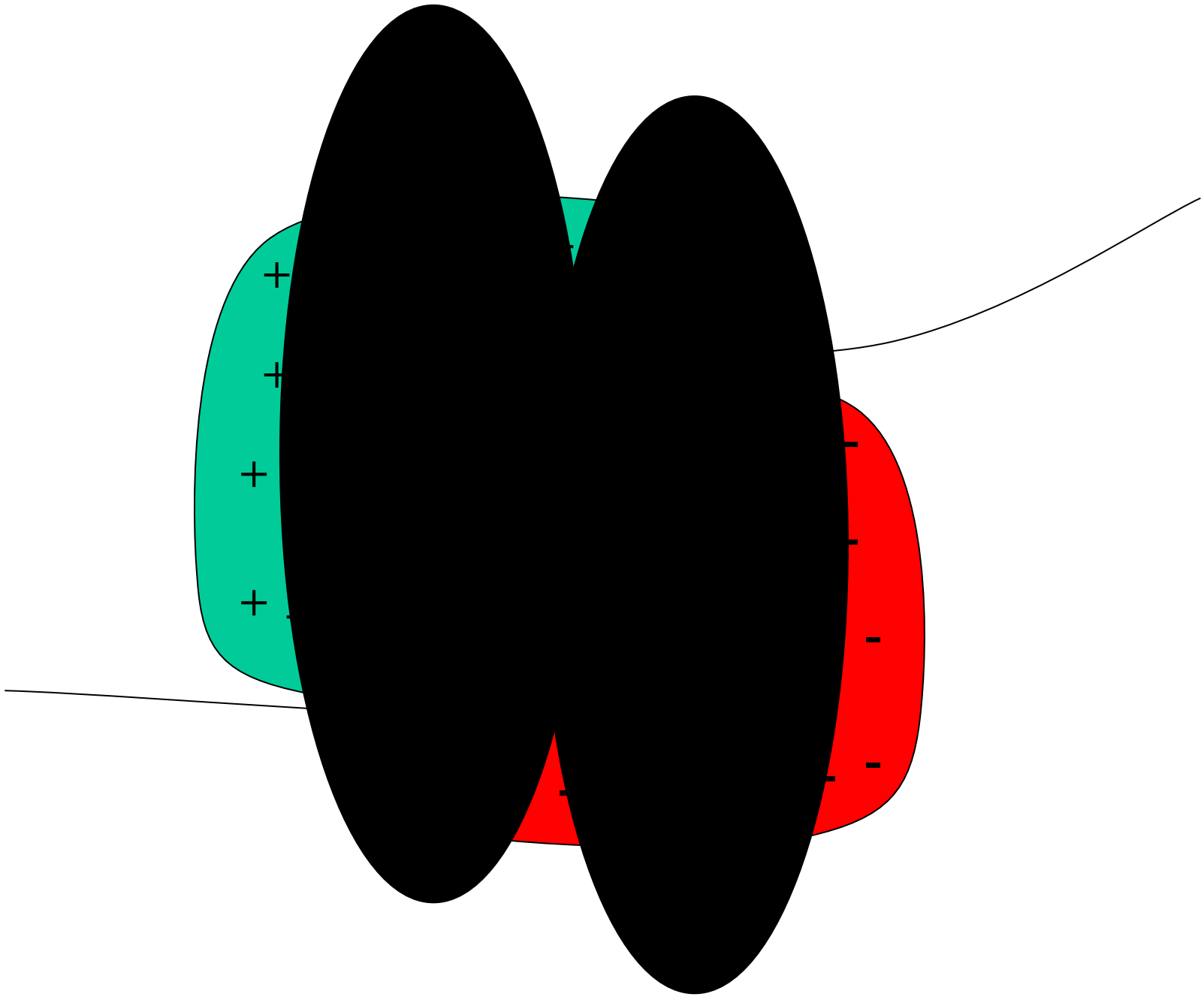
Kapitel 10
Die Support-Vektor-Maschine (SVM)

Ein statistischer Ansatz der Lerntheorie zum Entwurf eines optimalen Klassifikators

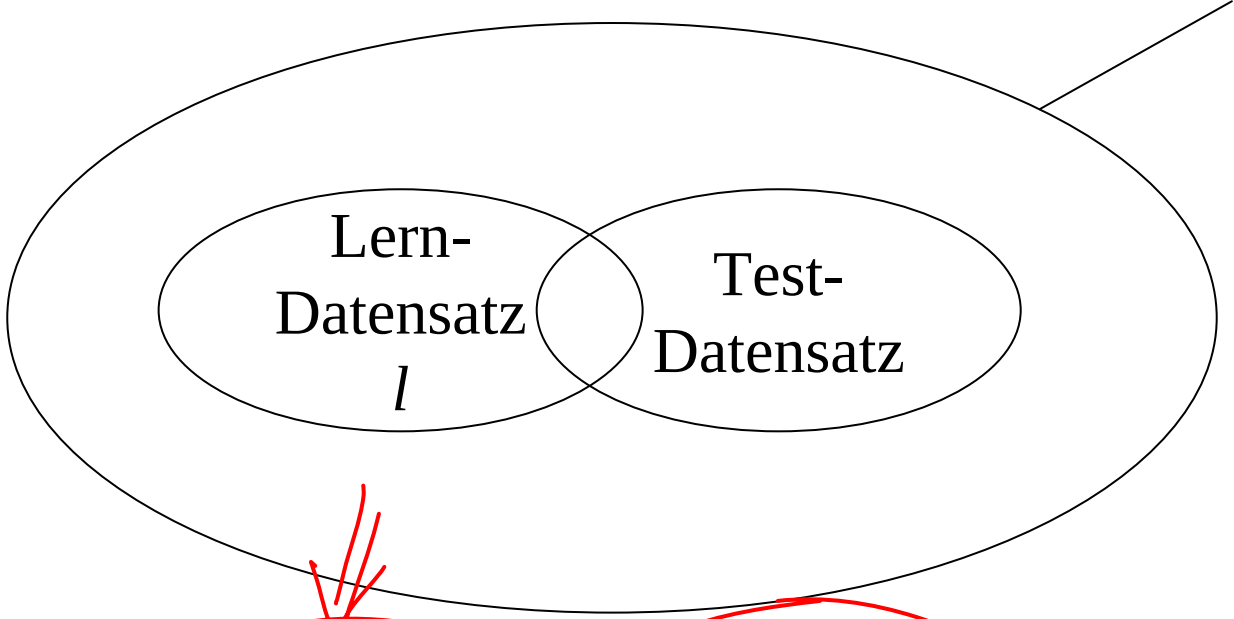
Inhalt:

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. Vorstellung von Forschungsprojekten:
   - Claus Bahlmann: Handschrifterkennung
   - Olaf Ronneberger: Autom. Erkennung von Blütenpollen
   - Bernard Haasdonk: Tangentendistanz und SVM

Durch Normalverteilungen schlecht darstellbare Klassen



Mustererkennungsexperiment



Stochastischer Prozess mit *unbekannter* Verbundverteilung:
$P(\mathbf{x},y)$

Gesucht ist eine Zuordnungsfunktion
$f(\mathbf{x},\alpha)$: $\mathbf{x}_i \rightarrow y_i$
welche durch die unbekannten Parameter des Vektors α charakterisiert wird, welche den Erwartungswert des Zuordnungsfehlers minimiert:

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

Empirischer Fehler

Test-Fehler (Generalisierungs-fähigkeit)

$R_{emp}(\alpha)$ $R_{test}(\alpha)$

Lerntheoretischer Ansatz

Überwachtes Lernen:

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

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

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:

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

$$= \int \tfrac{1}{2}\left|y - f(\mathbf{x},\boldsymbol{\alpha})\right| p(\mathbf{x},y)\,d\mathbf{x}dy \quad \text{falls Verbundverteilungsdichte bekannt}$$

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:

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

- Der Ansatz mit einem Neuronalen Netz und dem Backpropagation Learning z.Bsp. begnügt sich mit einer **E**mpirischen **R**isiko **M**inimierung (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 (**S**tructural **R**isc **M**inimization (SRM) anstatt Empirical Risc 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-η) gilt die folgende obere Abschätzung für den tatsächlichen Fehler (d.h. z.Bsp. mit η = 0,05 und damit mit einer Wahrscheinlichkeit von 95%):

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

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

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

Bemerkenswert ist, dass dieser Ausdruck unabhängig ist von der zugrundeliegenden Verbundverteilung P(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(**x**,α)), entscheidet man sich für die Maschine, welche den geringsten Wert bei gegebener VC-Dimension für die Konfidenz Φ liefert.

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

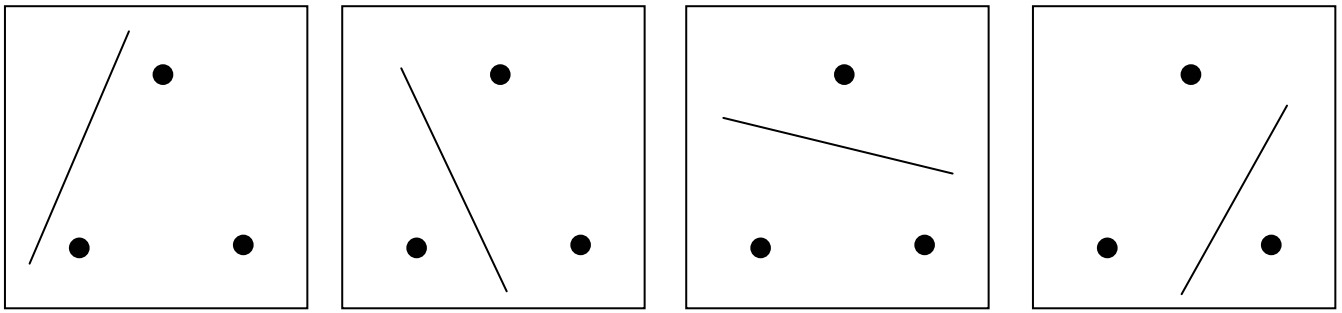
- Eine gegebene Menge von l Punkten kann für den Zweiklassen-Fall in 2^l mögliche Klassen aufgeteilt werden. Die VC-Dimension einer Menge von Funktionen $\{f(\alpha)\}$, ist definiert als die maximale Anzahl von Trainingspunkten, welche durch diese Klasse von Funktionen in allen möglichen Zuordnungen separiert werden können.

- Liefert Maße für „Kapazität“ von Funktionenklassen
  - Funktionenklassen z.B.
    - Menge der linearen Klassifikatoren
    - Menge der Klassifikatoren, die ein NN realisieren kann
    - Menge der Klassifikatoren, die eine SVM realisieren kann

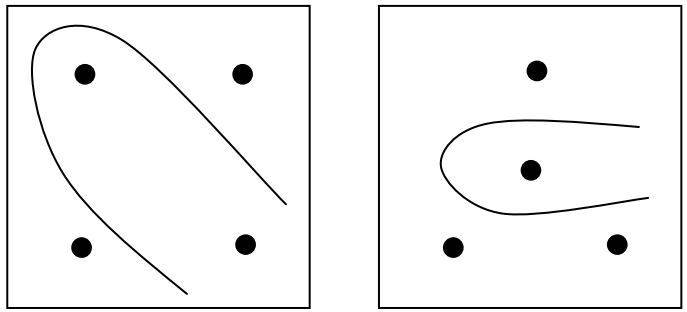
- Die VC-Konfidenz wächst monoton mit h: Demnach würde man bei einer Auswahl von Lernmaschinen, deren empirisches Risiko 0 ist, sich für diejenige entscheiden, deren assoziierte Menge von Abbildungsfunktionen minimale VC-Dimension besitzt.

VC-Dimension von Hyperebenen in $\mathbb{R}^2$

- 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



  - => Hyperebenen in $\mathbb{R}^2$: VCdim=3
  - Allgemein: Hyperebenen in $\mathbb{R}^N$: VCdim=N+1
  - die VC-Dimension bei Polynomen wächst mit ihrem Grad!

VC-Dimension von SVM

– Aussagen über VC-Dimensionen bei SVM möglich!
  - VC-Dimension der Menge der Hyperebenen in $\mathbb{R}^M$

$$h \leq \min(R^2 A^2, M) + 1$$

$$\|w\| \leq A, \;\; K(x_i, x_i) \leq R^2$$

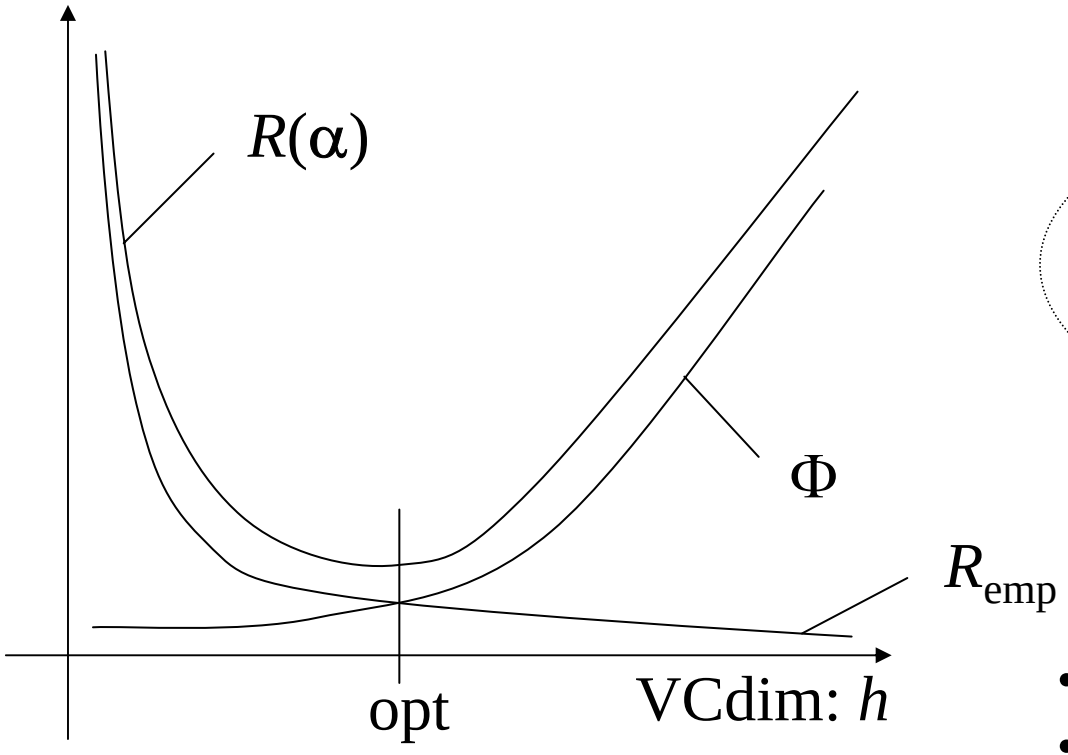
  - SVM mit Polynomkernen $K(x,y)=(x \cdot y)^p$

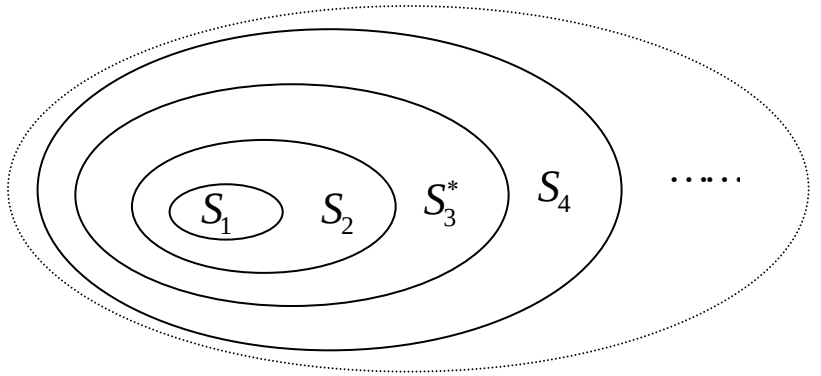
$$h = \binom{N + p - 1}{p} + 1$$

– 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 ($h_1<h_2<h_3<...$).
- Kompromiss zwischen empirischem Risiko und Generalisierungsfähigkeit





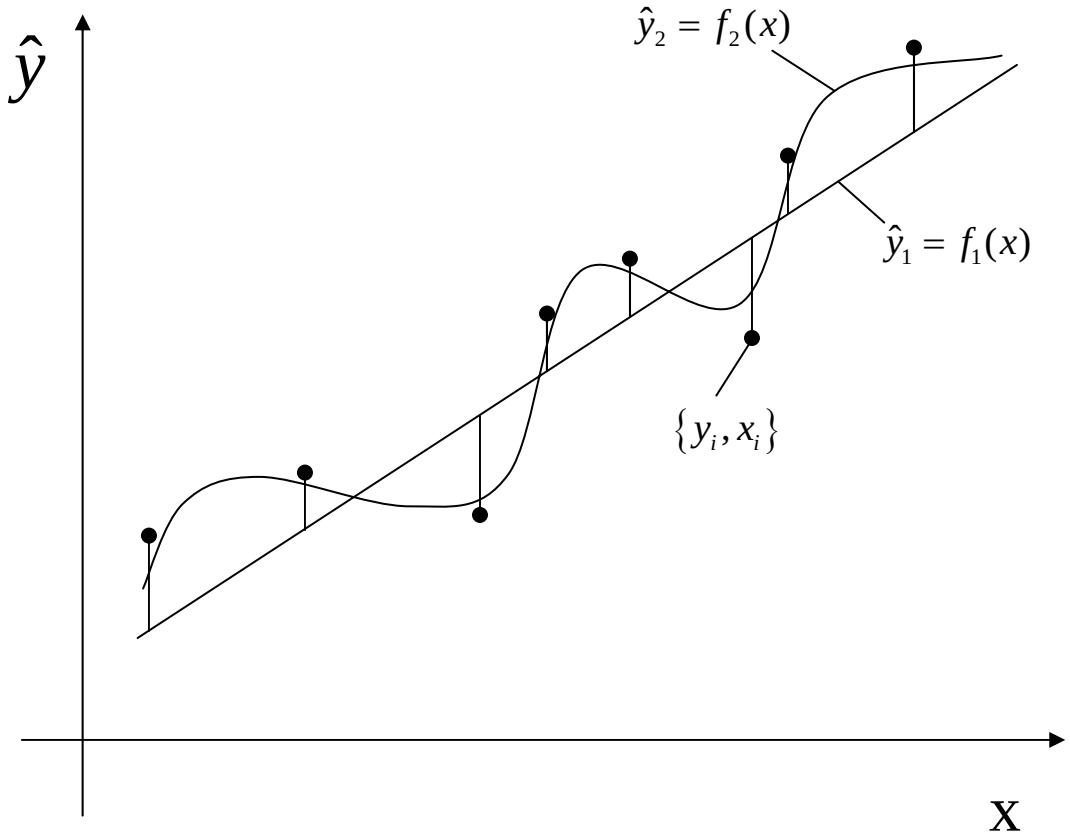
- Φ wächst an mit h
- R_{emp} wird kleiner mit h (overfitting)

Entwurfshinweise

Mehrere Ausdrücke für das gleiche Phänomen:

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

Bias-Varianz-Tradeoff



f_1 :

- Bias $\quad \frac{1}{n}\sum_{i=1}^{n}(y_i - f_1(x_i))$ klein
- Varianz $\quad \frac{1}{n}\sum_{i=1}^{n}(y_i - f_1(x_i))^2$ groß

gute Generalisierung (einfaches Modell)!

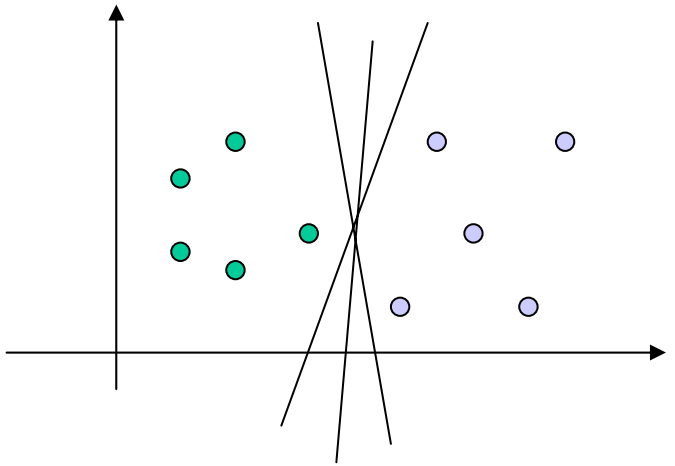
f_2 :

- Bias $\quad \frac{1}{n}\sum_{i=1}^{n}(y_i - f_2(x_i))$ groß
- Varianz $\quad \frac{1}{n}\sum_{i=1}^{n}(y_i - f_2(x_i))^2$ klein

schlechte Generalisierung (overfitting)!

Linear separierbare Klassen

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



  - Allgemeine Hyperebene $\langle \mathbf{w}, \mathbf{x} \rangle + b = 0$
  - Klassifikation via $f(x) = \mathrm{sgn}(\langle \mathbf{w}, \mathbf{x} \rangle + b)$
  - z.B. Rosenblatt´s Perceptron (1956)

    Iteratives Lernen, Korrektur nach jeder Fehlklassifikation

    ->keine Eindeutigkeit der Lösung

Optimale Hyperebene: Maximierung des Randes

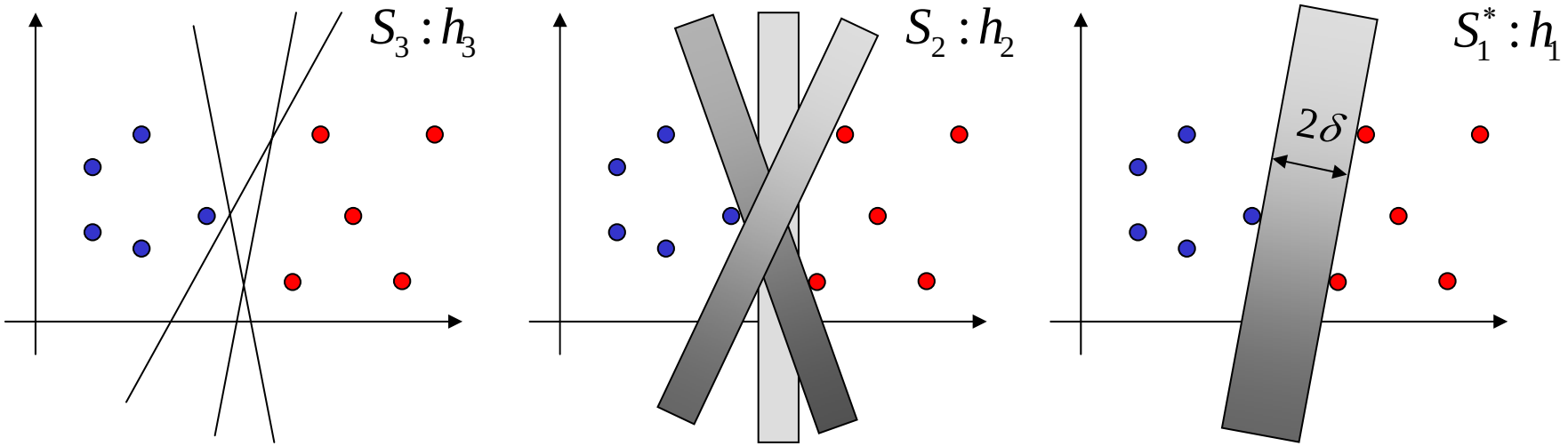
- Geht man von einem linear separierbarem Zweiklassenproblem aus, so erreichen alle Geraden, welche beide Klassen trennen einen empirischen Fehler Null
- Die Konfidenz wird minimiert durch ein Polynom minimaler VC-Dimension, nämlich eine Hyperebene
- Die VC-Dimension kann weiter abgesenkt werden durch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

• Trennende Hyperebene mit maximalem Rand		• Trennende Hyperebene mit minimaler VC-Dimension

- Dieses Ergebis ist plausibel. Bei konstanter Intraklassenstreuung, wächst die Klassifikationssicherheit mit wachsendem Interklassenabstand.
- oder: bei konstant gehaltenem Interklassenabstand (z.Bsp. 0) muss die Intraklassenstreung maxmial zulässig werden

“Large-margin“-Klassifikator

– Hyperebene mit größtem Rand [Vap63]



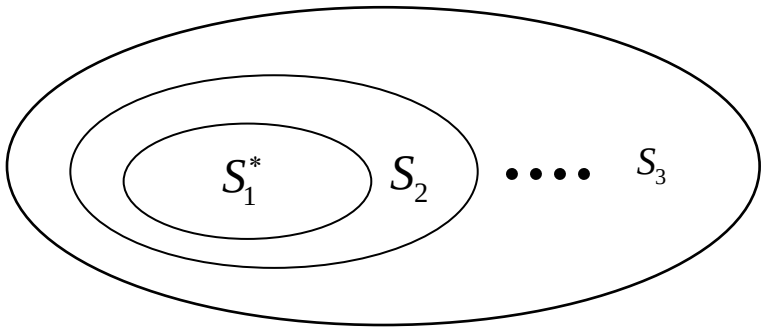
hohe VC-Dimension

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

mittlere VC-Dimension
Variabilität wird kleiner!

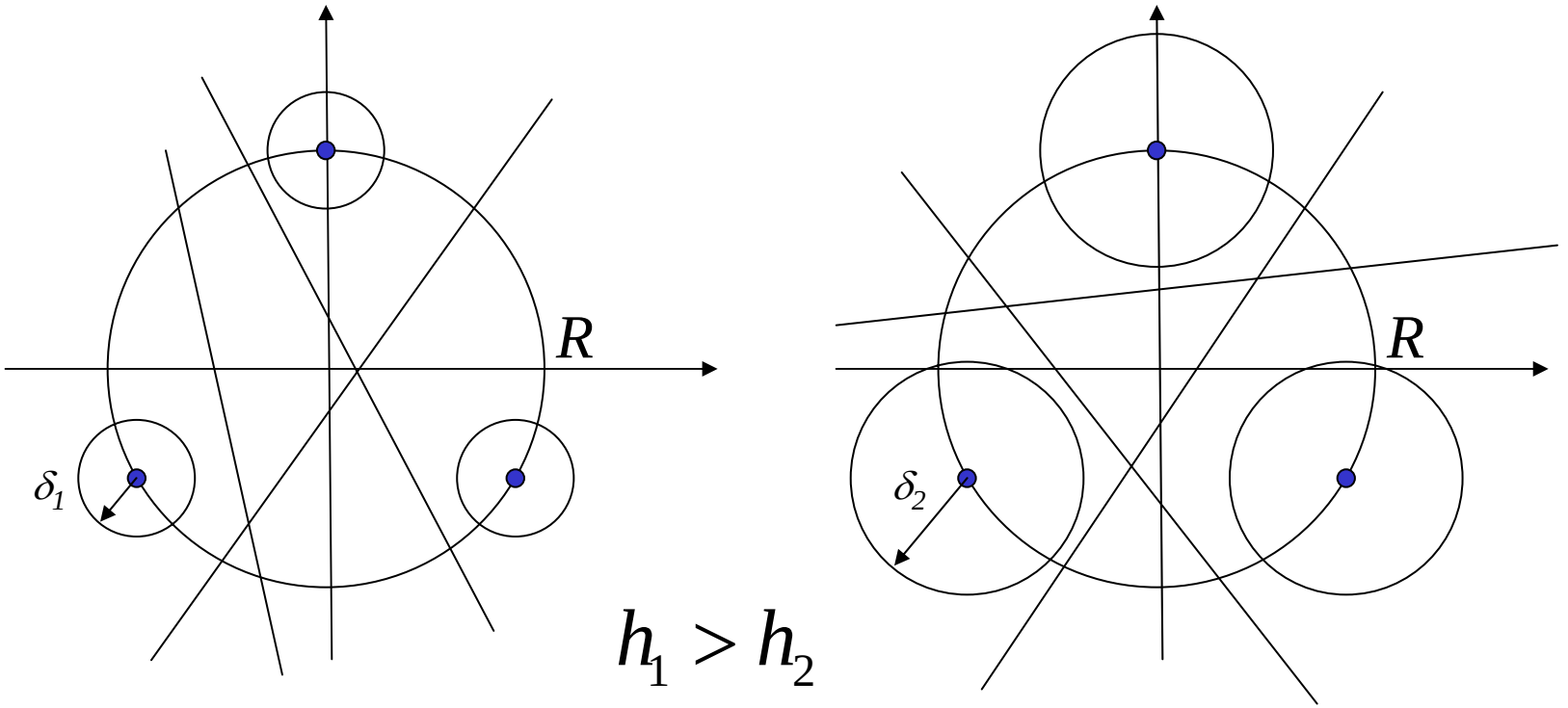
kleinste VC-Dimension
mit maximaler Breite
Variabilität gleich Null

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



$h_1 < h_2 < \ldots < h_3$

VC-Dimension von „breiten“ Hyperebenen



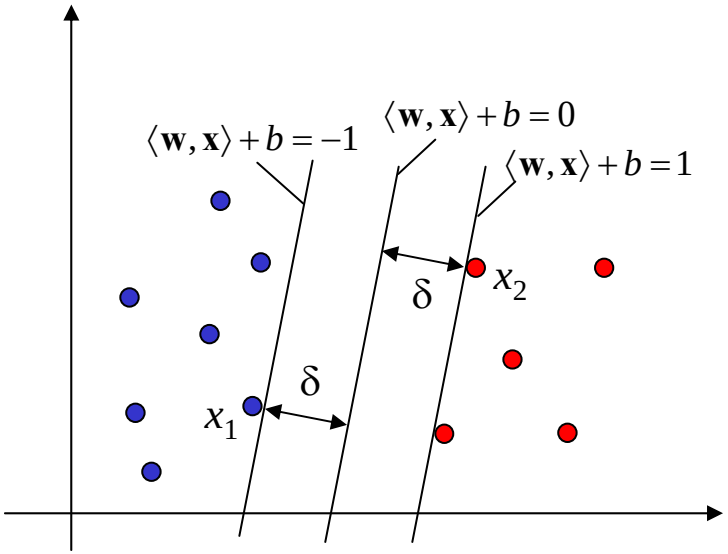
$$h_1 > h_2$$

Die VC-Dimension *h* von Hyperebenen mit einem Mindestabstand δ von den zu trennenden Punkten ist begrenzt durch:

$$VC\dim \leq \frac{R^2}{\delta^2} + 1$$

=> grosser Margin, kleine VCdim

Formalisierung



Die Daten werden korrekt klassifiziert, falls:

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

Dieser Ausdruck ist invariant gegenüber einer positiven Reskalierung:

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

Einführung von kanonischen Hyperebenen:

$$\begin{cases} \langle \mathbf{w}, \mathbf{x}_1 \rangle + b = -1 & \text{für die blaue Klasse} \\ \langle \mathbf{w}, \mathbf{x}_2 \rangle + b = +1 & \text{für die rote Klasse} \end{cases}$$

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

$$\frac{+(\langle \mathbf{w}, \mathbf{x}_1 \rangle + b = +1) \quad -(\langle \mathbf{w}, \mathbf{x}_2 \rangle + b = -1)}{\langle \mathbf{w}, (\mathbf{x}_1 - \mathbf{x}_2) \rangle = 2}$$

$$\Rightarrow \quad \underbrace{\langle \frac{\mathbf{w}}{\|\mathbf{w}\|}, (\mathbf{x}_1 - \mathbf{x}_2) \rangle}_{2\delta} = \frac{2}{\|\mathbf{w}\|}$$

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

Die Maximierung von δ ist gleichwertig mit einer Minierung von $\|\mathbf{w}\|^2$ =>

Optimierungsproblem

Primales OP:

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

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

Einführung einer Lagrangefunktion:

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

mit: $\alpha_i \geq 0$

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

Wolf-duale OP:

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

$$\text{unter der N.B.:} \quad \alpha_i \geq 0 \ \text{ und } \ \sum_{i=1}^{l} y_i \alpha_i = 0$$

Dies ist ein positiv semidefinites Problem, welches mit Hilfe der konvexen quadratischen Programmierung numerisch iterativ gelöst werden kann!

Lösung:

– Lösung des dualen Problems liefert eindeutig gewünschte Hyperebene

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

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

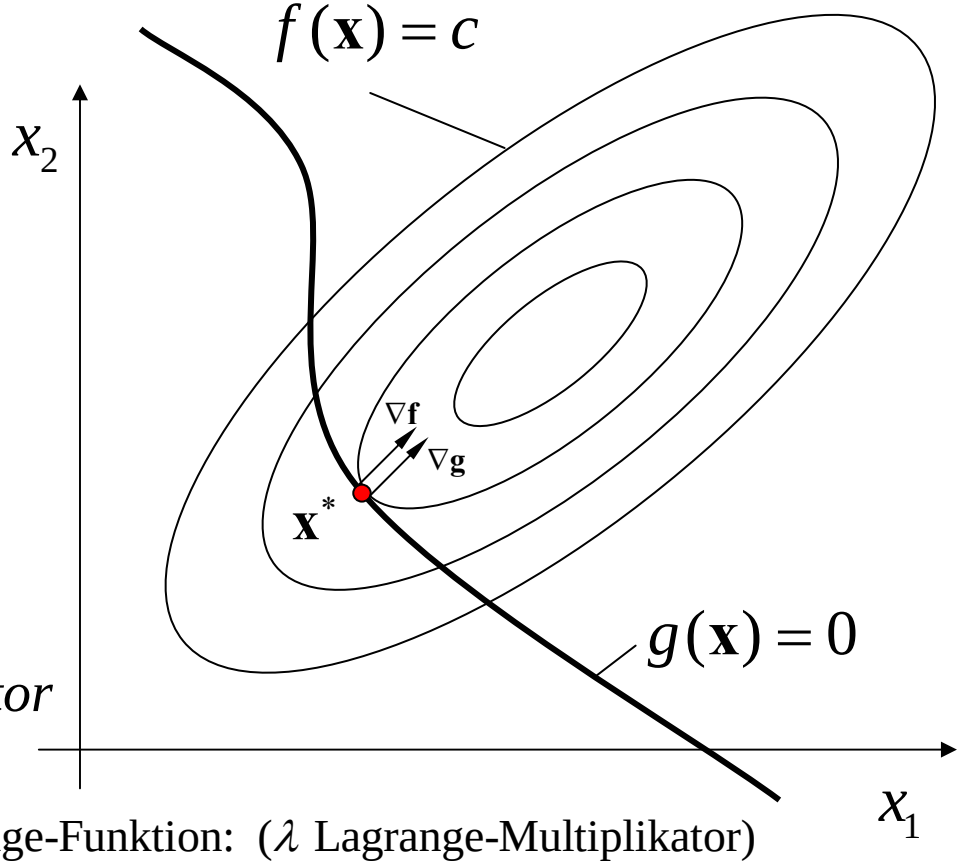
Klassifikation: $f(\mathbf{x}) = \text{sgn}(\langle \mathbf{w}^*, \mathbf{x} \rangle + b^*) = \text{sgn}(\sum_{\mathbf{x}_i \in SV} \alpha_i y_i \mathbf{x}_i) + b^*)$

Lösung nur abhängig von den Supportvektoren !!

Beobachtungen:

- Für die Support-Vektoren gilt: $0<\alpha_i<\infty$
- Für alle Beispiele ausserhalb des Randes ist $\alpha_i=0$
  -> 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



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

Gradient = *Normalenvektor*

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

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

Eine Lösung für das Optimierungsproblem zu finden ist gleichbedeutend mit der Aufgabe, stationäre Punkte x* zu finden, in denen gilt:

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

Aus der Lagrange-Funktion: (λ Lagrange-Multiplikator)

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

folgt mit der notwendigen Bedingung für ein Extremum:

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

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

Dies ist nur eine notwendige Bedingung für ein lokales Optimum; es kann mehrere Lösungen geben!

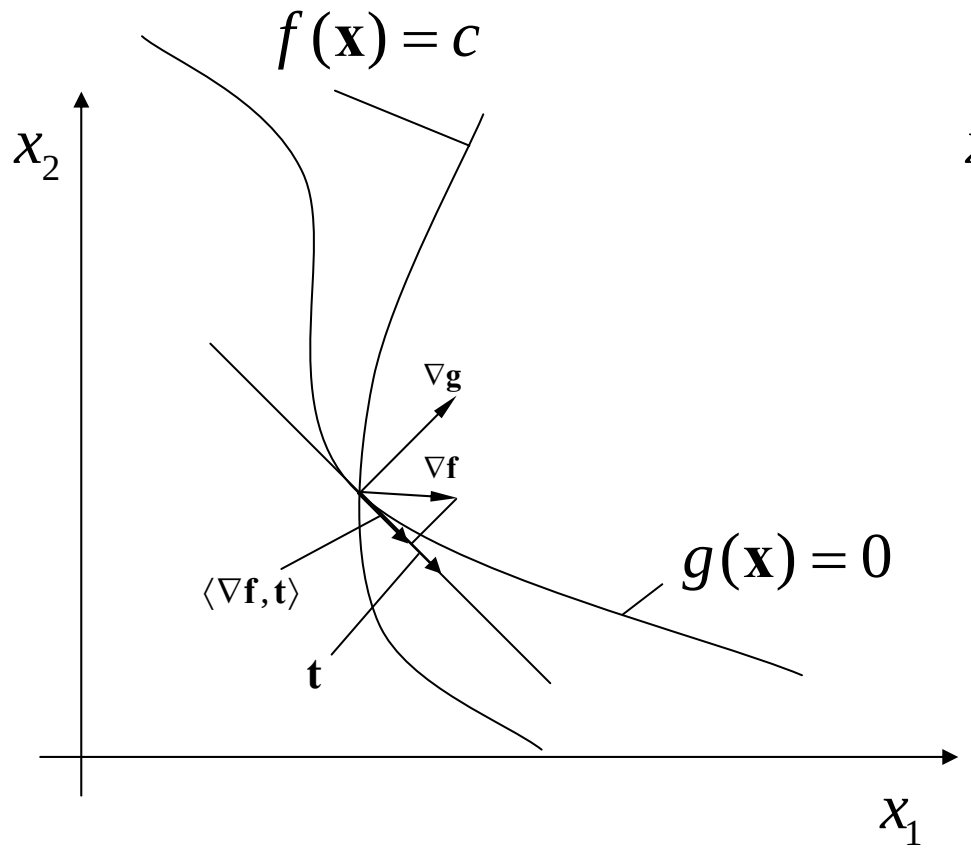
Der Gradient ist ein Normalenvektor an die Kurve $g(\mathbf{x})=0$

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

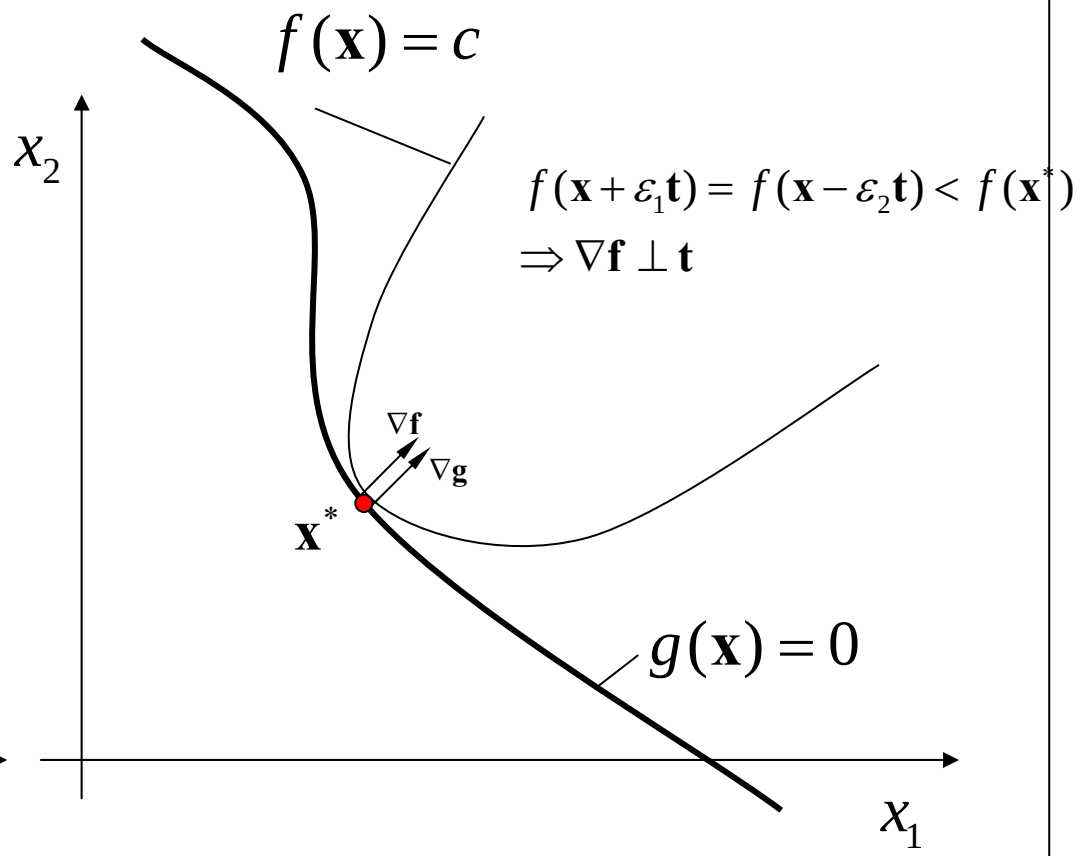
$$g(\mathbf{x}+\boldsymbol{\varepsilon}) = g(\mathbf{x}) + \boldsymbol{\varepsilon}^T \nabla \mathbf{g}$$

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

Bedingung für einen stationären Punkt **x***



a) Verhältnisse an einem nichtstationären Punkt



b) Verhältnisse an einem stationären Punkt

a) Enthält die Projektion von **∇f** auf die Tangentenrichtung **t** einen von Null verschiedenen Beitrag, so kann das Optimierungskriterium durch Bewegung in diese Richtung entlang der Kurve der NB verbessert werden! => kein stationärer Punkt!

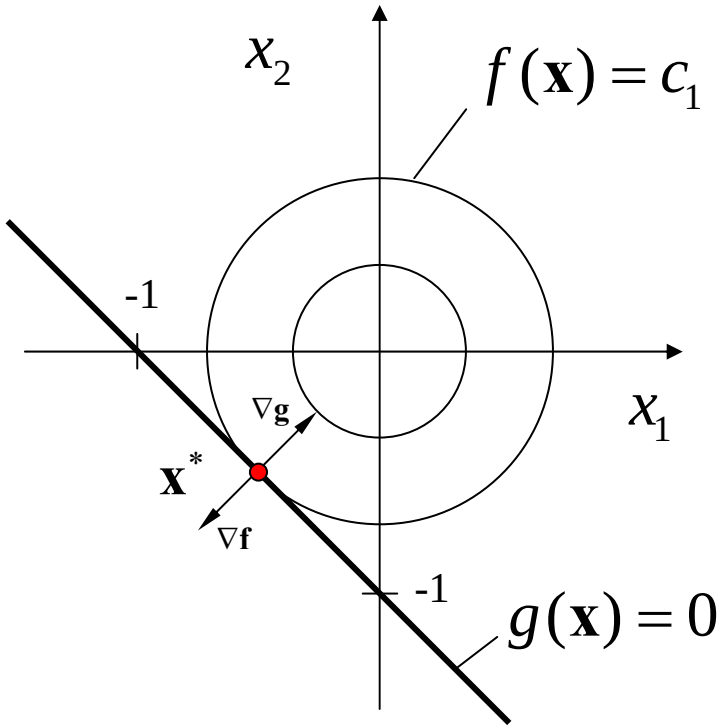
b) An einem stationären Punkt wird das Gütekriterium f(x) in beiden tangentialen Richtungen verschlechtert. **∇f** steht senkrecht auf **t**.

Beispiel:

$$\boxed{\begin{array}{ll} 1) & f(\mathbf{x}) = x_1^2 + x_2^2 \\ 2)\ \text{NB:} & g(\mathbf{x}) = x_1 + x_2 + 1 = 0 \end{array}}$$

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

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



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

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

notw. Bed. für ein Optimum:

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

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

$$\Rightarrow \quad \begin{array}{ll} 1) & 2x_1 + \lambda = 0 \\ & 2x_2 + \lambda = 0 \\ 2) & x_1 + x_2 + 1 = 0 \end{array}$$

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

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

Demos mit MATLAB

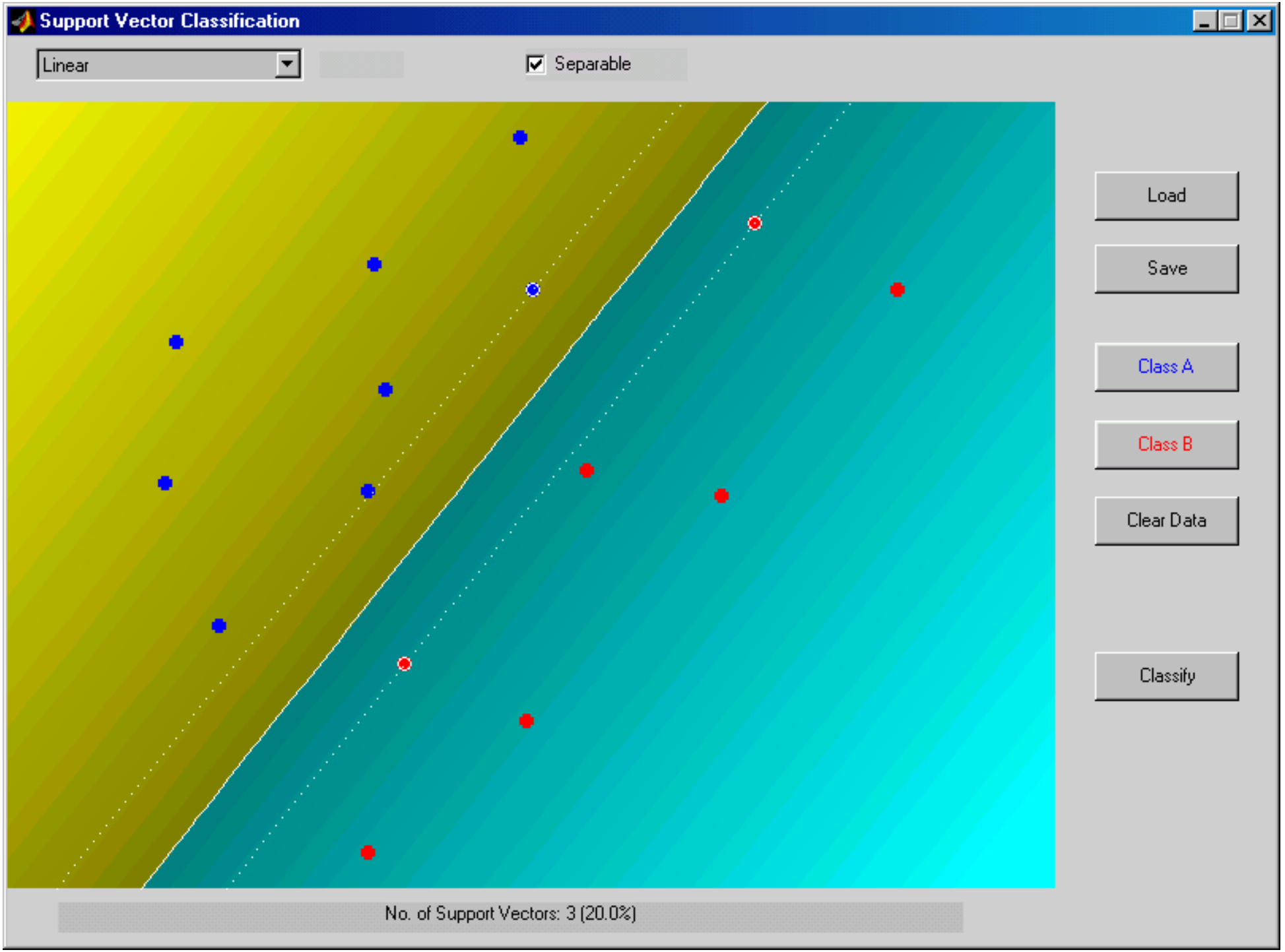
(svmmatlab, uiclass.m)

- Linearer Klassifikator
  - Harter Rand, Problem separierbar
  - Harter kleiner Rand wegen Ausreißer, Problem separierbar, aber schlechte Generalisierung => emp. Fehler vergrößern mit Gewinn bei Generalisierung => weichen Rand einführen
- 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
  - Polynomiale Separierung (p=2), harter Rand
  - Polynomiale Separierung (p=4), harter Rand
  - Bananenshape mit Polynomkernen
  - Bananenshape mit Gauss-Radialbasisfunktionen

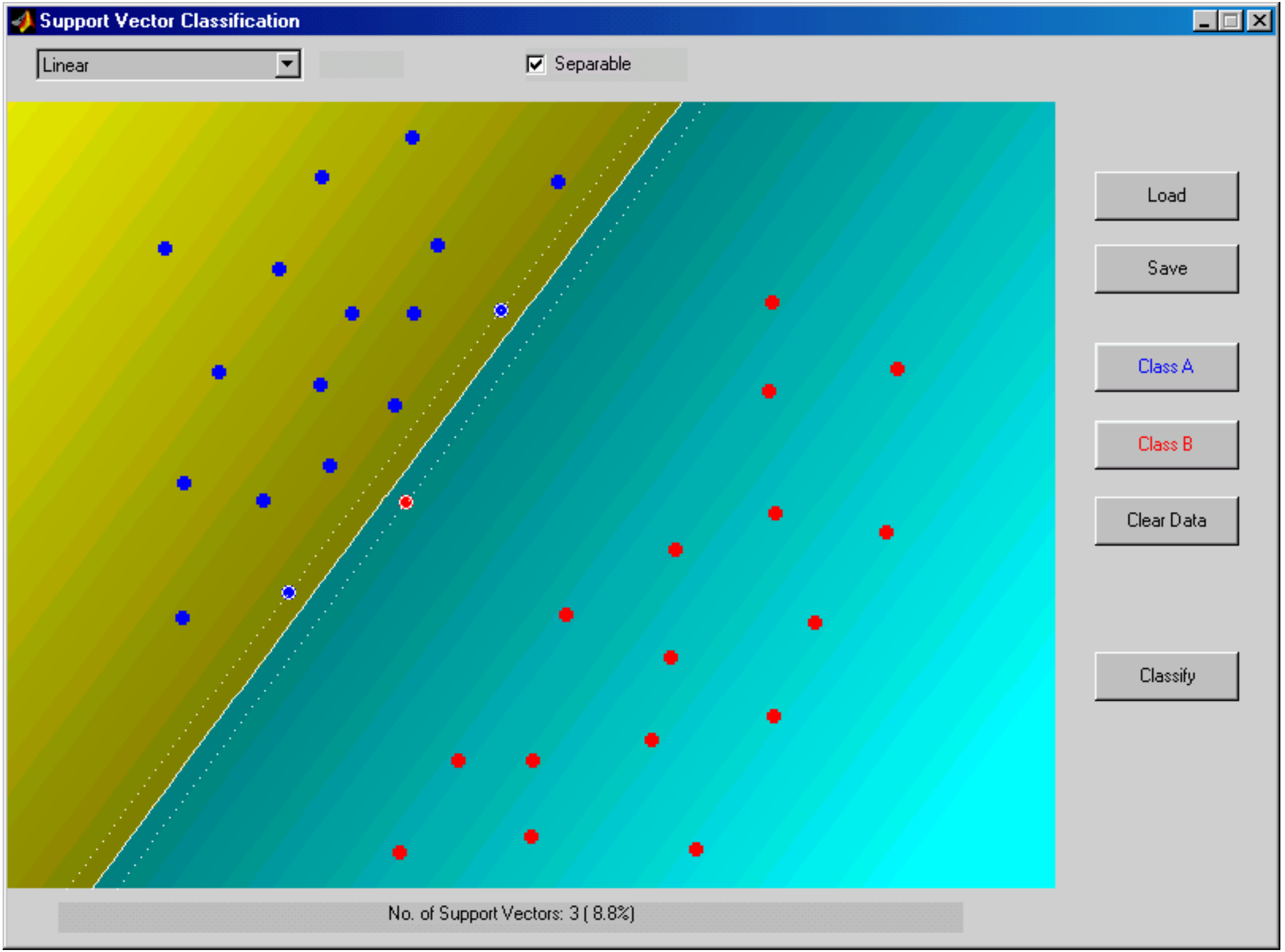
Start der Matlab-Demo

matlab-SVM.bat

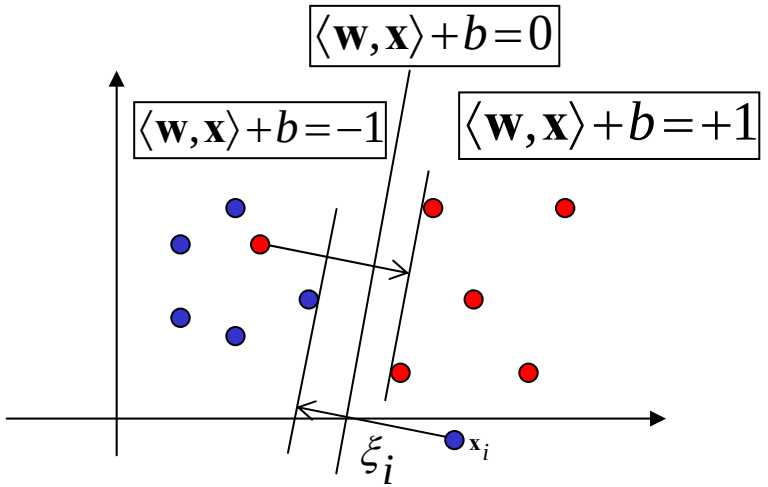
Linear separierbare Klassen; harter Rand



Linear separiebare Klassen; harter, kleiner Rand, schlechte Generalisierung



Nichtseparabler Fall



Bestrafen von Randverletzungen via "slack"-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 $\xi_i \geq 0$

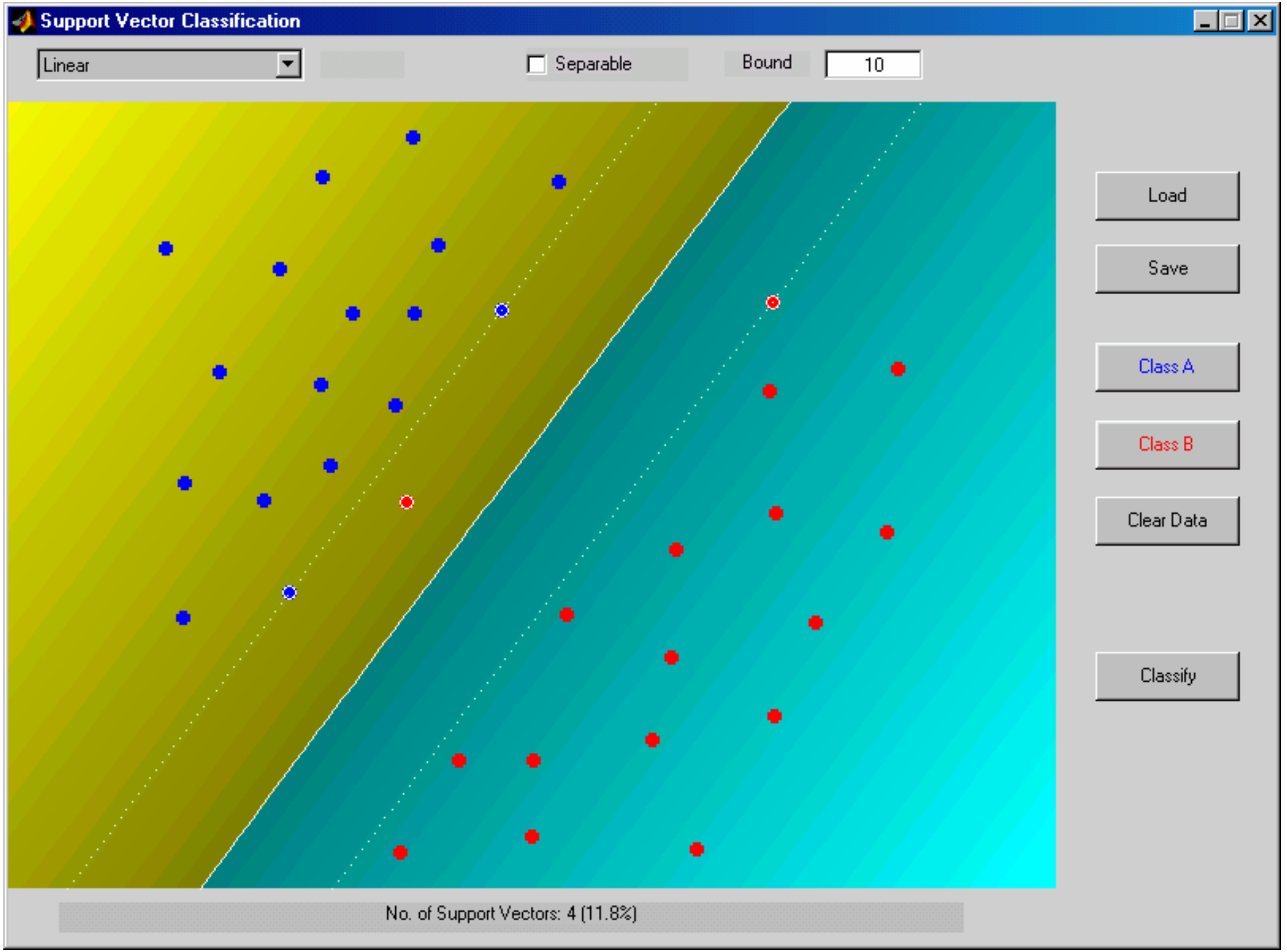
Gründe für diese Verallgemeinerung:

- Lösung nicht existent mit bisherigem Ansatz mit hartem Rand
- Verbesserung der Generalisierung bei Ausreißern in der Randzone

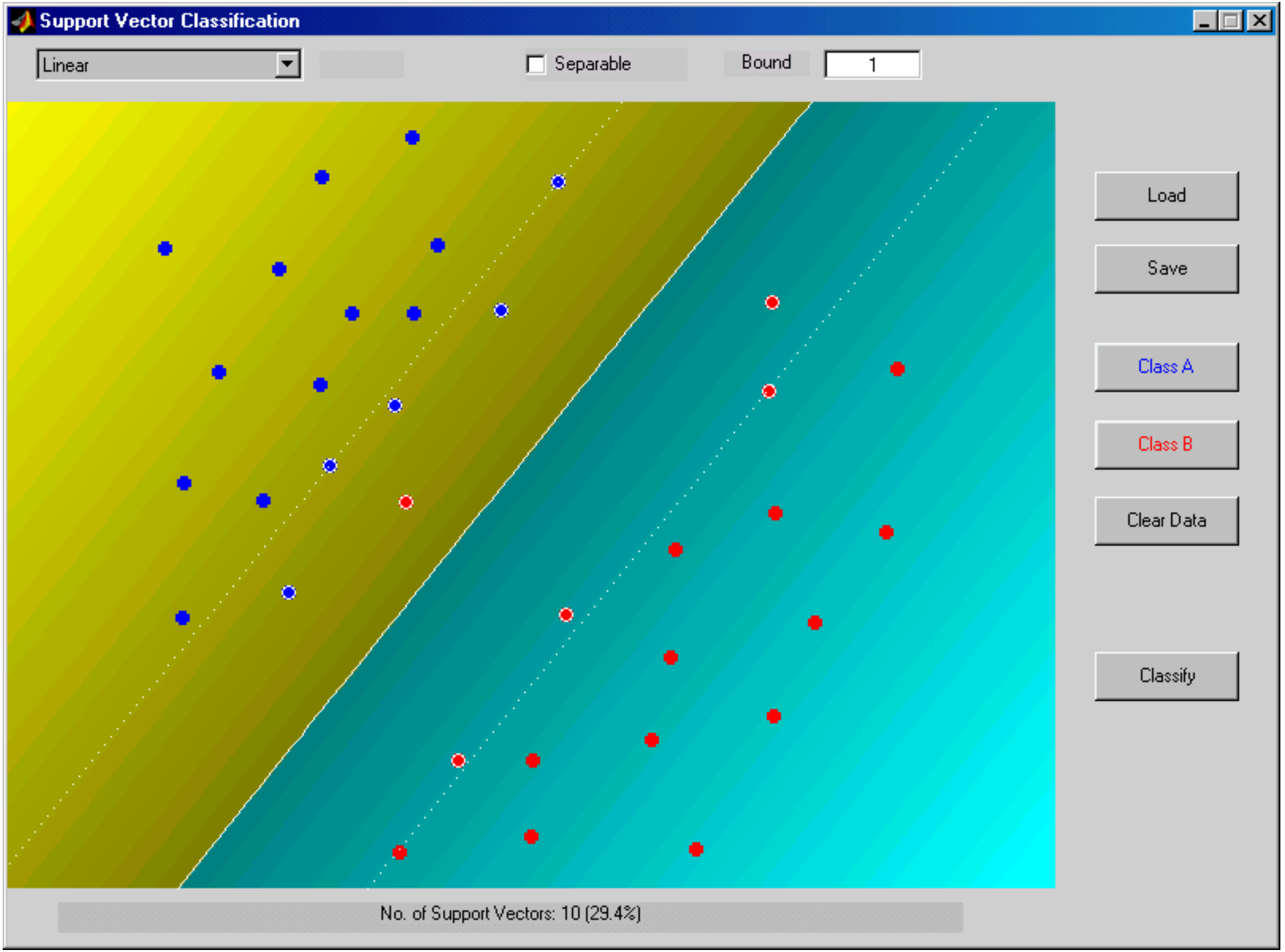
Fallunterscheidung:

- $0<\alpha_i<C \quad \Leftrightarrow$ SV mit $\xi_i=0$
- $\alpha_i=C \quad \Leftrightarrow$ SV mit $\xi_i>0$
- $\alpha_i=0 \quad \Leftrightarrow$ für die restlichen Vektoren $\mathbf{x}_i$

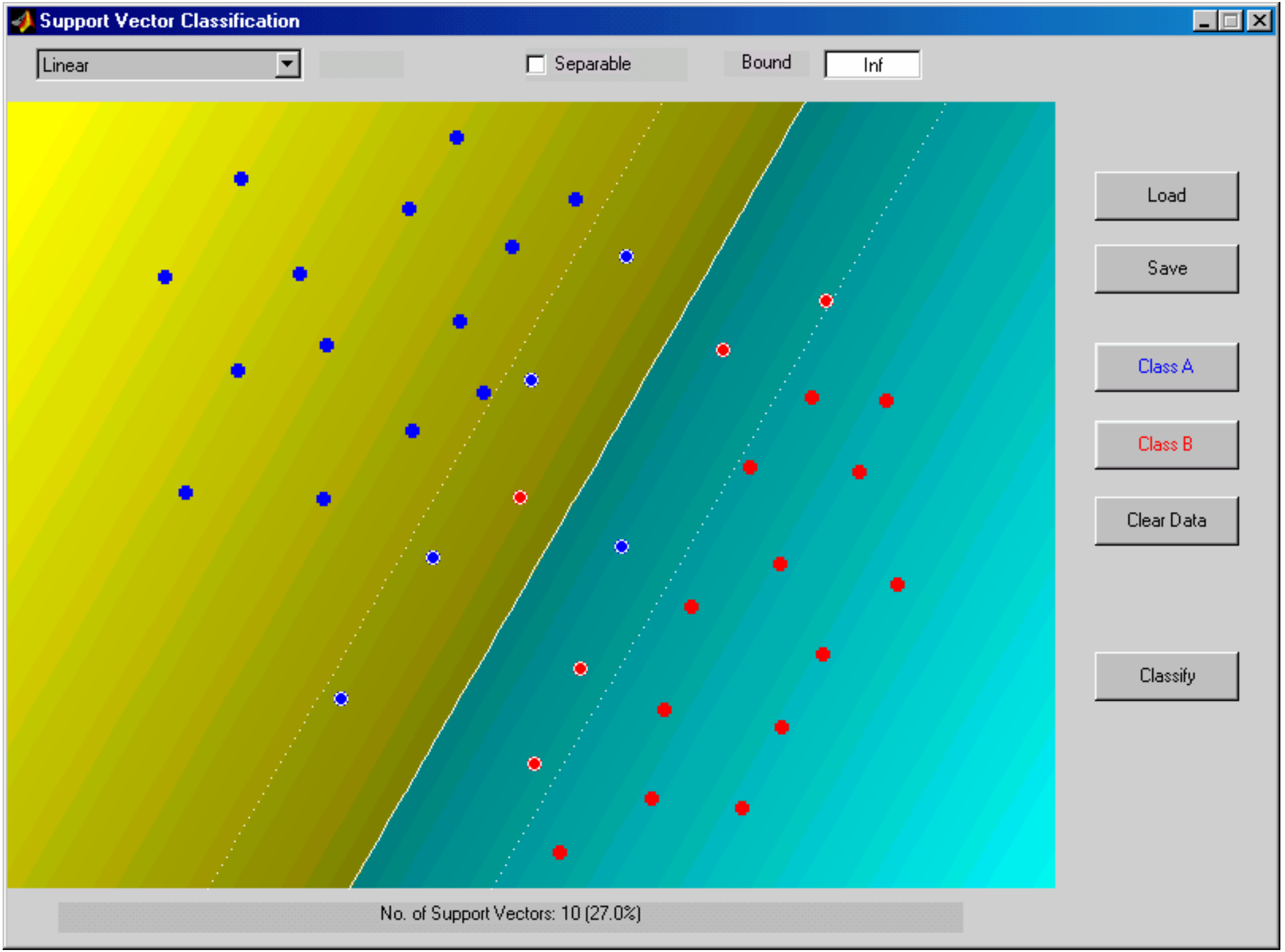
Linear sep. Klassen; weicher, breiter Rand => R_{emp} größer, gute Generalisierung



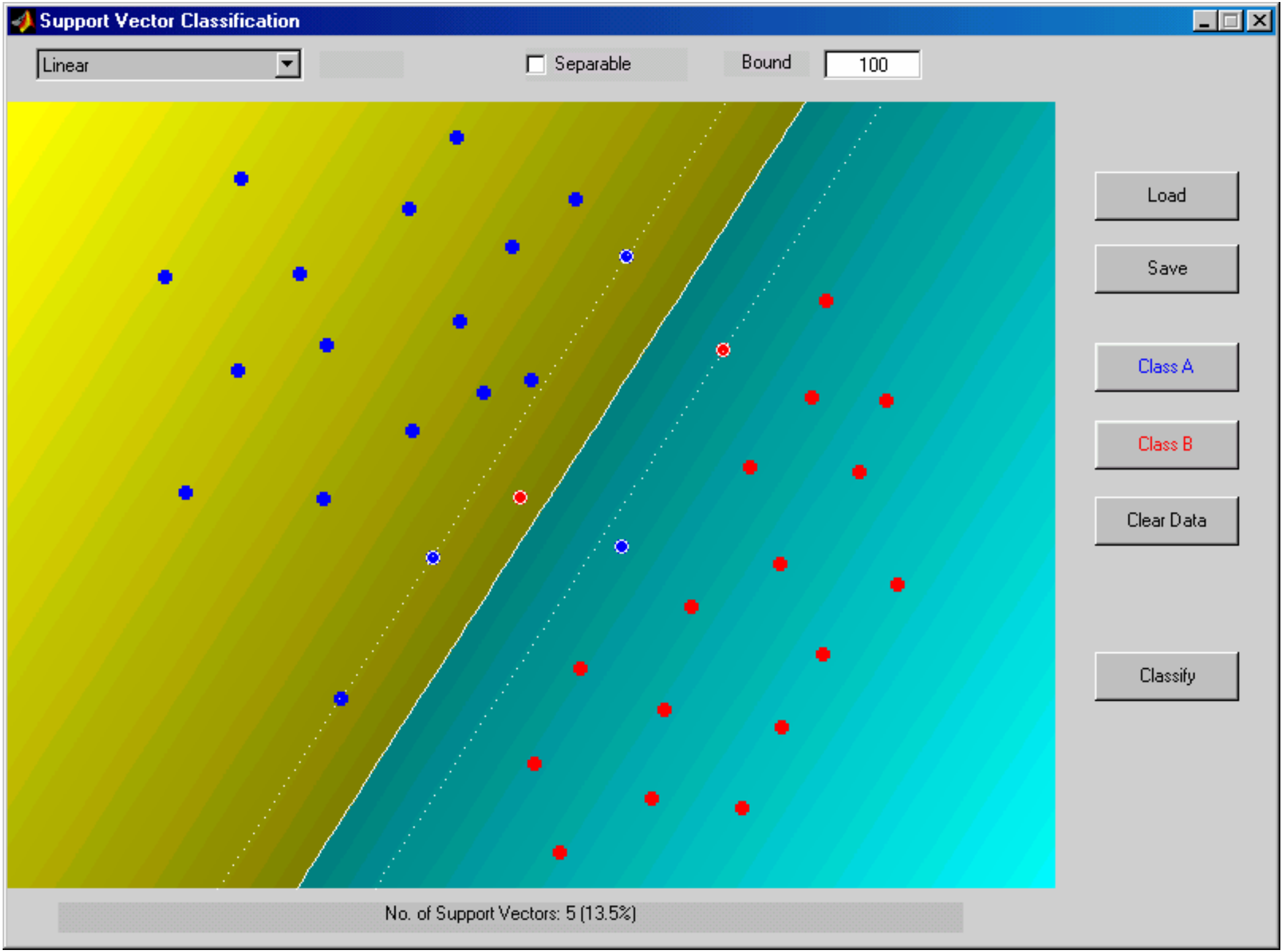
Linear sep. Klassen; weicher, breiter Rand => R_{emp} größer, gute Generalisierung



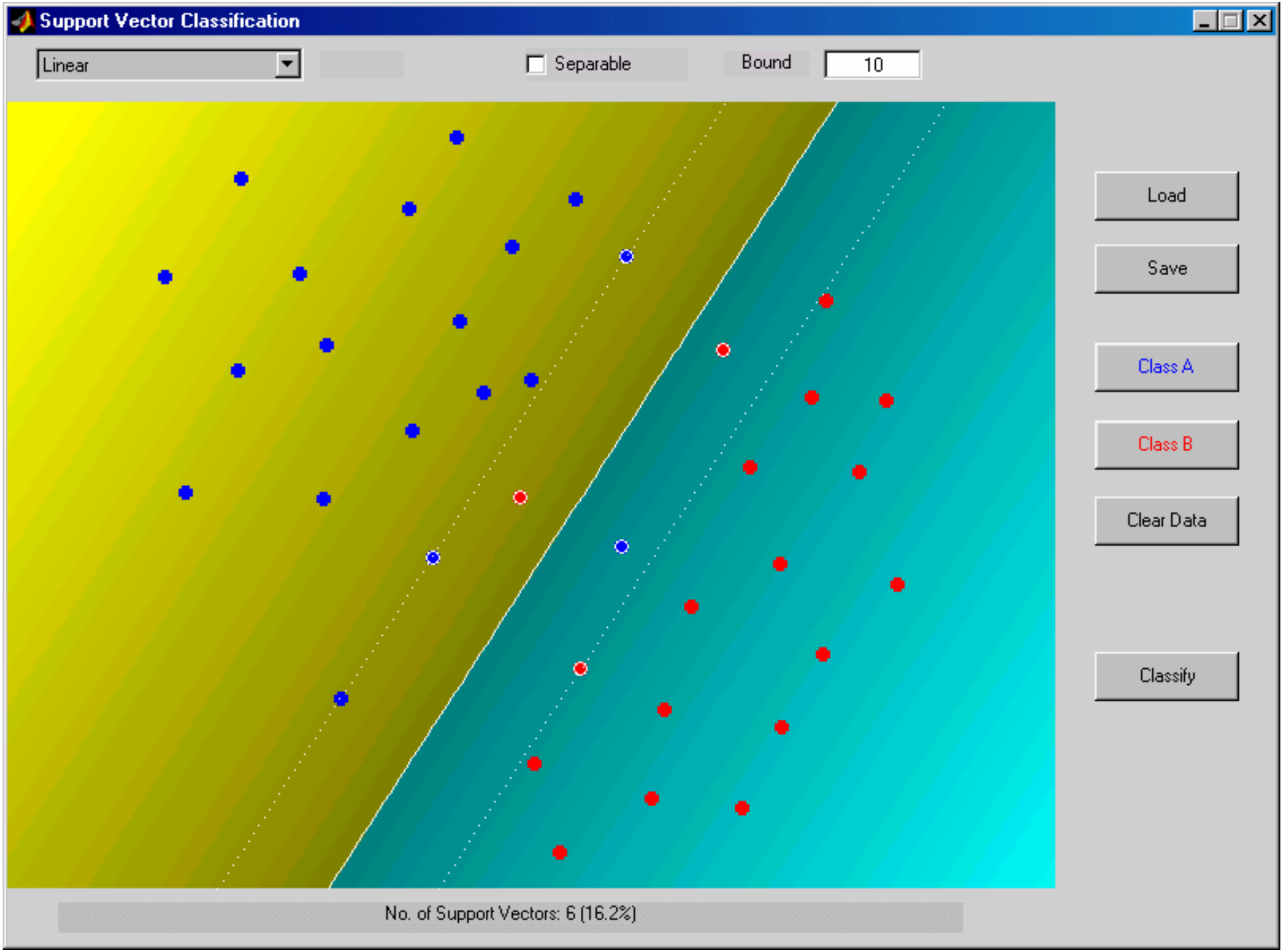
Linear separiebare Klassen; harter Rand



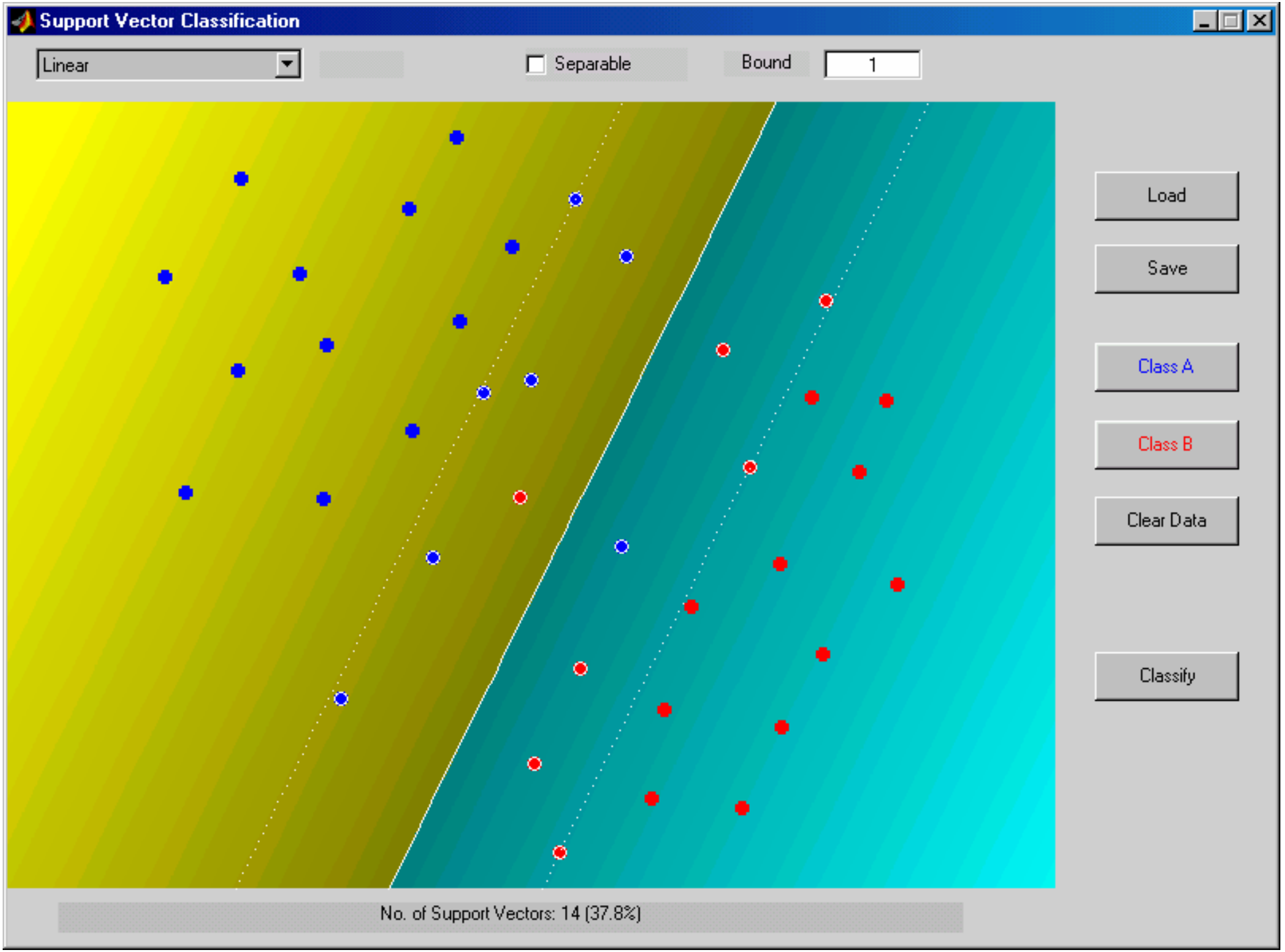
Linear separiebare Klassen; harter Rand



Linear separiebare Klassen; harter Rand

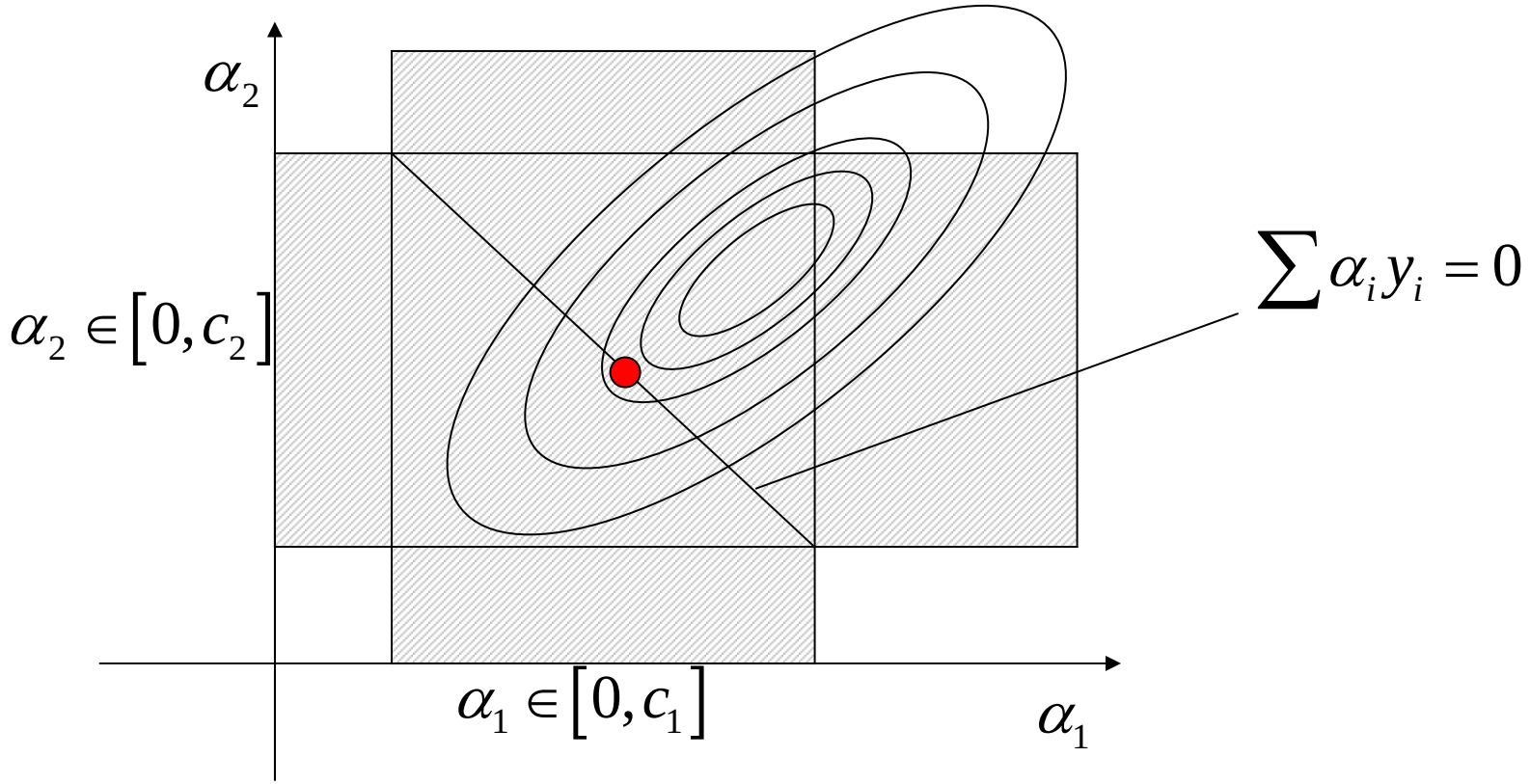


Linear separiebare Klassen; harter Rand



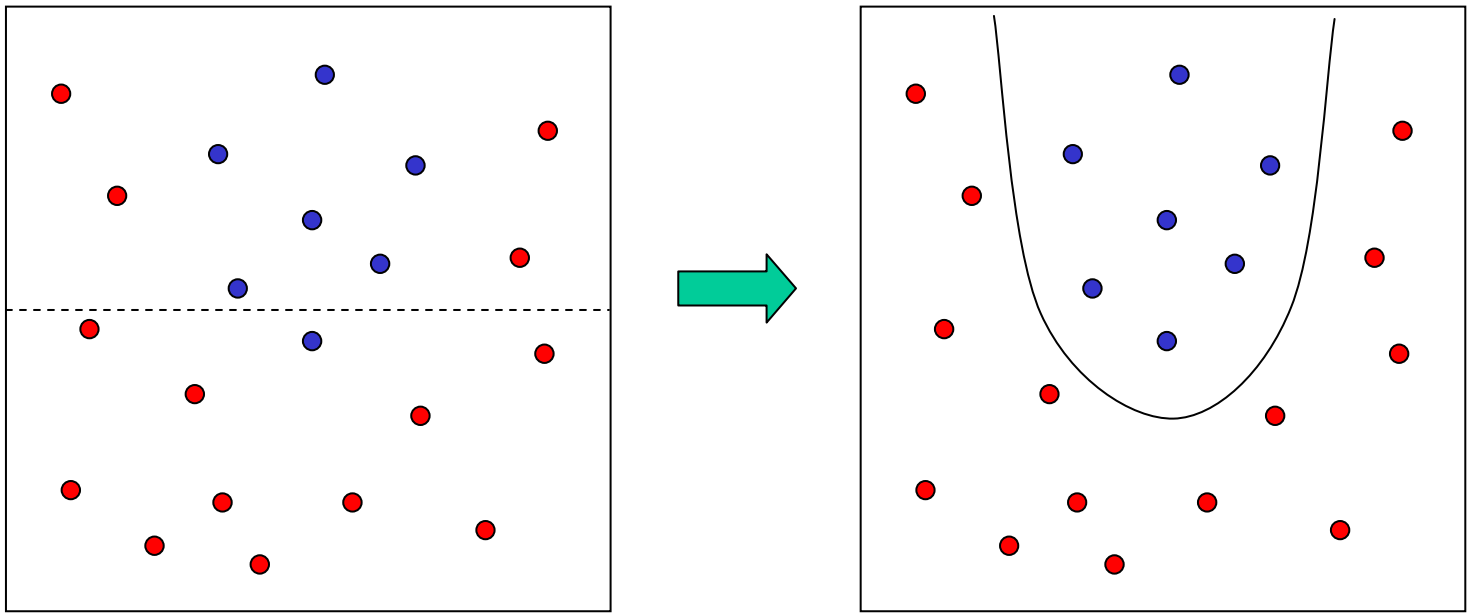
Globales, eindeutiges Optimum

- Optimierung eines quadratischen Problems in **w** (konvex)
- unter linearen Nebenbedingungen
- Lineare Nebenbedingungen miteinander geschnitten ergeben konvexes Gebiet; dieses geschnitten mit quadratischer Form => ergibt wiederum konvexes Gebiet => eindeutiges Minimum!



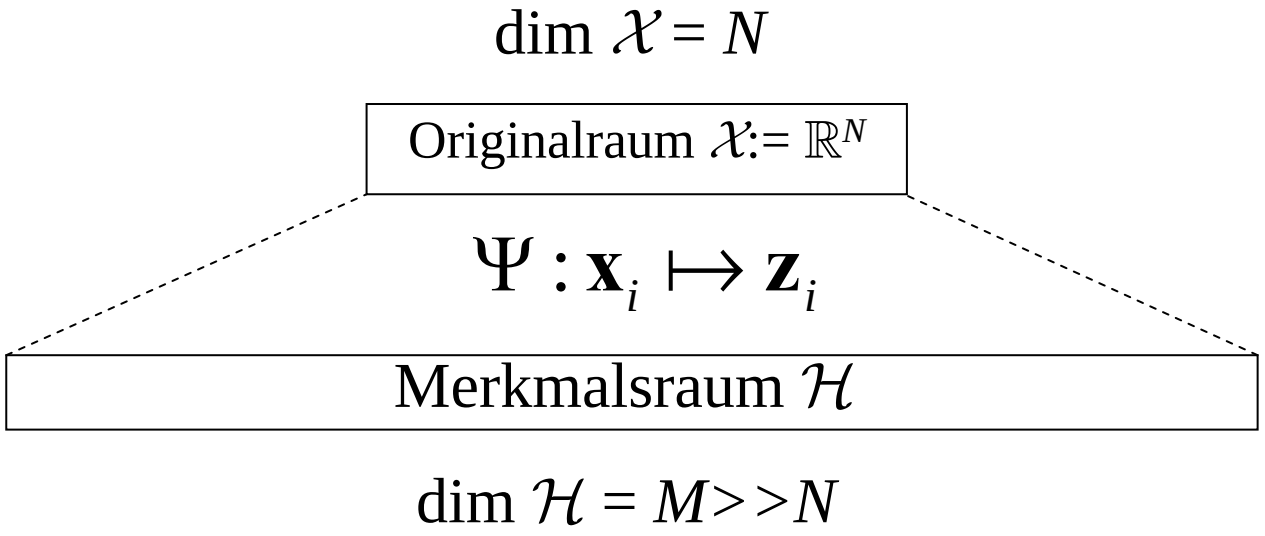
Nichtlineare Probleme

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



Erweiterung des Hypothesenraumes

Idee: Finde Hyperebene im höherdimensionalen Merkmalsraum

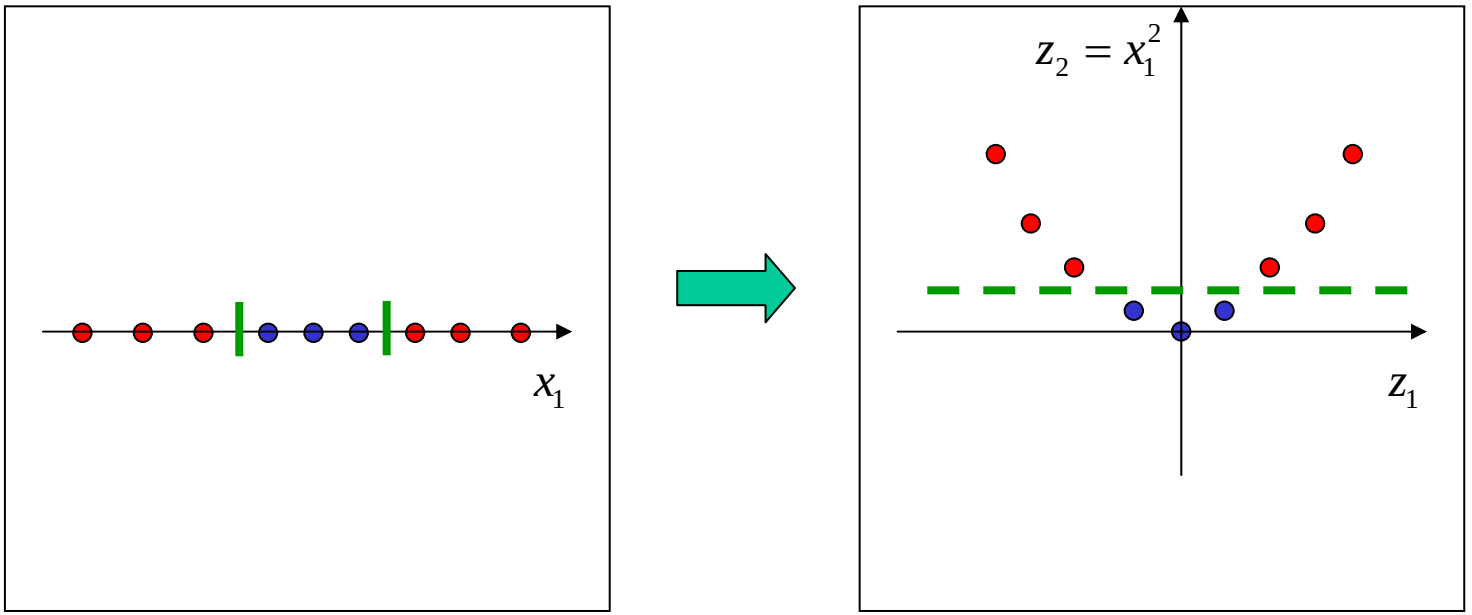


Die trennende Hyperebene im Merkmalsraum ist eine nichtlineare Trennfläche im Originalraum (siehe XOR-Problem mit Polynomklassifikator)

Nichtlineare Probleme

- Eindimensionaler Originalraum: x_1
- Zweidimensionaler Merkmalsraum:

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



linear nicht separierbar

lineare Separation

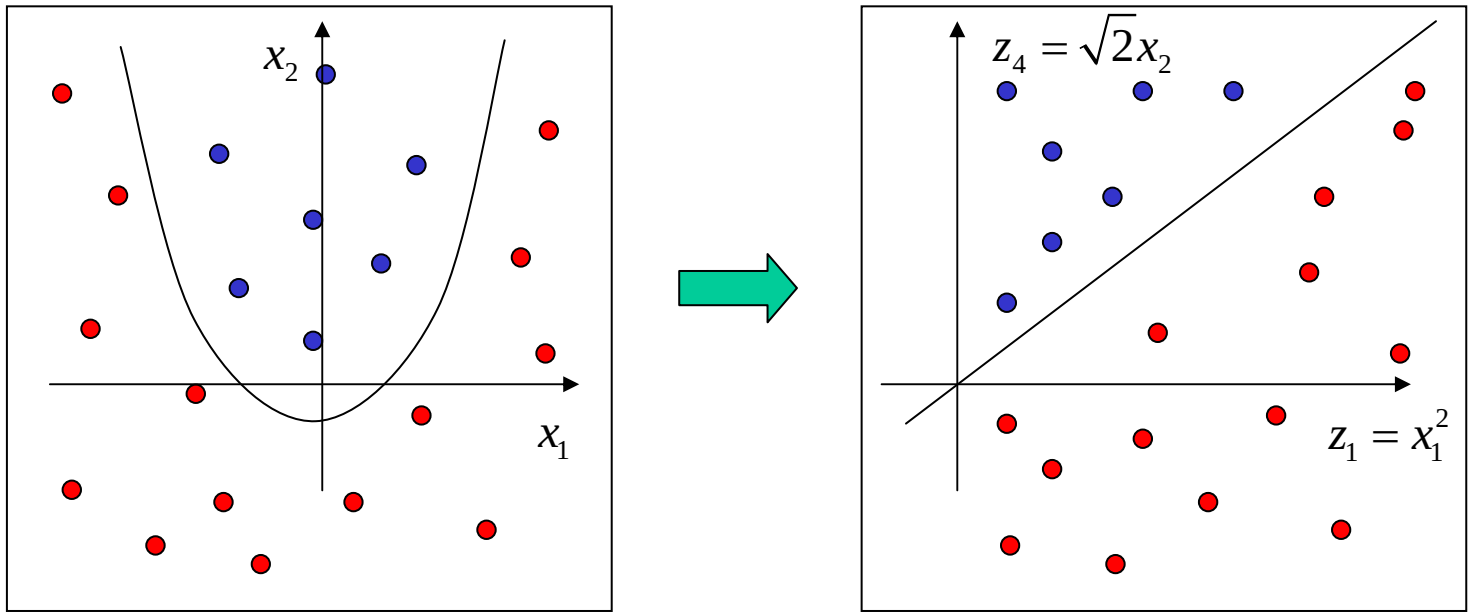
Nichtlineare Probleme

- Originalraum:

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

- Merkmalsraum:

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



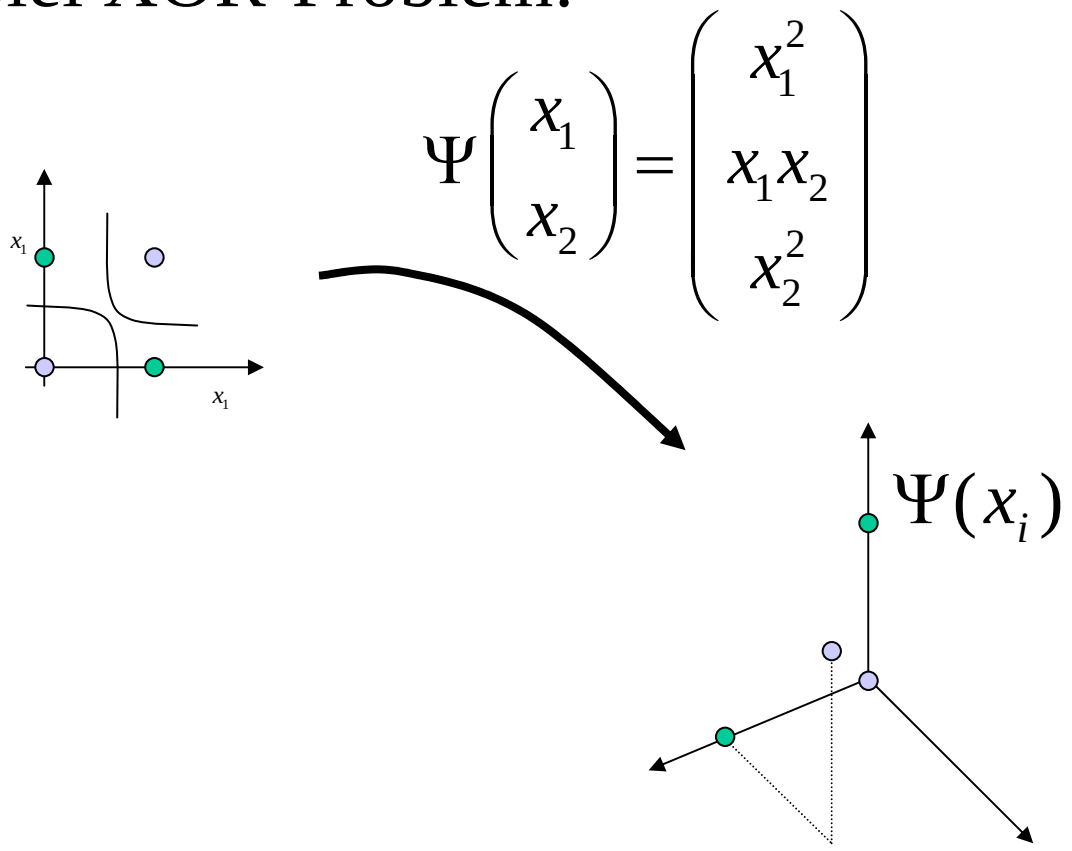
- $x_2 > x_1^2$ nichtlineare
- $x_2 < x_1^2$ Separation

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

Nichtlineare Erweiterung

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

- 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 $\mathcal{H}$ wir durch den Trick mit Kernfunktionen reduziert. Das Innenprodukt in H hat einen 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 des Originalraums führen zu $O(M=N^p)$ Dimensionen im Merkmalsraum!

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

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

Beispiel:

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

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

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 Basisfunkten 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 Φ 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 Φ 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 (K ist eine symmetrische, positiv semidefinite Funktion in x):

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

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:

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

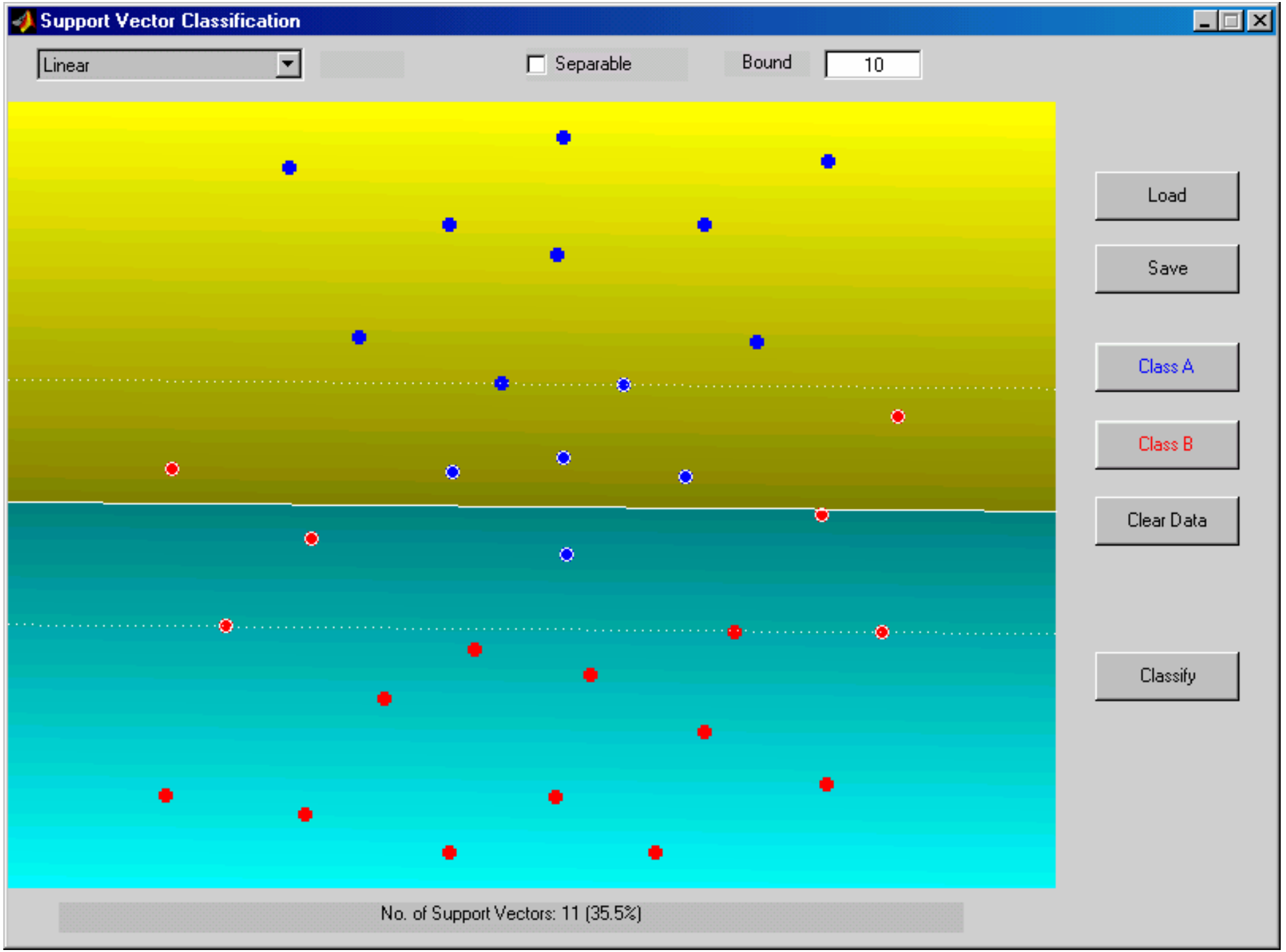
mit: $\sum_{i=1}^{l} y_i \alpha_i = 0 \quad \text{und} \quad 0 \le \alpha_i \le C$

– Klassifizieren ($\alpha_i \neq 0$ für alle *SV*):

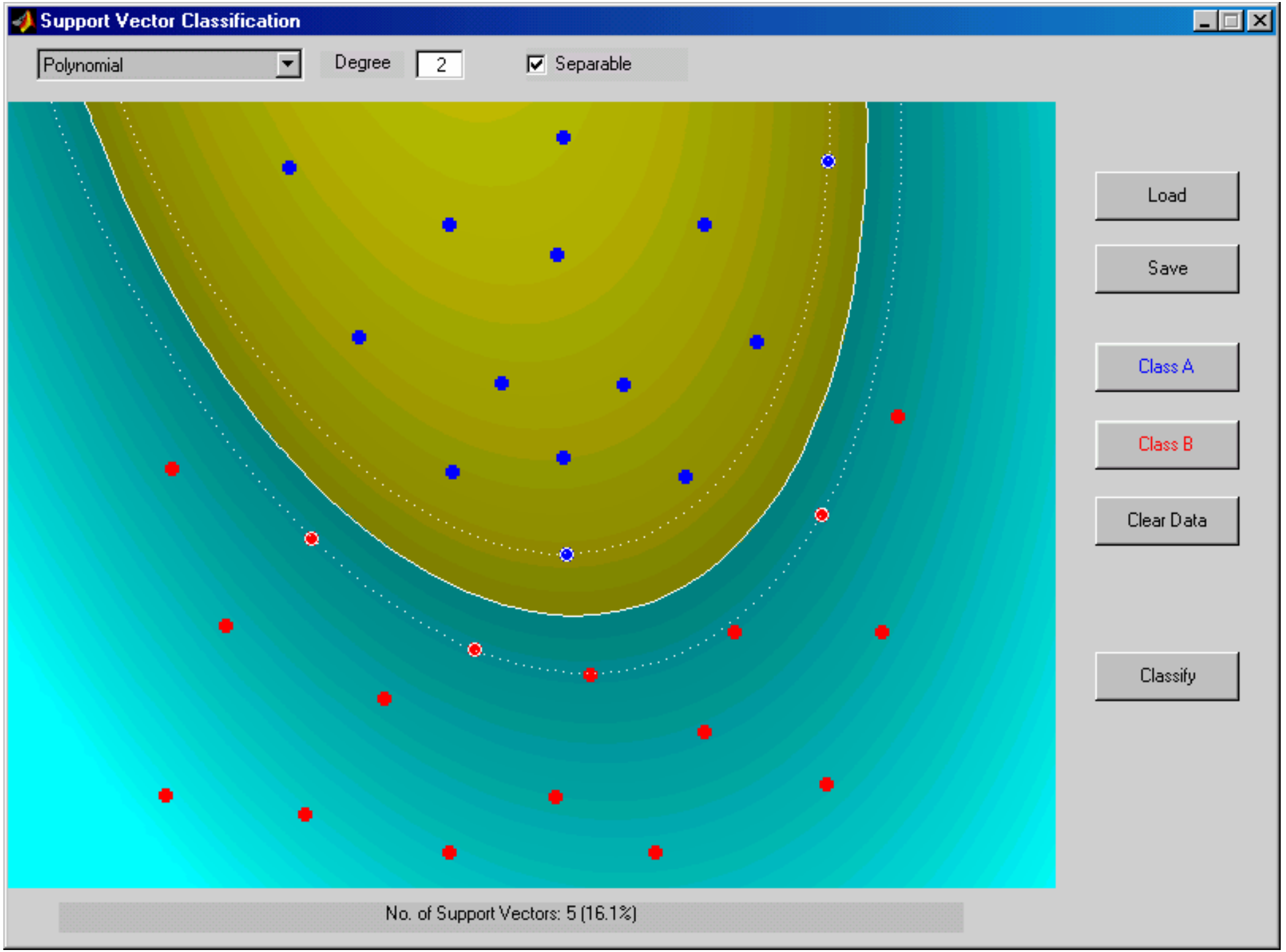
$$f(x) = \operatorname{sgn}\left(\sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}_j) + b \right) = \operatorname{sgn}\left(\sum_{\mathbf{x}_i \in SV} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}_j) + b \right)$$

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

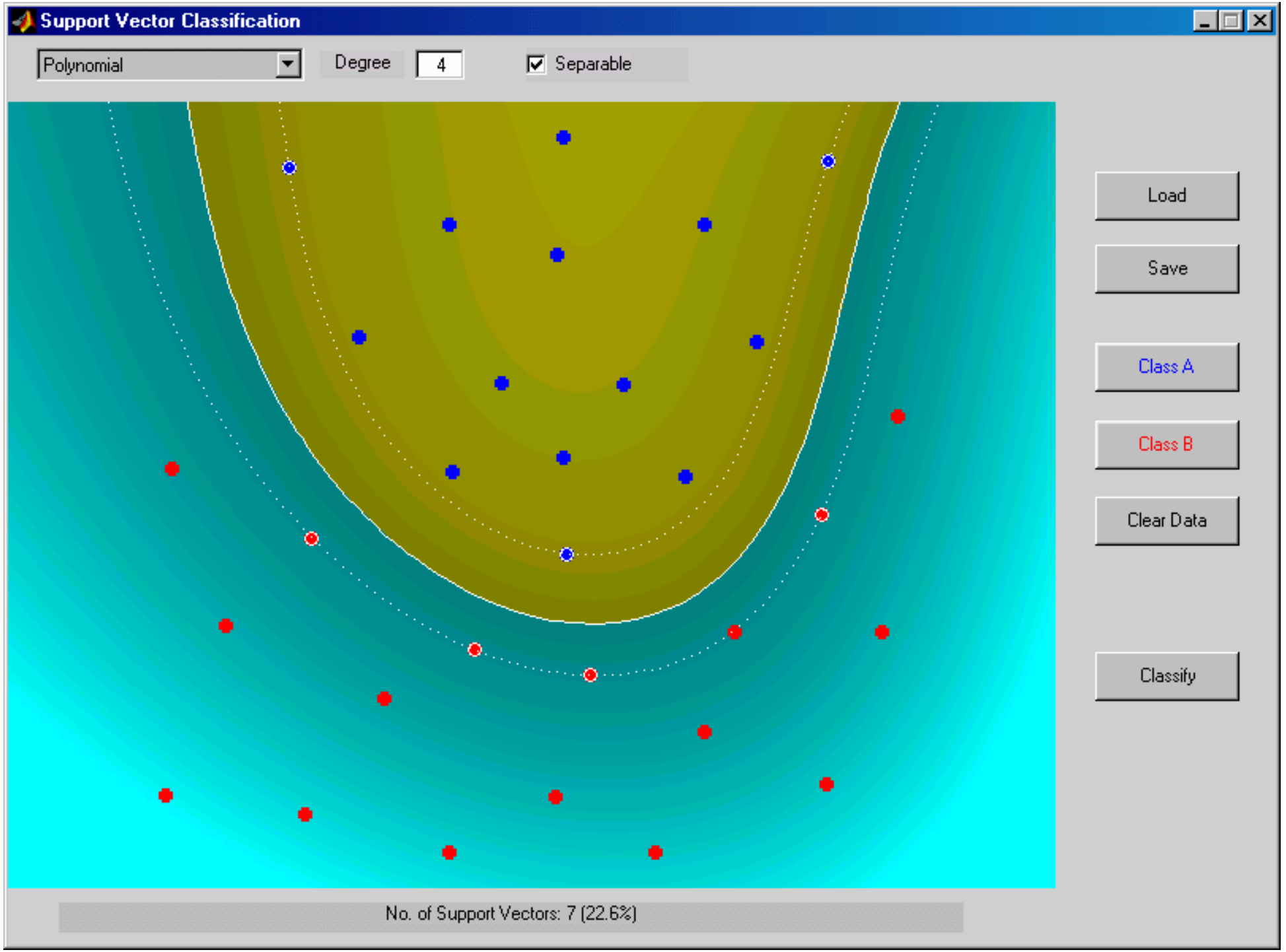
Lineare Separierung eines quadratischen Problems; weicher Rand



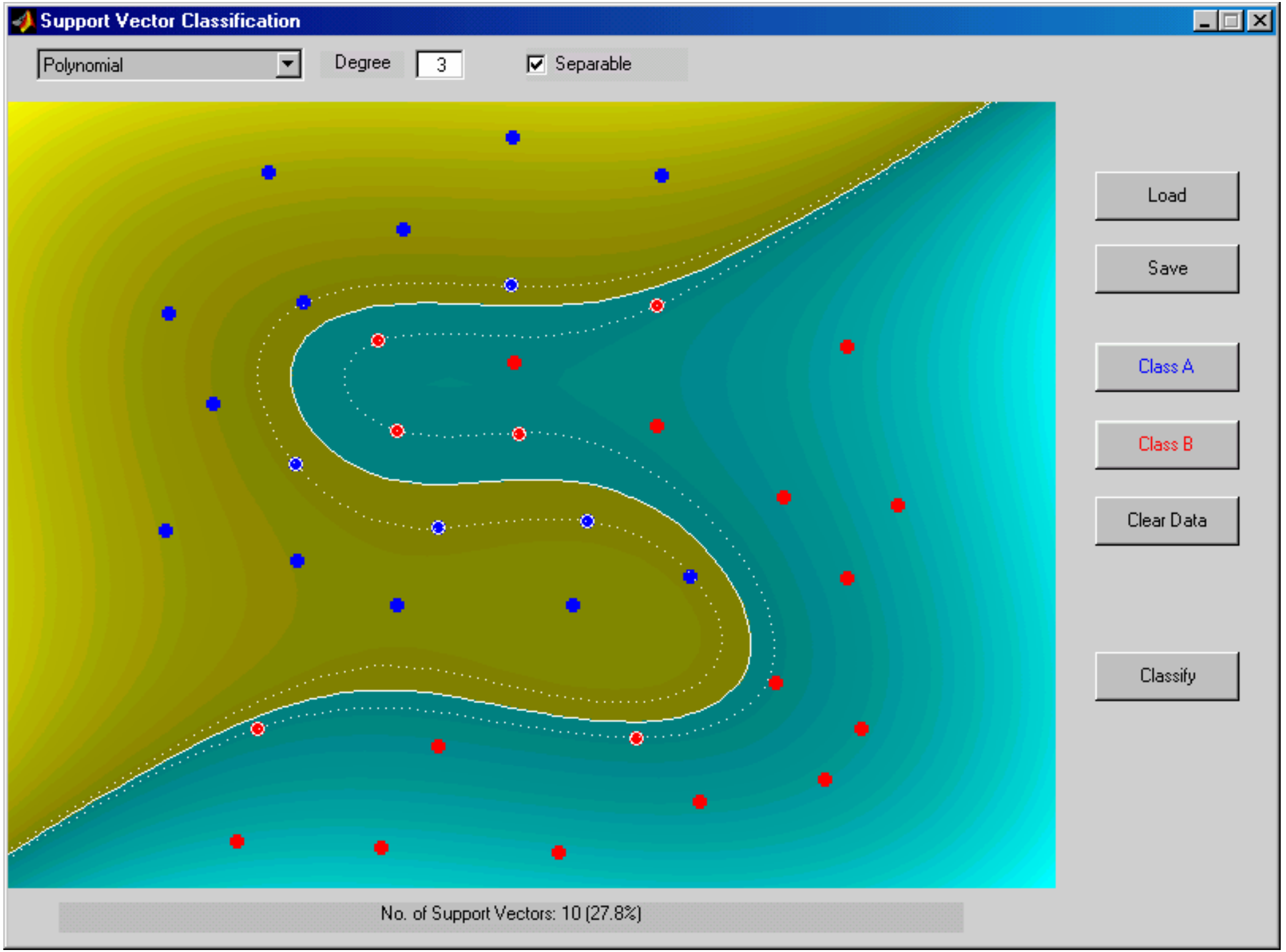
Polynomiale Separierung eines quadratischen Problems; harter Rand



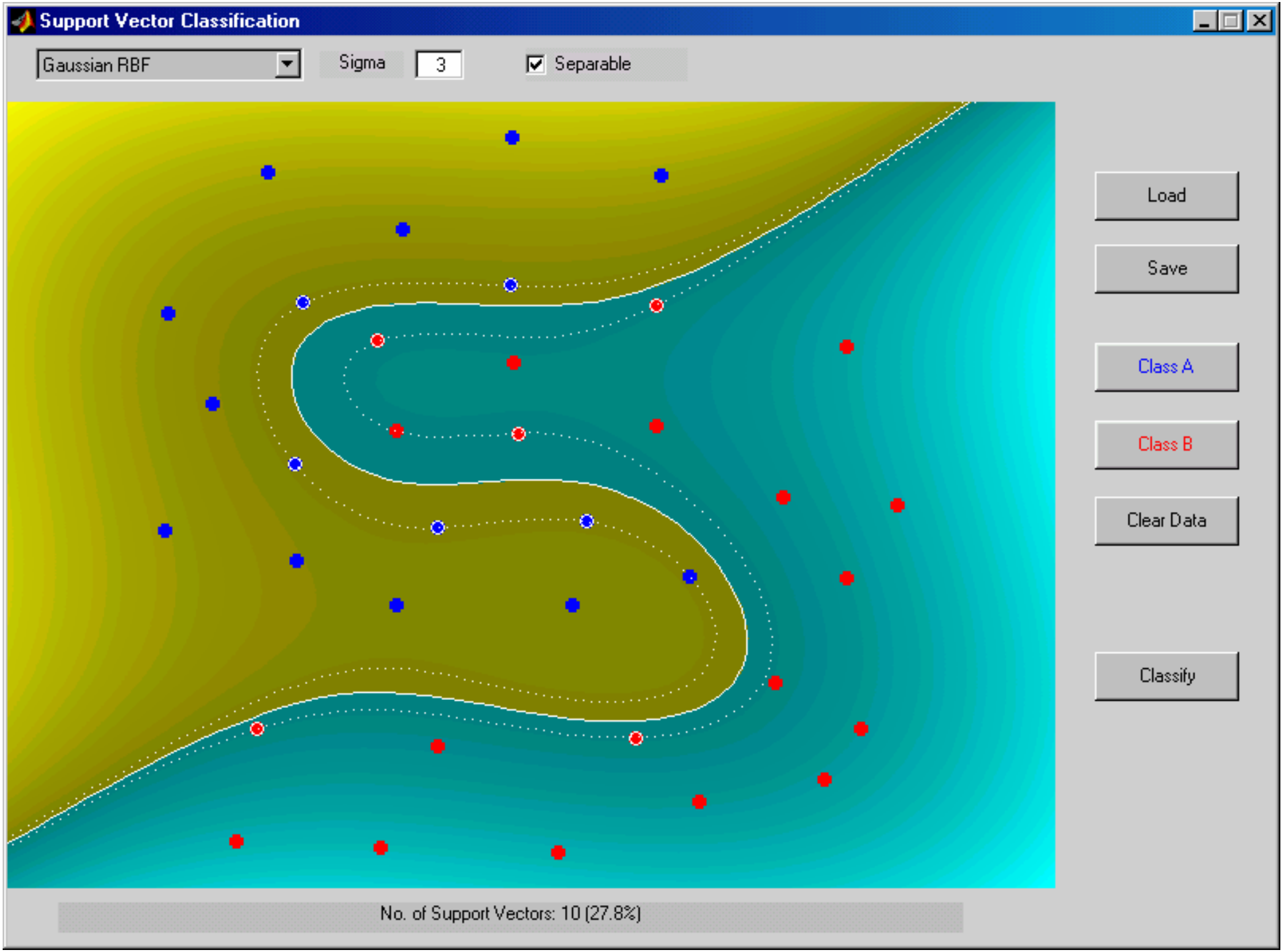
Polynomiale Separierung eines quadratischen Problems; harter Rand



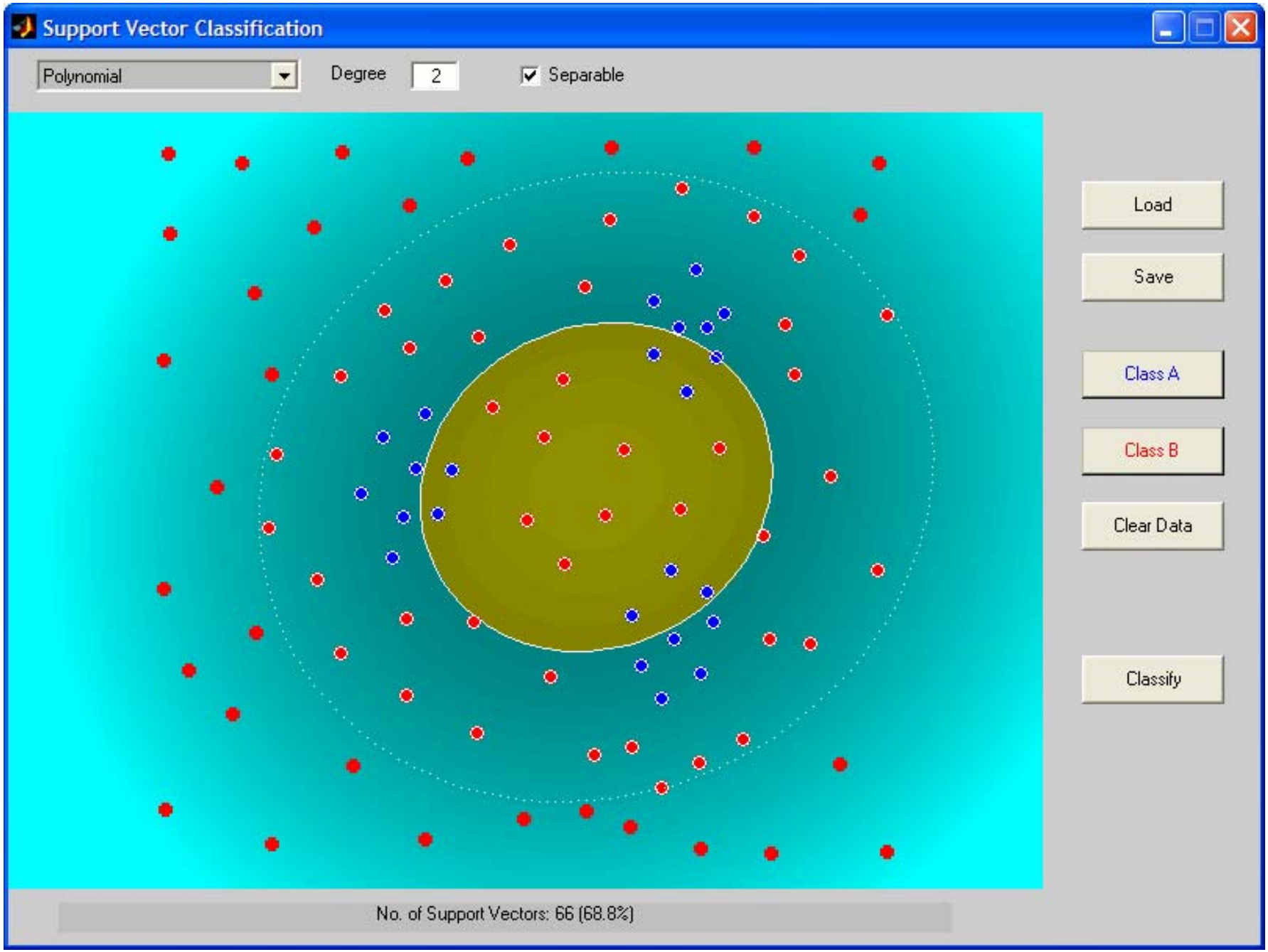
Nichtlinear separierbare Klassen; harter Rand



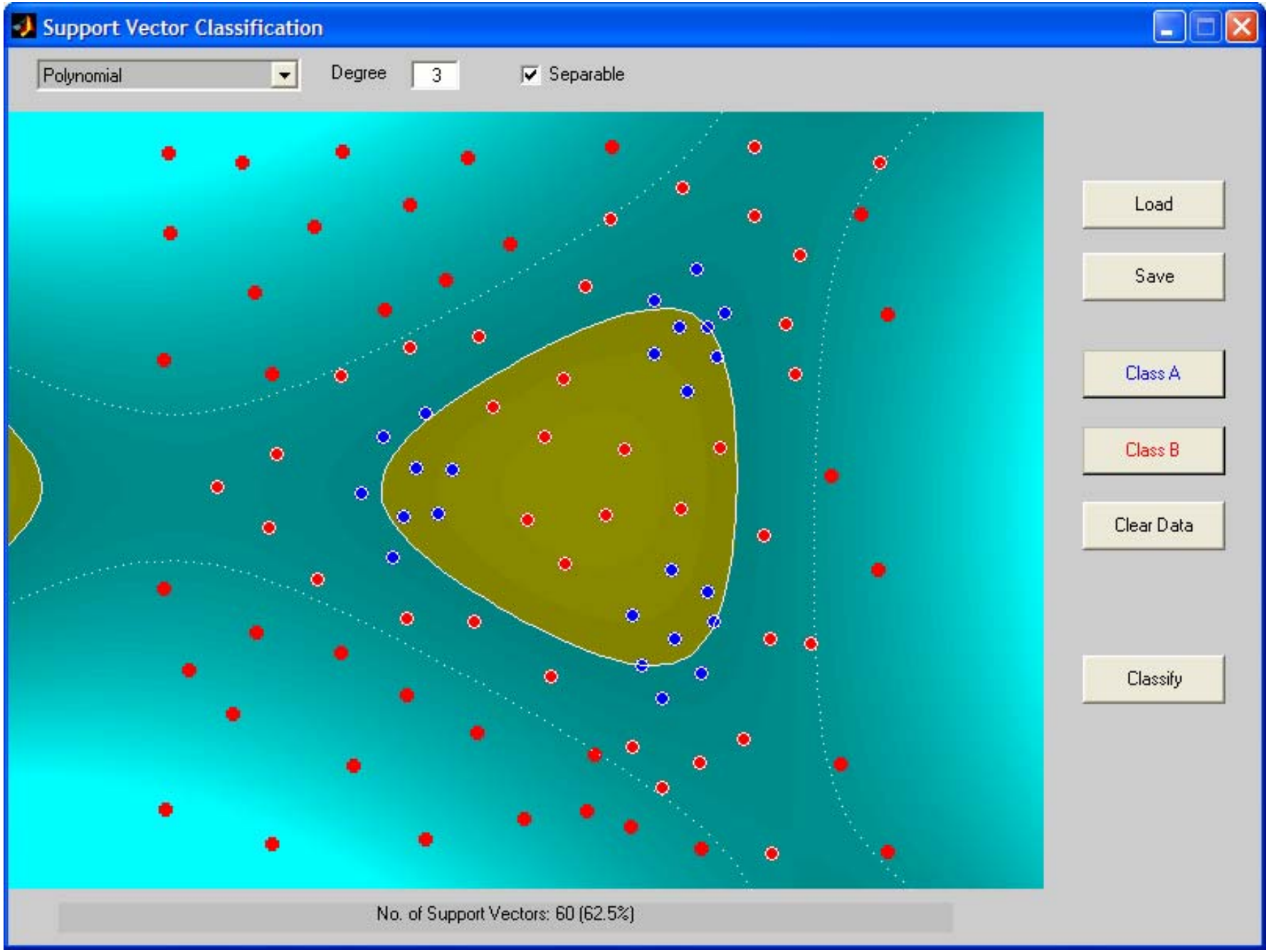
Nichtlinear separierbare Klassen; harter Rand



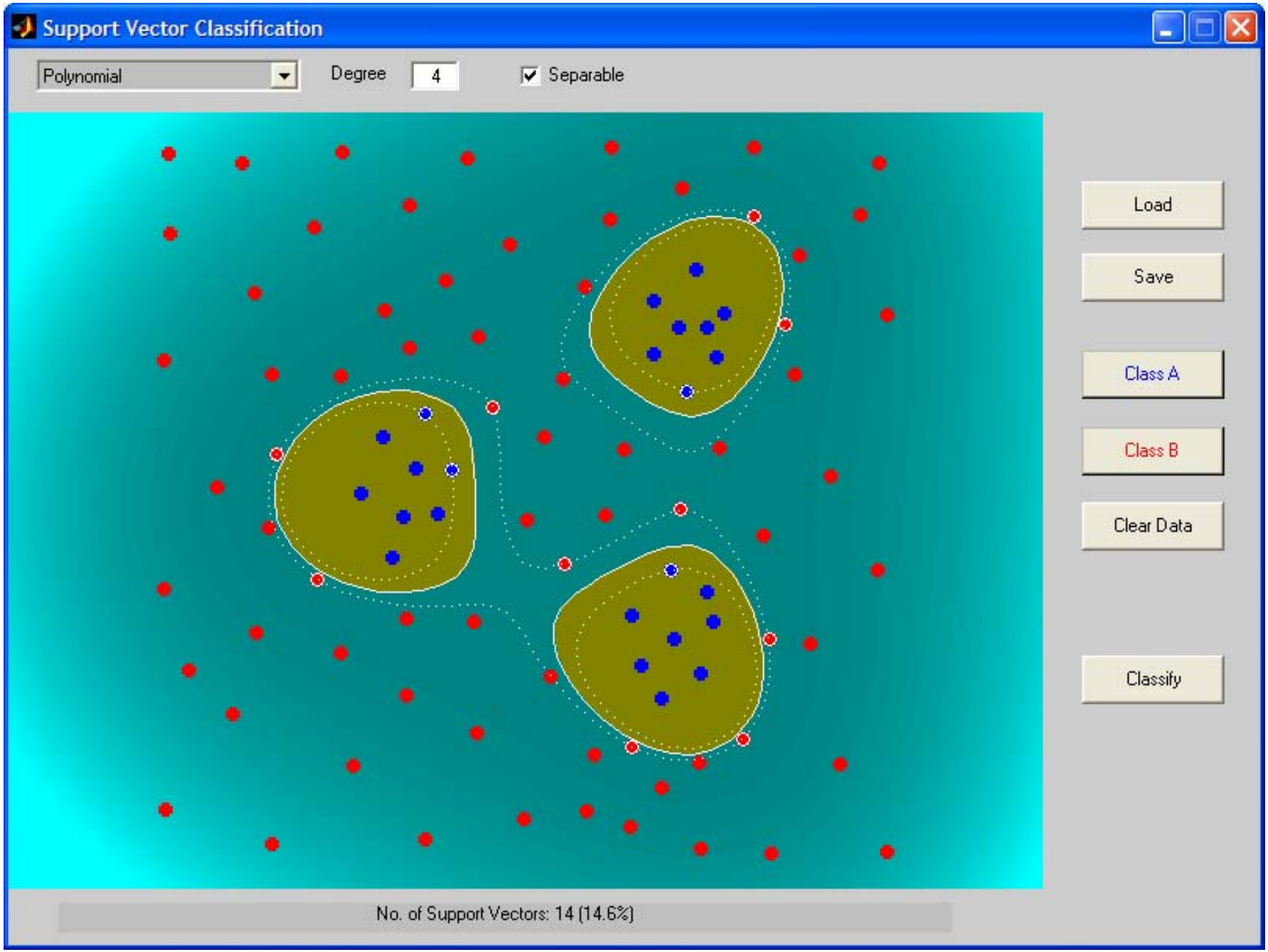
Beispiel: „Inseln“, Polynom mit $p = 2$



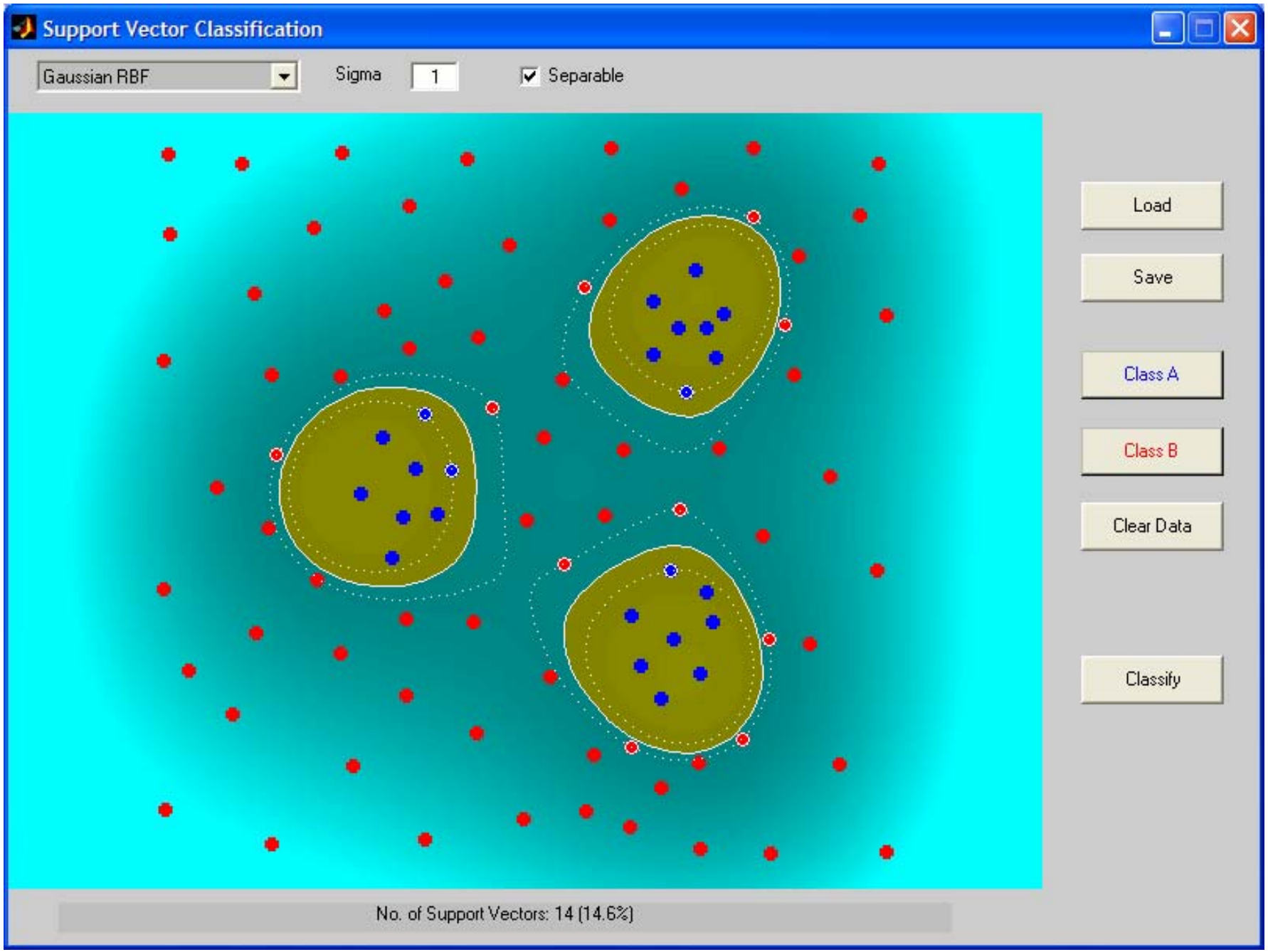
Beispiel: „Inseln“, Polynom mit $p = 3$

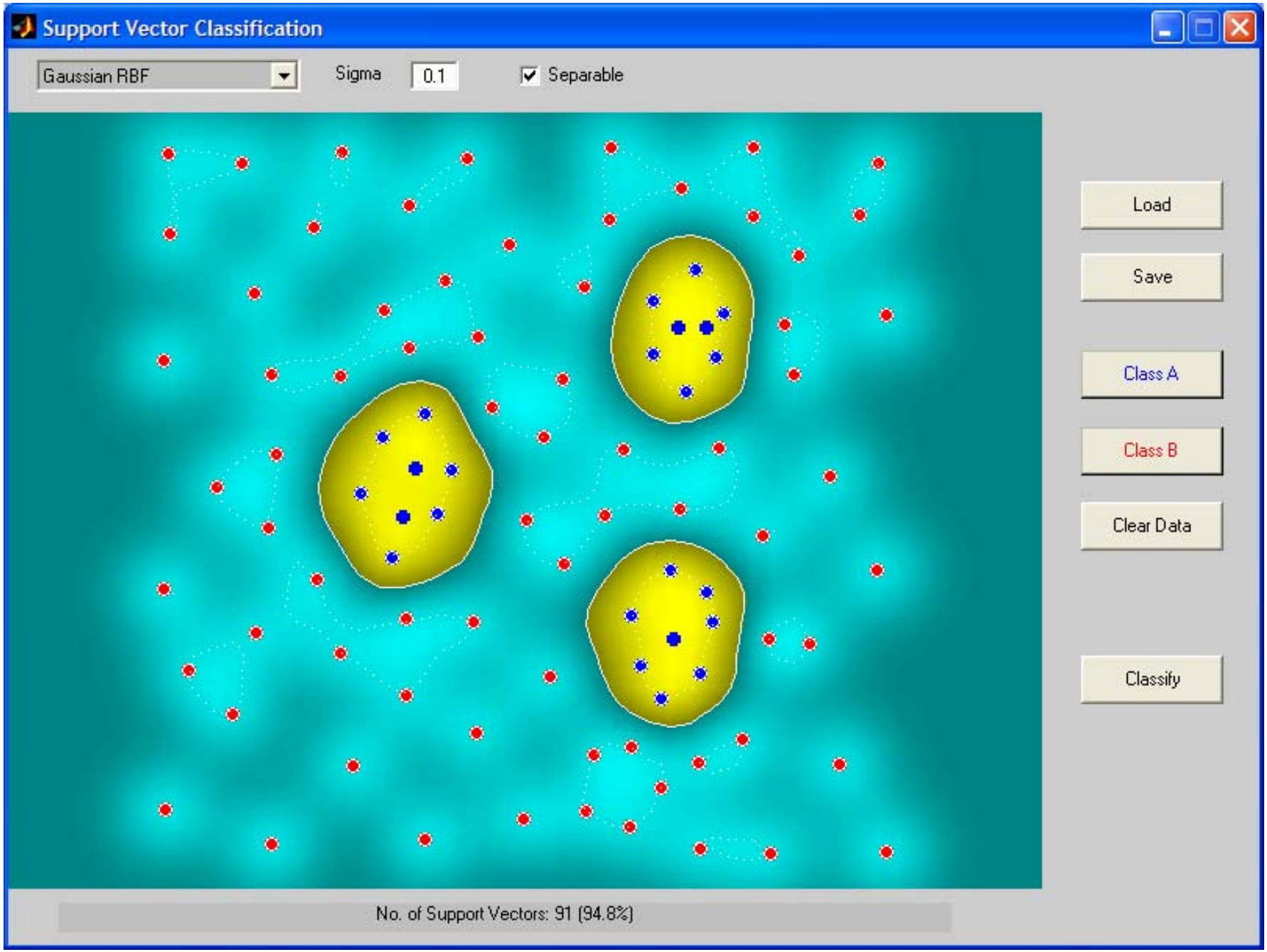


Beispiel: „Inseln“, Polynom mit $p = 4$



Beispiel: „Inseln“, RBFs mit 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%)

Anwendungsbeispiel: Zeichenerkennung

Beispiel: US Postal Service Digits [Schö95/96]

16x16 Grauwertbilder

7291 Training, 2007 Test-Samples

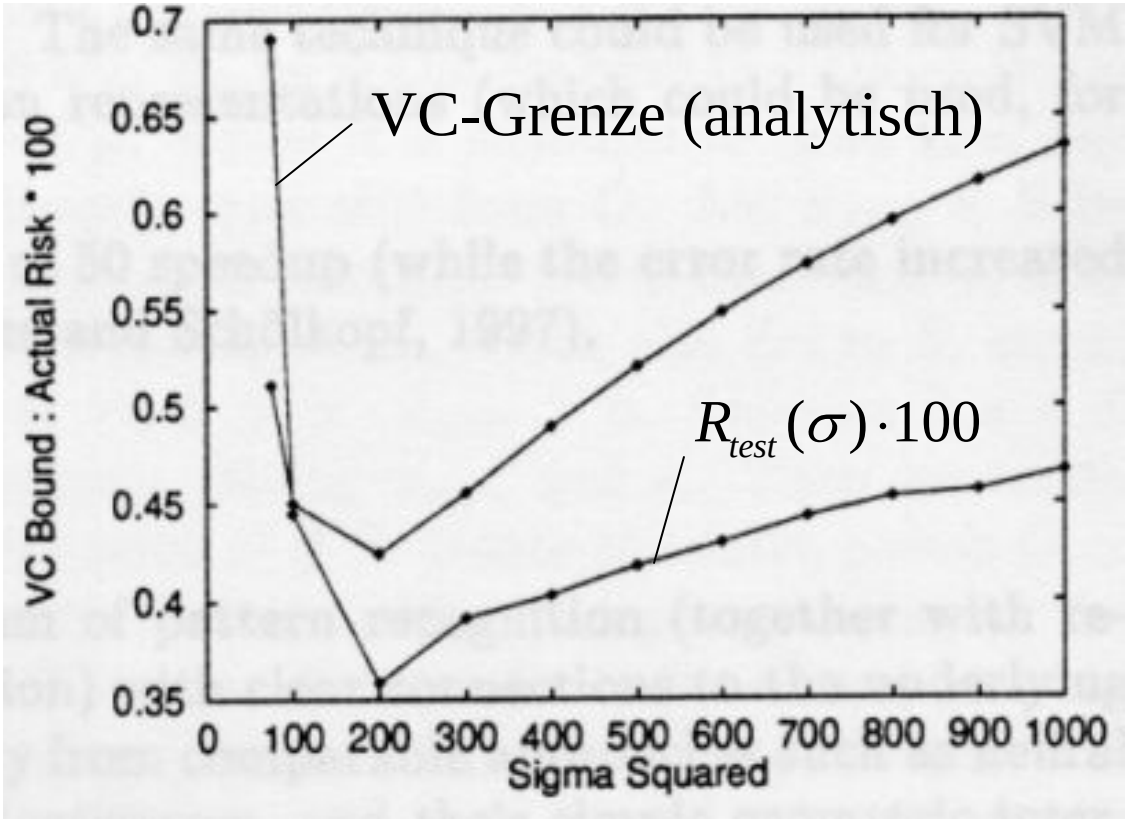
Ergebnisse:

Klassifikator	**Fehlerrate**
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 in alle Hauptrichtungen

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

(Nur ein Parameter: σ)



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

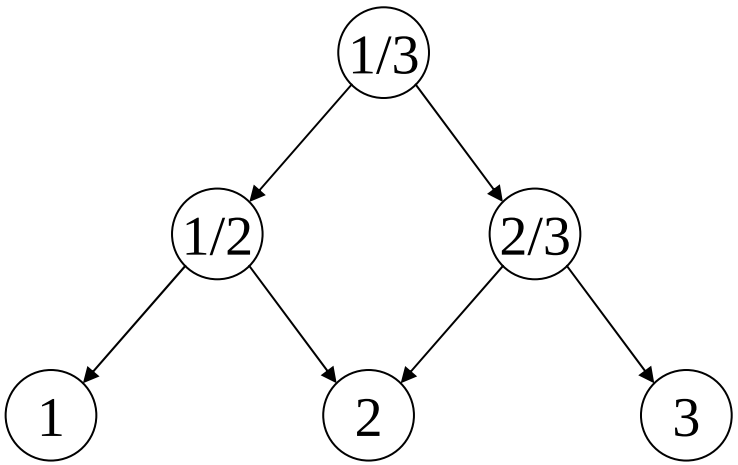
SVM: Eigenschaften und Berechnungskomplexität

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,σ), es wird kein a-priori-Wissen benötigt, kein Design); die genaue Wahl von (C,σ) ist jedoch zugleich eine der größten Schwächen im Entwurf
- Geometrisch anschauliche Funktionsweise
- Theoretische Aussagen über Ergebnis: globales Optimum, Generalisierungsfähigkeit
- SRM 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, d.h. Aufwand und Speicherbedarf steigt mit der Anzahl der Klassen



Reduktion eines Multiklassenproblems auf eine Reihe von binären Entscheidungsproblemen

Schwächen:

- Keine quantitative Qualitätsaussage der Klassifikation
- Langsames, speicherintensives Lernen:

  Laufzeit: $O(N_s^3)$ (Inversion der Hesse-Matrix)

  Speicher: $O(N_s^2)$

- Ansatz zur Verbesserung: Zerlegen in Teilprobleme
- VC-Abschätzung nur sehr ungenau und praktisch wenig verwertbar
- Langsames Klassifizieren mit $O(MN_s)$, wobei $M=O(\dim(\mathcal{H}))$ (d.h. im Fall ohne Kernfunktionen $M=N$), aber bei geeignet gewählten Kernfunktionen gilt auch hier: $M=N$.

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.