Kapitel 9

Polynomklassifikator
und
Radiale-Basis-Funktionen-Netzwerke
(RBFN)

Ansätze zum Entwurf eines Klassifikators

Es gibt zwei prinzipiell unterschiedliche Ansätze zum Entwurf eines Klassifikators:

1. Statistische parametrische Modellierung der Klassenverteilungen, dann MAP
2. Lösung eines Abbildungsproblems durch Funktionsapproximation (nichtlineare Regression)

$$\min E\left\{ \left\| \mathbf{f}(\mathbf{x}) - \mathbf{y} \right\|^2 \right\}$$

$\mathcal{X}$ – *Merkmalsraum*

$\mathcal{Y}$ – *Entscheidungsraum*

Zu 1.) Die zuerst beschriebene Vorgehensweise beim Entwurf eines Klassifikators beruht darauf, dass man die *klassenspezifischen Verteilungsdichten*

$$p(\mathbf{x}|\omega)$$

parametrisch durch statistische Modelle annähert (durch Schätzung der Parameter, z.Bsp. Einer Gauß-Verteilung) und durch eine Maximumselektion zu einer Entscheidung kommt. Lernen bedeutet hierbei: Verbesserung des Parameter-Fitting.

Zu 2.) Es gibt nun eine zweite Möglichkeit der Herangehensweise, welche auf die Auswertung der a-posteriori-Wahrscheinlichkeitsdichte

$$p(\omega|\mathbf{x})$$

aufbaut und durch ein Problem der *Funktionsapproximation* beschrieben werden kann.

Diese Funktionsapproximation kann z.B. durch eine *nichtlineare Regression mit Polynomen* oder auch mit Hilfe eines *künstlichen Neuronalen Netzwerkes* (NN) durchgeführt werden. Im Rahmen dieser Veranstaltung sollen die Grundlagen für beide Vorgehensweisen behandelt werden.

Grundsätzlich ist die Suche nach einer besten Näherungsfunktion ein *Variationsproblem*, welches durch die Wahl von Basisfunktionen auf ein *parametrisches Optimierungsproblem* zurückgeführt werden kann. Lernen bedeutet dann auch hier: Parameterfitting.

Die Gleichwertigkeit der beiden genannten Vorgehensweisen ergibt sich aus dem Bayes-Theorem:

$$p(\omega_i \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \omega_i) P(\omega_i)}{\cancel{p(\mathbf{x})}}$$

$p(\mathbf{x})$ unabhängig von ω

Die Gleichwertigkeit ergibt sich daraus, dass der Nenner unabhängig von ω ist.

Im folgenden soll nun die Überführung in ein Funktionsapproximations-problem durch eine Reihe begründet werden.

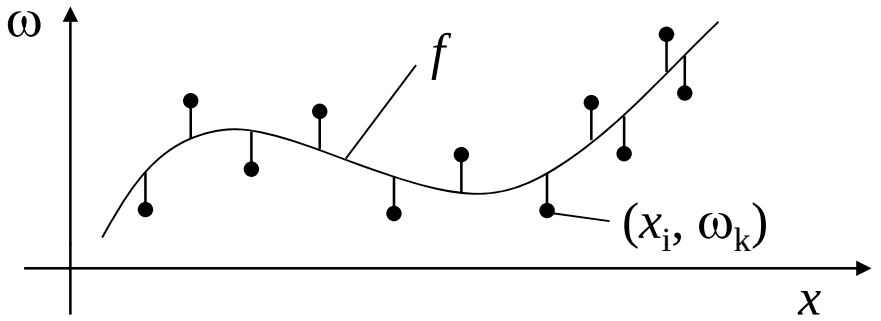
Bei bekannter a-posteriori-W. $p(\omega|\mathbf{x})$ würde für jedes kontinuierliche x bestmöglich ein ω zugeordnet werden (funktionale Zuordnung f: $\mathbf{x} \rightarrow \omega$). Gegeben sind jedoch nur *Stichproben* und man sucht nach einer Funktion f, welche bestmöglich die Einzelexperimente fittet und damit eine Abbildung realisert:

$$\mathbf{f}: \quad \underset{\mathbf{x}}{\text{Merkmalsraum}} \rightarrow \underset{\omega}{\text{Bedeutungsraum}}$$

Diese Aufgabenstellung kann mit Hilfe der Variationsrechnung gelöst werden.

Wählt man als Gütekriterium das minimale Fehlerquadrat, so geht es um die Minimierung von:

$$J = \min_{\mathbf{f}(\mathbf{x})} E\{\left\|\mathbf{f}(\mathbf{x}) - \mathbf{y}\right\|^2\}$$



Dabei entstehen die Zielvektoren $\{\mathbf{y}_i\}$ im *Entscheidungsraum* $\mathcal{Y}$
durch einfache Abbildung der skalaren Bedeutungswerte $\{\omega_i\}$

$$\Omega := \{\omega_1, \omega_2, \ldots, \omega_K\}$$

$$\updownarrow \quad \updownarrow \quad \updownarrow$$

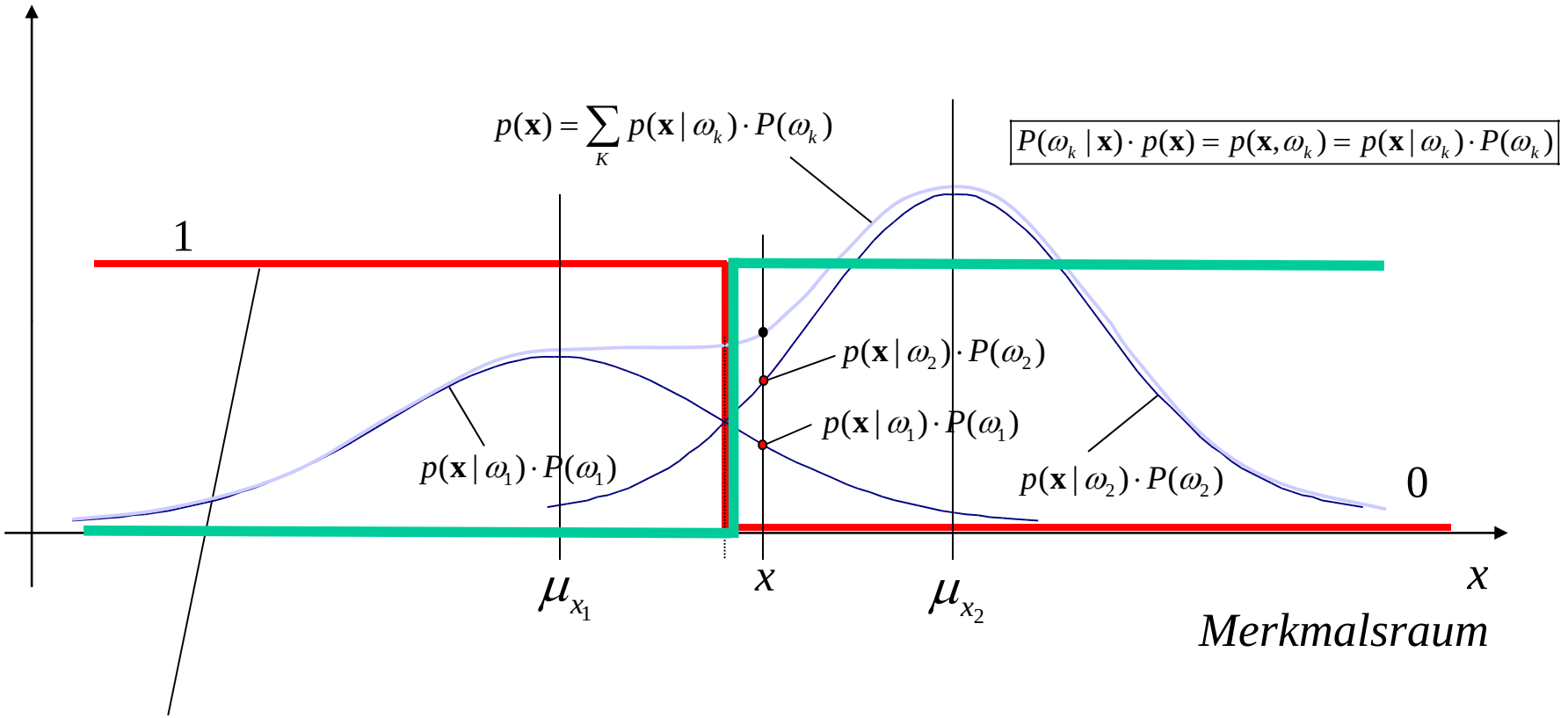
$$\mathcal{Y} := \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_K\}$$

mit:

$$\mathbf{y}_i = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

i-ter Einheitsvektor

Zwei-Klassen-Problem mit Gaußverteilungsdichte

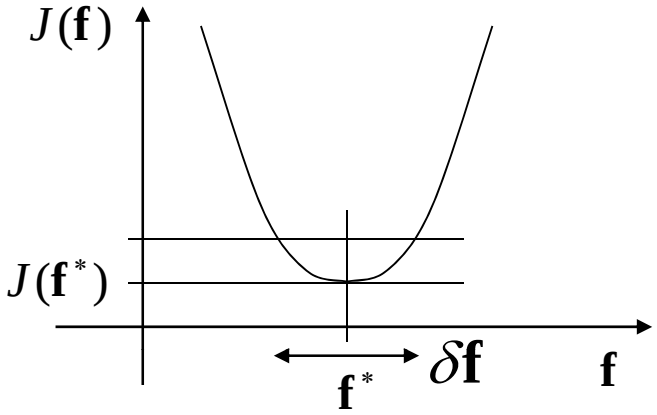


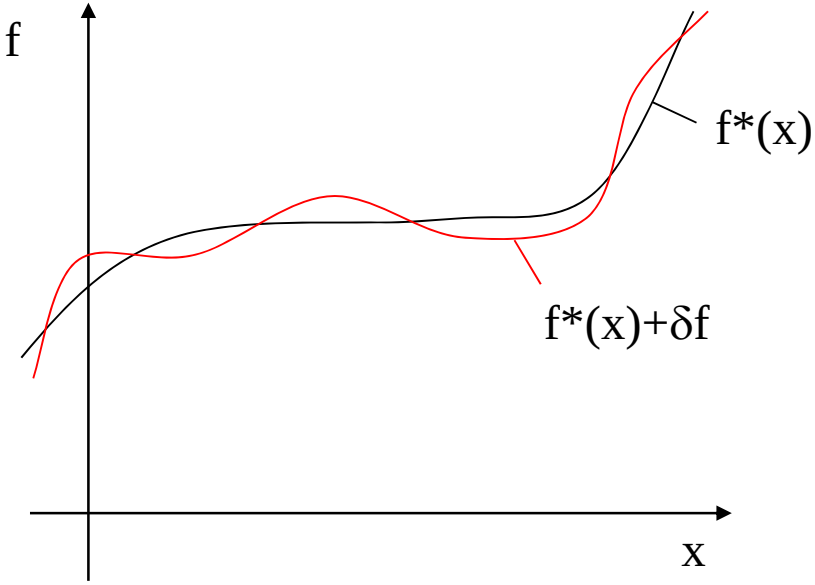
Entscheidungsfunktion für Klasse 1

Lösung des Variationsproblems

Die Lösung des Variationsproblems erhält man über die Annahme, dass man eine optimale Funktion **f***(**x**) bereits kennt. Dann muss jede Variation zu einer Verschlechterung des Gütekriteriums führen:

$$J(\mathbf{f}^* + \delta\mathbf{f}) > J(\mathbf{f}^*) \qquad \text{für} \quad \forall \delta\mathbf{f} \neq \mathbf{0}$$





Mit Hilfe der Definitionsgleichung ergibt sich:

$$J(\mathbf{f}^* + \delta\mathbf{f}) = E\left\{ \left\| \mathbf{f}^* + \delta\mathbf{f} - \mathbf{y} \right\|^2 \right\} = E\left\{ \langle (\mathbf{f}^* - \mathbf{y}) + \delta\mathbf{f}, (\mathbf{f}^* - \mathbf{y}) + \delta\mathbf{f} \rangle \right\}$$

$$= E\left\{ \left\| \mathbf{f}^* - \mathbf{y} \right\|^2 \right\} + 2E\left\{ \delta\mathbf{f}^T (\mathbf{f}^* - \mathbf{y}) \right\} + E\left\{ \left\| \delta\mathbf{f} \right\|^2 \right\}$$

$$\Rightarrow J(\mathbf{f}^*) = E\left\{ \left\| \mathbf{f}^* - \mathbf{y} \right\|^2 \right\} \quad (\text{für } \delta\mathbf{f} = \mathbf{0})$$

Eingesetzt in die Ungleichung $J(\mathbf{f}^* + \delta\mathbf{f}) - J(\mathbf{f}^*) > 0$ ergibt:

$$E\left\{ \left\| \delta\mathbf{f} \right\|^2 \right\} + 2E\left\{ \delta\mathbf{f}^T (\mathbf{f}^* - \mathbf{y}) \right\} > 0 \qquad \forall \delta\mathbf{f} \neq 0$$

Dies wird erfüllt, wenn der zweite Term verschwindet,
da der erste Term positiv definit ist (siehe nächste Folie!).

Daraus resultiert eine notwendige Optimalitätsbedingung:

$$\boxed{E\left\{ \delta\mathbf{f}^T (\mathbf{f}^* - \mathbf{y}) \right\} = 0}$$

Zwischenrechnung:

Mit

$a := E\left\{ \left\| \delta\mathbf{f} \right\|^2 \right\}$ und b:=$2E\left\{ \delta\mathbf{f}^T (\mathbf{f}^* - \mathbf{y}) \right\}$

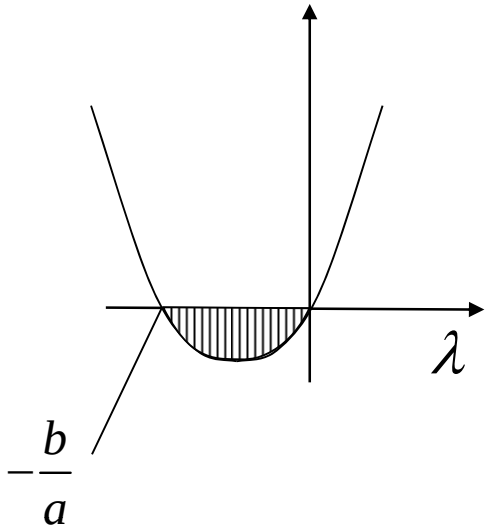
ergibt sich unter der Annahme einer Variation in

eine beliebige Raumrichtung von $\delta\mathbf{f}$:

$\delta\mathbf{f} \rightarrow \lambda\delta\mathbf{f}$

$a\lambda^2 + b\lambda > 0 \quad \forall\lambda$

$\lambda(a\lambda + b) > 0 \quad \forall\lambda$



Ist $b \neq 0$, so gilt für alle $\lambda \in (-b/a, 0)$:

$\lambda(a\lambda + b) < 0$

$\Rightarrow$ b muss Null sein, dann lautet die Parabel:

$a\lambda^2 > 0$

Der Erwartungswert kann mit der Verbundverteilungsdichte ausformuliert werden zu:

$$E\left\{\delta\mathbf{f}^T(\mathbf{f}^* - \mathbf{y})\right\} = \sum_{\mathbf{x}}\sum_{\mathbf{y}} \delta\mathbf{f}^T(\mathbf{f}^* - \mathbf{y}) \cdot p(\mathbf{x}, \mathbf{y})$$

$$= \sum_{\mathbf{x}} \delta\mathbf{f}^T \left[\sum_{\mathbf{y}} (\mathbf{f}^* - \mathbf{y}) \cdot p(\mathbf{y} \mid \mathbf{x})\right] p(\mathbf{x}) = \mathbf{0}$$

Dieser Ausdruck verschwindet, falls der Term in der eckigen Klammer Null wird.

Daraus resultiert die folgende Optimalitätsbedingung:

$$\sum_{\mathbf{y}} (\mathbf{f}^* - \mathbf{y}) \cdot p(\mathbf{y} \mid \mathbf{x}) = \mathbf{f}^* \underbrace{\sum_{\mathbf{y}} p(\mathbf{y} \mid \mathbf{x})}_{=1} - \sum_{\mathbf{y}} \mathbf{y} \cdot p(\mathbf{y} \mid \mathbf{x})$$

$$= \mathbf{f}^* - \sum_{\mathbf{y}} \mathbf{y} \cdot p(\mathbf{y} \mid \mathbf{x}) = \mathbf{0}$$

Die optimale Schätzfunktion ist die sogenannte Regressionsfunktion:

$$\mathbf{f}^*(\mathbf{x}) = \sum_{\mathbf{y}} \mathbf{y} \cdot p(\mathbf{y} \mid \mathbf{x}) = E\left\{\mathbf{y} \mid \mathbf{x}\right\}$$

Polynomiale Regression

Wahl von Polynomen als Basisfunktionen für die Funktionsapproximation.

Zunächst für eine skalare Funktion mit einem vektoriellen Argument **x**:

$$f(\mathbf{x}) = a_0 + a_1 x_1 + a_2 x_2 + \ldots + a_N x_N$$
$$+ a_{N+1} x_1^2 + a_{N+2} x_1 x_2 + a_{N+3} x_1 x_3 + \ldots \qquad \text{mit:} \quad N = \dim(\mathbf{x})$$
$$+ a_{\ldots} x_1^3 + a_{\ldots} x_1^2 x_2 + a_{\ldots} x_1^2 x_3 + \ldots$$

Dieses verallgemeinertes Polynom enthält eine Konstante a_0, gefolgt von N linearen Termen, $N(N+1)/2$ quadratischen Termen usw.

Ein Polynom vom *Grade G* und der *Dimension N* des Argumentvektors **x** hat

$$L = \binom{N+G}{G} = \frac{(N+G)!}{G!N!}$$

polynomiale Terme, welche als Basisfunktionen $f_i(\mathbf{x})$, i=1,2,...,L einer Funktionsentwicklung betrachtet werden können.

Obiges Polynom kann kompakt durch Einführung einer vektoriellen Abbildung eines N-dimensionalen Vektors $\mathbf{x}$ auf einen L-dimensionalen Vektor $\mathbf{p}$ beschrieben werden:

$$\mathbf{p}(\mathbf{x}) = \begin{bmatrix} 1 & x_1 & x_2 & \dots & x_N & x_1^2 & x_1 x_2 & x_1 x_3 & \dots & x_1^3 & x_1^2 x_2 & x_1^2 x_3 & \dots \end{bmatrix}^T$$

Wir führen einen Koeffizientenvektor $\mathbf{a}$ ein und erhalten für das skalare Polynom:

$$\boxed{f(\mathbf{x}) = \mathbf{a}^T \mathbf{p}(\mathbf{x}) = \sum_{i=0}^{L} a_i \, p_i(\mathbf{x})} \quad \text{mit: } L = \dim(\mathbf{p})$$

Für die Musterklassifikation benötigen wir für jede Klasse eine Polynomfunktion

$$f_k(\mathbf{x}) = \mathbf{a}_k^T \mathbf{p}(\mathbf{x}) \quad \text{verantwortlich für die Klassen}$$

$$k = 1, 2, \dots, K$$

Kombiniert man die klassenspezifischen Koeffizientenvektoren $\mathbf{a}_k$ zu einer Matrix

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \dots & \mathbf{a}_K \end{bmatrix}$$

so erhält man für die vektorielle Polynomfunktion:

$$\boxed{\mathbf{f}(\mathbf{x}) = \mathbf{A}^T \mathbf{p}(\mathbf{x})}$$

Optimale Anpassung der Matrix **A**

Es gilt nun, die Koeffizientenmatrix A während der Optimierungsprozedur bestmöglich anzupassen.

Dabei kann das Approximationsproblem direkt auf die Messungen oder aber auf die Merkmale eines Unterraumes (z.B. nach einer KLT) angewendet werden.

Adaption der Koeffizientenmatrix **A**:

$$J = \min_{\mathbf{f}(\mathbf{x})} E\left\{ \left\| \mathbf{f}(\mathbf{x}) - \mathbf{y} \right\|^2 \right\} \approx \min_{\mathbf{A}} E\left\{ \left\| \mathbf{A}^T \mathbf{p}(\mathbf{x}) - \mathbf{y} \right\|^2 \right\}$$

Das *Variationsproblem* mit der Suche nach einer unbekannten Funktion wird durch die Wahl polynomialer Basisfunktionen auf ein *Parameteroptimierungsproblem* reduziert!

Das Gütekriterium *J* minimiert die Varianz des Residuums, nämlich den Fehler zwischen y und der Approximation $\mathbf{A}^T\mathbf{p}(\mathbf{x})$.

Die Lösung wird bestimmt unter Ausnutzung des Variationsansatzes, was zu einer notwendigen Bedingung führt:

$$J(\mathbf{A}^* + \delta\mathbf{A}) \geq J(\mathbf{A}^*) \qquad \text{für} \quad \forall \delta\mathbf{A} \neq \mathbf{0}$$

Es gilt nun, die Koeffizientenmatrix **A** durch eine Optimierungsprozedur bestmöglich anzupassen.

Adaption der Koeffizientenmatrix **A**:

Wegen

$$J(\mathbf{A}) = E\left\{ \left\| \mathbf{A}^T\mathbf{p}(\mathbf{x}) - \mathbf{y} \right\|^2 \right\} = E\left\{ \left[\mathbf{A}^T\mathbf{p} - \mathbf{y} \right]^T \left[\mathbf{A}^T\mathbf{p} - \mathbf{y} \right] \right\}$$

und $\Rightarrow$ $\underbrace{\mathbf{a}^T\mathbf{b}}_{\text{Skalarprodukt}} = \text{Spur}(\underbrace{\mathbf{b}\mathbf{a}^T}_{\substack{\text{dyad.} \\ \text{Produkt}}})$

sowie $\text{Spur}(\mathbf{PQ}) = \text{Spur}(\mathbf{QP})$

erhält man:

$$J(\mathbf{A}) = E\left\{ \text{Spur}\left[\left[\mathbf{A}^T\mathbf{p} - \mathbf{y} \right] \left[\mathbf{A}^T\mathbf{p} - \mathbf{y} \right]^T \right] \right\}$$

$$= \text{Spur}\left[E\left\{ \mathbf{y}\mathbf{y}^T \right\} \right] - 2\,\text{Spur}\left[\mathbf{A}^T E\left\{ \mathbf{p}\mathbf{y}^T \right\} \right] + \text{Spur}\left[\mathbf{A}^T E\left\{ \mathbf{p}\mathbf{p}^T \right\} \mathbf{A} \right]$$

$$= E\left\{ \left\| \mathbf{y} \right\|^2 \right\} - 2\,\text{Spur}\left[\mathbf{A}^T E\left\{ \mathbf{p}\mathbf{y}^T \right\} \right] + \text{Spur}\left[\mathbf{A}^T E\left\{ \mathbf{p}\mathbf{p}^T \right\} \mathbf{A} \right]$$

Wie zuvor angenommen, kann der Zielvektor **y** o.B.d.A. als Einheitsvektor formuliert sein und damit gilt:

$$J(\mathbf{A}) = 1 - 2\,\text{Spur}\left[\mathbf{A}^T E\left\{\mathbf{p}\mathbf{y}^T\right\}\right] + \text{Spur}\left[\mathbf{A}^T E\left\{\mathbf{p}\mathbf{p}^T\right\}\mathbf{A}\right]$$

Mit dem Variationsansatz ergibt sich:

$$\begin{aligned}
J(\mathbf{A}^* + \delta\mathbf{A}) &= 1 - 2\,\text{Spur}\left[(\mathbf{A}^* + \delta\mathbf{A})^T E\left\{\mathbf{p}\mathbf{y}^T\right\}\right] \\
&\quad + \text{Spur}\left[(\mathbf{A}^* + \delta\mathbf{A})^T E\left\{\mathbf{p}\mathbf{p}^T\right\}(\mathbf{A}^* + \delta\mathbf{A})\right] \\
&= 1 - 2\,\text{Spur}\left[\mathbf{A}^{*T} E\left\{\mathbf{p}\mathbf{y}^T\right\}\right] - 2\,\text{Spur}\left[\delta\mathbf{A}^T E\left\{\mathbf{p}\mathbf{y}^T\right\}\right] \\
&\quad + \text{Spur}\left[\mathbf{A}^{*T} E\left\{\mathbf{p}\mathbf{p}^T\right\}\mathbf{A}^*\right] + 2\,\text{Spur}\left[\delta\mathbf{A}^T E\left\{\mathbf{p}\mathbf{p}^T\right\}\mathbf{A}^*\right] \\
&\quad + \text{Spur}\left[\delta\mathbf{A}^T E\left\{\mathbf{p}\mathbf{p}^T\right\}\delta\mathbf{A}\right]
\end{aligned}$$

Eingesetzt in die Ungleichung der notwendigen Bedingung für ein Optimum ergibt:

$$\text{Spur}\left[\delta\mathbf{A}^T E\left\{\mathbf{p}\mathbf{p}^T\right\}\delta\mathbf{A}\right] - 2\,\text{Spur}\left[\delta\mathbf{A}^T\left(E\left\{\mathbf{p}\mathbf{y}^T\right\} - E\left\{\mathbf{p}\mathbf{p}^T\right\}\mathbf{A}^*\right)\right] \geq 0$$

Da der erste Term positiv ist (wegen der Positiv-Definitheit von der Momentenmatrix $E\{\mathbf{p}\mathbf{p}^T\}$), muss der zweite Term verschwinden:

$$\text{Spur}\left[\delta\mathbf{A}^T\left(E\left\{\mathbf{p}\mathbf{y}^T\right\}-E\left\{\mathbf{p}\mathbf{p}^T\right\}\mathbf{A}^*\right)\right]=0 \qquad \forall\delta\mathbf{A}$$

und daraus schließlich das in **A*** *lineare* Gleichungssystem für den optimalen Polynomklassifikator:

$$\boxed{E\left\{\mathbf{p}\mathbf{p}^T\right\}\mathbf{A}^*=E\left\{\mathbf{p}\mathbf{y}^T\right\}}$$

$$\Rightarrow \boxed{\mathbf{A}^*=E\left\{\mathbf{p}\mathbf{p}^T\right\}^{-1}\cdot E\left\{\mathbf{p}\mathbf{y}^T\right\}=\mathbf{R}_{pp}^{-1}\cdot\mathbf{R}_{py}}$$

diese Gleichung garantiert ein minimales Residuum des Fehlervektors:

$$\Delta\mathbf{f}(\mathbf{x})=\mathbf{f}(\mathbf{x})-\mathbf{y}=\mathbf{A}^{*T}\mathbf{p}(\mathbf{x})-\mathbf{y}$$

Die Lösung des Problems führt auf die Aufgabe, die beiden Momentenmatrizen $E\{\mathbf{p}\mathbf{p}^T\}$ und $E\{\mathbf{p}\mathbf{y}^T\}$ aus dem Trainingsdatensatz zu schätzen und eine lineare Matrizengleichung zu lösen.

Der verbleibende Fehlervektor berechnet sich zu:

$$\boxed{J(\mathbf{A}^*)=1-\text{Spur}\left[\mathbf{A}^{*T}E\left\{\mathbf{p}\mathbf{p}^T\right\}\mathbf{A}^*\right]}$$

Orthogonalität des Schätzfehlervektors

Interpretation der Lösungsgleichung auf der Folie zuvor:

Die Bestapproximation fordert nach dem Projektionssatz, dass der Fehlervektor $\Delta\mathbf{f}$ senkrecht steht auf den Unterraum, welcher durch die polynomialen Basisfunkionen in $\mathbf{p}(\mathbf{x})$ aufgespannt wird.

In statistischer Notation bedeutet das, dass die Momentenmatrix, welche aus $\mathbf{p}$ und $\Delta\mathbf{f}$ aufgespannt wird, eine Nullmatrix sein muss:

$$E\left\{ \mathbf{p}\Delta\mathbf{f}^T \right\} = E\left\{ \mathbf{p}\left(\mathbf{A}^{*T}\mathbf{p} - \mathbf{y}\right)^T \right\} = E\left\{ \mathbf{p}\mathbf{p}^T \right\}\mathbf{A}^* - E\left\{ \mathbf{p}\mathbf{y}^T \right\} = \mathbf{0}$$

Diese Eigenschaft wird auch von $\mathbf{p}$ auf $\mathbf{f}$ vererbt, da $\mathbf{f}$ aus Linearkombinationen der Basisfunktionen in $\mathbf{p}(\mathbf{x})$ aufgebaut wird.

$$E\left\{ \mathbf{f}^*\Delta\mathbf{f}^T \right\} = E\left\{ \mathbf{A}^{*T}\mathbf{p}\Delta\mathbf{f}^T \right\} = \mathbf{A}^{*T}E\left\{ \mathbf{p}\Delta\mathbf{f}^T \right\} = \mathbf{0}$$

Erwartungstreue der Approximationsfunktion

Die Schätzfunktion ist erwartungstreu (unbiased). Der Schätzfehler (Residuum) hat den Erwartungswert **0**.

$$E\{\Delta\mathbf{f}\} = E\{\mathbf{f}(\mathbf{x}) - \mathbf{y}\} = \mathbf{0}$$

Daraus folgt, dass **f**(**x**) eine erwartungstreue Schätzung von **y** ist:

$$E\{\mathbf{f}(\mathbf{x})\} = E\{\mathbf{y}\}$$

Rekursive Lernregel für den Polynomklassifikator

(falls neue Werte in dem Experiment dazukommen können)

Für den auf das minimale Fehlerquadrat aufbauende Klassifikator ergibt sich die folgende rekursive Lernstrategie für die Koeffizientenmatrix **A**:

$$\mathbf{A}_n = \mathbf{A}_{n-1} - \alpha \underbrace{\left[\sum_{i=1}^{n} \mathbf{p}_i \mathbf{p}_i^T \right]^{-1}}_{\mathbf{G}^{-1}} \mathbf{p}_n \left[\mathbf{A}_{n-1}^T \mathbf{p}_n - \mathbf{y}_n \right]^T$$

Diese Lernregel beinhaltet folgende Schritte:

- Wahl eines Startwerts für **A**
- Die momentante Stichprobe $[\mathbf{x}, \mathbf{y}]$ wird abgebildet mit Hilfe der Polynombasis $\mathbf{p}(\mathbf{x})$ auf das Paar $[\mathbf{p}, \mathbf{y}]$
- Berechne die Schätzung $\mathbf{f}$ auf der Basis der gegebenen Beobachtung $\mathbf{p}$ und der momentanten Koeffizientenmatrix $\mathbf{A}$: $\mathbf{f} = \mathbf{A}^T \mathbf{p}$
- Berechne das Residuum: $\Delta \mathbf{f} = \mathbf{f} - \mathbf{y}$
- Korrigiere $\mathbf{A}_n$ auf der Grundlage obiger Rekursion

$$\mathbf{A}_n = \mathbf{A}_{n-1} - \alpha \mathbf{G}^{-1} \mathbf{p}_n \Delta \mathbf{f}^T$$

Die Gewichtsmatrix $\mathbf{G} = E\{\mathbf{p}\mathbf{p}^T\}$ ist dabei die Momentenmatrix auf der Basis der n zur Verfügung stehenden Stichproben.

Eine strikte Beachtung obiger Rekursion, würde sogar noch eine weitere Rekursion für die Inverse der Gewichtsmatrix erfordern gemäß:

$$\mathbf{G}_n^{-1} = \frac{1}{1-\alpha}\left[\mathbf{G}_{n-1}^{-1} - \alpha\frac{\mathbf{G}_{n-1}^{-1}\mathbf{p}_n\mathbf{p}_n^T\mathbf{G}_{n-1}^{-1}}{1+\alpha\left(\mathbf{p}_n^T\mathbf{G}_{n-1}^{-1}\mathbf{p}_n - 1\right)}\right]$$

$\mathbf{G}$ oder die echte zusätzliche Iteration ändern fast nichts am Ergebnis. $\mathbf{G}$ kann sogar noch vereinfacht werden zu:

$$\mathbf{G} = \underbrace{E\left\{\mathbf{p}\mathbf{p}^T\right\}}_{\approx\frac{1}{n}\sum_{i=1}^{n}\mathbf{p}_i\mathbf{p}_i^T} \approx \underbrace{E\left\{\left[\mathrm{diag}(\mathbf{p}_i)\right]^2\right\}}_{\approx\frac{1}{n}\sum_{i=1}^{n}\left[\mathrm{diag}(\mathbf{p}_i)\right]^2} \approx \mathbf{I}$$

ohne dass sich dabei die Konvergenzeigenschaften verschlechtern (Schürmann S. 174).

Herleitung der rekursiven Lernregel

$E\left\{\mathbf{p}\mathbf{p}^T\right\}\mathbf{A} = E\left\{\mathbf{p}\mathbf{y}^T\right\}$ opt. Lösung des Polynomkl.

Einführung von Schätzwerten:

$$\underbrace{\left(\frac{1}{n}\sum_{i=1}^{n}\mathbf{p}_i\mathbf{p}_i^T\right)}_{\mathbf{G}_n}\mathbf{A}_n = \underbrace{\frac{1}{n}\sum_{i=1}^{n}\mathbf{p}_i\mathbf{y}_i^T}_{\mathbf{H}_n}$$

$\mathbf{G}_n\mathbf{A}_n = \mathbf{H}_n$ und

$$\mathbf{G}_n = (1-\alpha)\mathbf{G}_{n-1} + \alpha\mathbf{p}_n\mathbf{p}_n^T$$
$$\mathbf{H}_n = (1-\alpha)\mathbf{H}_{n-1} + \alpha\mathbf{p}_n\mathbf{y}_n^T$$

$$\Rightarrow \mathbf{G}_n\mathbf{A}_n = (1-\alpha)\mathbf{H}_{n-1} + \alpha\mathbf{p}_n\mathbf{y}_n^T$$
$$= (1-\alpha)\mathbf{G}_{n-1}\mathbf{A}_{n-1} + \alpha\mathbf{p}_n\mathbf{y}_n^T$$
$$= \left(\mathbf{G}_n - \alpha\mathbf{p}_n\mathbf{p}_n^T\right)\mathbf{A}_{n-1} + \alpha\mathbf{p}_n\mathbf{y}_n^T$$
$$= \mathbf{G}_n\mathbf{A}_{n-1} - \alpha\mathbf{p}_n(\mathbf{p}_n^T\mathbf{A}_{n-1} - \mathbf{y}_n^T)$$
$$\Rightarrow \boxed{\mathbf{A}_n = \mathbf{A}_{n-1} - \alpha\mathbf{G}_n^{-1}\mathbf{p}_n\left[\mathbf{A}_{n-1}^T\mathbf{p}_n - \mathbf{y}_n\right]^T}$$

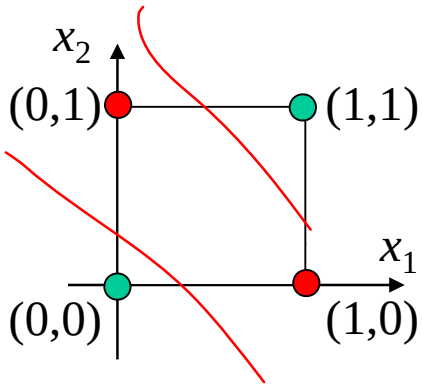
Polynomklassifikator für das XOR-Problem

Wir wählen als Regressionsfunktion ein Polynom mit Termen zweiten Grades in der Hoffnung, dass damit eine Lösung erreicht werden kann (eine lineare Lösung existiert nicht!):

$$f(\mathbf{x}) = a_1 x_1 + a_2 x_2 + a_3 x_1 x_2 = \mathbf{a}^T \mathbf{p}(\mathbf{x}) = \sum_{i=0}^{L} a_i \, p_i(\mathbf{x})$$

mit:

$$\mathbf{p} = \left[x_1, x_2, x_1 x_2 \right]^T \quad \text{und} \quad \mathbf{a} = \left[a_1, a_2, a_3 \right]^T$$

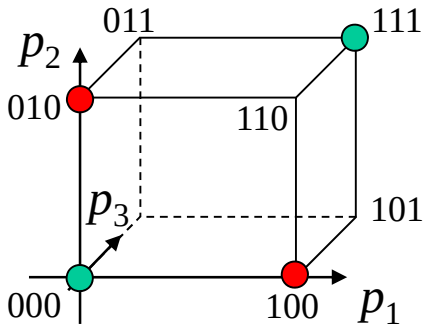


$\mathcal{X}$-Raum

Die Eckpunkte des Quadrats im *zweidimensionalen* Ursprungsraum $\mathcal{X}$ werden auf die Ecken eines (Hyper-) Würfels eines *dreidimensionalen* Merkmalraums $\mathcal{P}$ abgebildet und dieser schließlich auf den eindimensionalen Bedeutungsraum $\mathcal{Y}$:

$$\mathcal{X} \rightarrow \mathcal{P} \rightarrow \mathcal{Y}$$

$\mathcal{X}$	x_1	0	1	0	1
	x_2	0	0	1	1
$\mathcal{P}$	p_1	0	1	0	1
	p_2	0	0	1	1
	p_3	0	0	0	1
$\mathcal{Y}$	y	0	1	1	0



$\mathcal{P}$-Raum

Die optimalen Parameter der Regressionsfunktion **a*** ergeben sich aus:

$$\underbrace{E\left\{\mathbf{p}\mathbf{p}^T\right\}}_{\mathbf{R}_{pp}}\mathbf{a}^* = \underbrace{E\left\{\mathbf{p}\mathbf{y}^T\right\}}_{\mathbf{R}_{py}} \quad \text{bzw:} \quad \mathbf{a}^* = \mathbf{R}_{pp}^{-1}\cdot\mathbf{R}_{py}$$

mit: $\mathbf{R}_{pp} = E\left\{\mathbf{p}\mathbf{p}^T\right\} \approx \frac{1}{4}\sum_{i=1}^{4}\mathbf{p}_i\mathbf{p}_i^T$

$$= \frac{1}{4}\left\{\mathbf{0} + \left[\begin{array}{c|ccc} & 1 & 0 & 0 \\ \hline 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right] + \left[\begin{array}{c|ccc} & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right] + \left[\begin{array}{c|ccc} & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{array}\right]\right\} = \frac{1}{4}\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

und: $\mathbf{R}_{py} = E\left\{\mathbf{p}y\right\} \approx \frac{1}{4}\sum_{i=1}^{4}\mathbf{p}_i y_i = \frac{1}{4}\sum_{i=1}^{4} y_i\mathbf{p}_i$

$$= \frac{1}{4}\left\{0\cdot\begin{bmatrix}0\\0\\0\end{bmatrix} + 1\cdot\begin{bmatrix}1\\0\\0\end{bmatrix} + 1\cdot\begin{bmatrix}0\\1\\0\end{bmatrix} + 0\cdot\begin{bmatrix}1\\1\\1\end{bmatrix}\right\} = \frac{1}{4}\begin{bmatrix}1\\1\\0\end{bmatrix}$$

3 Parameter a_1, a_2, a_3 und 4 Beobachtungen => $\mathbf{R}_{pp}$ ist regulär!

Daraus folgt:

$$\mathbf{a}^* = \mathbf{R}_{pp}^{-1} \cdot \mathbf{R}_{py} = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \\ -1 & -1 & 3 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ -2 \end{bmatrix}$$

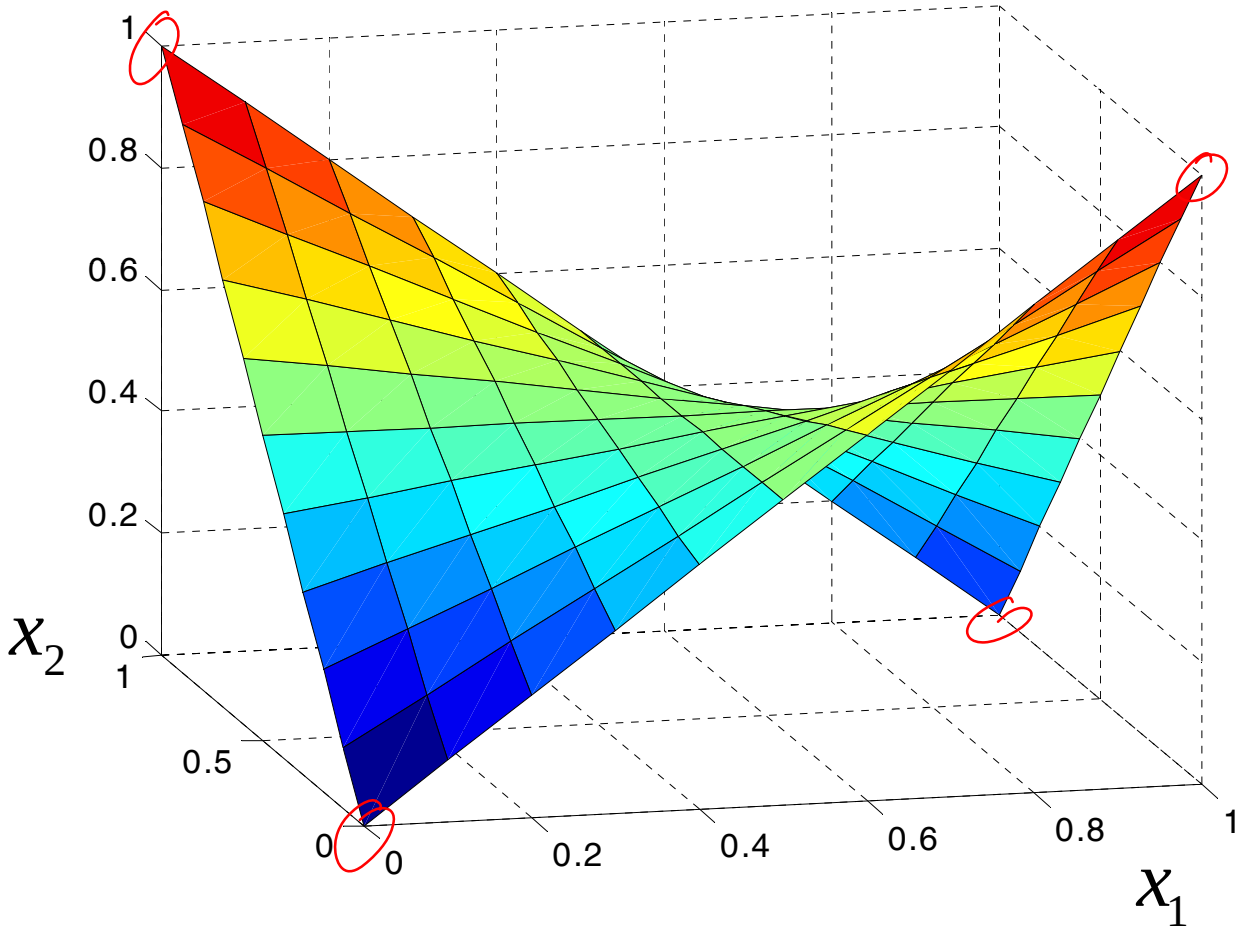
und damit eine Regressionsfunktion

$$\boxed{f(\mathbf{x}) = \left\langle \mathbf{a}^*, \mathbf{p} \right\rangle = a_1^* x_1 + a_2^* x_2 + a_3^* x_1 x_2 = x_1 + x_2 - 2x_1 x_2}$$

welche an den 4 Funktionswerten genau die Werte der Zielfunktion annimmt und ansonsten interpoliert!

Regressionsfunktion

$$f(\mathbf{x}) = x_1 + x_2 - 2x_1x_2$$



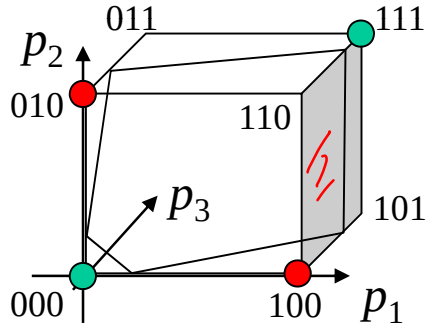
Matlab-Demo: XOR_poly.bat

Trennflächen für die beiden Klassen

Die Trennfläche für die beiden Bedeutungsklassen verläuft dort, wo die Regressionsfunktion $f(\mathbf{x})$ den arithmetischen Mittelwert der Zielfunktion, nämlich $c=(0+1)/2=0{,}5$ annimmt. Das nichtlinear separierbare XOR-Problem kann in dem dreidimensionalen Merkmalsraum durch eine *lineare* (Hyper-)Ebene separiert werden, was auch leicht geometrisch an dem dreidimensionalen Kubus nachvollziehbar ist:

$$f(\mathbf{x}) = p_1 + p_2 - 2p_3 - 0{,}5 = \langle \mathbf{a}^*, \mathbf{p} \rangle - 0{,}5 \begin{cases} > 0 \Rightarrow \mathbf{x} \in \mathcal{A} \\ < 0 \Rightarrow \mathbf{x} \in \mathcal{B} \end{cases}$$

Gleichung für *lineare* Trennfläche mit Normalenvktor $\mathbf{a}^*$



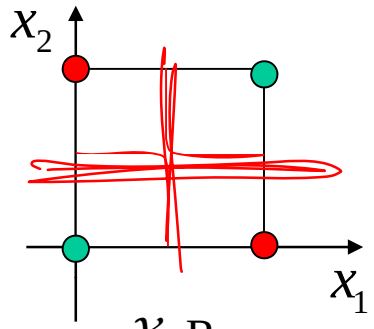
$\mathcal{P}$ -Raum

Nichtlineare Trennfunktion im Originalraum:
(Kegelschnitt, nämlich zwei Geraden)

$$f(\mathbf{x}) = x_1 + x_2 - 2x_1x_2 - 0{,}5 \begin{cases} > 0 \Rightarrow \mathrm{x} \in \mathrm{A} \\ < 0 \Rightarrow \mathrm{x} \in \mathrm{B} \end{cases}$$

$$\Rightarrow \left(x_1 + x_2 - 2x_1x_2 - 0{,}5\right) = -2(x_1 - 0{,}5)(x_2 - 0{,}5) = 0$$

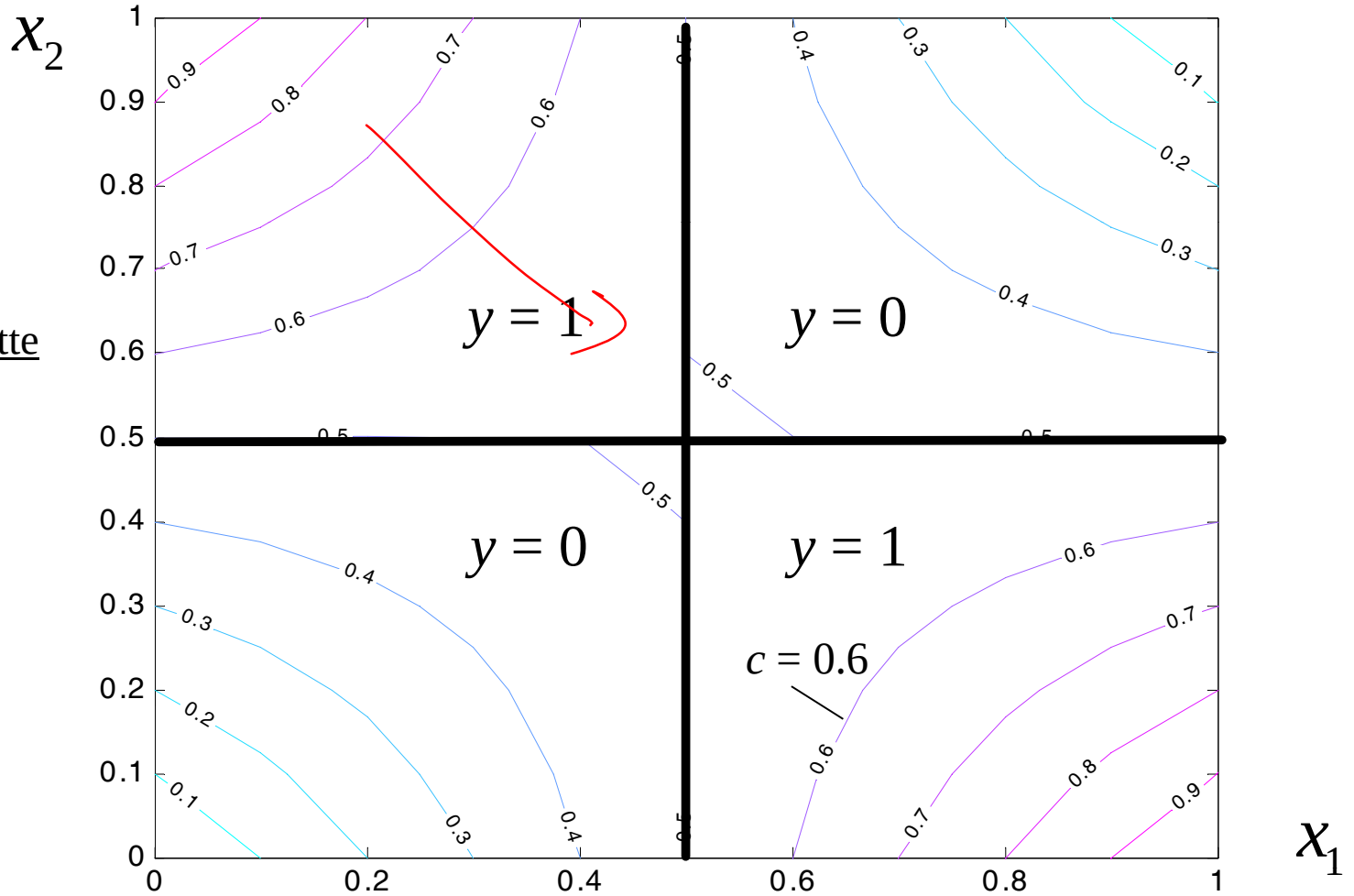
gilt für: $\begin{cases} x_1 = 0{,}5 & x_2 \text{ beliebig} \\ x_2 = 0{,}5 & x_1 \text{ beliebig} \end{cases}$ Euklidisch ideale Trennlinien



$\mathcal{X}$ -Raum

Ortskurven konstanter Wertigkeit im Originalraum (Hyperbeln) für unterschiedliche Schwellwerte c

$$f(\mathbf{x}) = x_1 + x_2 - 2x_1x_2 = c$$



Trennkurven: Kegelschnitte
Beim Übergang c→0,5 entarten die Hyperbeln in zwei Geraden

Trennflächen für die beiden Klassen

Wir erhalten einen

- *linearen* Klassifikator im **p**-Raum

und einen

- *nichtlinearen* Klassifikator im **x**-Raum

Polynomklassifikator für komplexere Beispiele

$$f(\mathbf{x}) = a_0 + \sum_{i=1}^{N} a_i x_i + \sum_{i=1}^{N-1} \sum_{m=i+1}^{N} a_{im} x_i x_m + \ldots = P(x_1) \times P(x_2) \times \ldots \times P(x_N)$$

kartesisches Produkt der eindimensionalen Polynome mit:

$\dim(\mathbf{x}) = N$ und Grad des Polynoms: G

d.h. der N-dimensionale Originalraum wird in einen L-dimensionalen Merkmalsraum abgebildet. Dieser enthält Terme der Art:

$$x_1^{k_1} x_2^{k_2} \cdots x_N^{k_N} \quad \text{mit:} \quad k_1 + k_2 + \ldots + k_N \leq G$$

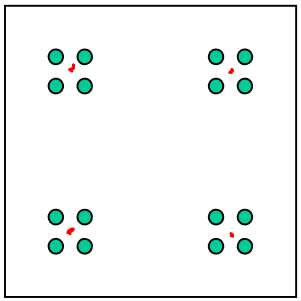
Für ein Polynom vom Grade G und ein Merkmalsvektor der Dimension N ergibt sich eine Dimension des Ausgangsraums L von:

$$L = \binom{N+G}{G} = \frac{(N+G)!}{G!N!}$$

Demo mit MATLAB
(Klassifikationgui.m)

- Polynomklassifikator für XOR-Problem
  - zuerst setmypath.m aufrufen, dann
  - C:\Home\ppt\Lehre\ME_2002\matlab\KlassifikatorEntwurf-WinXX\Klassifikationgui
  - Datensatz: samples/xor_test_exakt.mat laden
  - Polynomklassifikator vom Grad 2 anwenden
- Zweiklassenproblem mit Bananenshape
  - Datensatz: samples/banana_train_samples.mat laden
  - Versuch mit Polynomklassifikator vom Grad 2 und Grad 3

um genügend unabhängige Stichproben für das Polynom zu haben:



Man benötigt mindestens:

$$L = \binom{N+G}{G} = \frac{(N+G)!}{G!N!} = \binom{2+2}{2} = 6$$

unabhängige Beobachtungen.

Start der Matlab-Demo
matlab-Klassifikation_gui.bat

Man benötigt mindestens:

$$L = \binom{N+G}{G} = \frac{(N+G)!}{G!N!} = \binom{2+2}{2} = 6$$

unabhängige Beobachtungen.

Man benötigt mindestens:

$$L = \binom{N+G}{G} = \binom{2+3}{3} = 10$$

unabhängige Beobachtungen.

Eigenschaften des Polynomklassifikators:

- Vorteile:
  - Lineare Entwurfsgleichungen mit expliziter Lösung für ein globales Minimum des Approximationsfehlers
- Nachteile:
  - Die Entwicklung nach den global wirkenden Polynombasen zeigen eine schlechte Konvergenz. Lokale Besonderheiten von Merkmalsclustern können nur sehr schlecht approximiert werden, ohne dabei gleichzeitig nicht zu viel an Qualität bei der Generalisierungsfähigkeit zu verlieren.

Funktionsapproximation mit einem Radiale-Basis-Funktionen-Netzwerk (RBFN)

- Anstatt Polynome für die Regressionsaufgabe zu verwenden, kann man auch eine lineare Entwicklung in Radiale Basisfunktionen (RBF) in Erwägung ziehen. Damit erhält man zwar wiederum ein *nichtlineares* Parameterschätzproblem (wie bei NN), welches typischerweise nur iterativ gelöst werden kann, aber es ergeben sich bessere Konvergenzeigenschaften.
- RBFs als Interpolationsfunktionen sind Funktionen der Form:

$$p(\|\mathbf{x} - \mathbf{c}_i\|)$$

- D.h. im Argument einer jeden RBF steht die Euklidsche Distanz zwischen dem Eingangsvektor **x** und einem Klassen-Zentrum $\mathbf{c}_i$ (daraus erklärt sich der Name). Die RBFs wirken nur in *lokalen* Umgebungen (im Gegensatz zu Polynomen).

Beispiele für Radiale-Basis-Funktionen (RBFN)

$$p(\mathbf{x}) = \exp\left(-\frac{1}{2\sigma_i^2} \left\| \mathbf{x} - \mathbf{c}_i \right\|^2 \right) \quad \text{(Gauss-RBFs)}$$

$$p(\mathbf{x}) = \frac{\sigma_i^2}{\sigma_i^2 + \left\| \mathbf{x} - \mathbf{c}_i \right\|^2}$$

- Man kann nun zeigen, dass mit der folgenden Summe jede Funktion f(x) für hinreichend große L beliebig genau approximiert werden kann:

$$f(\mathbf{x}) = \mathbf{a}^T \mathbf{p}(\mathbf{x}) = a_0 + \sum_{i=1}^{L} a_i \, p_i(\mathbf{x}, \mathbf{c}_i, \sigma_i)$$

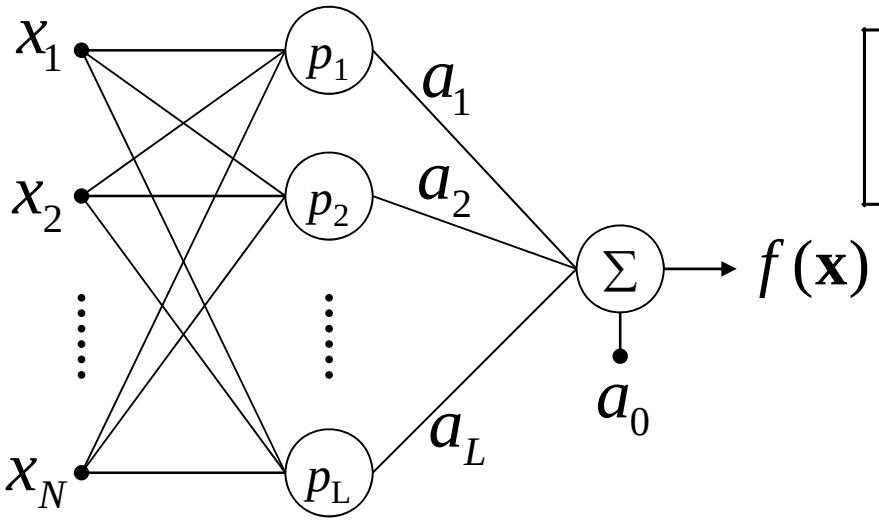
- Unbekannt sind die Parameter: $\{a_i\}, \{\mathbf{c}_i\}, \{\sigma_i\}$

  linear in $\{a_i\}$

  nichtlinear in $\{\mathbf{c}_i\}$ (Translation)

  nichtlinear in $\{\sigma_i\}$ (Radius)

RBF-Netzwerk als verallgemeinerter linearer Klassifikator



$$f(\mathbf{x}) = \mathbf{a}^T \mathbf{p}(\mathbf{x}) = \sum_{i=0}^{L} a_i \, p_i(\mathbf{x}, \mathbf{c}_i, \sigma_i)$$

Beim Perceptron wird der Originalraum in *Hyperebenen* aufgeteilt.

RBF-Netzwerk: Der in den neuen Variablen p_i lineare Klassifikator unterteilt den Originalraum in *Hypersphären*, da die $p_i(\mathbf{x})$ nichtlineare Funktionen von $||\mathbf{x}-\mathbf{c}_i||$ sind.

Der Polynomklassifikator läßt sich in der gleichen Art charakterisieren, nur mit dem Unterschied, dass er ebenfalls in **x** *nichtlinear*, aber in den unbekannten Parametern *linear* ist!

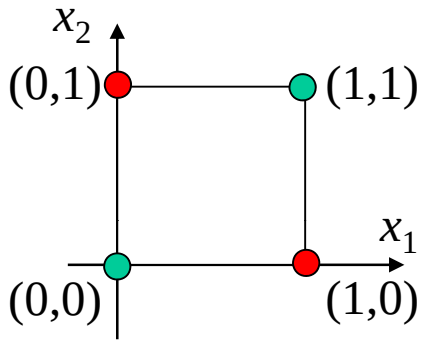
Lösung des XOR-Problems mit einem Gauss-RBFN

Wir wählen als Regressionsfunktion Gauss-RBFNs der Dimension L=2, welche auf die Stichproben zentriert sind:

$$f(\mathbf{x}) = \mathbf{a}^T \mathbf{p}(\mathbf{x}) = a_0 + \sum_{i=1}^{L} a_i p_i(\mathbf{x})$$

$$= a_0 + \sum_{i=1}^{L} a_i \exp\left(-\frac{1}{2\sigma_i^2} \left\| \mathbf{x} - \mathbf{c}_i \right\|^2 \right)$$

$$= a_0 + \sum_{i=1}^{L} a_i \exp\left(-\frac{(\mathbf{x} - c_i)^T (\mathbf{x} - c_i)}{2\sigma_i^2} \right)$$



$\mathcal{X}$-Raum

Mit $L = 2,$ den beiden Zentren

$\mathbf{c}_1 = \begin{bmatrix} 1 & 1 \end{bmatrix}^T$ sowie $\mathbf{c}_2 = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$

sowie $2\sigma_i^2 = 1$ ergibt sich:

$$\mathbf{p}(\mathbf{x}) = \begin{bmatrix} \exp\left(-\left\| \mathbf{x} - \mathbf{c}_1 \right\|^2 \right) \\ \exp\left(-\left\| \mathbf{x} - \mathbf{c}_2 \right\|^2 \right) \end{bmatrix}$$

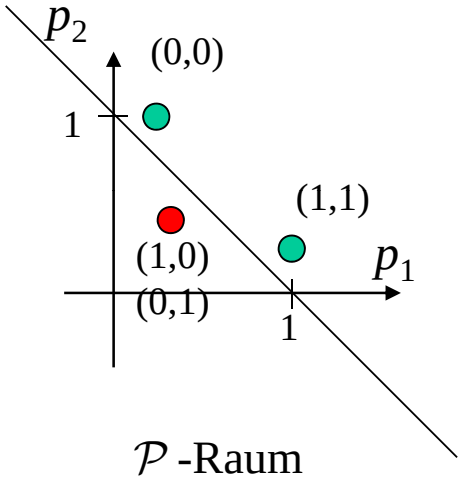
$\mathcal{X} \rightarrow \mathcal{P}$

x_1	0	1	0	1
x_2	0	0	1	1
p_1	$e^{-2} \approx 0.135$	$e^{-1} \approx 0.368$	$e^{-1} \approx 0.368$	1
p_2	1	$e^{-1} \approx 0.368$	$e^{-1} \approx 0.368$	$e^{-2} \approx 0.135$
$f(\mathbf{p})$	0.135	−0.264	−0.264	0.135

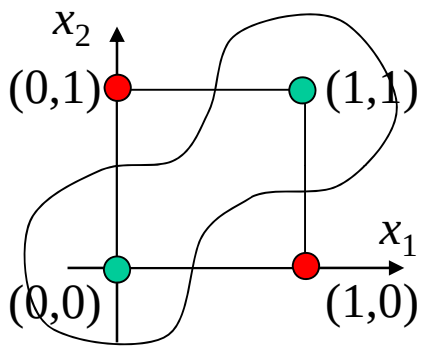
$$f(\mathbf{p}) = p_1 + p_2 - 1$$

Lösung des XOR-Problems mit einem Gauss-RBFN

Die Vektoren $\mathbf{x} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ und $\mathbf{x} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ werden auf den gleichen Punkt $\mathbf{p} = \begin{bmatrix} e^{-1} \\ e^{-1} \end{bmatrix}$ abgebildet.



$\mathcal{P}$-Raum



$\mathcal{X}$-Raum

Im $\mathcal{P}$-Raum sind die beiden Klassen nun offensichtlich linear separierbar:

Trenngerade im $\mathcal{P}$-Raum: $f(\mathbf{p}) = p_1 + p_2 - 1 = 0$

Trennkurve im $\mathcal{X}$-Raum: $f(\mathbf{x}) = \exp\left(-\left\|\mathbf{x} - \mathbf{c}_1\right\|^2\right) + \exp\left(-\left\|\mathbf{x} - \mathbf{c}_2\right\|^2\right) - 1 = 0$

Eigenschaften von RBFNs:

- Vorteile:
  - Im Unterschied zur Verwendung von Polynomen (Taylorreihen) können lokale Besonderheiten von Merkmalsclustern sehr gut approximiert werden, ohne Qualitätsverlust bei der Generalisierungsfähigkeit, d.h. man hat ein sehr gutes Konvergenzverhalten (falls optim. Lösung gefunden wird).
- Nachteile:
  - Es handelt sich um einen in den Parametern nichtlinearen Entwurf, welcher nur iterativ gelöst werden kann mit all den negativen Begleiterscheinungen evtl. langsamer Konvergenz der Iterationen und der Gefahr nur Nebenminima und damit nur suboptimale Lösungen zu finden.

Demo mit MATLAB

- Funktionsapproximation mit RBFNs:
  (Demo/Toolboxes/Neural Network/Radial Basis Networks)
  - Gute Wahl der σ_i (generalisierfähig)
    (Radial basis approximation, demorb1.m)
  - Wahl der σ_i zu klein (overfitting, d.h. neu hinzukommende Punkte werden nicht erreicht)
    (Radial basis underlapping neurons, demorb3.m)
  - Wahl der σ_i zu groß; die Basisfunktionen spannen den Raum nur sehr schlecht auf, da sie nahezu linaer abhängig voneinander sind.
    (Radial basis overlapping neurons, demorb4.m)
- Klassifikationsbeispiel mit RBFNs (PNN classification, demopnn1.m).

Start der Matlab-Demo
matlab-RBFNs-N.bat

(Batch-Mode, weiter mit Space-Taste)

Start der Matlab-Demo
matlab-RBFNs.bat

(Interaktive-Mode, weiter mit Next-Taste)