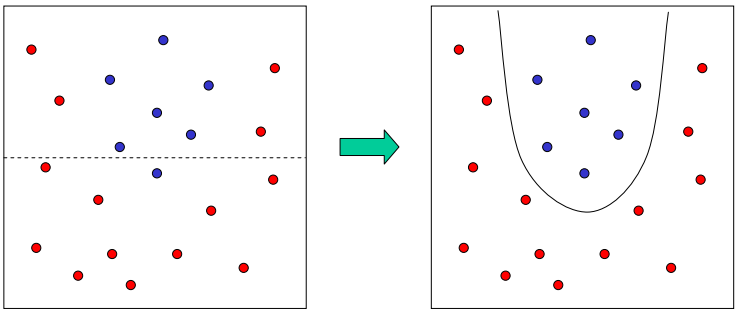
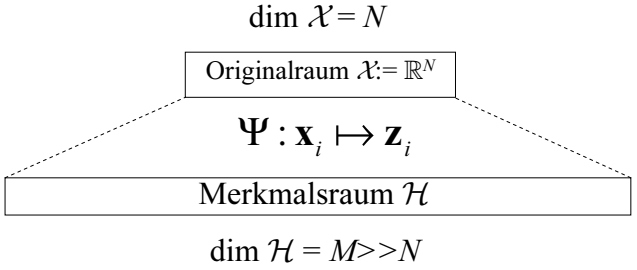
Nichtlineare Probleme

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



Erweiterung des Hypothesenraumes

Idee: Finde Hyperebene im höherdimensionalen Merkmalsraum

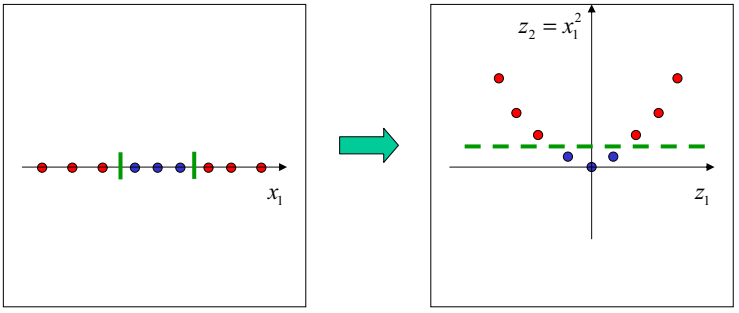


Die trennende Hyperebene im Merkmalsraum ist eine nichtlineare Trennfläche im Originalraum (siehe XOR-Problem mit Polynomklassifikator)

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

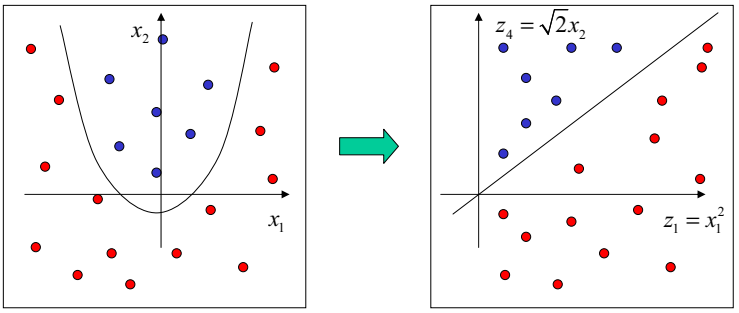
lineare Separation

Nichtlineare Probleme

- Originalraum:

  $\mathbf{x}=(x_1,x_2)$ (zweidimensional)
- Merkmalsraum:

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



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

nichtlineare Separation

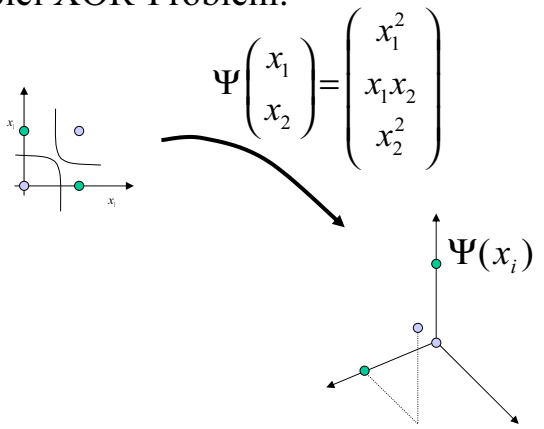
- $z_4 > \sqrt{2}z_1$
- $z_4 < \sqrt{2}z_1$

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

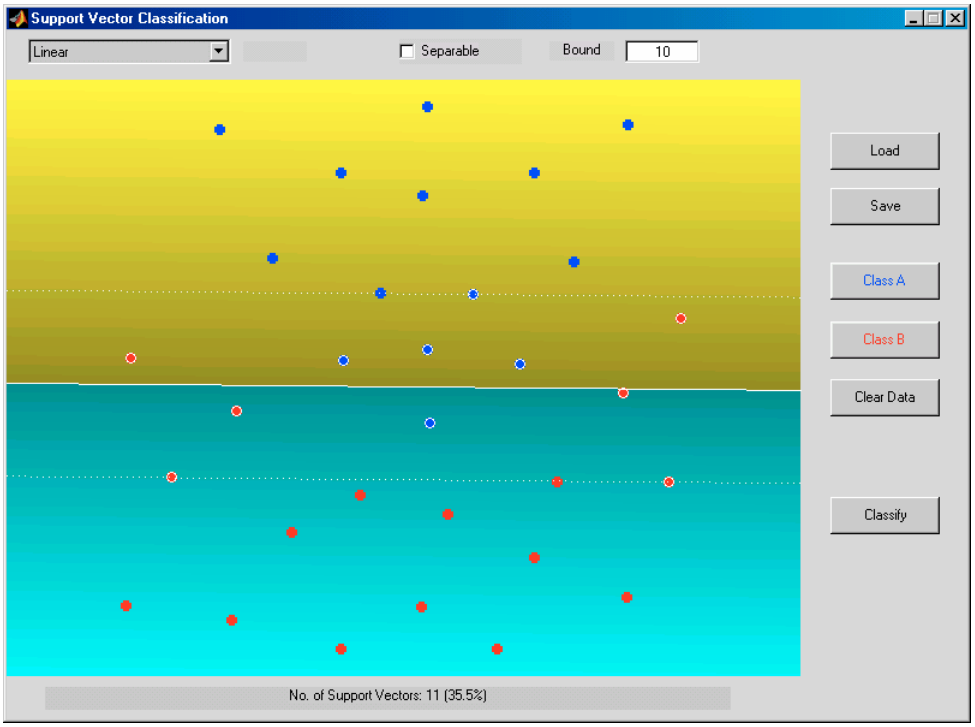
mit: $\sum_{i=1}^{l} y_i \alpha_i = 0 \quad \text{und} \quad 0 \leq \alpha_i \leq C$

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

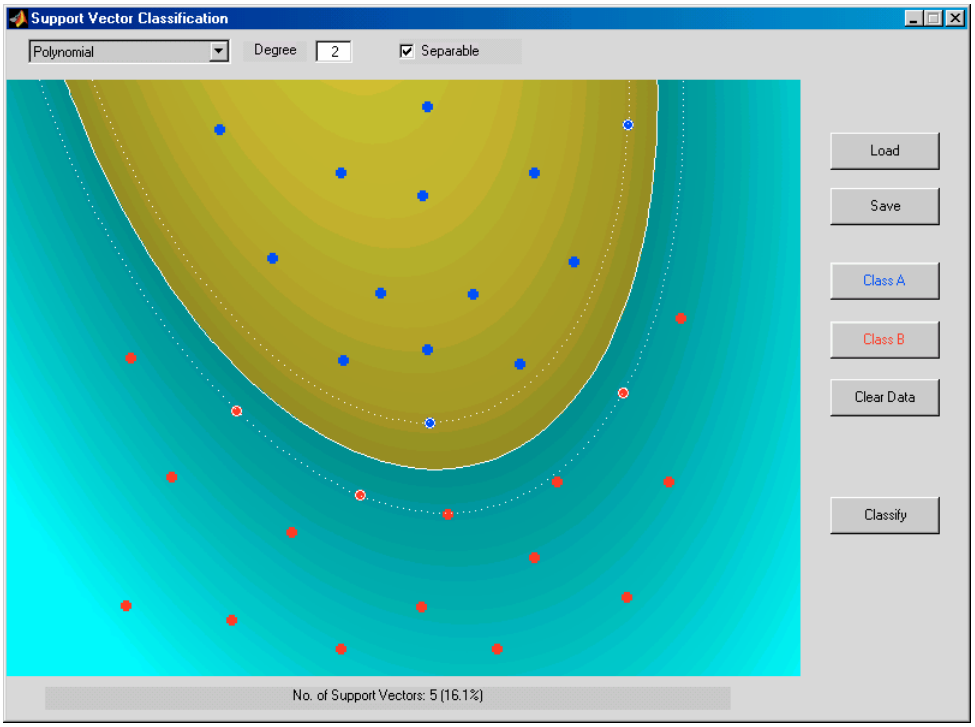
$$f(x) = \operatorname{sgn}\left(\sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}_j) + b \right) = \operatorname{sgn}\left(\sum_{\mathbf{x}_i \in SV} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}_j) + b \right)$$

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

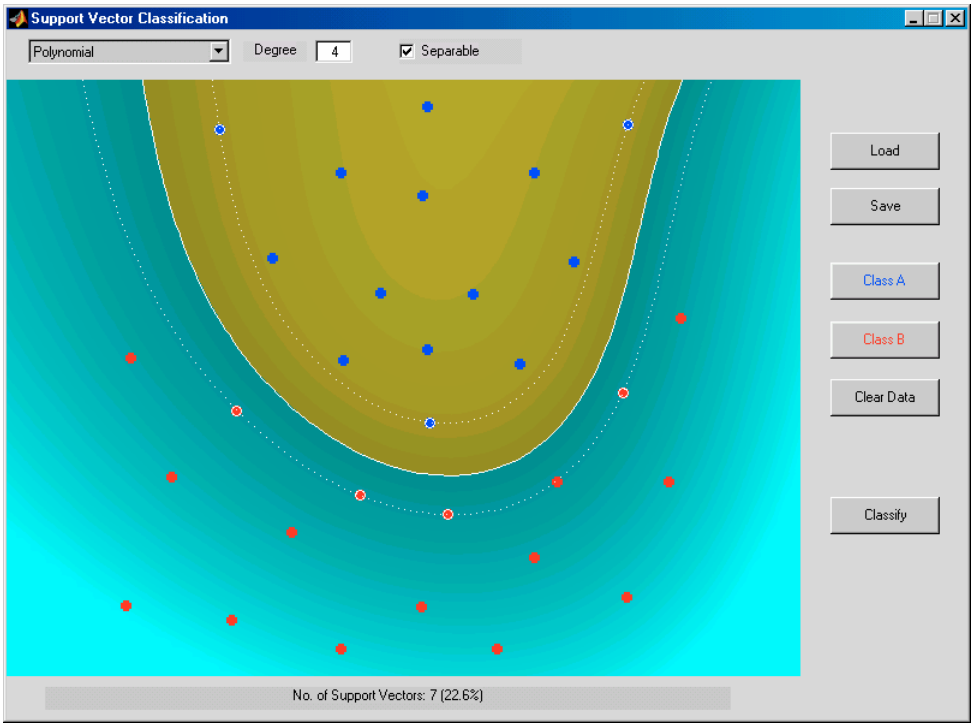
Lineare Separierung eines quadratischen Problems; weicher Rand



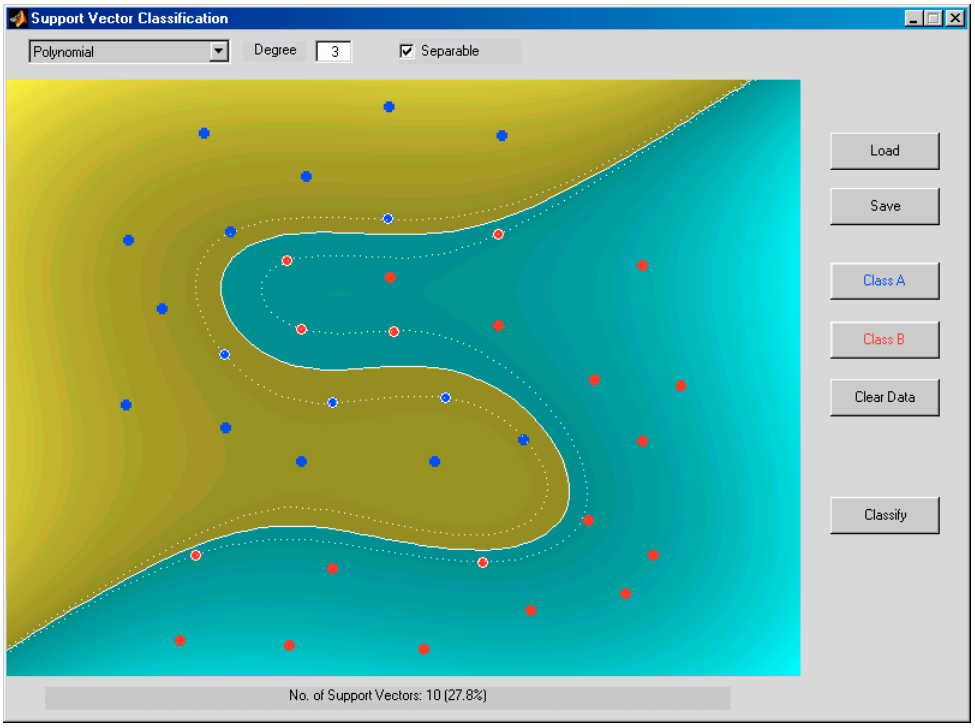
Polynomiale Separierung eines quadratischen Problems; harter Rand



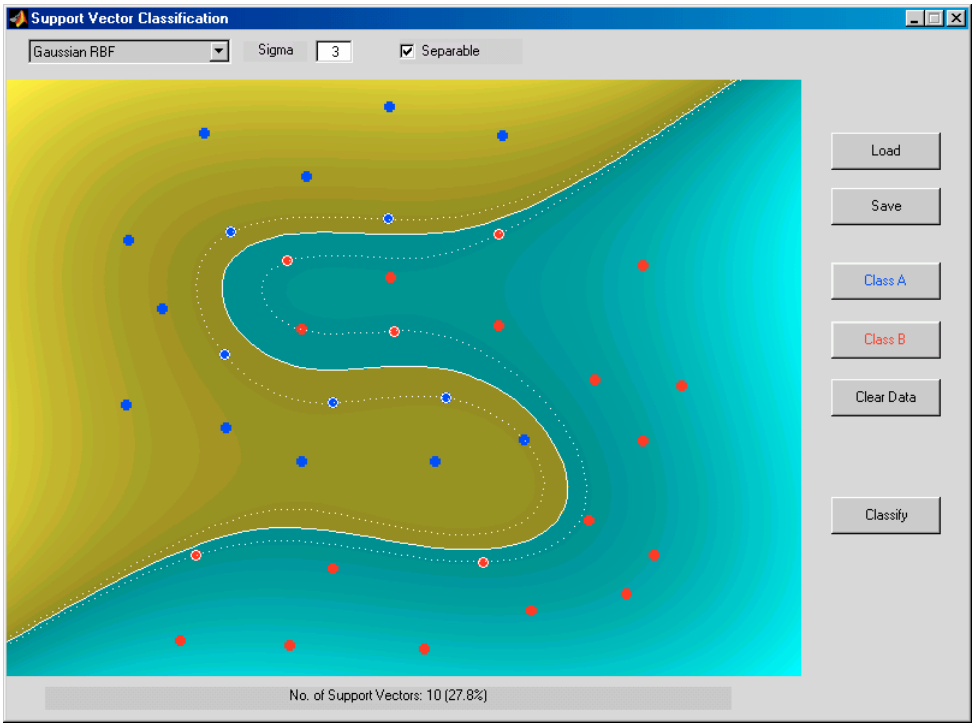
Polynomiale Separierung eines quadratischen Problems; harter Rand



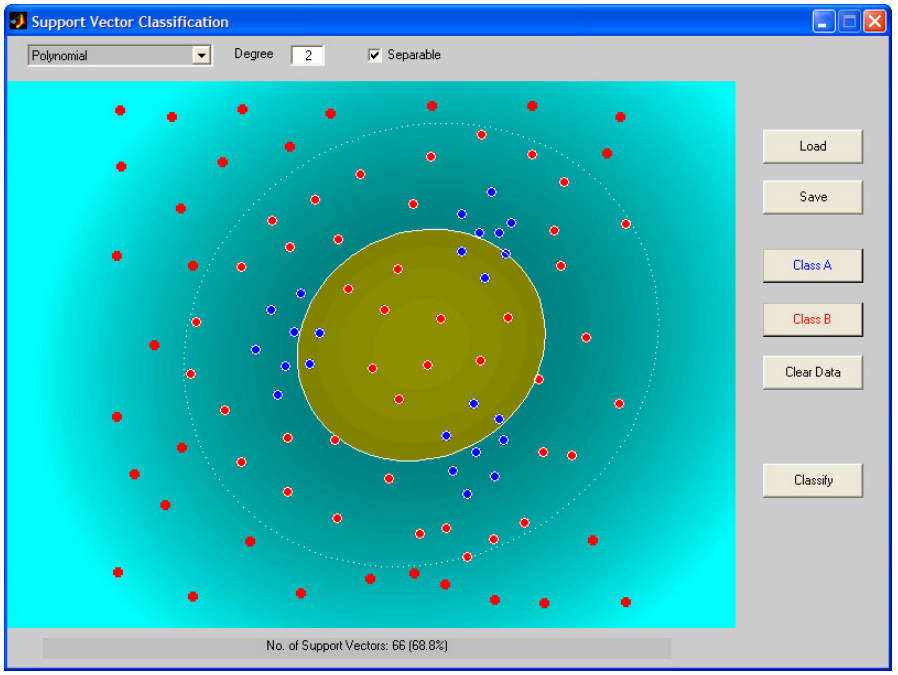
Nichtlinear separierbare Klassen; harter Rand



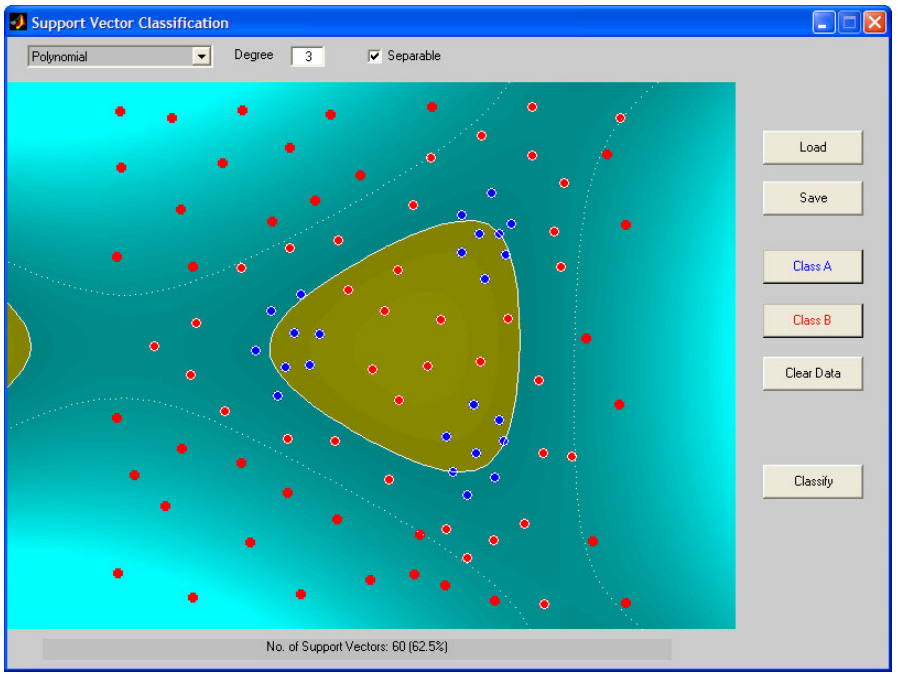
Nichtlinear separierbare Klassen; harter Rand



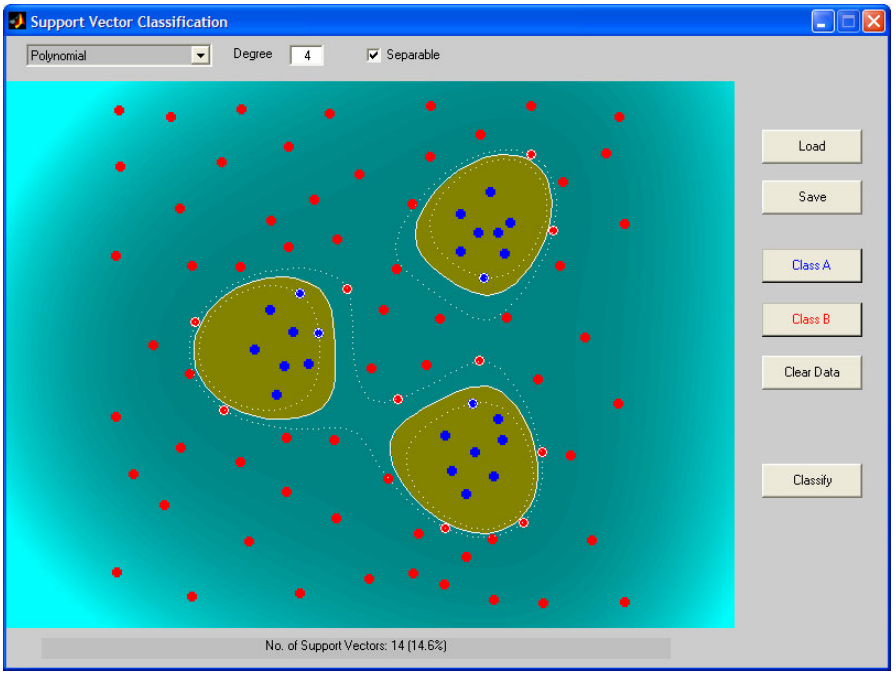
Beispiel: „Inseln“, Polynom mit $p = 2$



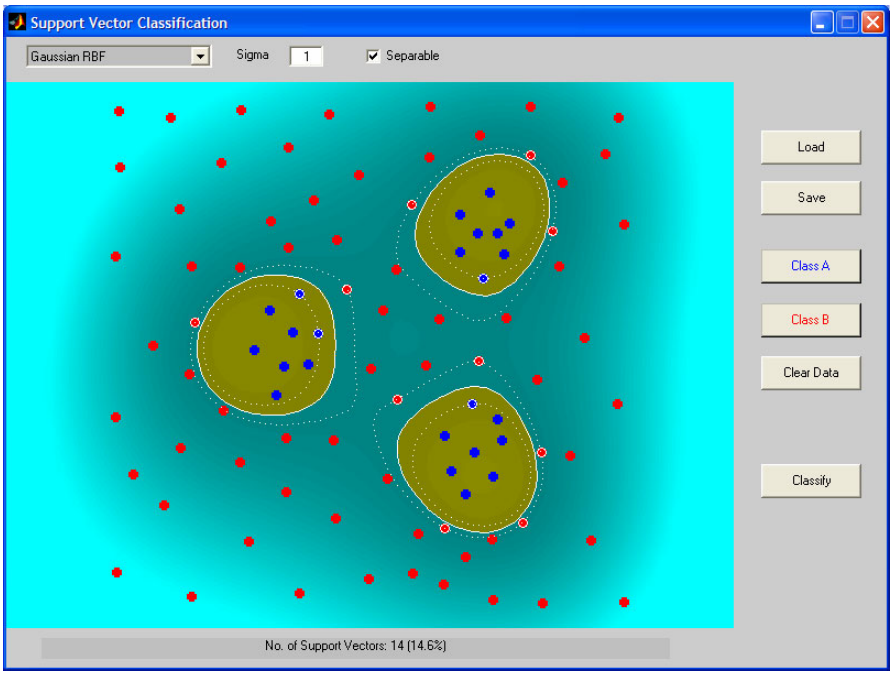
Beispiel: „Inseln“, Polynom mit $p = 3$

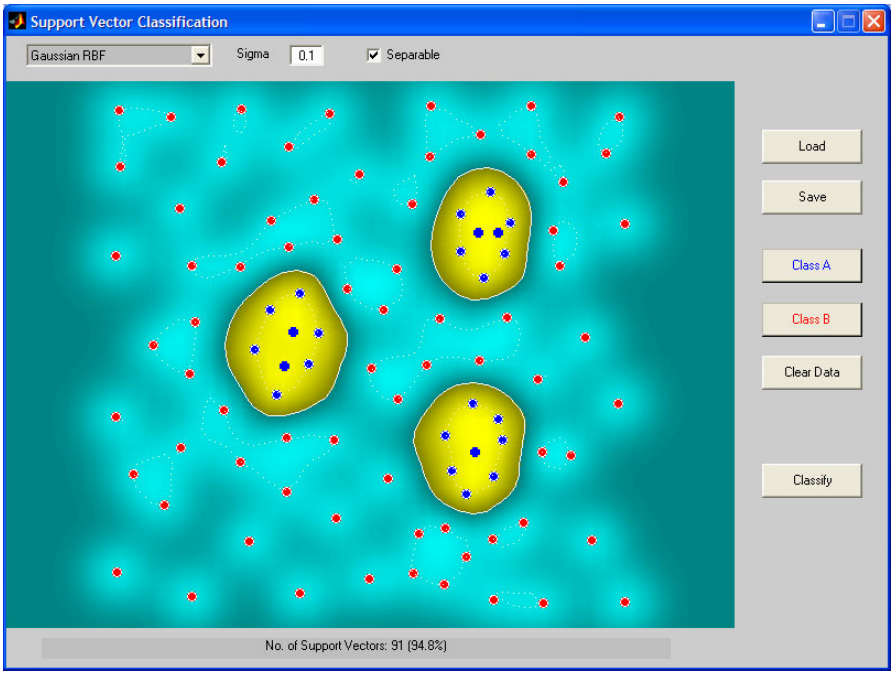


Beispiel: „Inseln“, Polynom mit $p = 4$



Beispiel: „Inseln“, RBFs mit σ=1





Anwendungsbeispiel: Zeichenerkennung

Beispiel: US Postal Service Digits [Schö95/96]

16x16 Grauwertbilder

7291 Training, 2007 Test-Samples

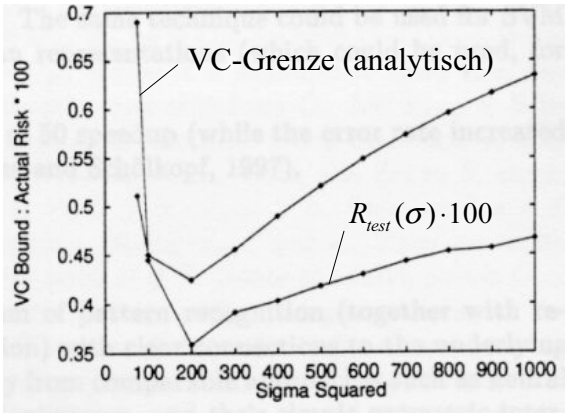
Ergebnisse:

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 (Aufwand kann reduziert werden auf SV, da nur diese einen Einfluss auf die gemessene Fehlerrate haben).

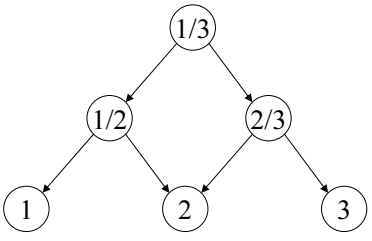
SVM: Eigenschaften und Berechnungskomplexität

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.