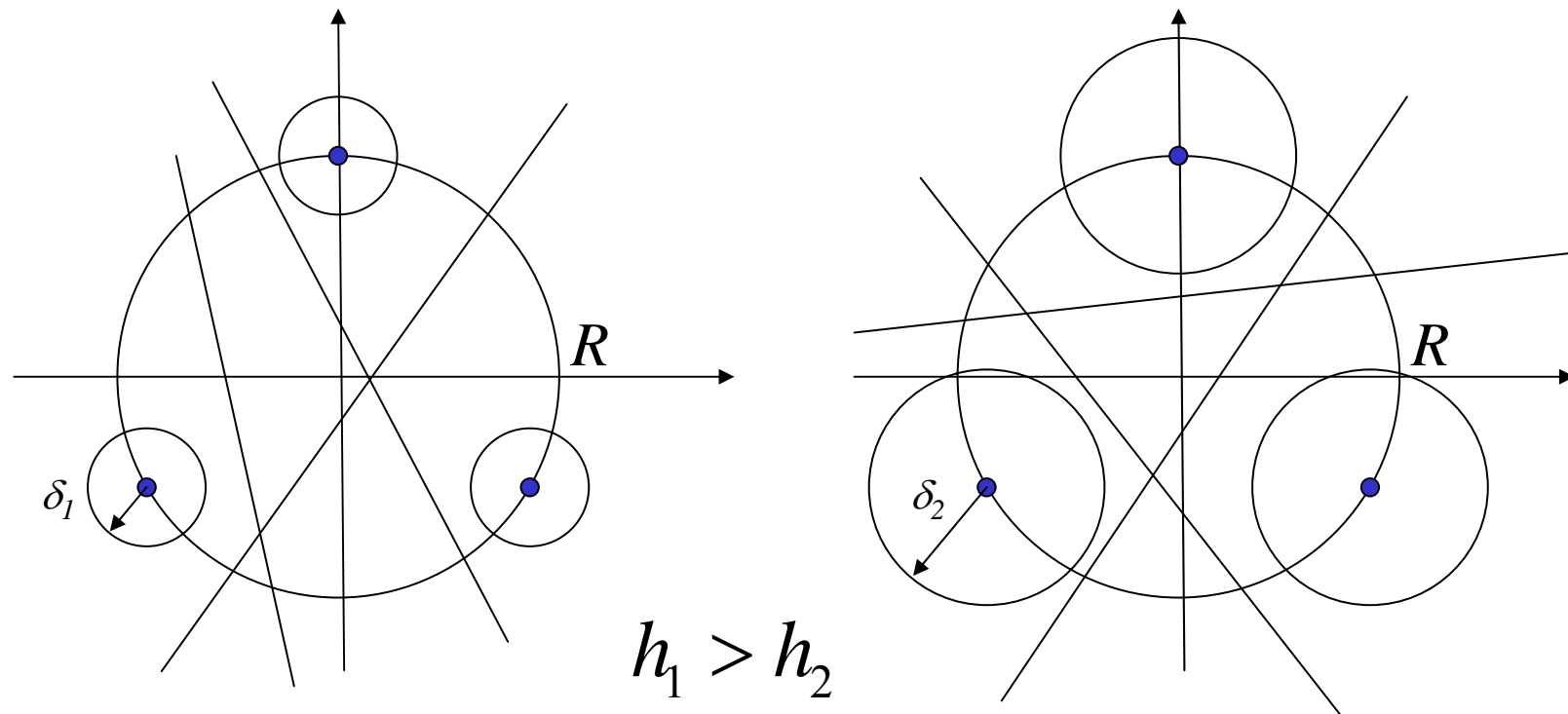
kleinste VC-Dimension  
mit maximaler Breite  
Variabilität gleich Null

- Anschaulich sinnvoll
- theoretisch begründet
- Lösung abhängig von wenigen Daten:  
=>“Support-Vektoren“



$$h_1 < h_2 < \dots < h_3$$

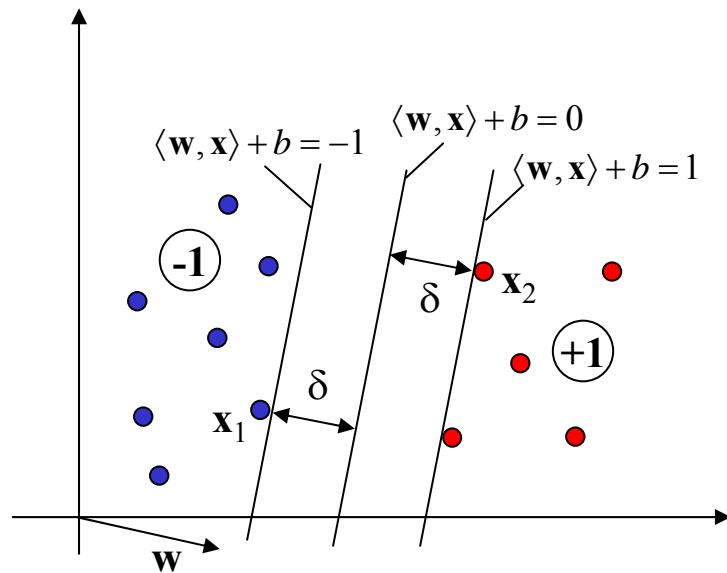
# VC-Dimension von „breiten“ Hyperebenen



Die VC-Dimension  $h$  von Hyperebenen mit einem Mindestabstand  $\delta$  von den zu trennenden Punkten ist begrenzt durch:

$$h \leq \frac{R^2}{\delta^2} + 1 \quad \Rightarrow \text{grosser Margin } \delta, \text{ kleine VCdim } h$$

# Formalisierung: gesucht ist Trennebene mit maximalem Margin $\delta$



Normalenvektor

Abstand eines Punktes von der Trennebene:

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

Der Abstand zwischen den kanonischen Hyperebenen ergibt sich durch Projektion von  $\mathbf{x}_1 - \mathbf{x}_2$  auf den Normalenvektor  $\mathbf{w}/\|\mathbf{w}\|$  unter Beachtung der Nebenbedingung:

Die Daten werden korrekt klassifiziert, falls:

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

**NB:** Einführung von *kanonischen Hyperebenen* durch die der Trennebene nächstgelegenen Punkte beider Klassen (Supportvektoren  $\mathbf{x}_1, \mathbf{x}_2$ ). Der Ausdruck ist invariant gegenüber einer positiven Reskalierung mit dem Faktor  $a$ :

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

$a$  wird so gewählt, dass gilt:

$$\langle \mathbf{w}, \mathbf{x}_1 \rangle + b = -1 \quad \text{Rand für blaue Klasse}$$

$$\langle \mathbf{w}, \mathbf{x}_2 \rangle + b = +1 \quad \text{Rand für rote Klasse}$$

$$\{\mathbf{x}_1, \mathbf{x}_2\} \quad \text{Supportvektoren}$$

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

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

Die Maximierung von  $\delta$  ist gleichwertig mit einer Minimierung von  $\|\mathbf{w}\|^2 \Rightarrow$

# Quadratisches Optimierungsproblem für $l$ Trainingsstichproben $(\mathbf{x}_i, y_i)$

Primales OP: 
$$\begin{aligned} &\text{minimiere } J(\mathbf{w}, b) = \frac{1}{2} \|\mathbf{w}\|^2 = \frac{1}{2} \langle \mathbf{w}, \mathbf{w} \rangle \\ &\text{unter der N.B.: } \forall i [y_i (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1] \end{aligned} = 1 \text{ für SVn}$$

Einführung einer Lagrangefunktion mit den Lagrange-Faktoren  $\alpha_i$ :

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

mit:  $\alpha_i \geq 0$

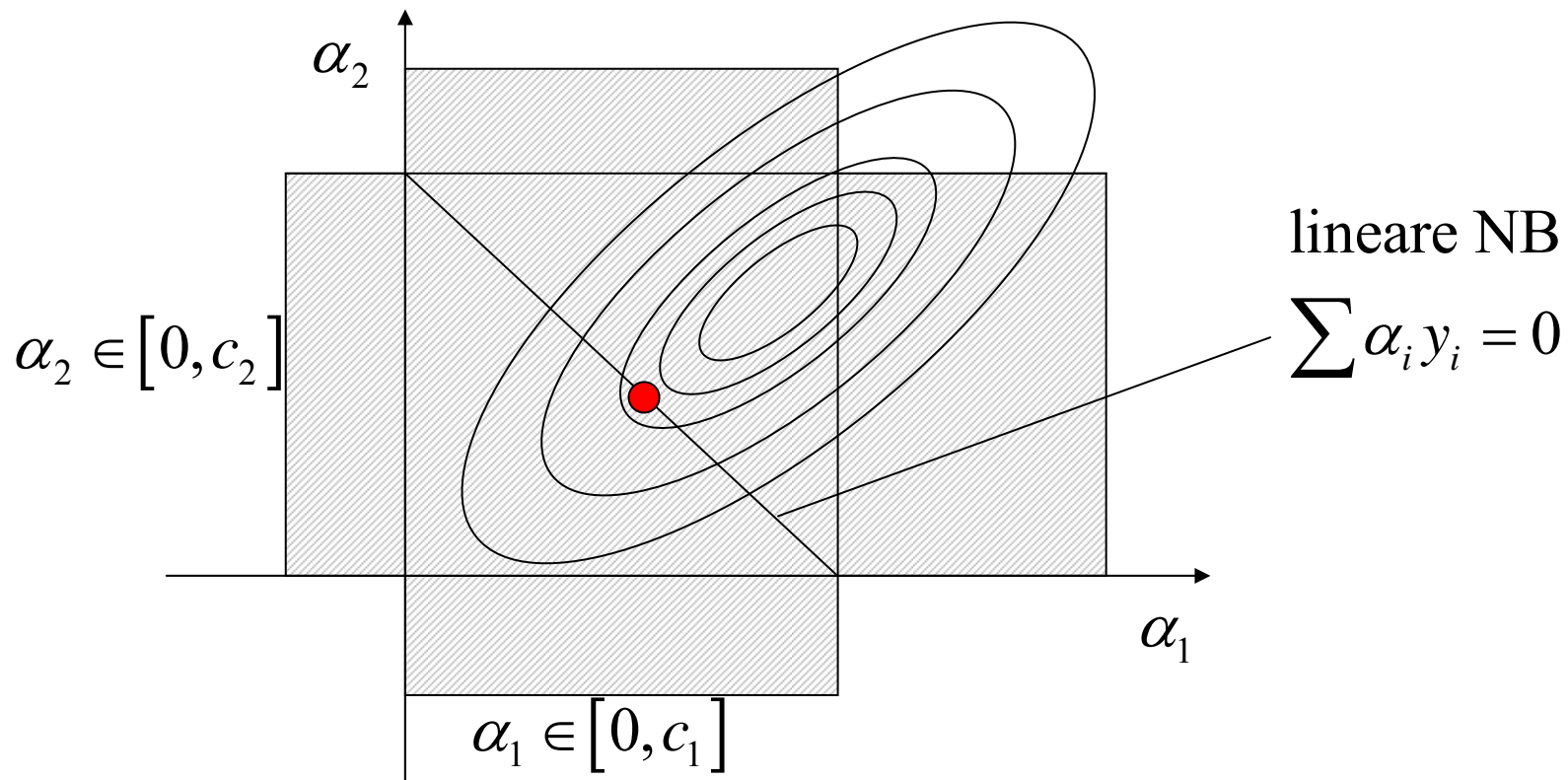
Die partiellen Ableitungen nach  $w_i$ ,  $b$  und den  $\alpha_i$  führen nach Einsetzen in das primale OP auf das äquivalente:

$$\begin{aligned} &\text{maximiere } L'(\alpha) = \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i=1}^l \sum_{j=1}^l y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle = \alpha^T \mathbf{1} - \frac{1}{2} \alpha^T \mathbf{Q} \alpha \\ &\text{Wolf-duale OP: unter den N.B.: } 0 \leq \alpha_i \leq C \text{ (} C=\infty \text{ für den Fall ohne "slack")} \\ &\text{und } \sum_{i=1}^l y_i \alpha_i = \alpha^T \mathbf{y} = 0 \end{aligned}$$

Dies ist ein positiv semidefinites quadr. Problem für  $\alpha$  mit lin. NB, welches mit Hilfe der konvexen quadratischen Programmierung numerisch iterativ gelöst werden kann!

# Globales, eindeutiges Optimum

- Optimierung eines quadratischen Problems in  $w$  (konvex)
- unter linearen Nebenbedingungen ergibt sich ein globales Optimum
- beschränkt zugelassene Lösungsgebiete sind konvexe Mengen; mehrere Beschränkungen ergeben als Schnitt wieder konvexe Mengen => diese geschnitten mit ursprünglichen Problem erhalten die Eigenschaften eines globalen Optimums, selbst wenn dieses jetzt auch am Rand liegen kann!



# Lösung:

Lösung des dualen Problems für  $\alpha^*$  liefert eindeutig die gewünschte Hyperebene (Gln. gelten auch für den Fall mit Slack-Variablen!)

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

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

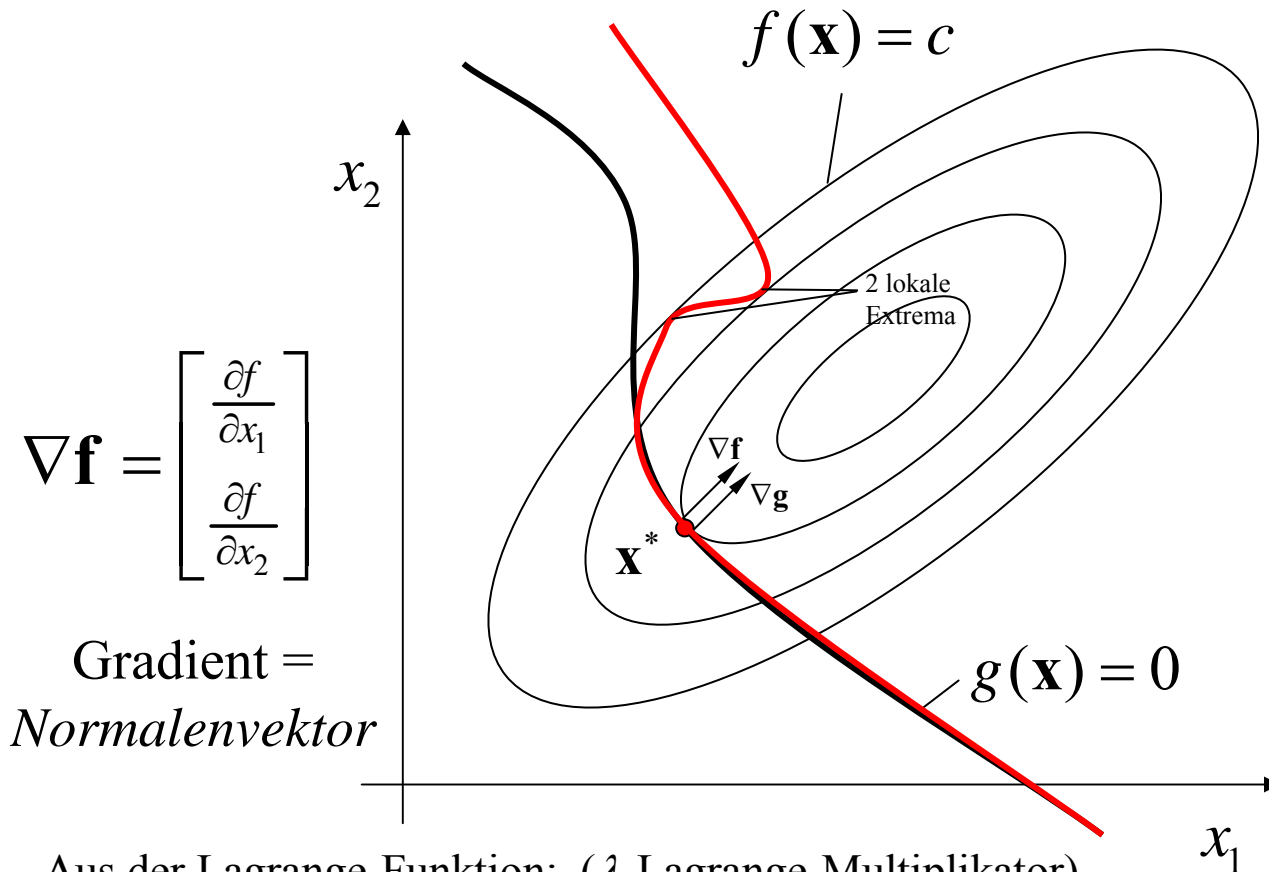
Klassifikator:  $f(\mathbf{x}) = \text{sgn}(\langle \mathbf{w}^*, \mathbf{x} \rangle + b^*) = \text{sgn} \left( \sum_{\mathbf{x}_i \in SV} \alpha_i^* y_i \langle \mathbf{x}_i, \mathbf{x} \rangle + b^* \right)$

***Lösung nur abhängig von den Supportvektoren ( $\alpha_i > 0$ ) !!***

Beobachtungen:

- Für die Support-Vektoren gilt:  $0 < \alpha_i < \infty$
- Für alle Nicht-Support-Vektoren gilt:  $\alpha_i = 0$   
-> Support-Vektoren, “sparse“-Darstellung der Lösung
- Eindeutigkeit der Ebene, globales Optimum!!

# Optimierung von $f(\mathbf{x})$ unter gegebenen Nebenbedingungen $g(\mathbf{x})=0$ mit Hilfe des Lagrange-Ansatzes



$$f(\mathbf{x}) = \min_{\mathbf{x}}$$

$$\text{NB: } g(\mathbf{x}) = 0$$

Eine Lösung für das Optimierungsproblem zu finden ist gleichbedeutend mit der Aufgabe, stationäre Punkte  $\mathbf{x}^*$  zu finden, in denen gilt:

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

Dies ist nur eine notwendige Bedingung für ein lokales Optimum; es kann mehrere Lösungen geben!  
Bei linearer NB gibt es nur eine Lösung!

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

Gradient =  
Normalenvektor

Aus der Lagrange-Funktion: ( $\lambda$  Lagrange-Multiplikator)

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

folgt mit der notwendigen Bedingung für ein Extremum:

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

$\Rightarrow \boxed{\nabla \mathbf{f} \parallel \nabla \mathbf{g}}$  dies ist aber genau die Bedingung für einen stationären Punkt!



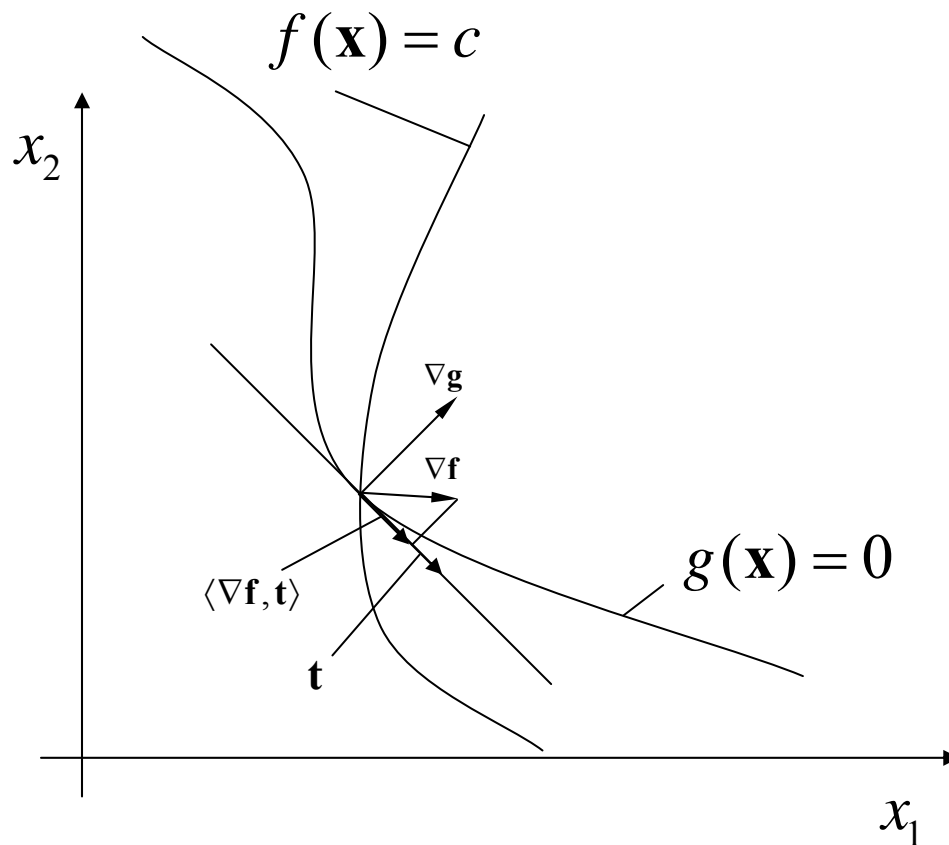
# Der Gradient ist ein Normalenvektor an die Kurve $g(\mathbf{x})=0$

Betrachten wir die Taylorentwicklung von  $g$  bzgl. einer kleinen vektoriellen Störung  $\mathbf{t}$  an einer Stelle  $\mathbf{x}$  auf der Kurve  $g(\mathbf{x})=0$ , so erhält man:

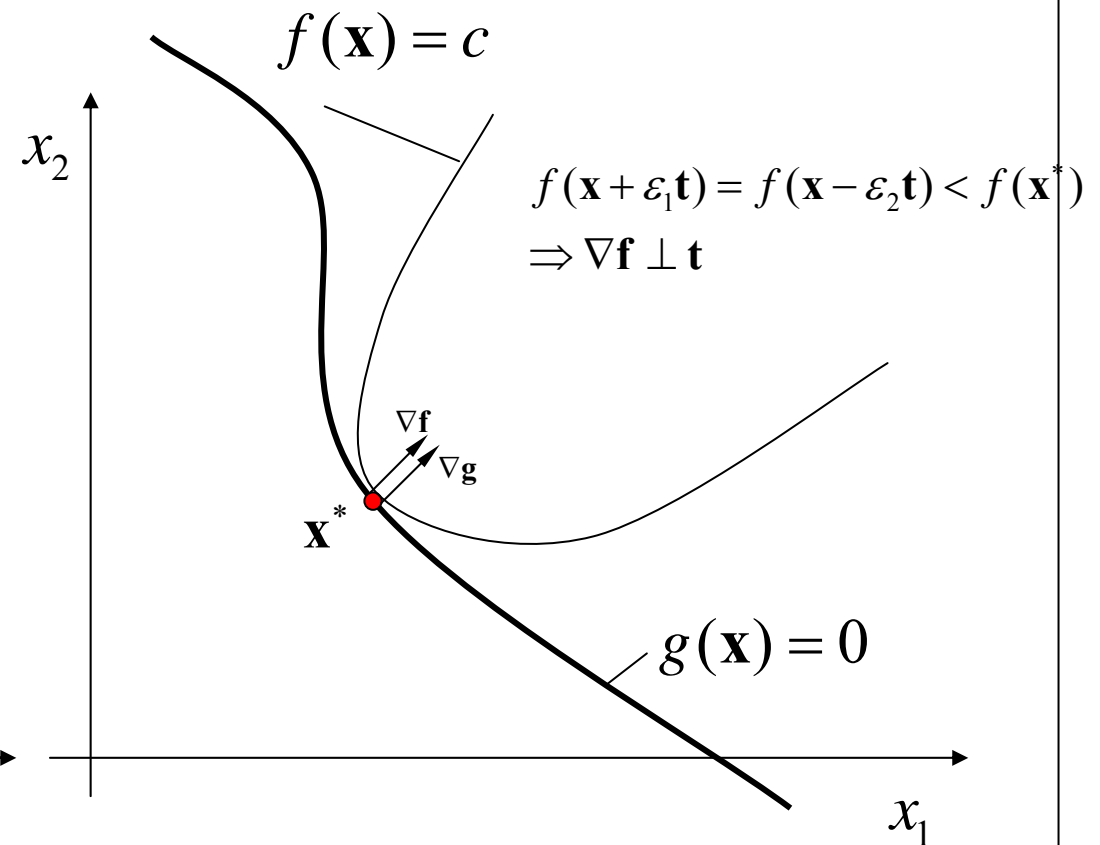
$$g(\mathbf{x} + \mathbf{t}) = g(\mathbf{x}) + \mathbf{t}^T \nabla \mathbf{g} + \dots$$

Bewegt sich die Störung  $\mathbf{t}$  tangential entlang der Kurve, so gilt  $g(\mathbf{x} + \mathbf{t}) = g(\mathbf{x})$  und somit auch  $\mathbf{t}^T \nabla \mathbf{g}(\mathbf{x}) = 0$ . Daraus erkennen wir, dass der Gradient (Normalenvektor) senkrecht steht auf den Tangentenvektor entlang der Oberfläche mit  $g(\mathbf{x})=0$ .

# Bedingung für einen stationären Punkt $\mathbf{x}^*$



a) Verhältnisse an einem nichtstationären Punkt



b) Verhältnisse an einem stationären Punkt

a) Enthält die Projektion von  $\nabla \mathbf{f}$  auf die Tangentenrichtung  $\mathbf{t}$  einen von Null verschiedenen Beitrag, so kann das Optimierungskriterium durch Bewegung in diese Richtung entlang der Kurve der NB verbessert werden!  $\Rightarrow$  kein stationärer Punkt!

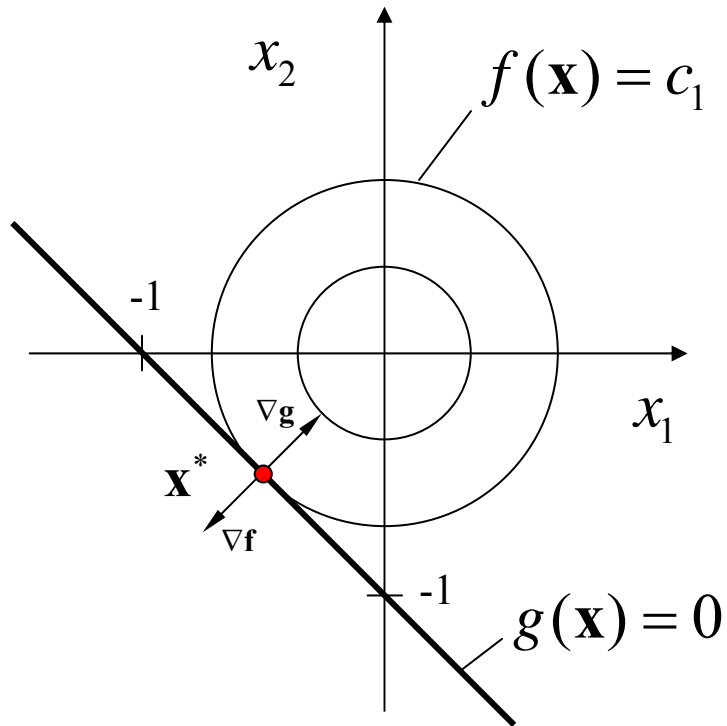
b) An einem stationären Punkt wird das Gütekriterium  $f(\mathbf{x})$  in *beiden* tangentialen Richtungen verschlechtert.  $\nabla \mathbf{f}$  steht senkrecht auf  $\mathbf{t}$ .

Beispiel:

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

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

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



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

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

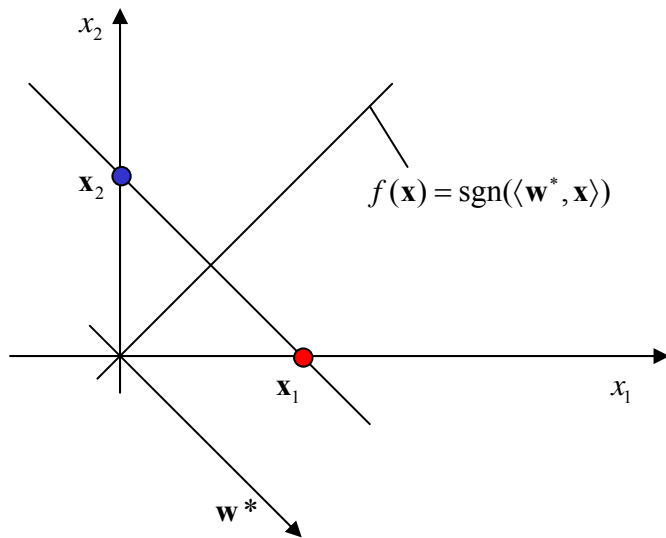
notw. Bed. für ein Optimum:

$$\begin{aligned} 1) \quad \nabla L = \frac{\partial L}{\partial \mathbf{x}} = \nabla \mathbf{f} + \alpha \nabla \mathbf{g} = \mathbf{0} & \quad \Rightarrow \quad \begin{aligned} 1) \quad 2x_1 + \alpha &= 0 \\ 2) \quad 2x_2 + \alpha &= 0 \end{aligned} \\ 2) \quad \frac{\partial L}{\partial \alpha} = g(\mathbf{x}) = 0 & \quad \Rightarrow \quad 2) \quad x_1 + x_2 + 1 = 0 \end{aligned}$$

$$\Rightarrow \alpha^* = 1 \quad \text{und} \quad \mathbf{x}^* = - \begin{bmatrix} 1/2 \\ 1/2 \end{bmatrix}$$

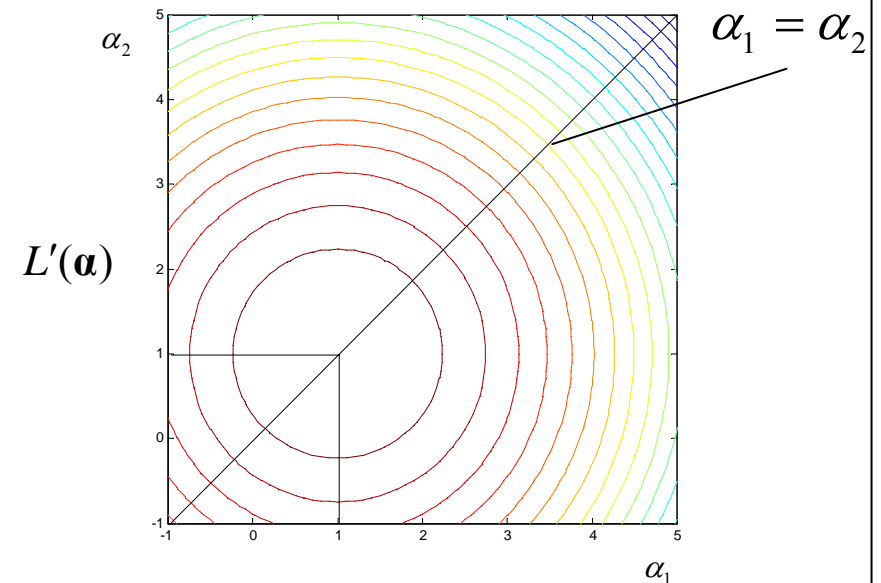
$$\Rightarrow \nabla \mathbf{f}(\mathbf{x}^*) = \begin{bmatrix} -1 \\ -1 \end{bmatrix} = -\alpha^* \nabla \mathbf{g}(\mathbf{x}^*) = -1 \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

# Beispiel: Entwurf einer SVM für 2 Punkte



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

Beide Vektoren sind Supportvektoren!



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

2) unter den N.B.:  $0 \leq \alpha_i < \infty$

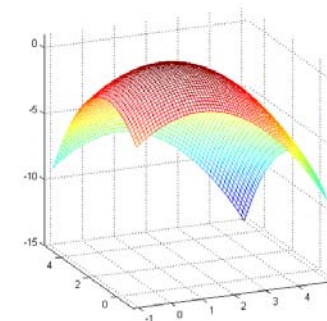
3) und  $\sum_{i=1}^2 y_i \alpha_i = (\alpha_1 - \alpha_2) = 0$

Lösung:  $\alpha_1 = \alpha_2 = 1$

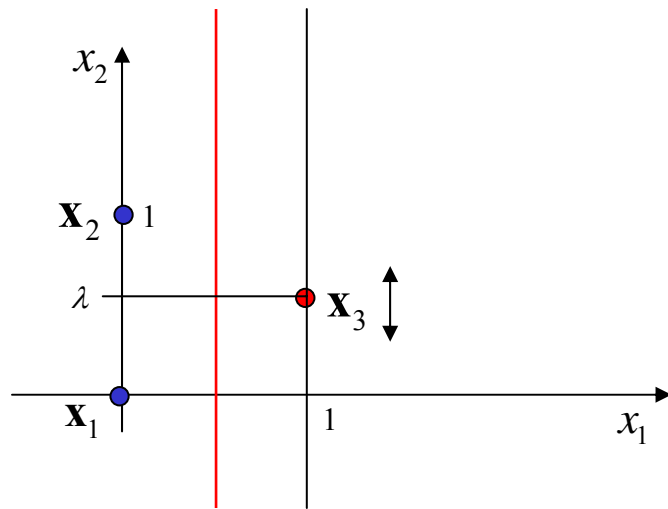
$$\mathbf{w}^* = \sum_{\mathbf{x}_i \in SV} \alpha_i y_i \mathbf{x}_i = \mathbf{x}_1 - \mathbf{x}_2 = \begin{bmatrix} +1 \\ -1 \end{bmatrix}$$

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

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

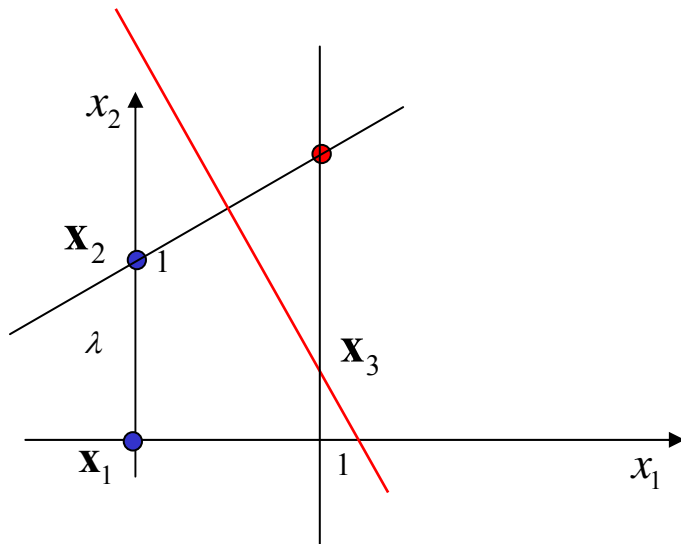


# Beispiel: Entwurf einer SVM für 3 Punkte



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

Fall I: für  $0 \leq \lambda \leq 1$  drei Supportvektoren



Fall II: für sonst. Werte: zwei Supportvekt.

# Beispiel: Entwurf einer SVM für 3 Punkte

1) maximiere  $L'(\boldsymbol{\alpha}) = \sum_{i=1}^3 \alpha_i - \frac{1}{2} \sum_{i=1}^3 \sum_{j=1}^3 y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle = (\alpha_1 + \alpha_2 + \alpha_3) - \frac{1}{2}(\alpha_2^2 - 2\lambda\alpha_2\alpha_3 + (1+\lambda^2)\alpha_3^2)$

2) unter den N.B.:  $0 \leq \alpha_i < \infty$

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

3) Setzt man die lin. NB 3) in 1) ein so ergibt sich:

4) maximiere  $L'(\alpha_2, \alpha_3) = 2\alpha_3 - \frac{1}{2}(\alpha_2^2 - 2\lambda\alpha_2\alpha_3 + (1+\lambda^2)\alpha_3^2)$

5) unter den N.B.:  $0 \leq \alpha_i < \infty$

Die Lösung von 4) ohne Beachtung von 3) (unbeschränkte Lösung) erhält man mit:

$$\frac{\partial L'(\alpha_2, \alpha_3)}{\partial \alpha_2} = -\alpha_2 + \lambda\alpha_3 = 0 \Rightarrow \alpha_2 = \lambda\alpha_3$$

$$\frac{\partial L'(\alpha_2, \alpha_3)}{\partial \alpha_3} = 2 + \lambda\alpha_2 - (1+\lambda^2)\alpha_3 = 0 \Rightarrow 2 + \lambda^2\alpha_3 - (1+\lambda^2)\alpha_3 = 0$$

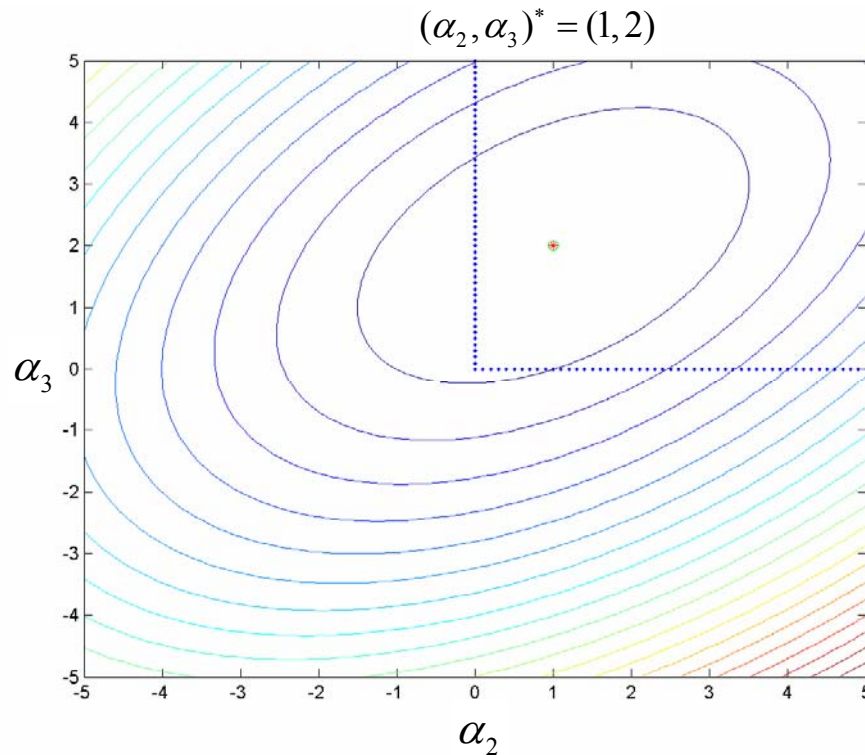
und daraus schließlich:

$$\alpha_1 = 2(1-\lambda) \quad \alpha_2 = 2\lambda \quad \alpha_3 = 2 \quad \text{für } 0 \leq \lambda \leq 1 \quad \text{sind alle } \alpha_i \geq 0 \quad (\text{Fall I})$$

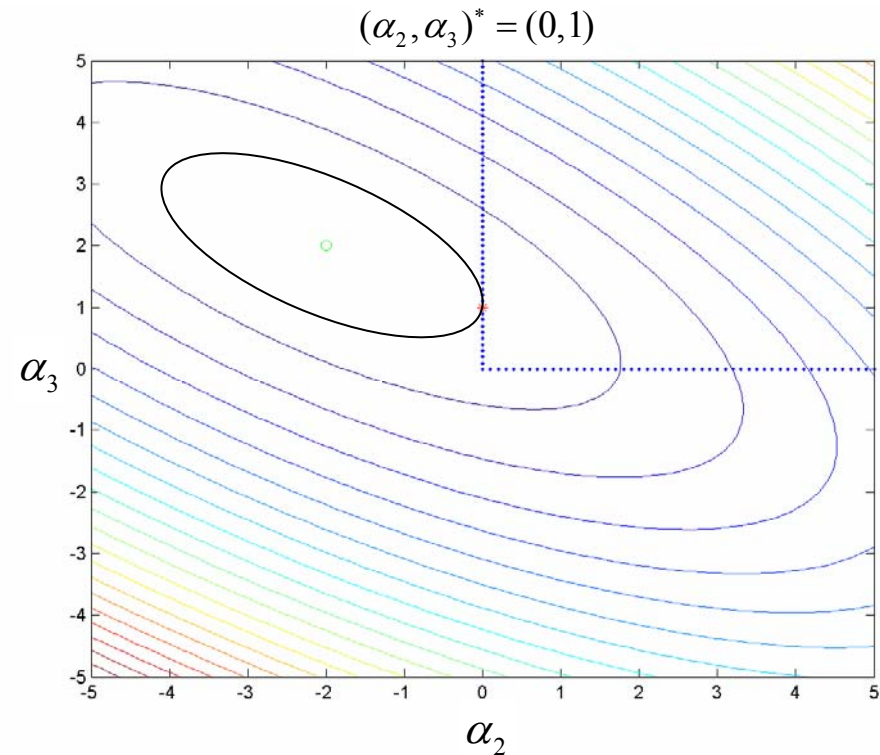
dafür gilt:  $\mathbf{w}^* = \sum_{\mathbf{x}_i \in SV} \alpha_i y_i \mathbf{x}_i = 2\lambda \begin{bmatrix} 0 \\ 1 \end{bmatrix} - 2 \begin{bmatrix} 1 \\ \lambda \end{bmatrix} = \begin{bmatrix} -2 \\ 0 \end{bmatrix}$

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

Fall I:  $\lambda=0,5$  absolutes Maximum im Bereich  $\alpha_i \geq 0$



Fall II:  $\lambda = -1$  absolutes Maximum außerhalb des Bereichs  $\alpha_i \geq 0$ , richtige Lösung am Rand



unbeschränkte Lösung mit  $\lambda = 0,5$  :

$$\alpha_1 = 2(1 - \lambda) = 1$$

$$\alpha_2 = 2\lambda = 1$$

$$\alpha_3 = 2$$

Hier liegt die Lösung offensichtlich auf dem Rand mit  $\alpha_2 = 0$ , d.h.  $\mathbf{x}_2$  ist kein Support-Vektor! Das Maximum von

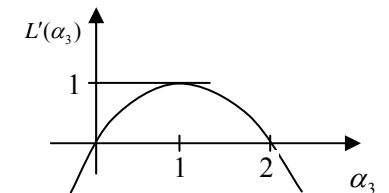
$$L'(\alpha_2, \alpha_3)|_{\alpha_2=0, \lambda=-1} = 2\alpha_3 - \frac{1}{2}((1 + \lambda^2)\alpha_3^2) = \alpha_3(2 - \alpha_3)$$

ergibt sich aus:

$$\frac{dL'(\alpha_3)}{d\alpha_3} = 2 - 2\alpha_3 = 0 \Rightarrow \alpha_3 = 1$$

und damit:  $(\alpha_1, \alpha_2, \alpha_3) = (1, 0, 1)$

Das absolute Maximum liegt hingegen bei:  $(\alpha_1, \alpha_2, \alpha_3) = (4, -2, 2)$



# Demos mit MATLAB

(svmmatlab, uiclass.m)

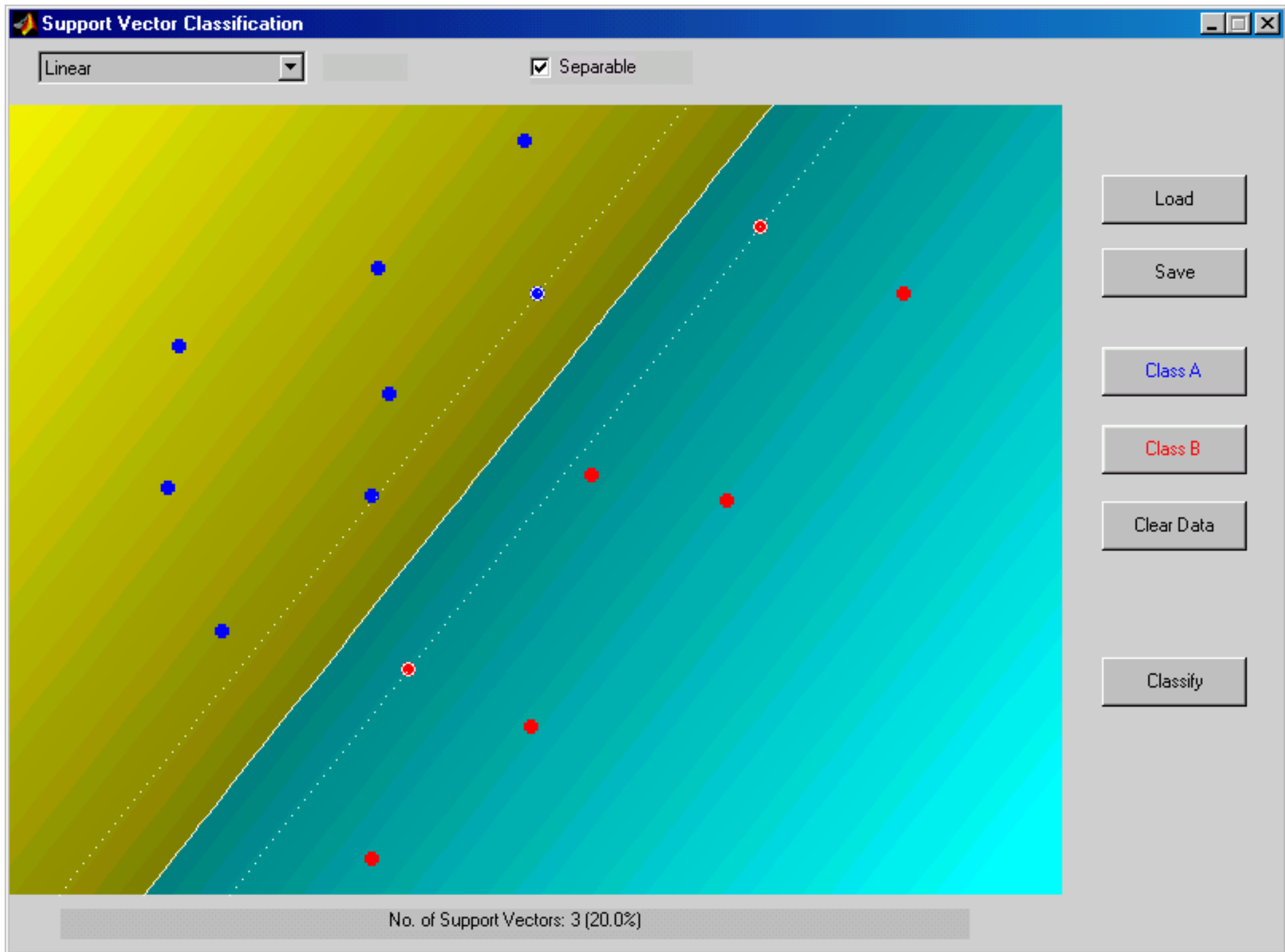
- **Linearer Klassifikator**
  - Problem linear separierbar
    - Harter Rand, Problem separierbar
    - Harter kleiner Rand wegen Ausreißer, Problem separierbar, aber schlechte Generalisierung => emp. Fehler vergrößern mit Gewinn bei Generalisierung => weichen Rand einführen
  - Problem nicht linear separierbar
    - Weichen Rand verwenden; Kontrolle mit  $C$
- **Nichtlinearer Klassifikator**
  - Für nichtlineare Probleme muss VC-Dimension der trennenden Hyperflächen und damit ihre Kapazität vergrößert werden!
  - Lineare Separierung eines quadratischen Problems mit weichem Rand; nicht zufriedenstellend
  - Polynomiale Separierung ( $p=2$ ), harter Rand
  - Polynomiale Separierung ( $p=4$ ), harter Rand
  - Bananenshape mit Polynomkernen
  - Bananenshape mit Gauss-Radialbasisfunktionen

Start der Matlab-Demo

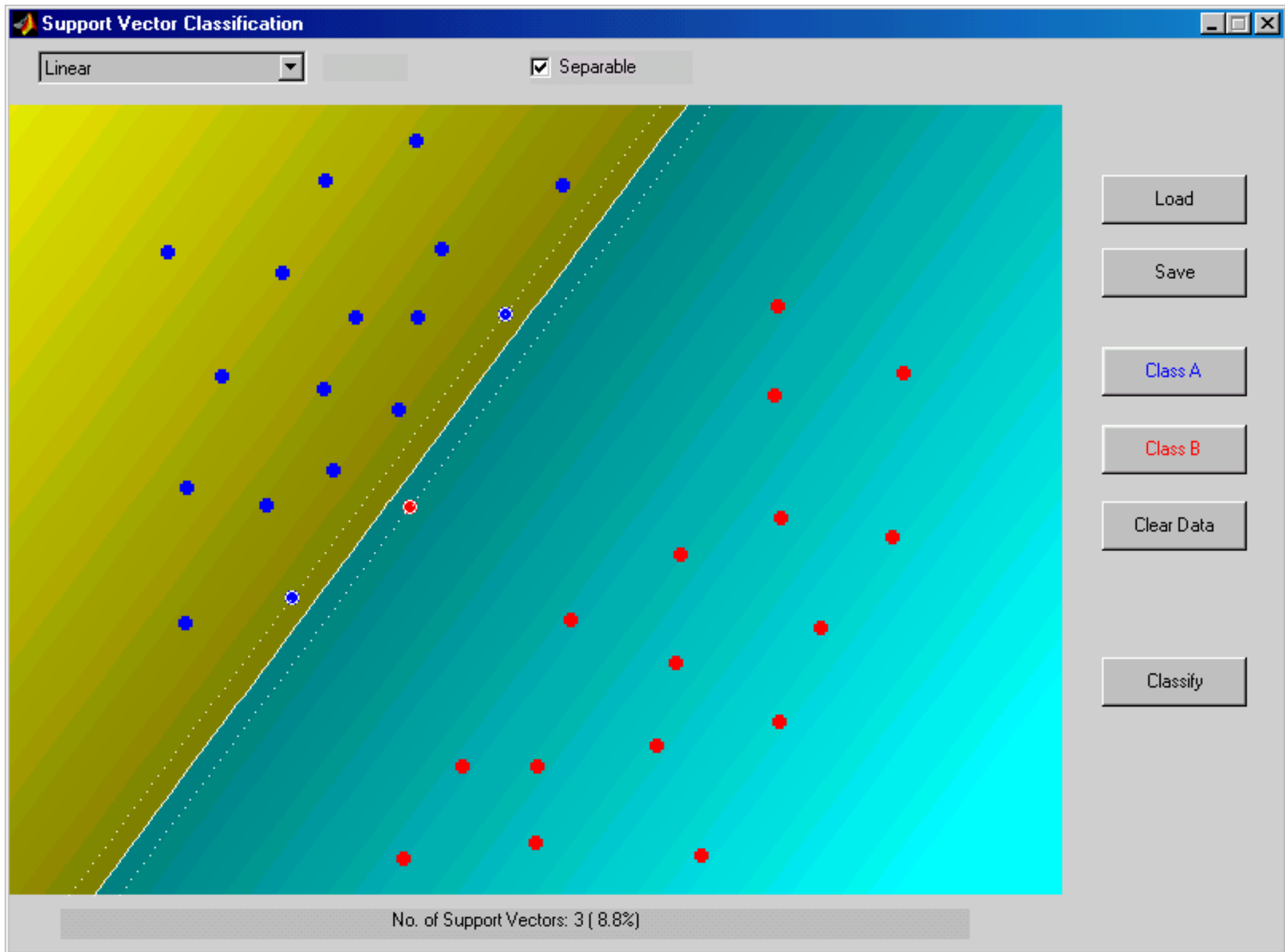
[matlab-SVM.bat](#)



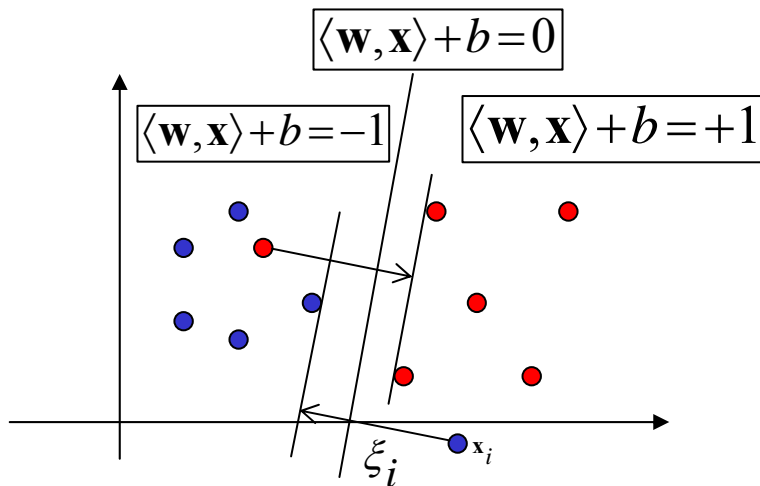
# Linear separierbare Klassen; harter Rand ( $C = \infty$ )



# Linear separable classes; harter, kleiner Rand, schlechte Generalisierung



# Nichtseparabler Fall



Bestrafen von Randverletzungen mit “slack“(Schlupf)-Variablen [Smith68] ->“Soft-Margin“ SVM

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

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

- $C$  kontrolliert Separationsweite u. Trainingsfehler; hohes  $C$  bedeutet hohe Bestrafung!
- $\sum \xi_i$  ist obere Schranke für Trainingsfehler

Gründe für diese Verallgemeinerung:

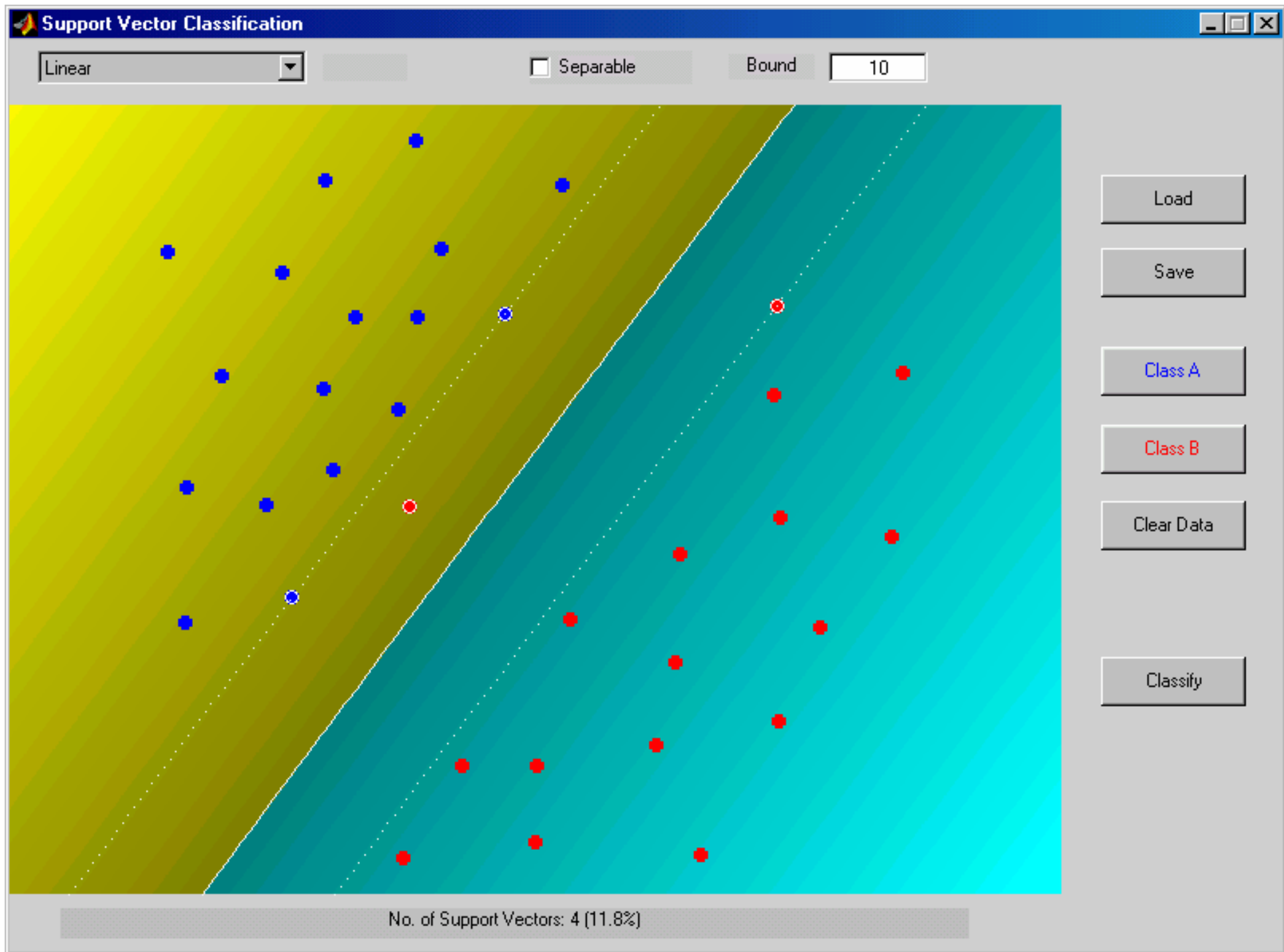
- Lösung nicht existent mit bisherigem Ansatz mit hartem Rand
- Verbesserung der Generalisierung bei Ausreißern in der Randzone auf Kosten eines etwas erhöhten empirischen Fehlers

Es sind jetzt drei Fälle zu unterscheiden:

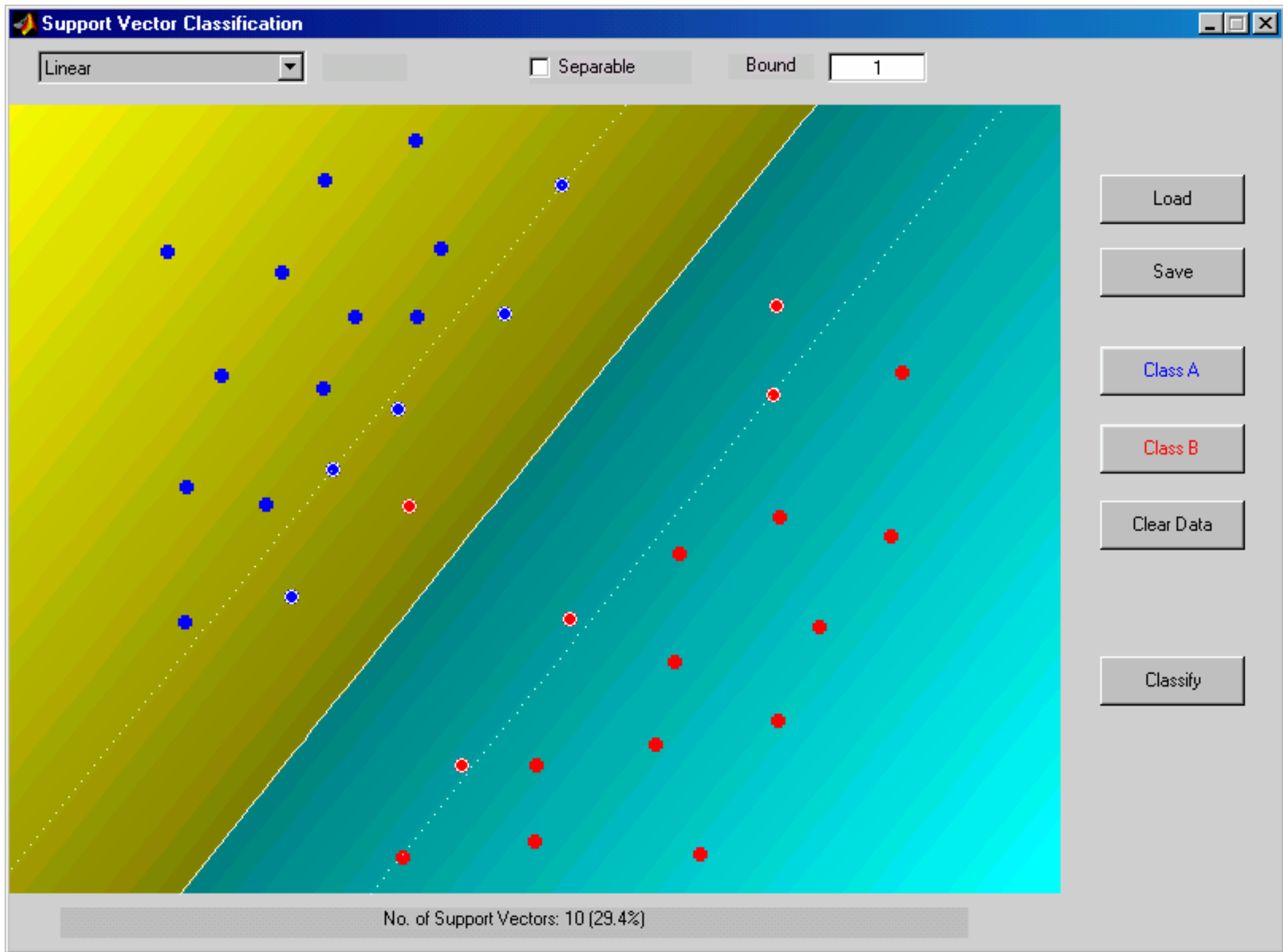
- $0 < \alpha_i < C \iff$  SV mit  $\xi_i = 0$  (liegen auf kanon. Hyperebenen)
- $\alpha_i = C \iff$  weitere SV mit  $\xi_i > 0$  (falsch zugeordnet)
- $\alpha_i = 0 \iff$  für die restlichen Vektoren  $\mathbf{x}_i$

$C \rightarrow \infty$  entspricht separierbarem Ansatz und hartem Rand!

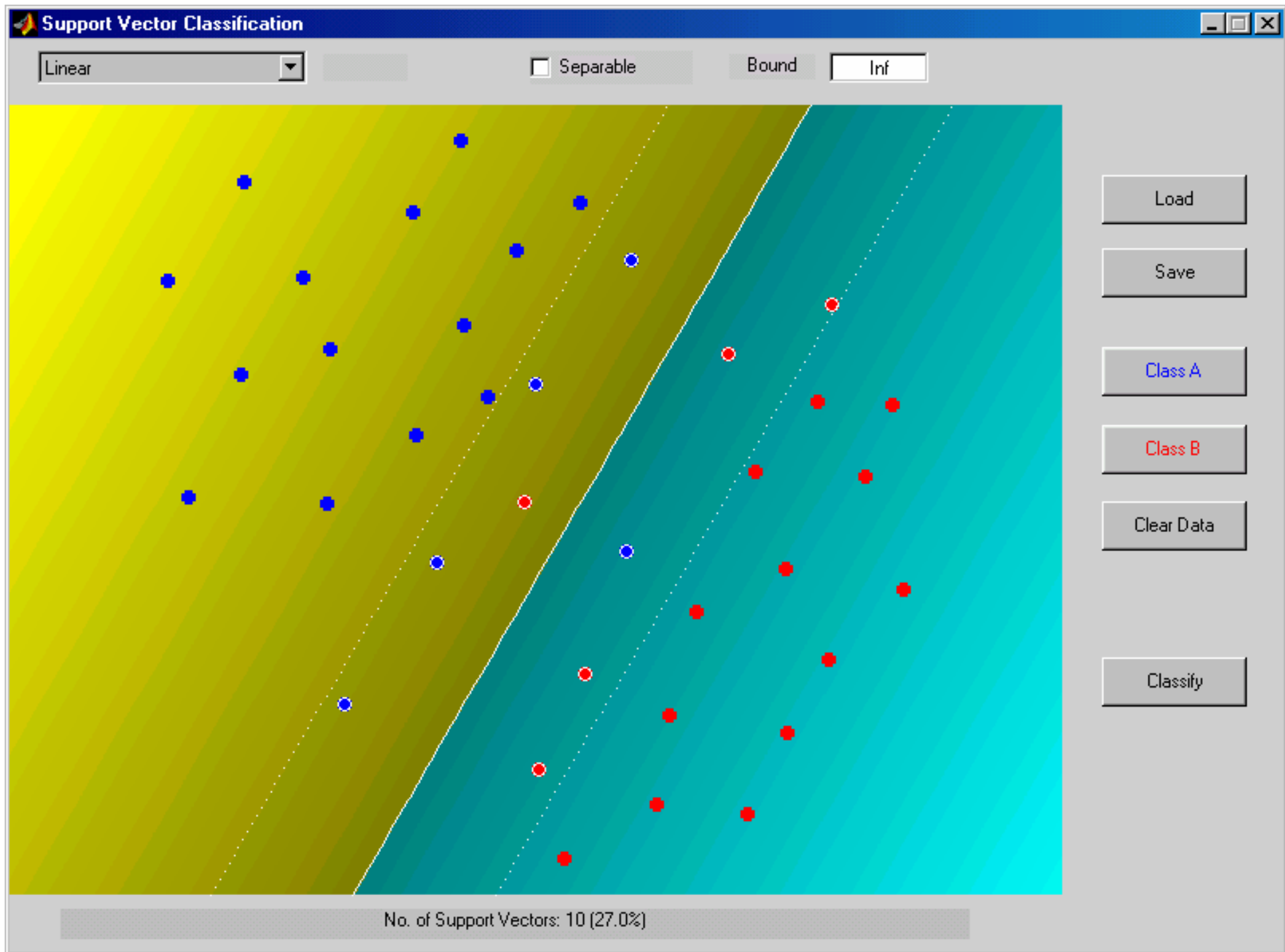
Linear sep. Klassen; weicher, breiter Rand =>  $R_{emp}$  größer, aber gute Generalisierung



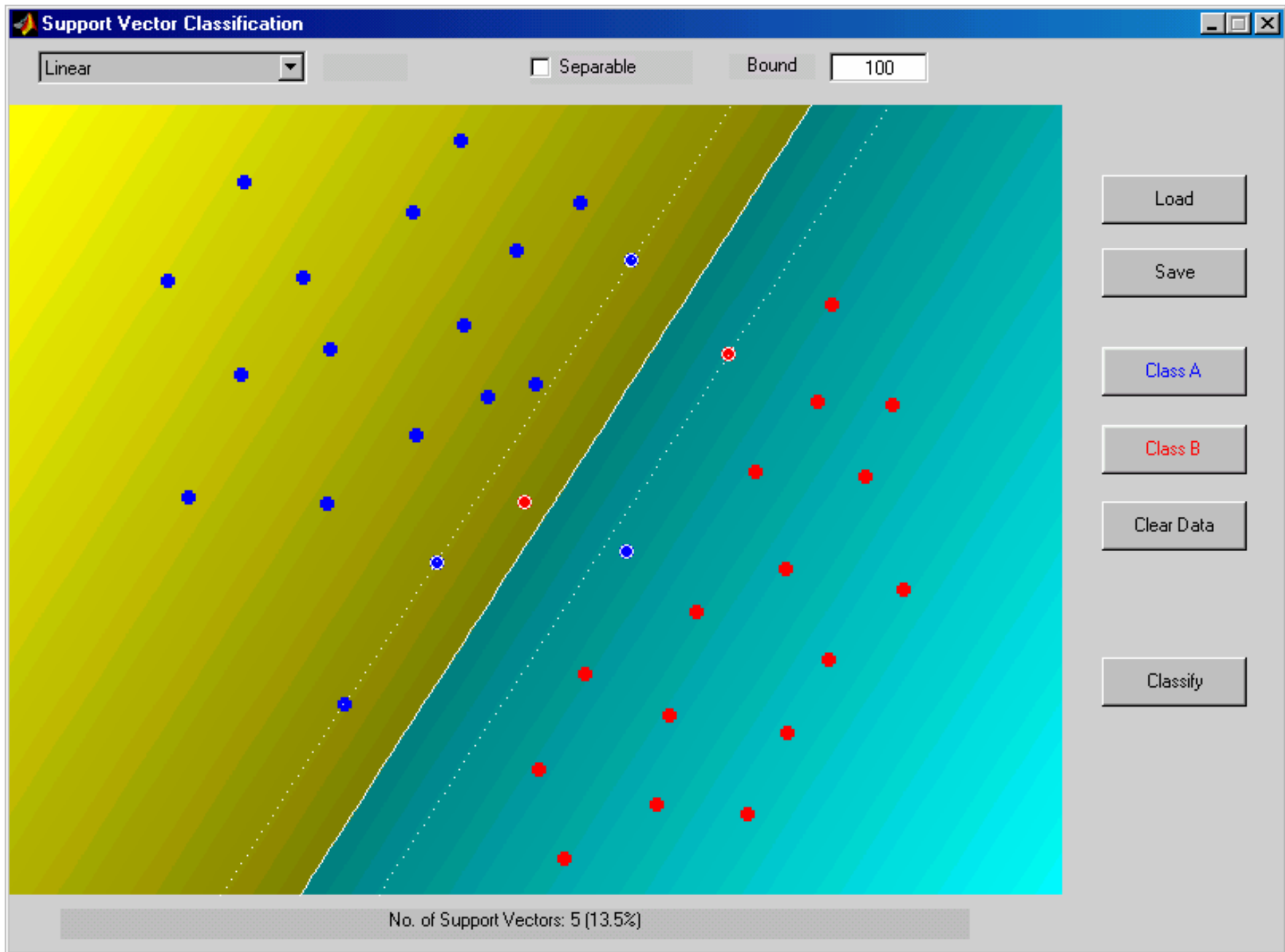
Linear sep. Klassen; weicher, breiter Rand =>  $R_{emp}$  größer, gute Generalisierung



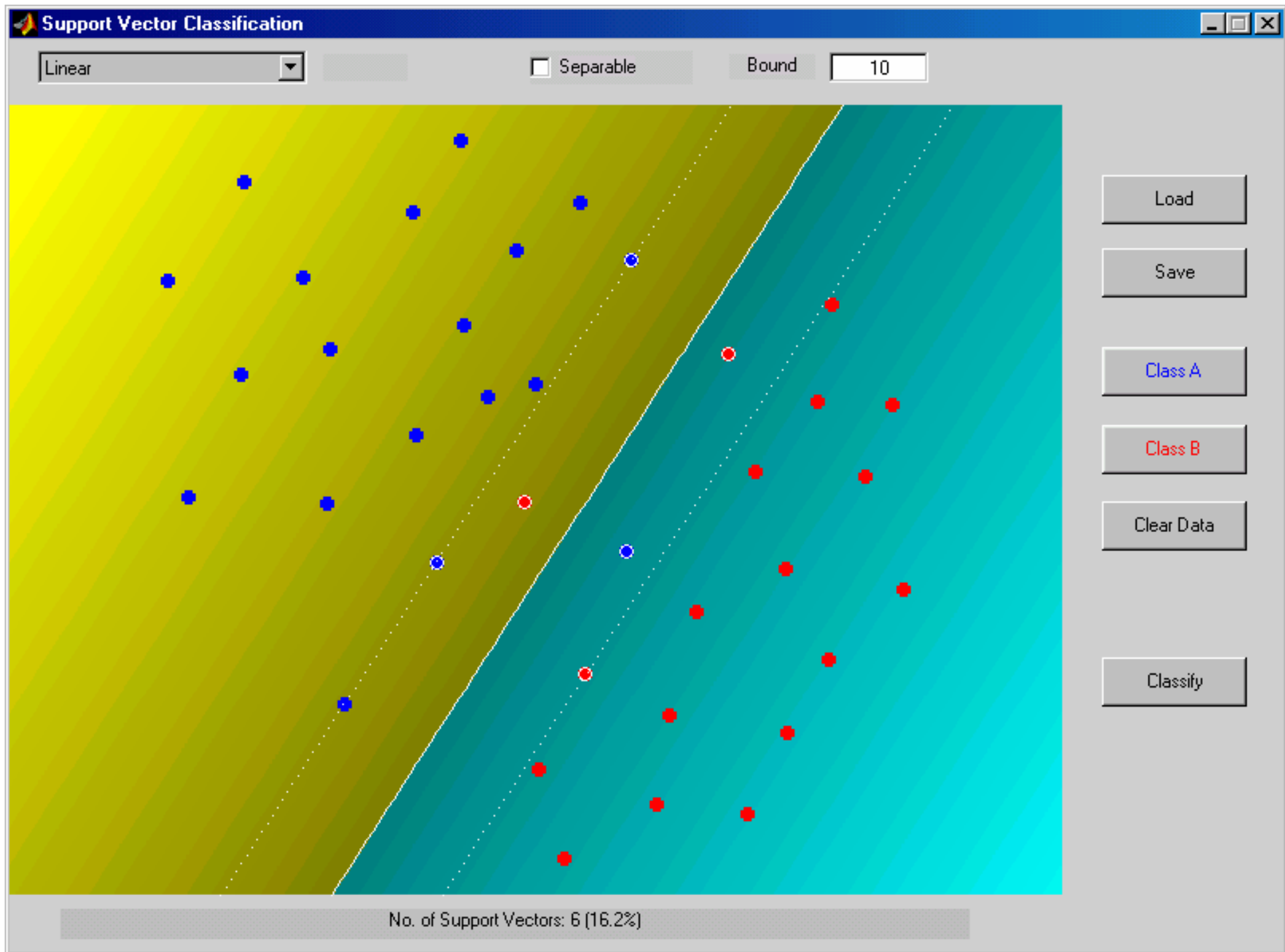
# Nicht linear separierbare Klassen; harter Rand



# Nicht linear separierbare Klassen; breiter Rand

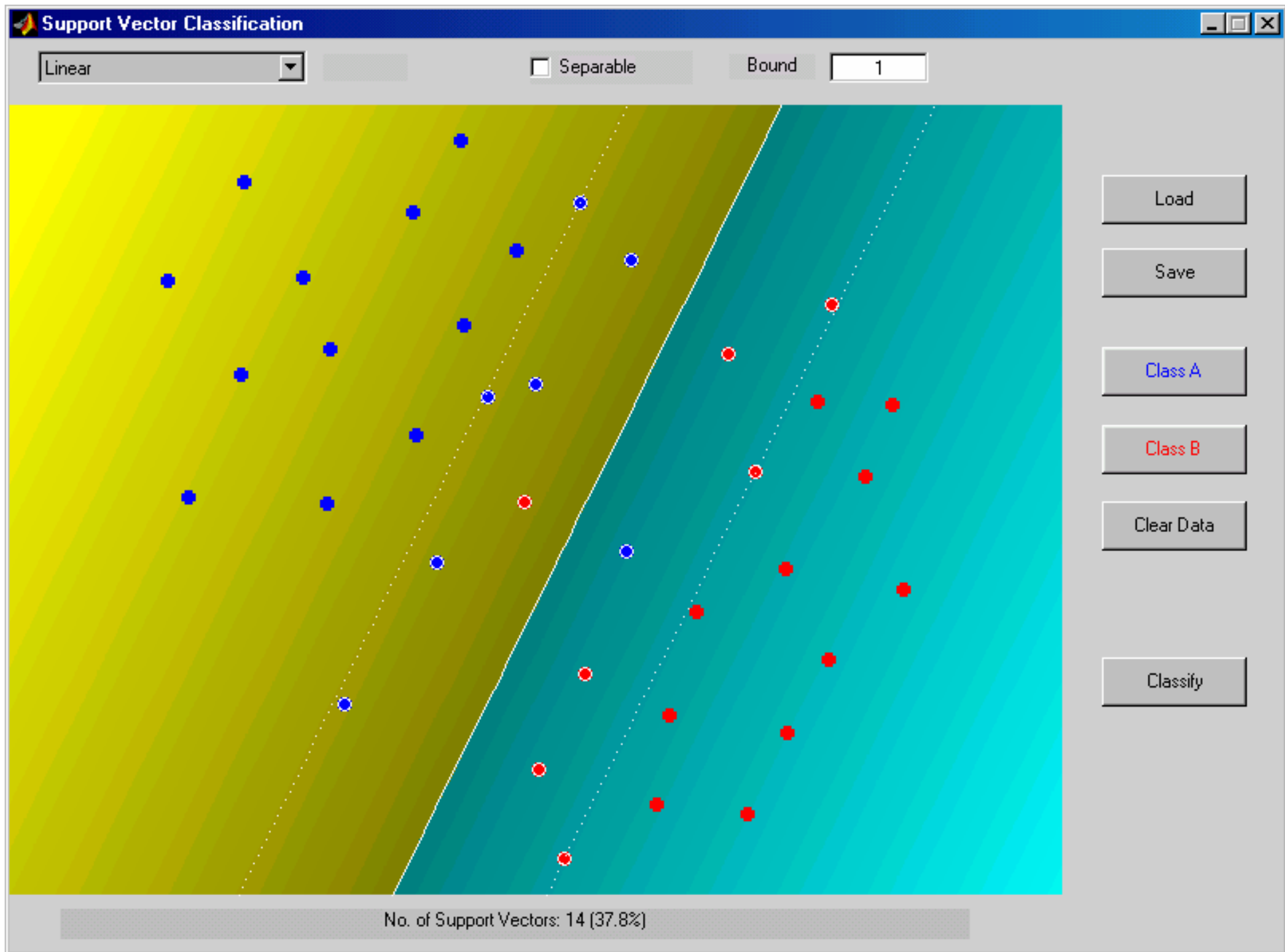


# Linear separable Klassen; breiter Rand



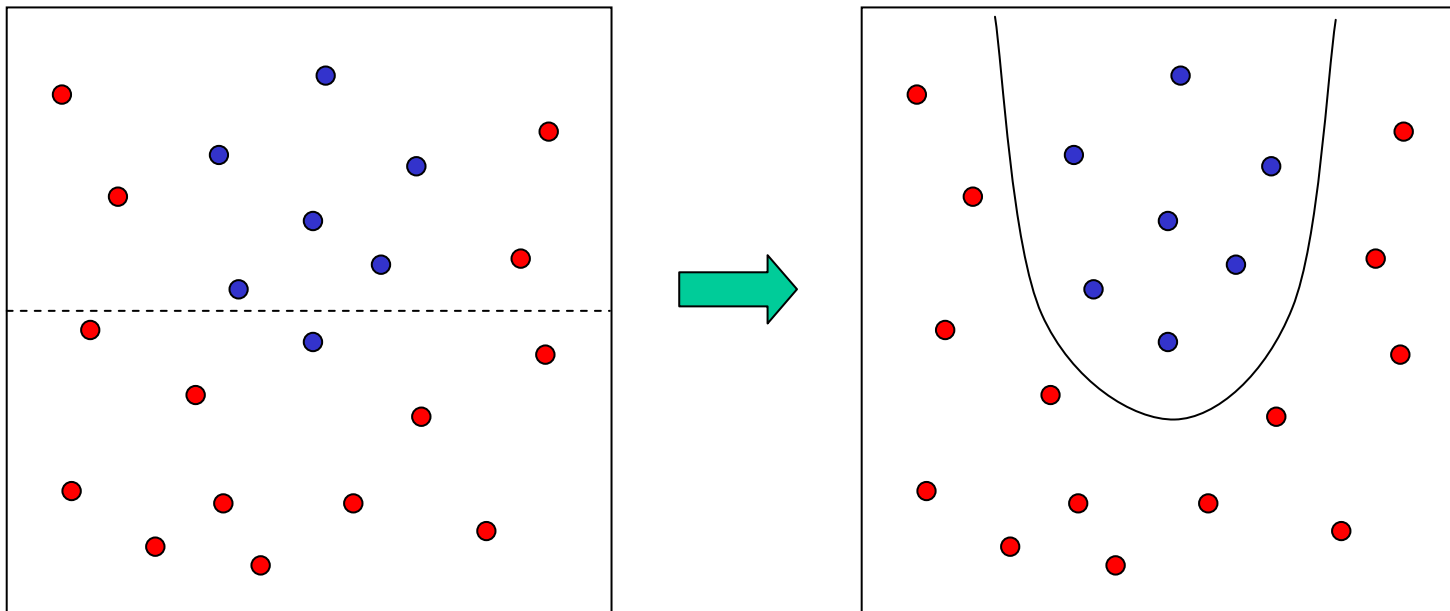


# Nicht linear separierbare Klassen; breiter Rand



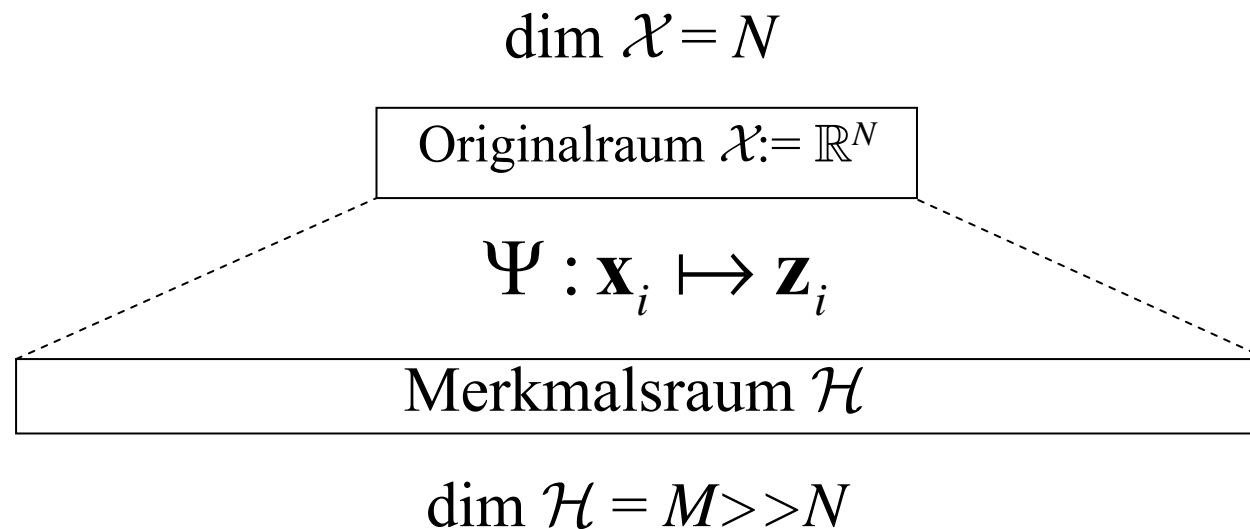
# Nichtlineare Probleme

- manche Probleme haben nichtlineare Klassengrenzen
- Hyperebenen erreichen keine zufriedenstellende Genauigkeit



# Erweiterung des Hypothesenraumes

Idee: Finde Hyperebene im höherdimensionalen Merkmalsraum

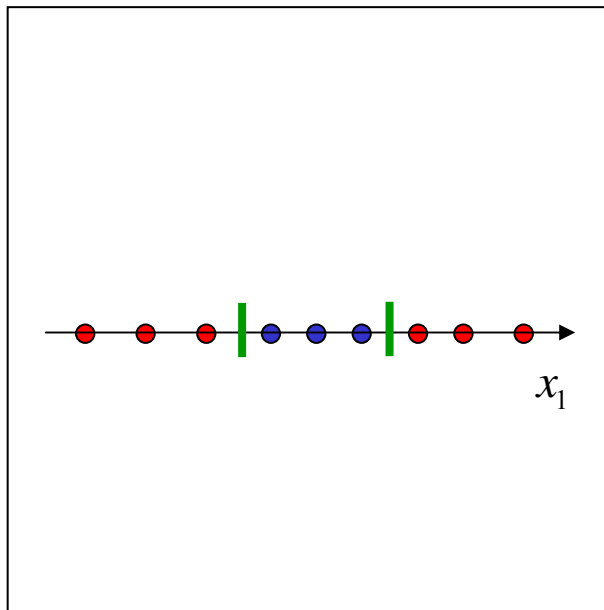


Die trennende Hyperebene im Merkmalsraum ist eine nichtlineare Trennfläche im Originalraum (siehe XOR-Problem mit Polynomklassifikator)

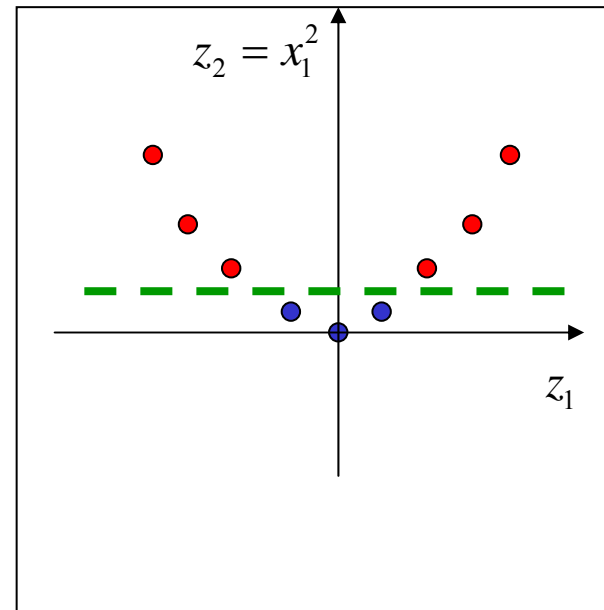
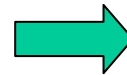
# Nichtlineare Probleme

- Eindimensionaler Originalraum:  $x_1$
- Zweidimensionaler Merkmalsraum:

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



linear nicht  
separierbar



lineare  
Separation

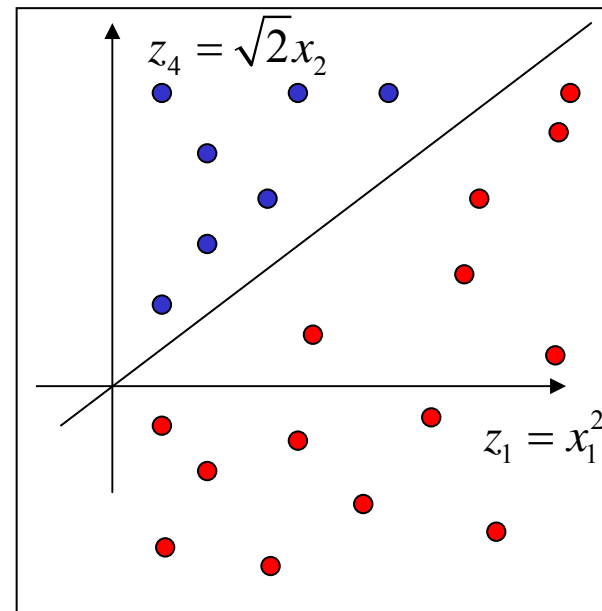
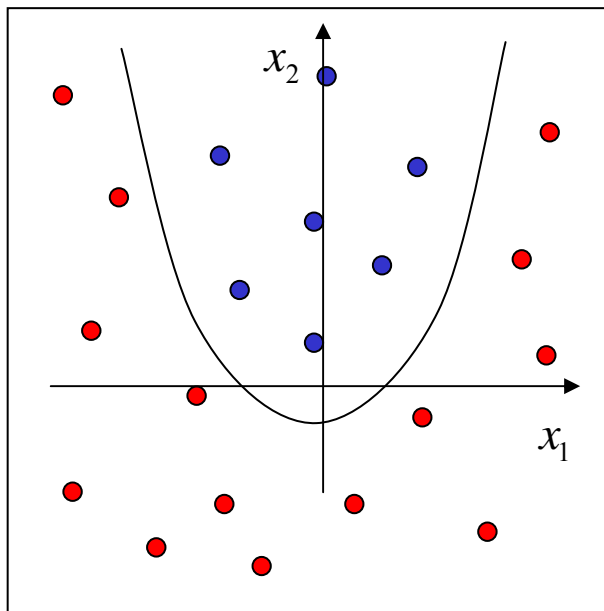
# Nichtlineare Probleme

- Originalraum:

$$\mathbf{x}=(x_1,x_2) \text{ (zweidimensional)}$$

- Merkmalsraum:

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



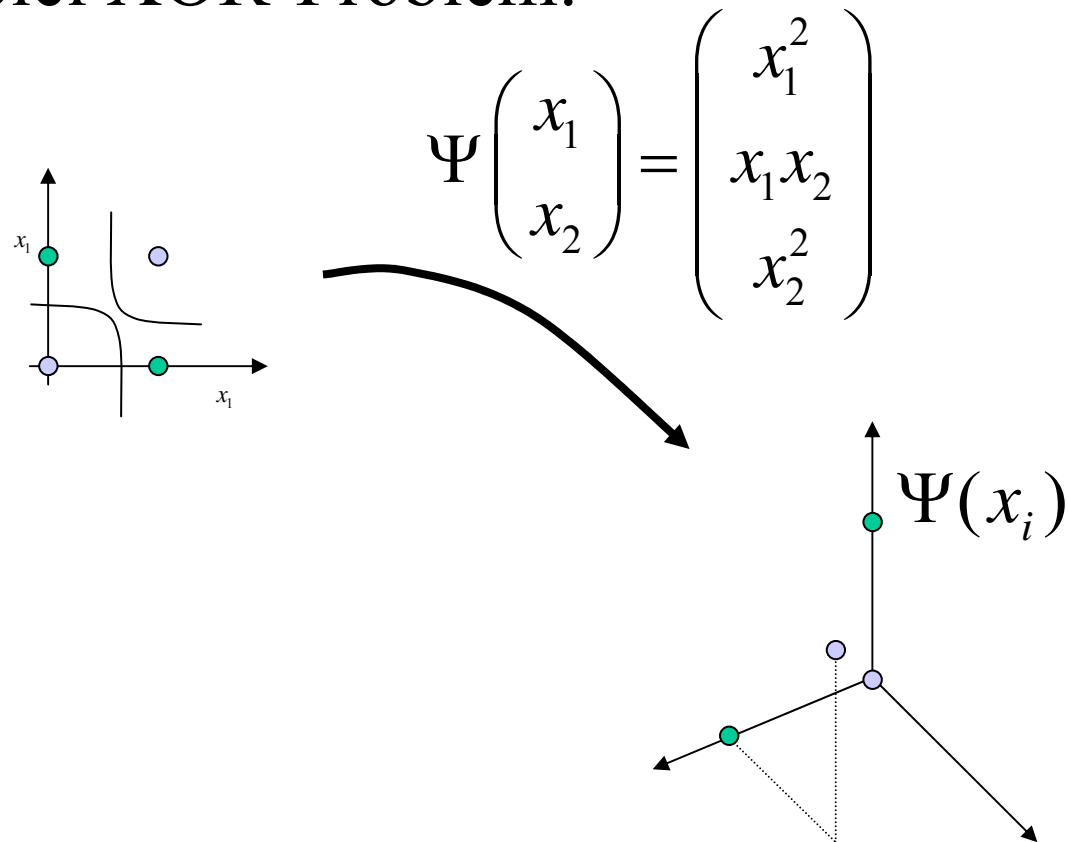
•  $x_2 > x_1^2$  nichtlineare  
•  $x_2 < x_1^2$  Separation

•  $z_4 > \sqrt{2}z_1$  lineare  
•  $z_4 < \sqrt{2}z_1$  Separation

# Nichtlineare Erweiterung

Nichtlineare Abbildung vorschalten  $\Psi(\mathbf{x}) : \mathbb{R}^N \rightarrow \mathcal{H}$

- Beispiel XOR-Problem:



# Konsequenzen

- Effekt:
  - Steigerung der Separabilität
  - Trennfläche im Ursprungsraum nichtlinear
- Fragen:
  1. Optimalität der Hyperebene?
  2. Hoher Rechenaufwand in hochdimensionalen Räumen?
- Zu 1: Optimalität bleibt erhalten, erneut positiv semidefinite Form, da in der zu optimierenden Funktion die gleichen Skalarprodukte auftauchen, nur in einem neuen Raum  $\mathcal{H}$
- Zu 2: der hohe Aufwand in den hochdimensionalen Merkmalsraum  $\mathcal{H}$  wird durch den Trick mit Kernfunktionen reduziert. Das Innenprodukt in  $H$  hat eine äquivalente Formulierung mit Kernfunktionen im Originalraum  $\mathcal{X}$

# Der Trick mit Kernfunktionen

Problem: Sehr hohe Dimension des Merkmalraumes! Polynome p-ten Grades über der Dimension  $N=\dim(\mathbf{x})$  des Originalraums führen zu  $O(M=N^p)$  Dimensionen im Merkmalsraum!

Lösung: Im dualen OP tauchen nur Skalarprodukte  $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$  auf. Im korrespondierenden Problem im Merkmalsraum tauchen dann ebenfalls nur Skalarprodukte in  $\langle \Psi(\mathbf{x}_i), \Psi(\mathbf{x}_j) \rangle$  auf. Diese müssen nicht explizit ausgerechnet werden, sondern können mit reduzierter Komplexität mit Kernfunktionen im Originalraum  $\mathcal{X}$  ausgedrückt werden:

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

Beispiel für  $N = 2$ :

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

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

das Skalarprodukt im Merkmalsraum.



# Häufig verwendete Kernfunktionen

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

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

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

Die resultierenden Klassifikatoren sind vergleichbar mit Polynomklassifikatoren, radialen Basisfunktionen und mit Neuronalen Netzen (sie werden allerdings anders motiviert).

Allgemeine Anforderung: Mercer's Bedingung. Sie garantiert, dass eine bestimmte Kernfunktion tatsächlich auch ein Skalarprodukt in irgendeinem Raum ist, aber sie erklärt nicht wie das dazugehörige Abbildung  $\Psi$  aussieht und wie der Raum  $\mathcal{H}$  beschaffen ist.

Ausserdem: Linearkombinationen von gültigen Kernen liefern neue Kerne (die Summe 2er pos. def. Fkten ist wieder pos. def.)

# Das Theorem von Mercer

Es existiert eine Abbildung  $\Psi$  und eine Entwicklung .

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

genau dann, wenn für ein *beliebiges*  $g(\mathbf{x})$  mit

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

gilt, dass  $K$  eine symmetrische, positiv semidefinite Funktion in  $\mathbf{x}$  ist:

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

Es gibt allerdings auch Fälle, wo Kernfunktionen die Mercer-Bedingung nicht erfüllen, aber für einen *bestimmten* Trainingsdatensatz zu einer positiv semidefiniten Hesse-Matrix führen und damit zu einem globalen Optimum konvergieren.

# Endformulierung

– Trainieren:

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

$$\text{mit: } \sum_{i=1}^l y_i \alpha_i = 0 \quad \text{und} \quad 0 \leq \alpha_i \leq C \quad (\text{harter Rand: } C = \infty)$$

C wägt ab zwischen emp. Fehler und Generalisierung

– Klassifizieren eines unbekanntes Objekts  $\mathbf{x}$  ( $\alpha_i \neq 0$  für alle  $SV$ ):

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

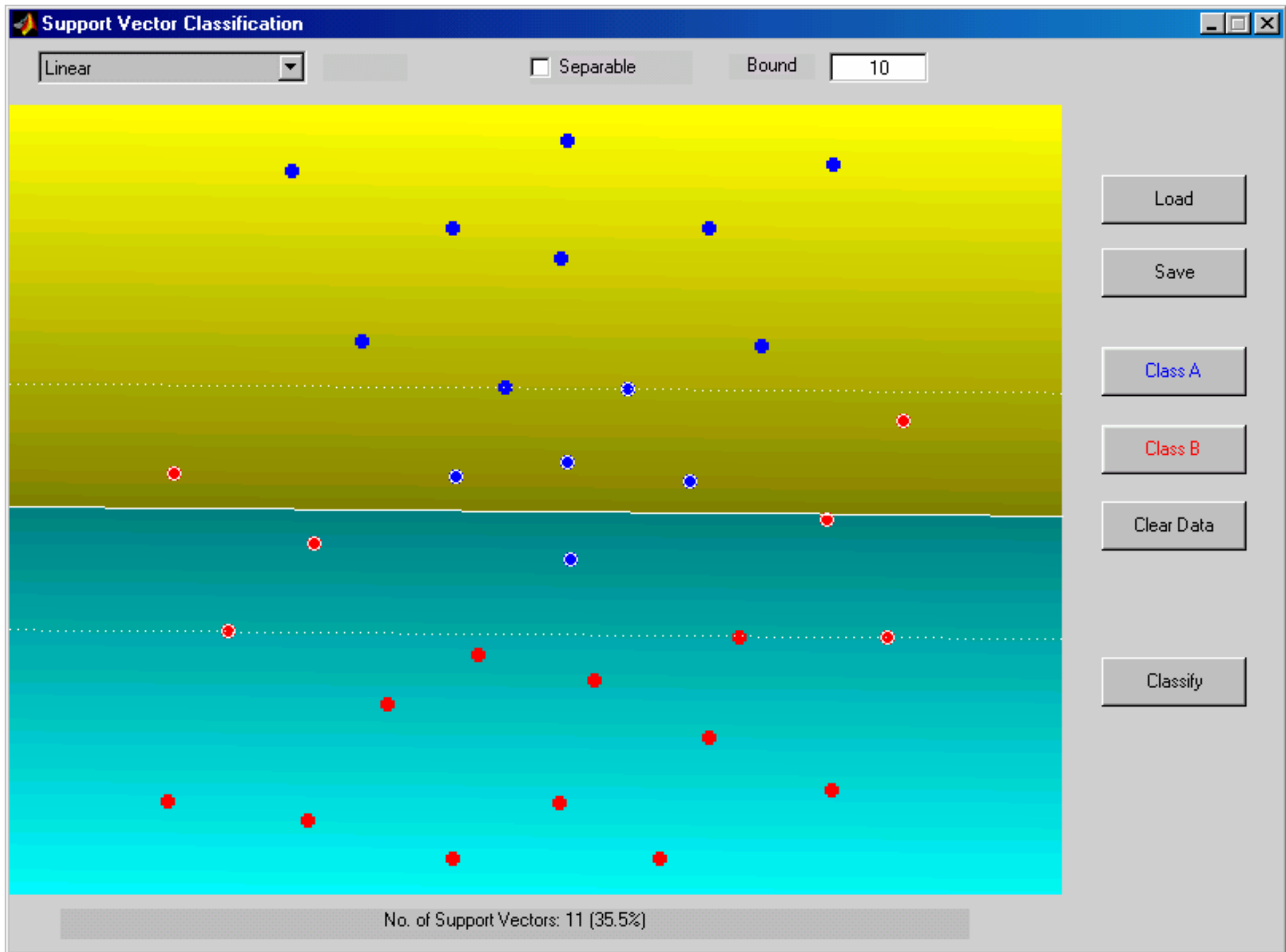
$$b^* = -\frac{1}{2} \left( \max_{i, y_i = -1} \left( \sum_{j=1}^l y_j \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \right) + \min_{i, y_i = +1} \left( \sum_{j=1}^l y_j \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \right) \right)$$

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

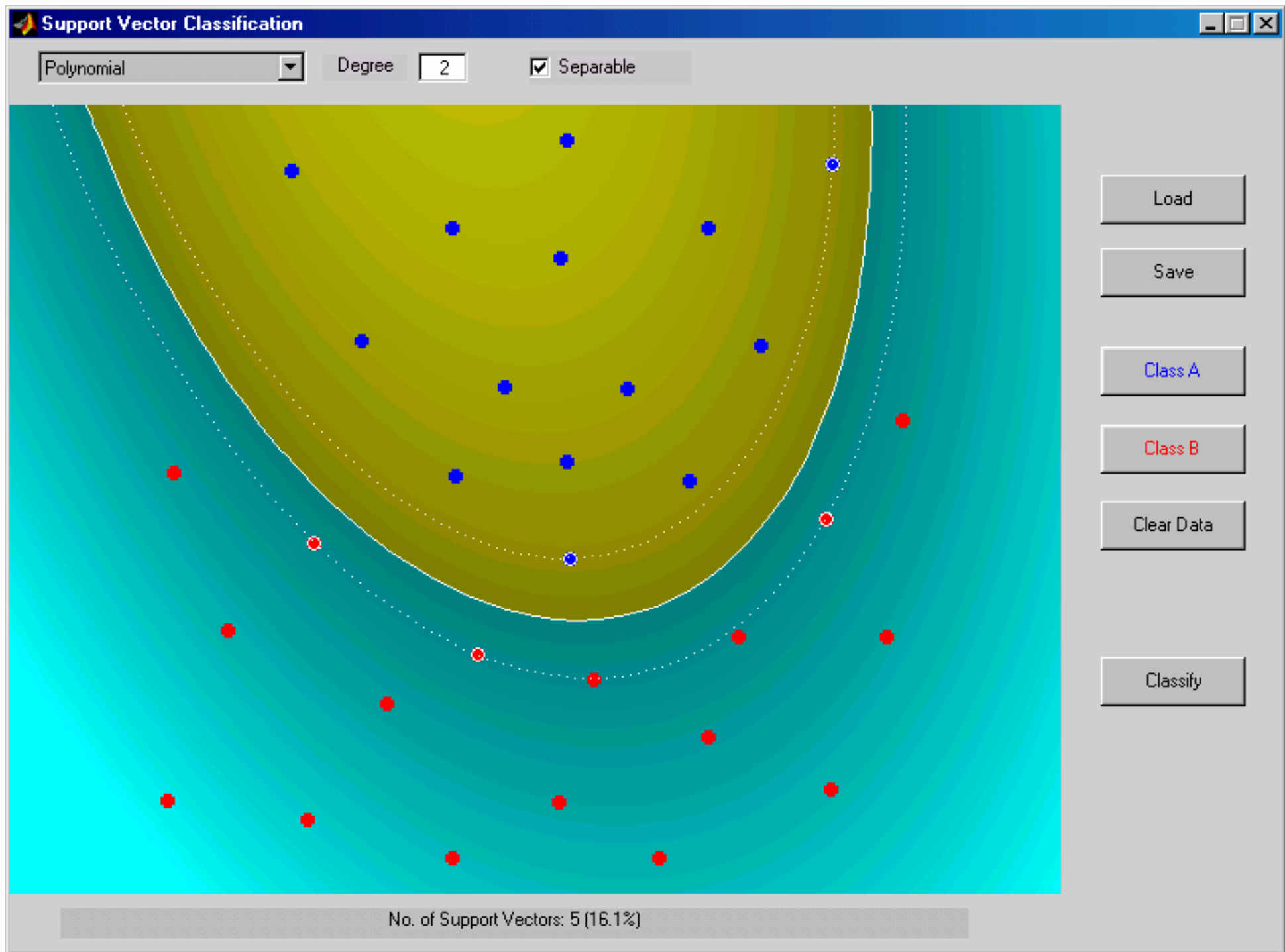
$$f(\mathbf{x}) = \text{sgn}(\langle \mathbf{w}^*, \Psi(\mathbf{x}) \rangle + b^*) = \text{sgn} \left( \sum_{\mathbf{x}_i \in SV} \alpha_i y_i \langle \Psi(\mathbf{x}_i), \Psi(\mathbf{x}) \rangle + b^* \right) \quad (\text{wird so nicht berechnet!})$$

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

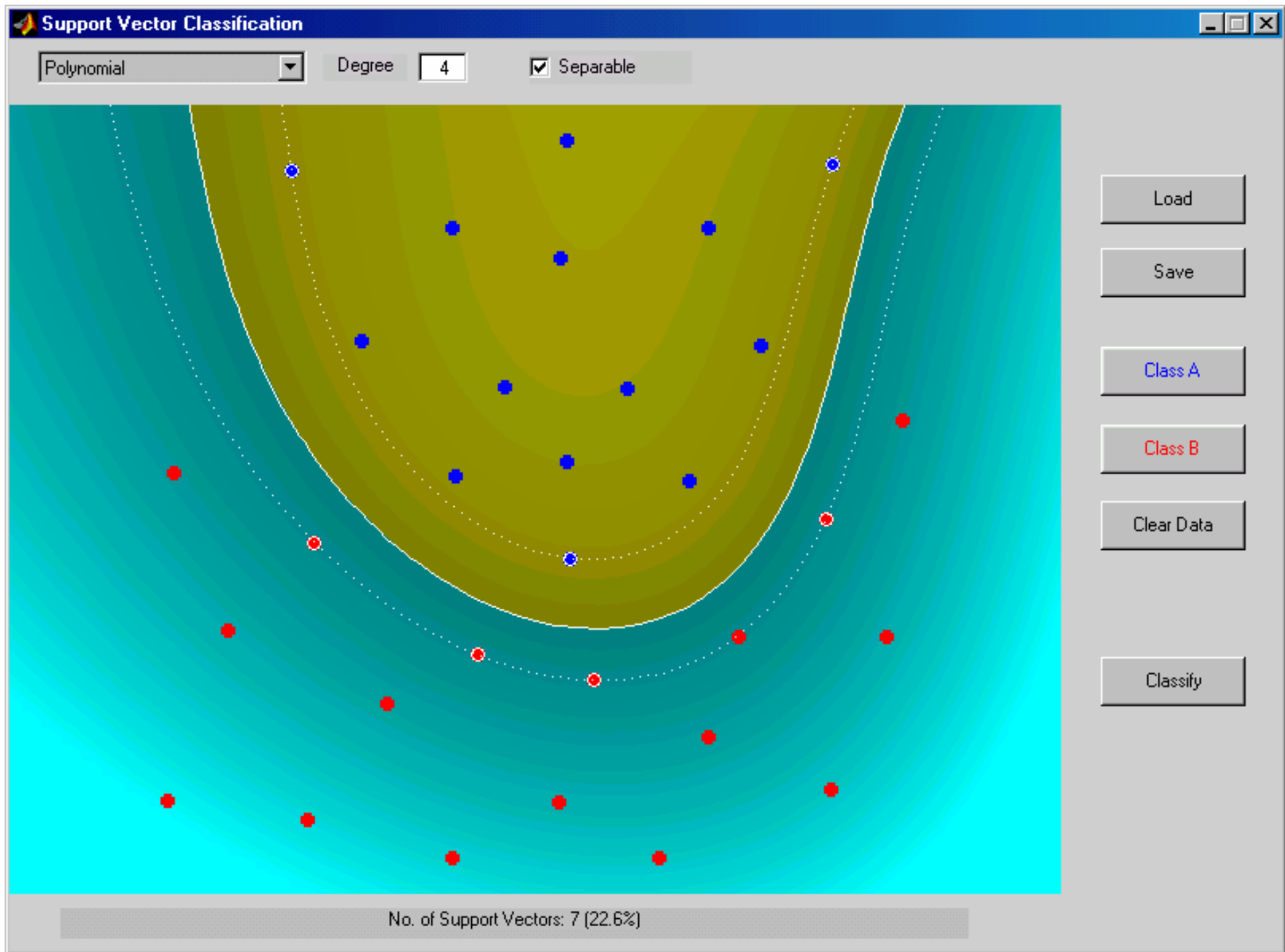
# Lineare Separierung eines quadratischen Problems; weicher Rand



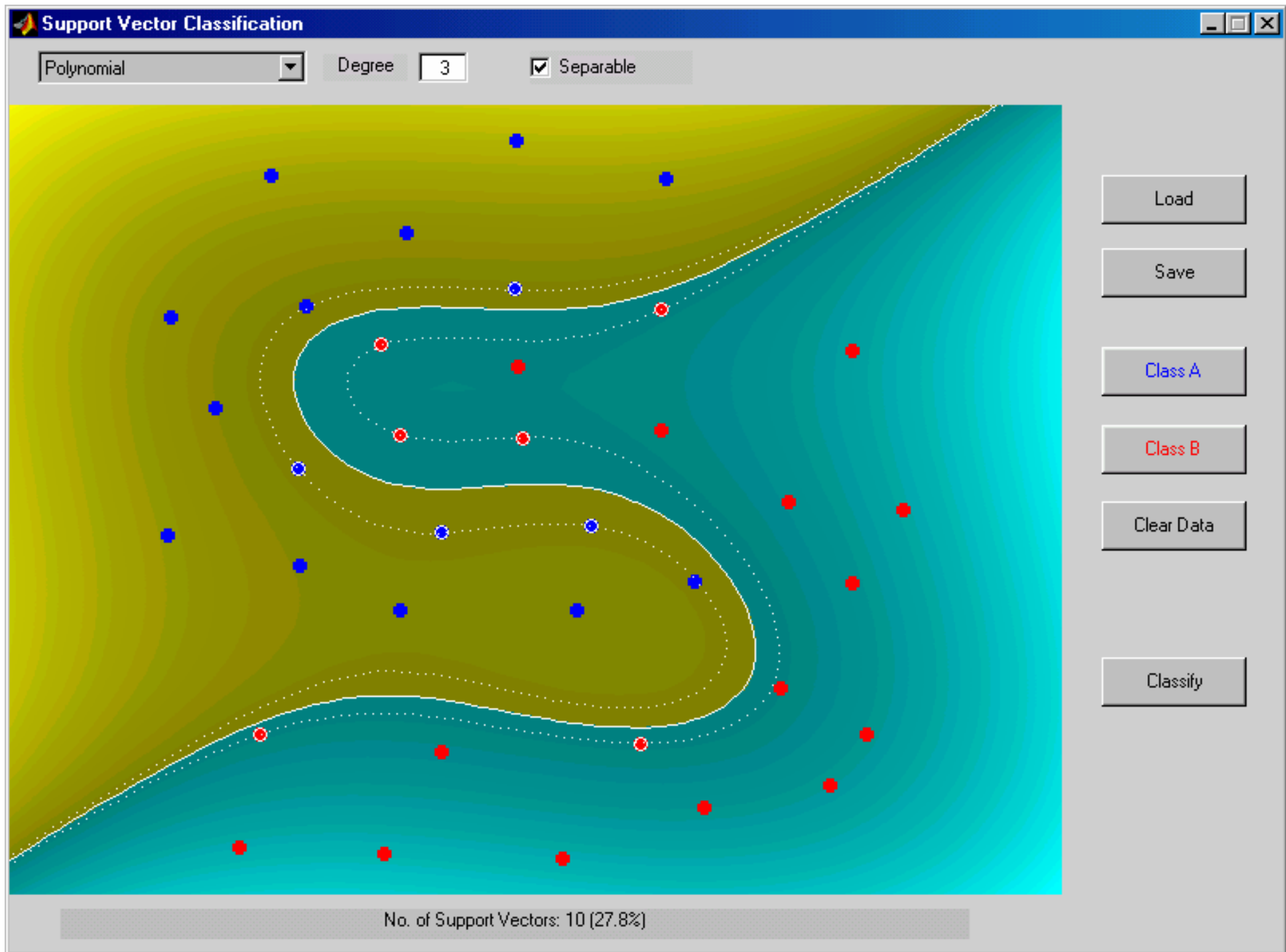
# Polynomiale Separierung eines quadratischen Problems; harter Rand



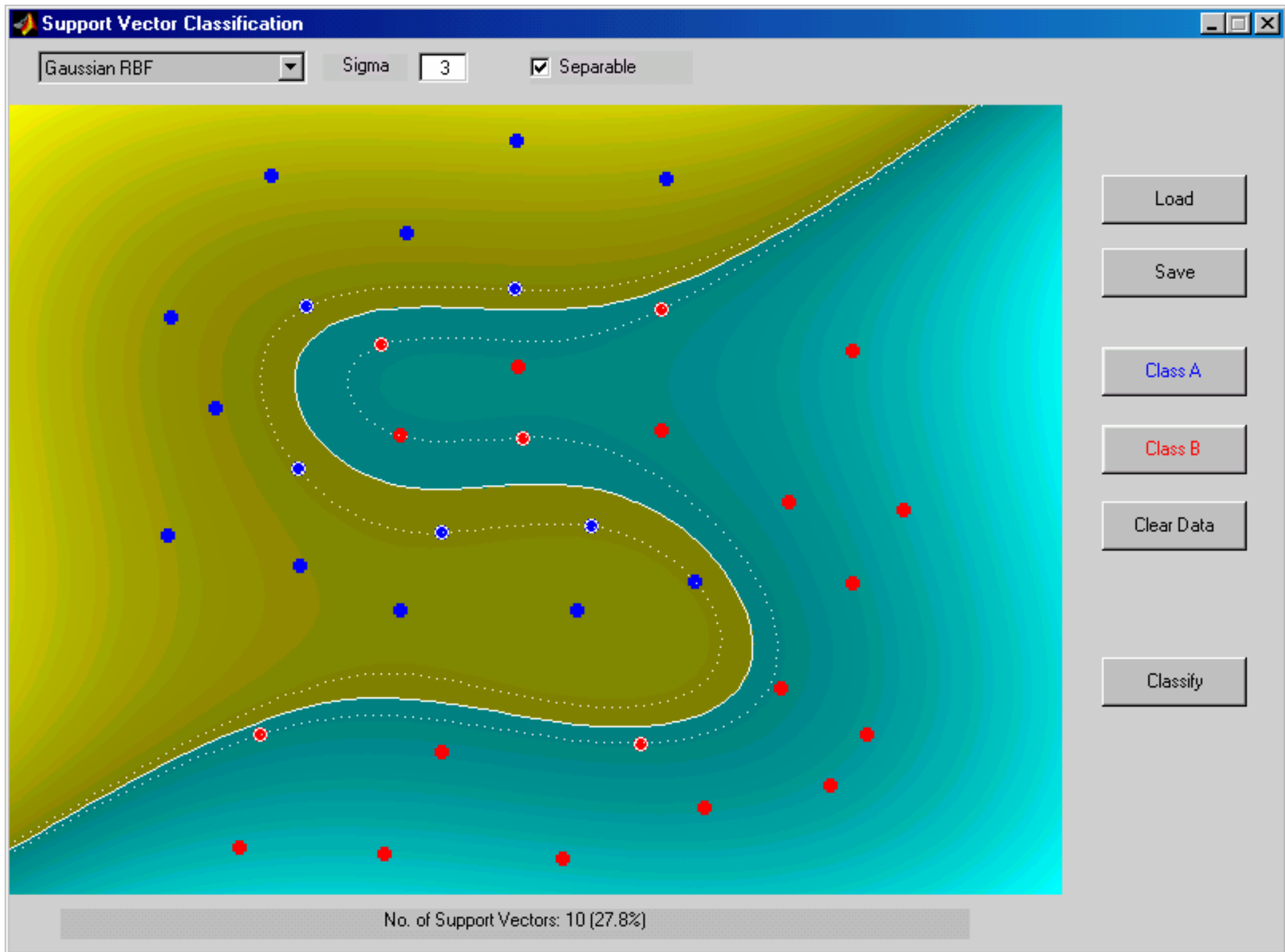
# Polynomiale Separierung eines quadratischen Problems; harter Rand



# Nichtlinear separierbare Klassen; harter Rand

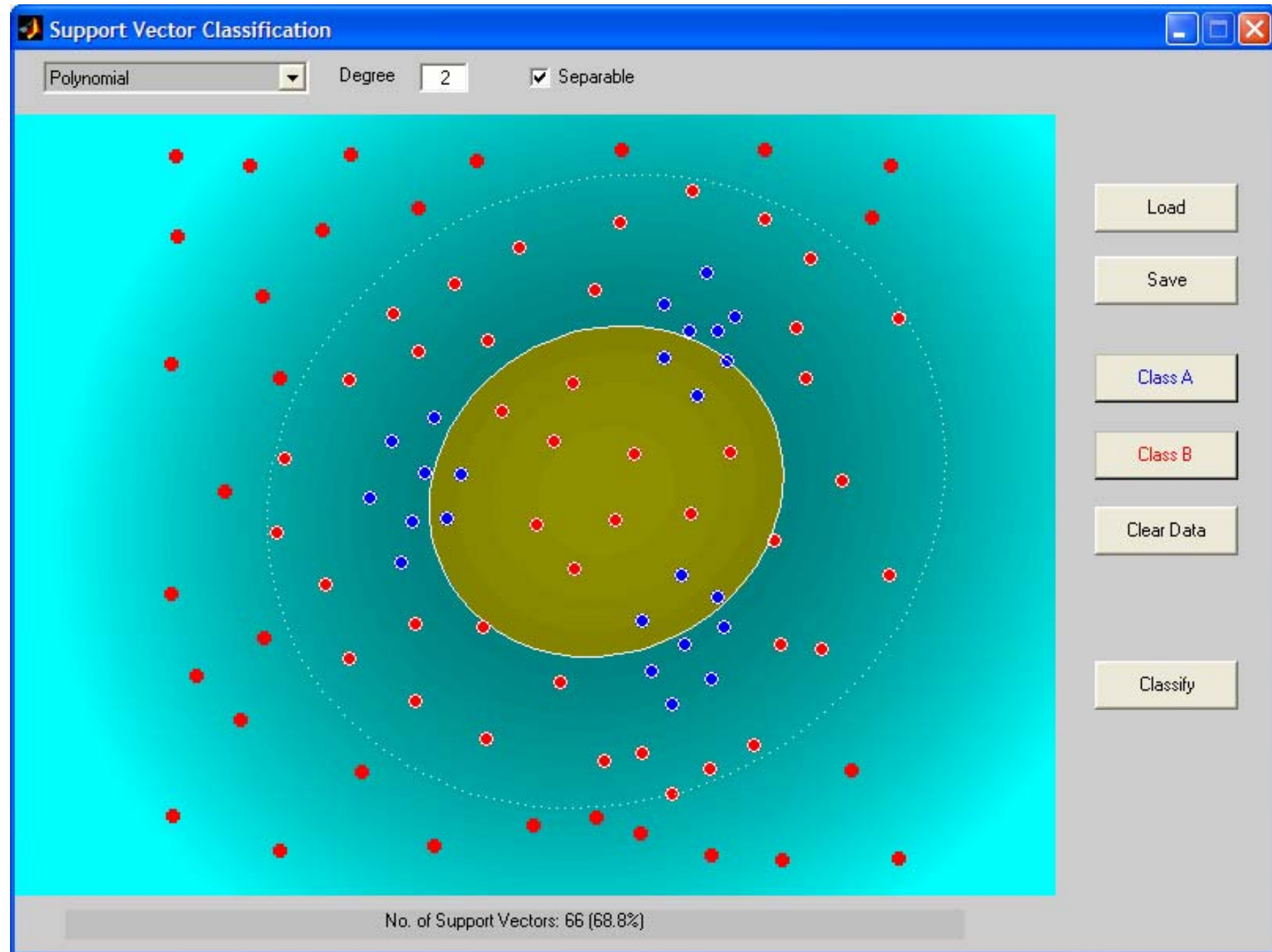


# Nichtlinear separierbare Klassen; harter Rand

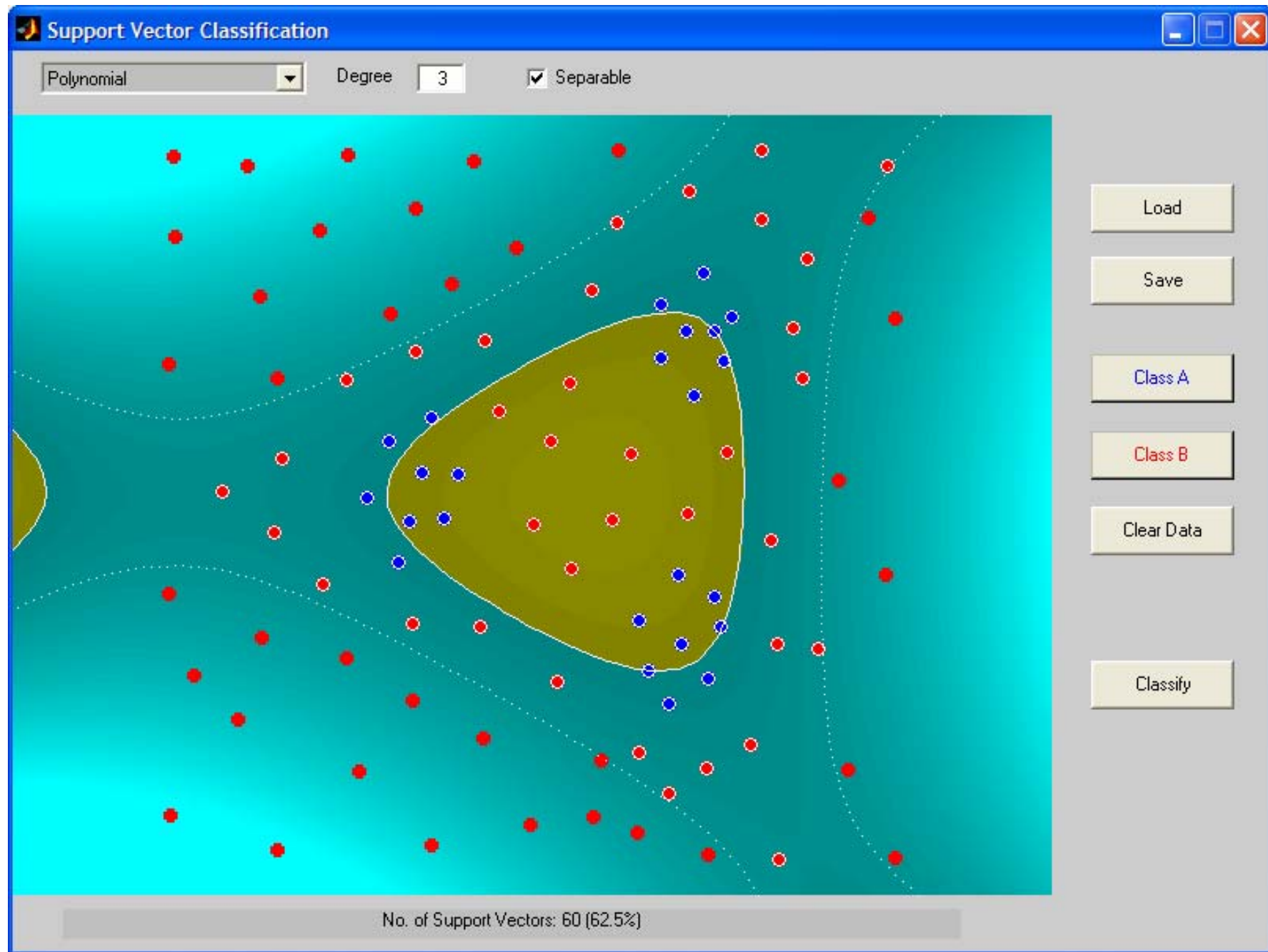




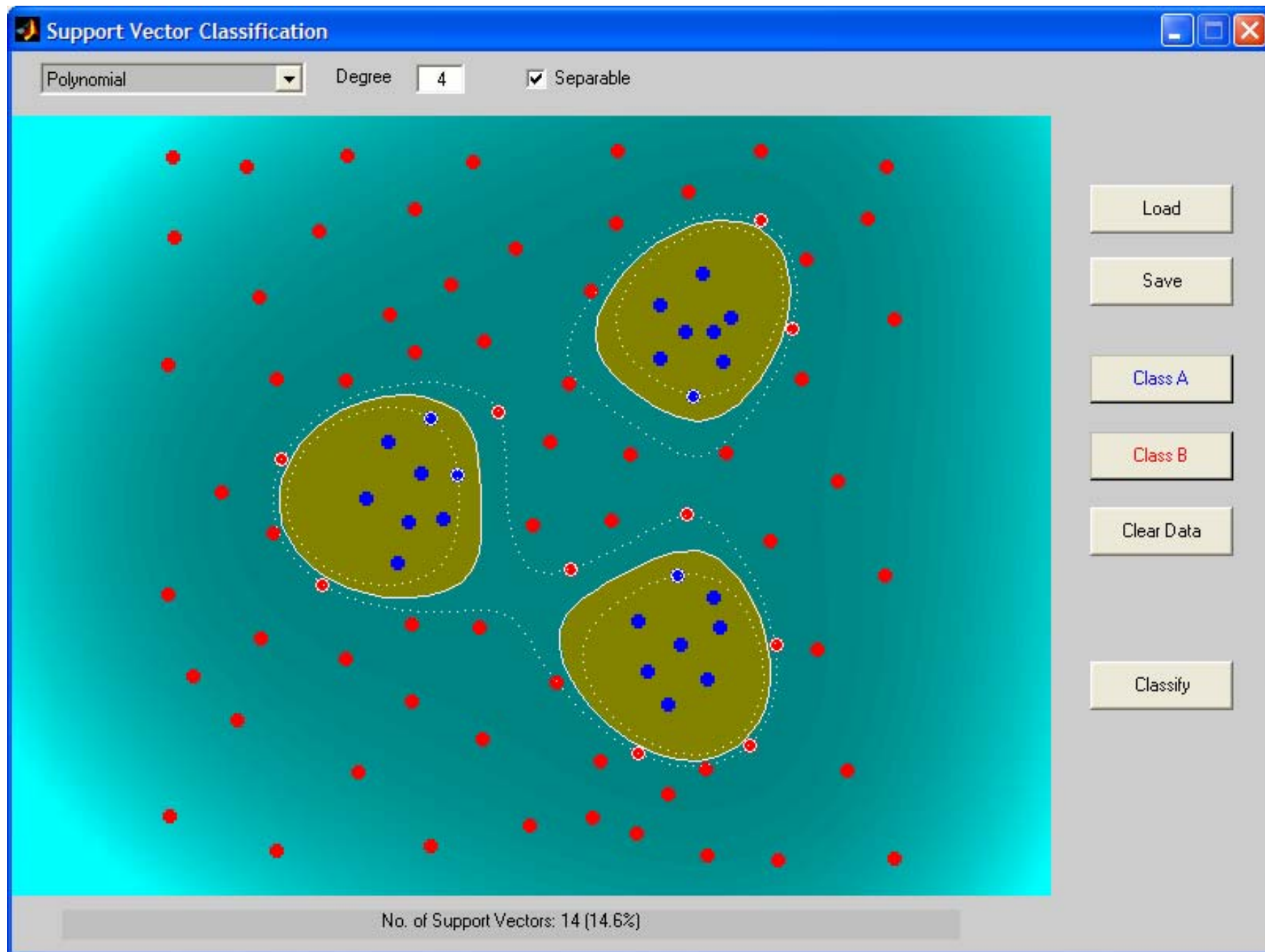
# Beispiel: „Inseln“, Polynom mit $p = 2$



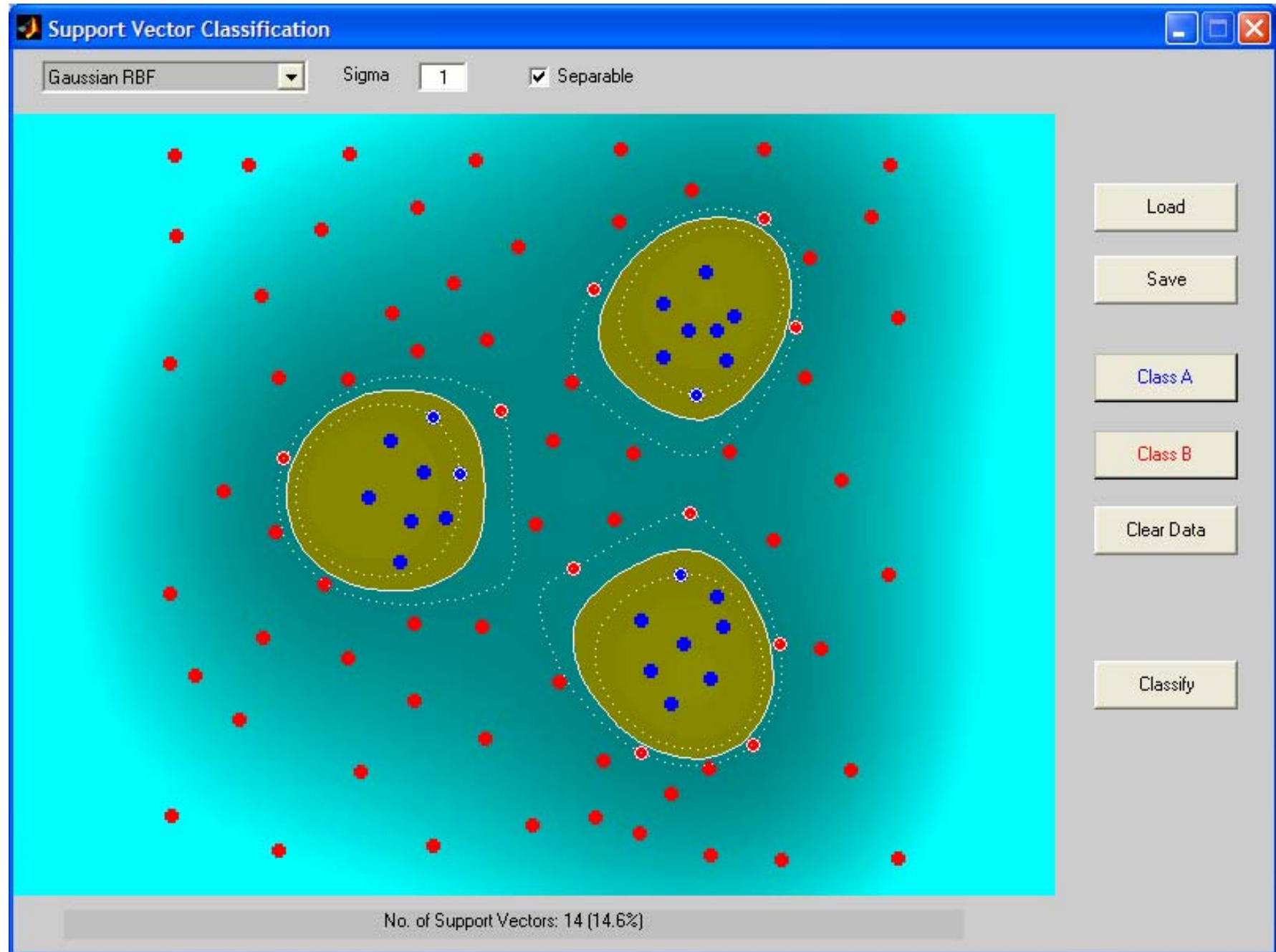
# Beispiel: „Inseln“, Polynom mit $p = 3$



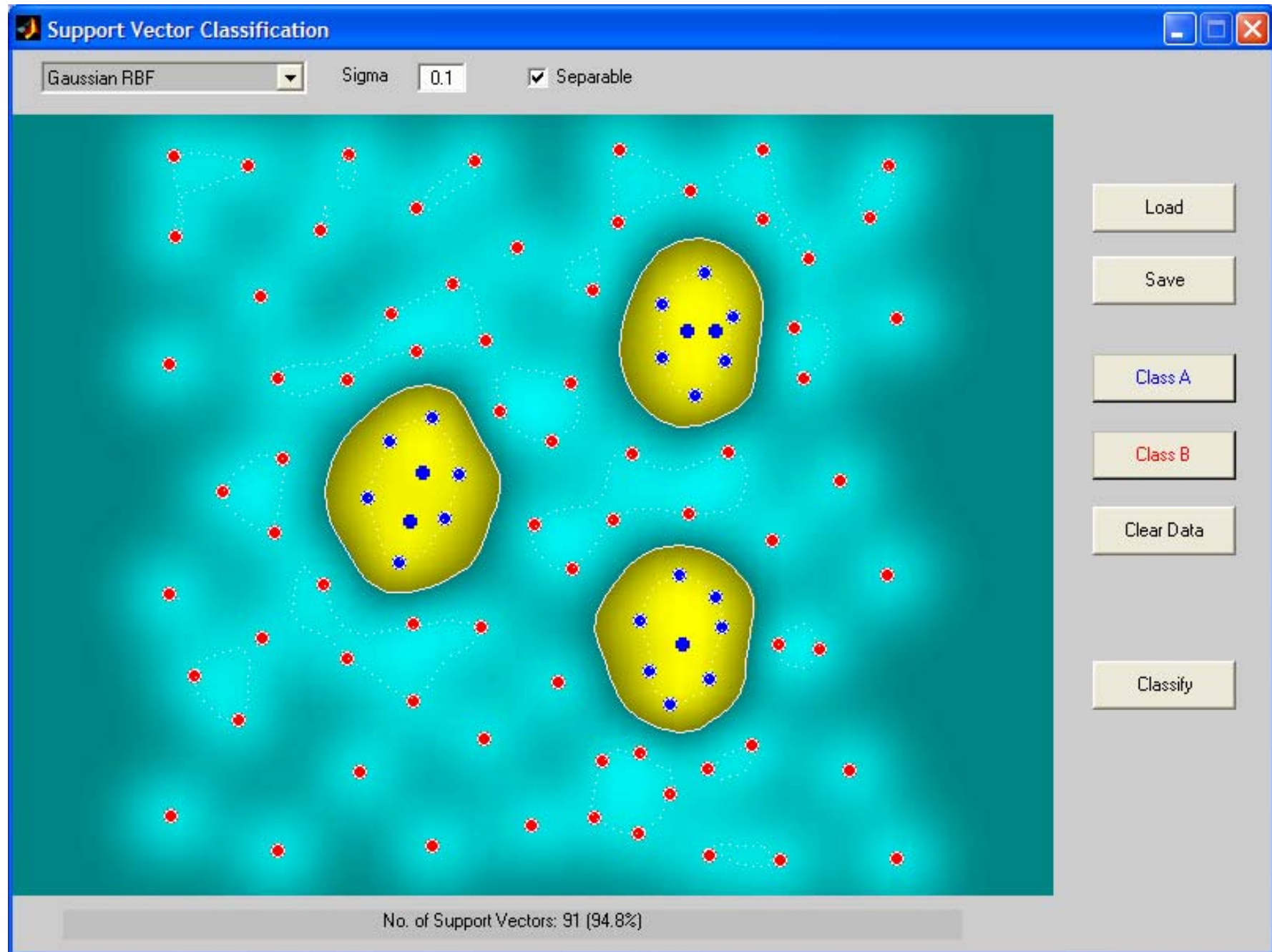
# Beispiel: „Inseln“, Polynom mit $p = 4$



# Beispiel: „Inseln“, RBFs mit $\sigma=1$



Bsp: „Inseln“, RBFs mit  $\sigma=0,1$ ; overfitting! Zu viele SVen, nämlich alle  $x_i$  !



# Visualisierung von Support-Vektoren

(aus Diss. Claus Bahlmann, S. 123)

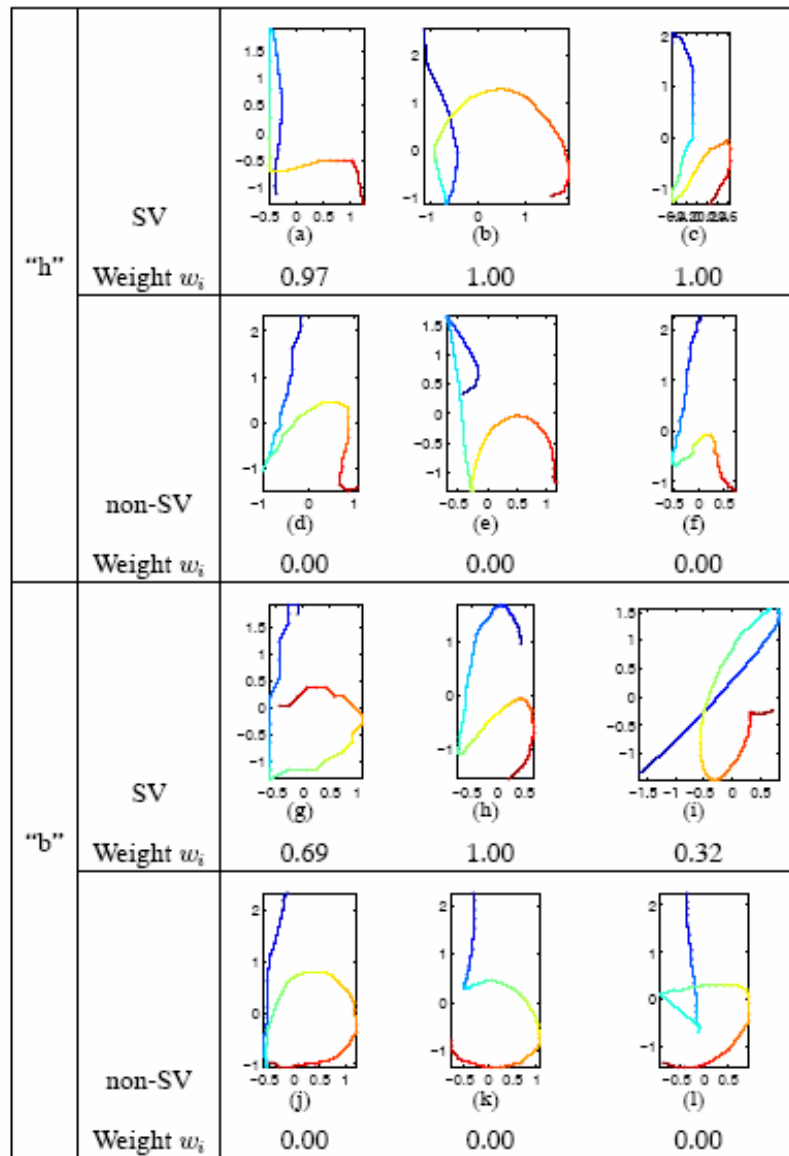


FIGURE 7.5: Support vectors (SV) and non-support vectors (non-SV) of a two-class SVM "h"  $\leftrightarrow$  "b". The patterns are represented through the feature subspace according to the two features  $\hat{x}$  and  $\hat{y}$ . Note that the tangent slope angle  $\theta$  is also used in the SVM but not illustrated here.

# Anwendungsbeispiel: Zeichenerkennung

Beispiel: US Postal Service Digits [Schö95/96]

16x16 Grauwertbilder

7291 Training, 2007 Test-Samples

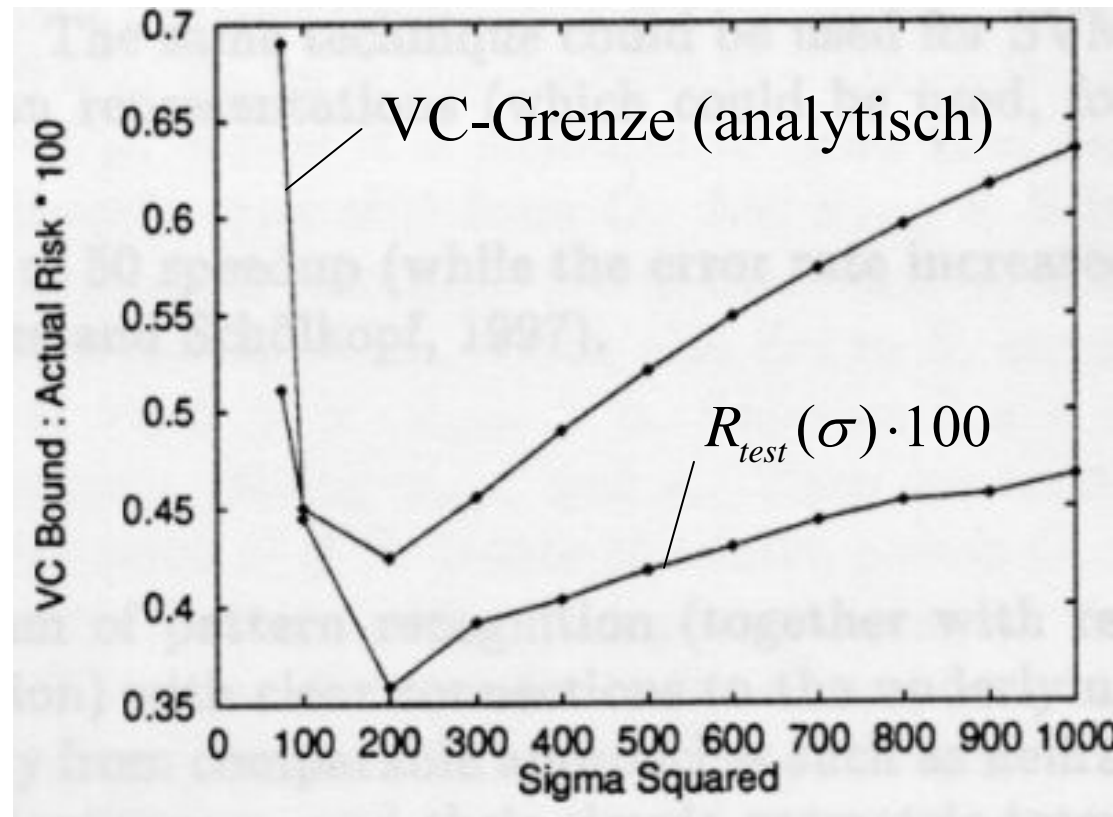
**Ergebnisse:**

<b>Klassifikator</b>	<b>Fehlerrate</b>
Mensch	2.5%
2-Schicht NN	5.9%
5-Schicht NN	5.1%
SVM (Polynom Grad 3)	4.0%
SVM + Invarianz	3.2%

SVM+Invarianz: Ursprüngliche Datenmenge verfünffachen durch Shift der Grauwertmuster in 4 Hauptrichtungen (N,S,O,W) um einen Pixel.

# SVM mit RBF, VC-Grenze im Vergleich zur tatsächlichen Testfehlerrate

(Nur ein Parameter:  $\sigma$ )



- Theorie: leider nur sehr grobe Abschätzungen, aber qualitativ aussagekräftig (Minimum an der gleichen Stelle).
- **Alternative:** Suche nach  $\sigma$  durch Testfehlerminimierung durch **Kreuzvalidierung**. Bildet man z.B. den Erwartungswert über alle leave-one-out Experimente, so erhält man eine Abschätzung des echten Risikos.
- Der Aufwand beim leave-one-out Experiment kann reduziert werden auf die Überprüfung der SV, da nur diese einen Einfluss auf die gemessene Fehlerrate haben.



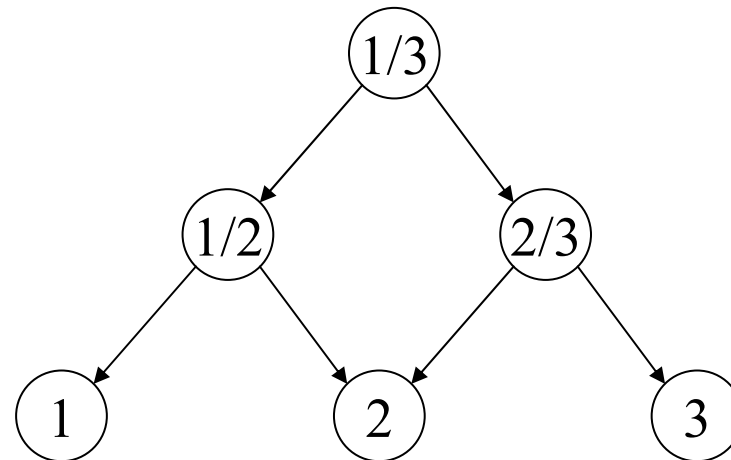
# SVM: Eigenschaften und Berechnungskomplexität

## Stärken:

- Die SVM liefert nach den derzeitigen Erkenntnissen sehr gute Ergebnisse und findet (unter gewissen Voraussetzungen) ein globales Minimum (Bei NN erhält man in der Regel nur suboptimale Lösungen)
- Sparse-Darstellung der Lösung über  $N_s$  Support-Vektoren
- Leicht anwendbar (wenig Parameter  $(C, \sigma)$ , es wird kein a-priori-Wissen benötigt, kein Design); die genaue Wahl von  $(C, \sigma)$  ist jedoch zugleich eine der größten Schwächen im Entwurf
- Geometrisch anschauliche Funktionsweise
- Theoretische Aussagen über Ergebnis: globales Optimum, Generalisierungsfähigkeit
- Structural-Risc-Minim. möglich, wenn auch schwer kontrollierbar
- auch Probleme in großen Merkmalsräumen lösbar (100.000 Trainings- und Testmuster)
- Bei Semidefinitheit des Problems: das duale Problem hat dann viele Lösungen, aber alle liefern die gleiche Hyperebene!

# Schwächen:

- VC-Abschätzung nur sehr ungenau und praktisch wenig verwertbar
- Multiklassenansatz noch Gegenstand der Forschung
  - Ansatz: eine SVM pro Klasse (one versus rest), d.h. Aufwand und Speicherbedarf steigt mit der Anzahl der Klassen
  - Reduktion eines Multiklassenproblems auf eine Reihe von binären Entscheidungsproblemen



# Schwächen:

- Keine quantitative Qualitätsaussage über Klassifikation
- Langsames, speicherintensives Lernen:  
Laufzeit:  $O(N_s^3)$  (Inversion der Hesse-Matrix/Newton-Alg.)  
(in der Praxis jedoch häufig weniger als quadratisch)  
Speicher:  $O(N_s^2)$
- Ansatz zur Verbesserung: Zerlegen in Teilprobleme (Klassifikatorhierarchien)
- Langsames Klassifizieren mit  $O(MN_s)$ , wobei  $M = \dim(\mathcal{H})$  (d.h. im Fall ohne Kernfunktionen  $M=N=\dim(\mathbf{x}_i)$ ), aber bei geeignet gewählten Kernfunktionen gilt auch hier:  $M=N$ .

Hinweis auf LIBSVMTL:

<http://lmb.informatik.uni-freiburg.de/lmbsoft/libsvm/tml/index.en.html>

Objektorientierte C++ Support Vector Machine library (aufbauend auf die LIBSVM von Chih-Jen Lin.

# Literatur:

- (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.