6. Netze radialer Basisfunktionen

6.1 Beiträge zur Konzeptbildung

Es sei eine Funktionsapproximation gewünscht.

Überblick:

- Satz von Kolmogorov (1957)
- Interpolation nach Powell (1987)
- Neuronale Informationsverarbeitung

Satz von Kolmogorov (1957)

Gegeben sei eine I-dimensionale stetige Funktion f.

Dann existieren ein-dimensionale stetige Funktionen g_0 und g_j, $j \in \{1, \ldots, (2I+1)\}$, sowie Konstanten α_i, $i \in \{1, \ldots, I\}$, so daß gilt:

$$f(x_1, \ldots, x_I) \stackrel{!}{=} \sum_{j=1}^{2I+1} g_0 \left(\sum_{i=1}^{I} \alpha_i g_j(x_i) \right)$$

Satz von Kolmogorov (1957)

Deutung: Jede multi-dimensionale stetige Funktion kann durch Kombination von ein-dimensionalen stetigen Funktionen repräsentiert werden.

Problem: Satz ist nicht konstruktiv.
Es ist unbekannt, wie $\boldsymbol{g_0}$ und $\boldsymbol{g_j}$, $\boldsymbol{j \in \{1, \ldots, (2I+1)\}}$,
gefunden werden.

Interpolation nach Powell (1987)

Gegeben seien M unterschiedliche Punkte
$\{x^m \in \mathbb{R}^I | m = 1, \ldots, M\}$
und reelle Zahlen $\{r^m \in \mathbb{R} | m = 1, \ldots, M\}$.

Dann findet man eine Funktion $f : \mathbb{R}^I \to \mathbb{R}$ gemäß nachfolgender Definition, so daß die Interpolationsbedingung erfüllt ist:

$$f(x^m) := \sum_{j=1}^{M} w_j h_j(x^m) = r^m ; \quad m \in \{1, \ldots, M\}$$

Interpolation nach Powell (1987)

Als Interpolationsfunktionen $\boldsymbol{h_j}$ unterscheidet man lokalisierende und nicht-lokalisierende Basisfunktionen $\boldsymbol{h}$, die nicht-linear sind.
Dabei sei definiert: $\boldsymbol{h_j(x) := h(\|x - x^j\|) =: h(d)}$

Lokalisierende Basisfunktionen:

$$\boldsymbol{d \to \infty \Rightarrow h(d) \to 0}$$

Beispiele:

$\boldsymbol{h(d) := e^{-\left(\frac{d}{\sigma}\right)^2}}$; Gauß-Funktion

$\boldsymbol{h(d) := \frac{1}{(\gamma^2 + d^2)^\zeta}}$; $\boldsymbol{\zeta > 0}$

Interpolation nach Powell (1987)

Nicht-lokalisierende Basisfunktionen:

$$d \to \infty \Rightarrow h(d) \to \infty \text{ oder}$$
$$h(d) \to \text{Konstante } (\neq 0)$$

Beispiele:

$$h(d) := (\gamma^2 + d^2)^\zeta\,; \quad \zeta > 0, \quad h(d) \to \infty$$
$$h(d) := \tfrac{1}{1+e^{-d}}\,; \quad h(d) \to 1$$

Interpolation nach Powell (1987)

Bestimmung der Koeffizienten $\boldsymbol{w_j}$ zur Interpolation.

Angenommen, Basisfunktion $\boldsymbol{h}$ und zugehörige Matrix $\boldsymbol{H}$ seien wie folgt gegeben:

$$\boldsymbol{H} := \begin{pmatrix} \boldsymbol{h_{11}} & \cdots & \boldsymbol{h_{1M}} \\ \vdots & & \vdots \\ \boldsymbol{h_{M1}} & \cdots & \boldsymbol{h_{MM}} \end{pmatrix}$$

$$\boldsymbol{h_{mj} := h(\|x^m - x^j\|)}$$

$$\boldsymbol{m \in \{1, \ldots, M\}, \quad j \in \{1, \ldots, M\}}$$

Interpolation nach Powell (1987)

Weiterhin seien die Sollwerte gegeben:

$$\boldsymbol{R} := \begin{pmatrix} \boldsymbol{r^1} \\ \vdots \\ \boldsymbol{r^M} \end{pmatrix}$$

Gesucht ist Vektor der Koeffizienten

$$\boldsymbol{w} := \begin{pmatrix} \boldsymbol{w_1} \\ \vdots \\ \boldsymbol{w_M} \end{pmatrix}$$

Lösung: $\boldsymbol{R} = \boldsymbol{H} \cdot \boldsymbol{w} \Longrightarrow \boldsymbol{w}^* := \boldsymbol{H}^{-1} \cdot \boldsymbol{R}$

Neuronale Informationsverarbeitung

- Lokale Informationsverarbeitung
  (Neuronen sind sensitiv für lokalisierte Teilräume des Eingaberaumes mit Überlappungsgebieten)
- Partitionierung des Lernproblems

6.2 Definition von RBF-Netzen

Überblick:

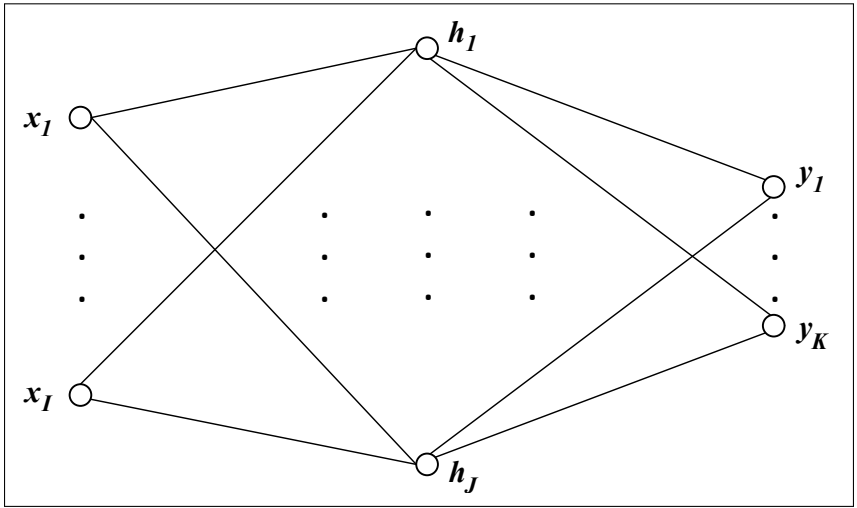
- Topologie und Basisfunktionen
- Funktion eines RBF-Netzes
- Funktionsapproximation durch RBF-Netz
- Parameter eines RBF-Netzes
- Nicht-isotrope Basisfunktionen
- Einschub: Zufallsvariable, Mittelwert, Varianz, Kovarianz, Kovarianzmatrix

Topologie und Basisfunktionen

Ein RBF-Netz ist ein zweischichtiges Netz (d.h. eine verdeckte Schicht), deren Output-Knoten eine (gewichtete) Linearkombination von Projektionen auf radiale Basisfunktionen berechnen.

Die Basisfunktionen gehören zu den verdeckten Knoten, die eine lokalisierte Antwort auf den Input erzeugen.

Topologie und Basisfunktionen



Funktion eines RBF-Netzes

Lineare Kombination von nicht-linearen Basisfunktionen:

$$y_k(x) := \sum_{j=1}^{J} w_{jk} h_j(x)\,; \quad k \in \{1, \cdots, K\}$$

Wichtigste Basisfunktion:

Radial-symmetrischer Gauß (synonym Isotroper Gauß)

$$h_j(x) := h(\|x - \mu^j\|) := e^{-\frac{\|x-\mu^j\|^2}{2{\sigma_j}^2}}$$

Funktionsapproximation durch RBF-Netz

RBF-Netz repräsentiert eine stetige Funktion, wobei die Anzahl der Basisfunktionen von der Komplexität der zu repräsentierenden Funktion abhängt.

Anzahl $\boldsymbol{J}$ der Basisfunktionen $\boldsymbol{h}_j$ ist klein im Vergleich zur Anzahl $\boldsymbol{M}$ der Trainingselemente.

$\Rightarrow$ Approximation statt Interpolation.

Parameter eines RBF-Netzes

Zwei Parameterarten.

- Parameter der einzelnen RBFs:
  Zentren $\boldsymbol{\mu}^j$
  Distanzen $\boldsymbol{\sigma}_j$, je zw. Zentrum und Wendepunkt einer Gauß-Kurve
- Parameter zur Kombination der RBFs:
  Faktoren (Gewichte) $\boldsymbol{w}_{jk}$

Parameter eines RBF-Netzes

Basisfunktionen sind nicht mehr auf Trainingselementen lokalisiert.

$\Rightarrow$ Lokalisierung $\boldsymbol{\mu}^j$ ist Teil des Lernens.

Hinweis: Ergänzend zur Lokalisierung $\boldsymbol{\mu}^j$, wird auch die Anzahl $\boldsymbol{J}$ der Basisfunktionen gelernt. D.h., die Anzahl der Knoten der verdeckten Schicht wird gelernt.

Parameter eines RBF-Netzes

Jede Basisfunktion hat eine individuelle Einzugsweite, charakterisiert durch $\boldsymbol{\sigma_j}$.

Falls $\boldsymbol{\sigma_j}$ klein/groß, dann ist die Basisfunktion für einen kleinen/großen Teilbereich des Eingaberaums zuständig.

Weiterhin gilt: Falls die $\boldsymbol{\sigma_j}$ der Basisfunktionen klein/groß, dann beeinflußen wenige/viele Basisfunktionen das Ergebnis der Netzanwendung auf ein bestimmtes Eingabelement.

$\Rightarrow$ Anpassung von $\boldsymbol{\sigma_j}$ ist Teil des Lernens.

Parameter eines RBF-Netzes

Faktor $\boldsymbol{w_{jk}}$ gewichtet die Rolle der Basisfunktion $\boldsymbol{h_j}$ bei der Berechnung der $\boldsymbol{k}$.ten Output-Komponente.

$\Rightarrow$ Ermittlung dieser Gewichte ist auch Teil des Lernens.

Nicht-isotrope Basisfunktionen

Verallgemeinerung des radial-symmetrischen Gauß zu einem nicht-isotropen Gauß als Basisfunktion.

Definition der Basisfunktion mit Kovarianzmatrix $\boldsymbol{S_j}$:

$$\boldsymbol{h_j(x) := e^{-\frac{1}{2}(x-\mu^j)^T(S_j)^{-1}(x-\mu^j)}}$$

Im Falle eines $\boldsymbol{I}$-dimensionalen Input-Vektors hat jede Basisfunktion dann $\boldsymbol{I(I+3)/2}$ justierbare Parameter: $\boldsymbol{I}$ Parameter in $\boldsymbol{\mu^j}$ und $\boldsymbol{I(I+1)/2}$ Parameter in $\boldsymbol{S^j}$.

Balance erforderlich zwischen kleiner Zahl von (nicht-isotropen) Basisfunktionen mit vielen justierbaren Parametern und großer Zahl von (isotropen) Basisfunktionen mit wenigen Parametern.

Einschub: Zufallsvariable, Mittelwert

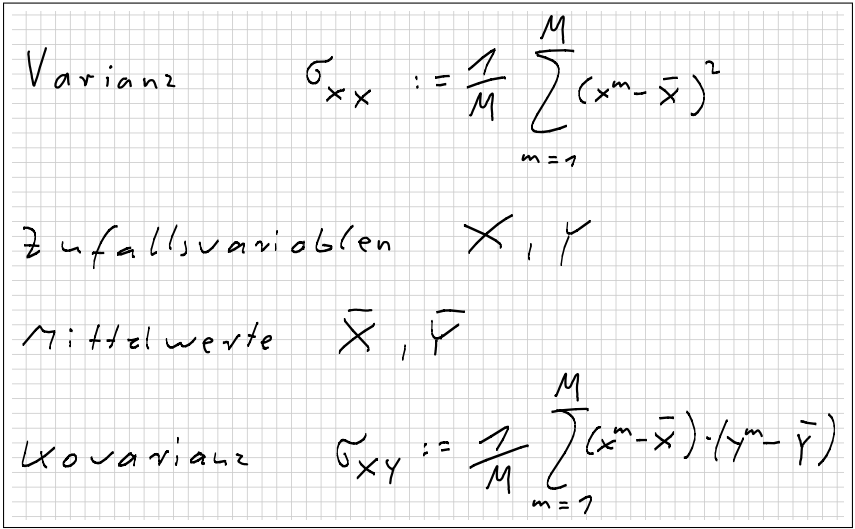
Zufallsvariable X

M mögliche Realisierungen x^m

z.B. beim Würfelspiel sechs mögliche Realisierungen $1, \dots, 6$

Mittelwert $\bar{x} := \frac{1}{M} \sum_{m=1}^{M} x^m$

Einschub: Varianz, Kovarianz



Einschub: Kovarianzmatrix

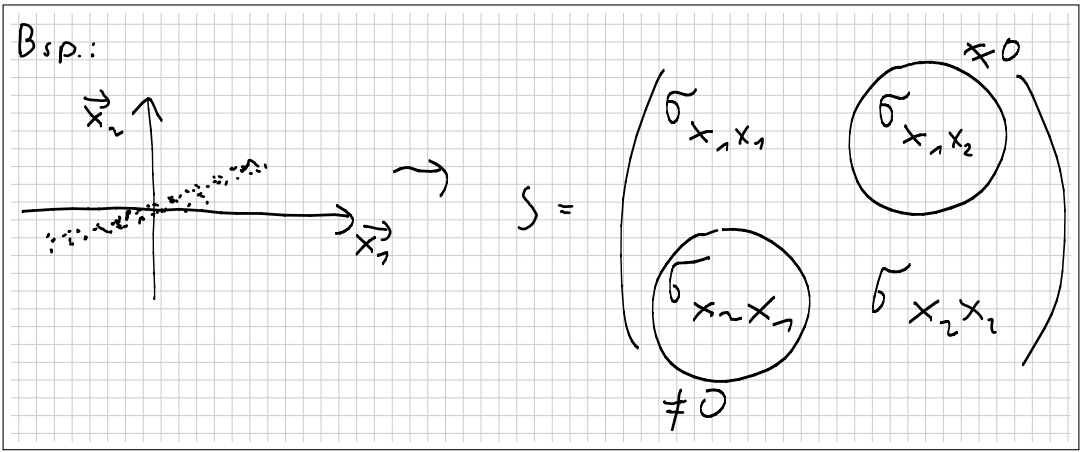
Zufallsvariablen $X_1, \ldots, X_I$

Kovarianz $\sigma_{X_i X_j}$

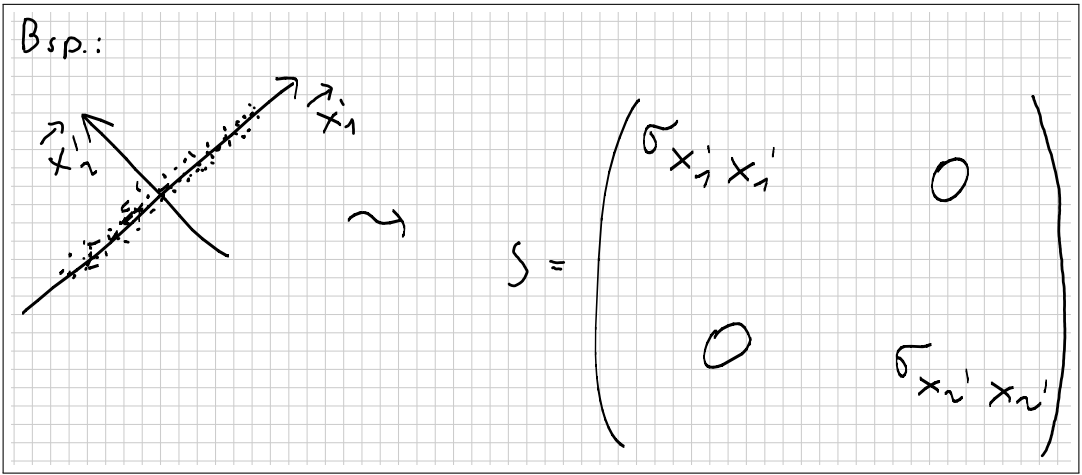
Kovarianzmatrix $S = \begin{pmatrix} \sigma_{X_1 X_1} & \cdots & \sigma_{X_1 X_I} \\ \vdots & & \vdots \\ \sigma_{X_I X_1} & \cdots & \sigma_{X_I X_I} \end{pmatrix}$

S ist symmetrisch, und hat $I \cdot (I+1)$ Parameter.

Einschub: Kovarianzmatrix



Einschub: Kovarianzmatrix



6.3 Zweischichtig, sequentielles Lernen

Überblick:

- Annahmen für das Lernen
- Art des Lernens in den Schichten
- MEANS Cluster-Algorithmus
- ISODATA Cluster-Algorithmus
- Lineare Regression durch Pseudo-Inverse

Annahmen für das Lernen

Es sei eine Funktionsapproximation gewünscht.

Seien $(\boldsymbol{x}^m, \boldsymbol{r}^m) \in \mathbb{R}^I \times \mathbb{R}$ die Trainingselemente, mit $\boldsymbol{m} \in \{1, \ldots, M\}$.

Seien $\boldsymbol{h_j}(\boldsymbol{x})$ die Symbole für die Basisfunktionen, mit $\boldsymbol{j} \in \{1, \ldots, J\}$.

Sei $\boldsymbol{f}(\boldsymbol{x})$ das Symbol für den reellen Netzwerk-Output (oBdA skalar).

Art des Lernens in den Schichten

1. Schicht: Unüberwachtes Lernen der Basisfunktionen bzgl. Eingabedaten $\boldsymbol{x}^m$ (keine Berücksichtigung von $\boldsymbol{r}^m$).
Gelernt werden $\boldsymbol{\mu}^j, \boldsymbol{\sigma}_j, \boldsymbol{J}$.

2. Schicht: Überwachtes Lernen der Faktoren $\boldsymbol{w}_j$ für die Linearkombination der Basisfunktionen $\boldsymbol{h}_j$, durch Lineare Regression unter Berücksichtigung von $\boldsymbol{r}^m$.

MEANS Cluster-Algorithmus

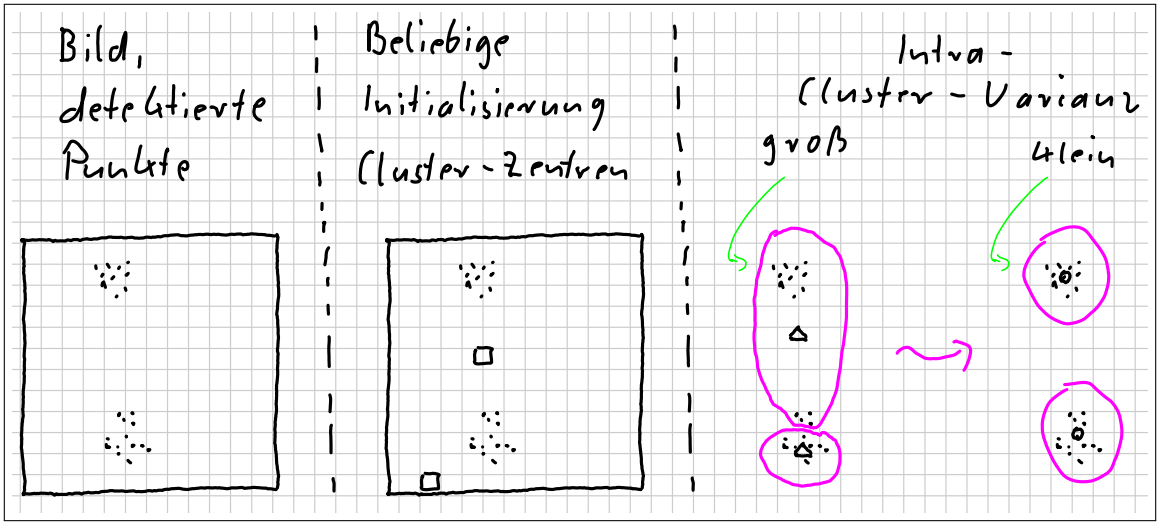
Ziel: Unüberwachtes Lernen, d.h. Clustern von Trainingsdaten zur Bestimmung von Anzahl, Lage, Ausdehnung der Basisfunktionen.

Methode: Partitionierung von $\boldsymbol{\Omega_T}$ in $\boldsymbol{J}$ disjunkte Teilmengen $\boldsymbol{Cl_j}$, durch Minimierung der Intra-Cluster-Varianz $\boldsymbol{V_{IC}}$.

$$\boldsymbol{V_{IC} := \frac{1}{J} \sum_{j=1}^{J} \left(\frac{1}{M_j} \sum_{x^m \in Cl_j} \|x^m - \mu^j\|^2 \right)}$$

$$\boldsymbol{\mu^j := \frac{1}{M_j} \sum_{x^m \in Cl_j} x^m, \quad M_j := |Cl_j|}$$

MEANS Cluster-Algorithmus



MEANS Cluster-Algorithmus

Algorithmus im Überblick:

- Initialisierung der Zentren der Mengen $\boldsymbol{Cl_j}$.
- Zugehörigkeiten von Elementen $\boldsymbol{x^m}$ zu den Mengen $\boldsymbol{Cl_j}$ festlegen bzw. ändern.
- Berechnung bzw. Neuberechnung der Zentren $\boldsymbol{\mu^j}$.

MEANS Cluster-Algorithmus

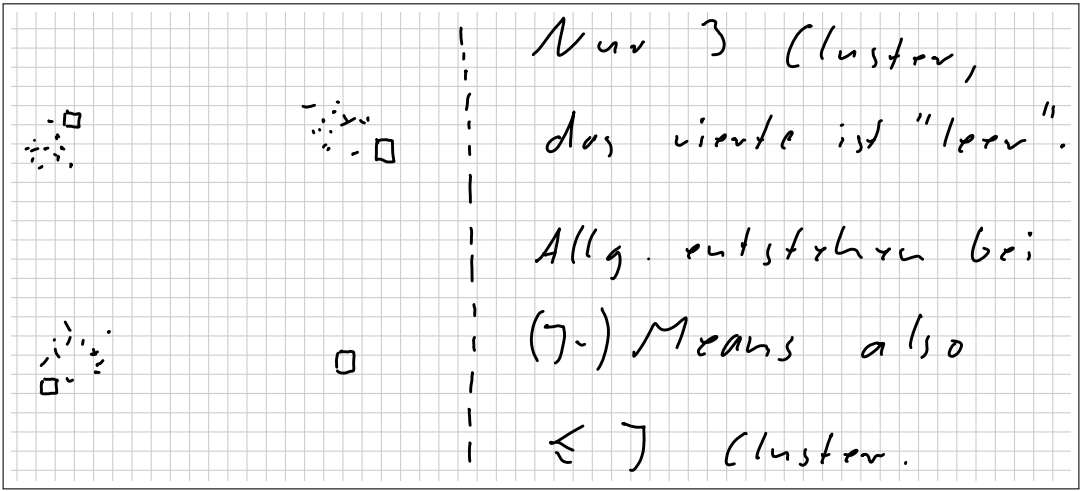
```
proc MEANS
{ Für alle j initialisiere mu[j] beliebig
  Wiederhole solange sich Cl[j] ändern
  {
    Für alle x[m] aus Omega[T] // Gruppierung
    { j_b = arg min {||x[m]−mu[j]||}
      füge x[m] zu Cl[j_b] hinzu
    }
    Für alle Cl[j]
    {  M[j] = Anzahl(Cl[j]) // Anzahl Elemente
       mu[j] = 1/M[j] * Summe(Cl[j]) // Zentrum
    }
  }
}
```

MEANS Cluster-Algorithmus

Bemerkungen:

- Bei jeder Iteration wird garantiert, daß $\boldsymbol{V_{IC}}$ nicht wächst.
- Manche Cluster bleiben eventuell leer.
- Resultierende Cluster hängen leider von Initialisierung ab.
- Zentren $\boldsymbol{\mu_j}$ der Cluster als Zentren der Basisfunktionen verwenden.
- Die Kovarianzmatrizen $\boldsymbol{S_j}$ der Cluster zur Definition von nicht-isotropen Basisfunktionen verwenden. Oder die Varianzen $\boldsymbol{\sigma_j}$ der Cluster zur Definition von isotropen Basisfunktion verwenden.

MEANS Cluster-Algorithmus



ISODATA Cluster-Algorithmus

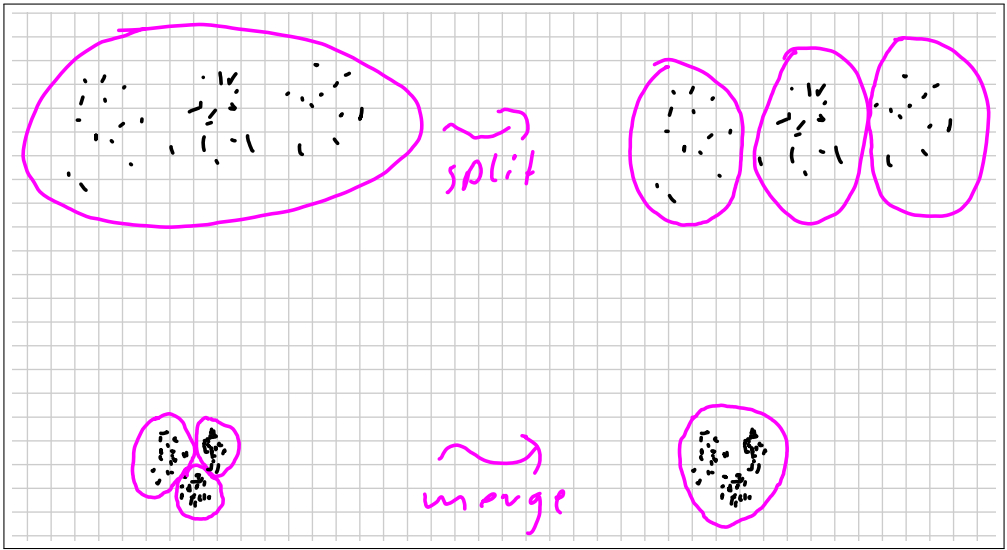
Iterative **S**elf-**O**rganizing **DATA** Analysis Technique.

Algorithmus im Überblick:

- Die Anzahl der Cluster ist variabel.
- Start mit Vorgabe niedriger Zahl von Clustern.
- Variieren der Größe und Gestalt der Cluster.
- Anfangsclusterung mit MEANS.
- Split-Operation falls Streuung im Cluster zu groß.
- Merge-Operation falls die Zentren der Cluster zu eng benachbart.

Funktioniert besser als MEANS (reduzierte Abhängigkeit von Initialisierung).

ISODATA Cluster-Algorithmus



Lineare Regression durch Pseudo-Inverse

- Gewichtsvektoren zwischen verdeckter Schicht und Ausgabeschicht werden unter der Annahme gelernt, dass die Basisfunktionen der verdeckten Schicht vorliegen.
- Diese Basisfunktionen werden auf die Trainingselemente angewendet.
- Weil die Knoten der Ausgabeschicht als Propagierungsfunktion den Linearen Assoziator und als Aktivierungsfunktion die Identität haben, ist nur noch eine Lineare Regression erforderlich.
- Beim Batch-Lernen erfolgt die Lineare Regression mit der Pseudo-Inversen (siehe Unterkapitel 4.5), und führt zur Minimierung des Mean-Squared-Error.

Lineare Regression durch Pseudo-Inverse

Funktion des RBF-Netzes: $\boldsymbol{f(x) := \sum_{j=1}^{J} w_j h_j(x)}$

Trainingsdaten: $\boldsymbol{(x^m, r^m) \in \Omega_T \subset \mathbb{R}^I \times \mathbb{R}}$

Fehlerfunktion: $\boldsymbol{D(w) := \frac{1}{2} \sum_{m=1}^{M} (f(x^m) - r^m)^2}$

Optimaler Gewichtsfektor: $\boldsymbol{w^* := \underbrace{(H^T \cdot H)^{-1} \cdot H^T}_{\text{Pseudo-Inverse von } H} \cdot R}$

Es ist $\boldsymbol{w^*}$ der Vektor mit den $\boldsymbol{J}$ optimalen Gewichten,

$\boldsymbol{H}$ die $(\boldsymbol{M} \times \boldsymbol{J})$-Matrix mit Komponenten $\boldsymbol{h_j(x^m)}$,

und $\boldsymbol{R}$ der Vektor mit den $\boldsymbol{M}$ Solldaten.

6.4 Weitere Lernvariante für RBF-Netze

Überblick:

- Diskussion über das zweistufige Lernen
- Integriertes Lernen aller Parameter
- Zweistufiges und integriertes Lernen

Diskussion über das zweistufige Lernen

Vorteile:

- Partitionierung des Lernproblems durch zweistufiges Lernen von zwei Parameterarten. Dies bewirkt eine Effizienzsteigerung beim Lernen.
- Transformation der Eingabe-(Roh-)Daten in einen anderen Raum, in welchem das Problem linearer Natur ist.
- Lernen der Gewichtsvektoren (in der zweiten Stufe) durch einfache, lineare Regression.

Diskussion über das zweistufige Lernen

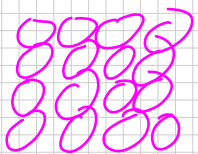
Nachteile:

- Die unabhängige Optimierung der Basisfunktionen von Solldaten $\boldsymbol{r}^m$ kann ein Problem sein.
- Zur Abtastung eines $\boldsymbol{I}$-dimensionalen Eingaberaumes wächst die notwendige Zahl der Basisfunktionen exponentiell mit $\boldsymbol{I}$.
- Oft haben bestimmte Dimensionen in den mehr-dimensionalen Eingabedaten keinen Einfluß auf die Ausgabedaten. Dies führt aber nicht zur Verringerung der Zahl der Basisfunktionen.

Diskussion über das zweistufige Lernen

Abtastung Eingabe Bereich

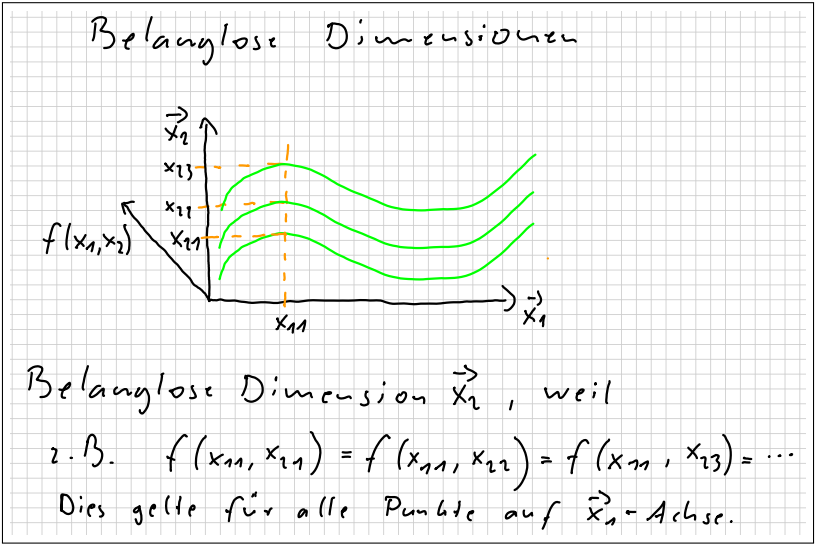
z.B. mit 4 Basisfunktionen je Dimension,
somit 4×4 bei 2 Dimensionen



Bzw. 4^3 bei 3 Dimensionen.

Exponentieller Zuwachs mit Dimensionszahl.

Diskussion über das zweistufige Lernen

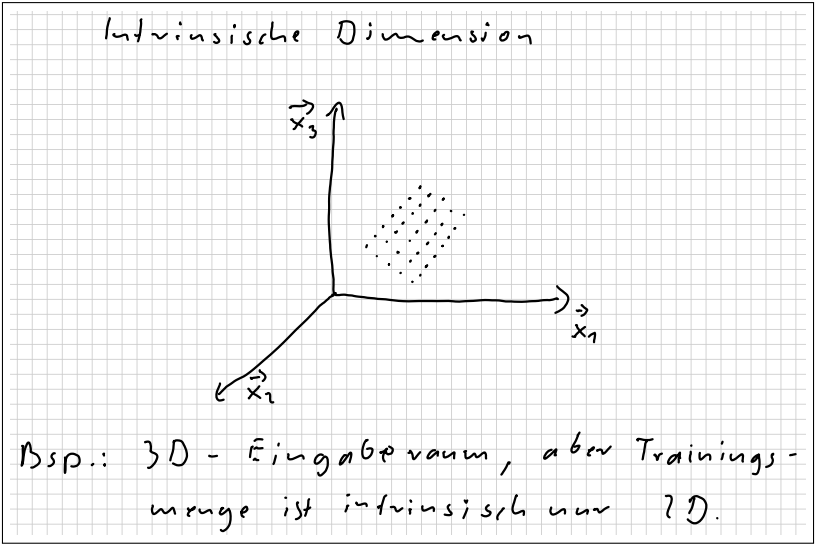


Diskussion über das zweistufige Lernen

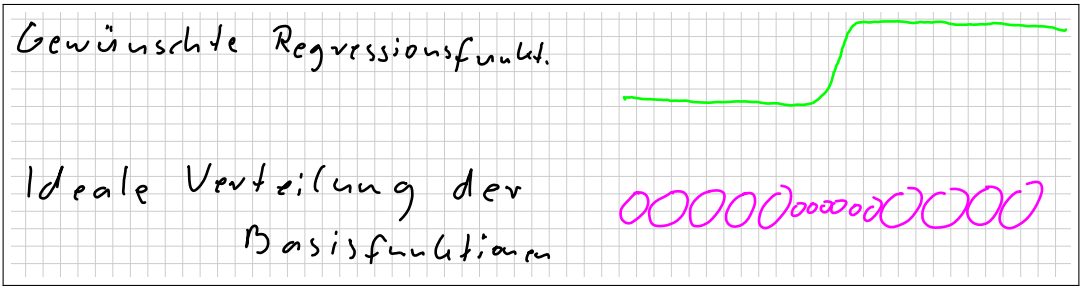
Nachteile:

- Wenn die intrinsische Dimensionalität der Eingabedaten und die Lage des eingebetteten Unterraumes bekannt wäre, dann könnte man besser geeignete Basisfunktionen definieren, um deren Anzahl zu reduzieren.
- Die Anzahl der Basisfunktionen sollte in denjenigen Teilgebieten des Eingaberaumes dicht verteilt sein, wo der Abbildungfehler groß ist, z.B. wo Funktionen stark gekrümmt sind.

Diskussion über das zweistufige Lernen



Diskussion über das zweistufige Lernen



Integriertes Lernen aller Parameter

- Integriertes, überwachtes Lernen aller Parameter der zwei Parameterarten, d.h. keine Organisation in zwei Stufen, auch Basisfunktionen überwacht lernen.
- Gradientenabstiegsverfahren in nicht-konvexer Fehlerfunktion.
- Realisiert eine ganzheitliche, nicht-lineare Regression.
- Verfahren ermittelt eventuell nur lokales Minimum (statt globales) der Fehlerfunktion.
- Verfahren würde eine konstante Zahl von Basisfunktionen zu Grunde legen, also die Anzahl der verdeckten Knoten nicht lernen.

Zweistufiges und integriertes Lernen

- Die Vorteile des zweistufigen Lernens und des integrierten Lernens könnten in folgendem Verfahren kombiniert werden.
- Erst zweistufiges Lernen, dann integriertes Lernen anschließen.
- Die Ergebnisse des zweistufigen Lernens dienen zur Initialisierung des Gradientenabstiegs beim integrierten Lernen.
- Dadurch wird Effizienz gesteigert und die Chance auf Erreichen des globalen Minimums der Fehlerfunktion erhöht.

6.5 Approximation und Regularisierung

Überblick:

- Gut oder schlecht gestellte Probleme
- Beispiele für schlecht gestellte Probleme
- Regularisierung
- Lineare Regression und Regularisierung

Gut oder schlecht gestellte Probleme

Bezugnahme auf Hadamard (1923).

Gut gestelltes (well-posed) Problem:

- Es existiert eine Lösung, und die
- Lösung ist eindeutig, und die
- Lösung hängt kontinuierlich von Eingabedaten ab.

Ist eine dieser Bedingungen nicht erfüllt, so ist das Problem schlecht gestellt (ill-posed).

Beispiele für schlecht gestellte Probleme

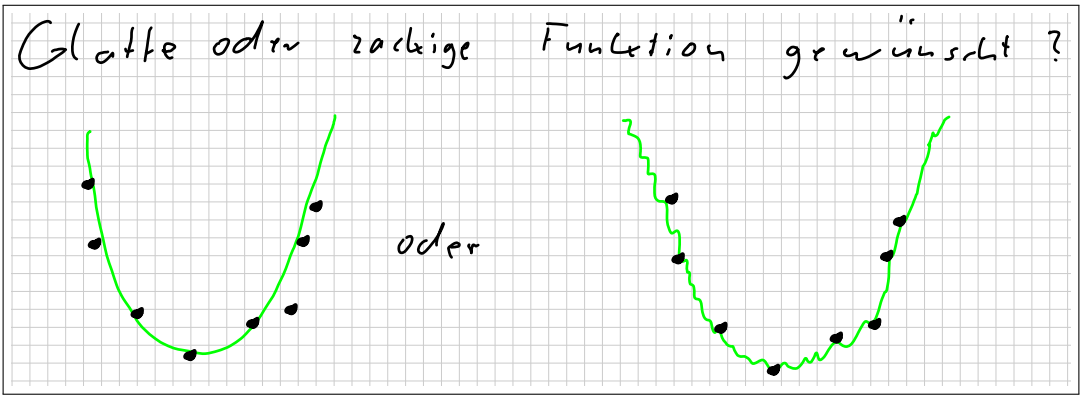
- Lineare Algebra: $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} \overset{?}{\Longrightarrow} \boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{y}$
- Regelungstheorie: Prozeßidentifikation aus Beobachtung
- Computer Vision: Szenenrekonstruktion aus Bildern

Regularisierung

Änderung/Verbesserung der Problemeigenschaften
(ill-posed $\Rightarrow$ well-posed) durch:

- mehr Meßergebnisse,
- Vorverarbeitung der Meßergebnisse,
- Definition von Zwängen durch zusätzliches Wissen.

Regularisierung



Regularisierung

Bezugnahme auf Poggio, Girosi (1990).

$$G(f) := \frac{1}{2} \sum_{m=1}^{M} (f(x^m) - r^m)^2 + \frac{\lambda}{2} \int |R_f|^2 dx$$

Generischer Regularisierungsoperator R_f für die Funktion f, der z.B. die Glattheit von f mißt.

Rolle (Wichtigkeit) des Regularisierungsterms wird festgelegt mit Hilfe des Regularisierungsparameters λ.

Bemerkung: Es gibt auch Regularisierungsterme mit mehreren Regularisierungsparametern.

Lineare Regression und Regularisierung

Verwendung der empirischen Fehlerfunktion $\boldsymbol{D(w)}$ in der Definition einer allgemeineren Kostenfunktion (loss function) $\boldsymbol{G(w)}$:

$$\boldsymbol{G(w) := D(w) + \underbrace{\frac{1}{2}\sum_{j=1}^{J} \lambda_j w_j^2}_{\text{Bestrafungsterm}}}$$

Der zusätzliche Term bewirkt einen "Zwang" bezüglich der zu lernenden Funktion $\boldsymbol{f}$, z.B. "bestraft" große Gewichte $\boldsymbol{w_j}$.
Die Regularisierungsparameter $\boldsymbol{\lambda_j \geq 0}$ kontrollieren das Ausmaß der Bestrafung:

$\boldsymbol{\lambda_j}$ klein $\Rightarrow$ dichte Anpassung an Messungen
$\boldsymbol{\lambda_j}$ groß $\Rightarrow$ Verzicht auf dichte Anpassung

Lineare Regression und Regularisierung

Durch "Bestrafungsterm" kann die Art der zu lernenden Funktion beeinflußt werden.

Verwendung 3 Gauße, |w| groß, dichte Anpassung an Daten

Verwendung 3 Gauße, |w| kleiner, grobe Anpassung an Daten.

Lineare Regression und Regularisierung

$$\frac{\partial G}{\partial w_j} = \underbrace{\sum_{m=1}^{M} (f(x^m) - r^m) \overbrace{\frac{\partial f}{\partial w_j}(x^m)}^{h_j(x^m)}}_{\frac{\partial D}{\partial w_j}} + \lambda_j w_j \stackrel{!}{=} 0\,;\ j \in \{1, \ldots, J\}$$

$$\sum_{m=1}^{M} f(x^m) h_j(x^m) + \lambda_j w_j = \sum_{m=1}^{M} r^m h_j(x^m)$$

Lineare Regression und Regularisierung

Vektor-Notation:

$$\vec{h}^{sp_j,T} \cdot \boldsymbol{F} + \boldsymbol{\lambda}_j \boldsymbol{w}_j = \vec{h}^{sp_j,T} \cdot \boldsymbol{R}$$

mit

$$\vec{h}^{sp_j} := (\boldsymbol{h}_j(\boldsymbol{x}^1), \ldots, \boldsymbol{h}_j(\boldsymbol{x}^M))^T$$

$$\boldsymbol{F} := (\boldsymbol{f}(\boldsymbol{x}^1), \ldots, \boldsymbol{f}(\boldsymbol{x}^M))^T$$

$$\boldsymbol{R} := (\boldsymbol{r}^1, \ldots, \boldsymbol{r}^M)^T$$

Nun Stapel von $\boldsymbol{J}$ Gleichungen zusammenfassen.

Lineare Regression und Regularisierung

Matrix-Notation:

$$\boldsymbol{H}^T \cdot \boldsymbol{F} + \boldsymbol{L} \cdot \boldsymbol{w} = \boldsymbol{H}^T \cdot \boldsymbol{R}$$

mit

$$\boldsymbol{H} := \begin{pmatrix} h_1(x^1) & \cdots & h_J(x^1) \\ \vdots & & \vdots \\ h_1(x^M) & \cdots & h_J(x^M) \end{pmatrix} =: \begin{pmatrix} \vec{h}^{ze_1} \\ \vdots \\ \vec{h}^{ze_M} \end{pmatrix}$$

$$=: \begin{pmatrix} \vec{h}^{sp_1} & \cdots & \vec{h}^{sp_J} \end{pmatrix}$$

und

$$\boldsymbol{L} := \begin{pmatrix} \lambda_1 & \cdots & 0 \\ & \ddots & \\ 0 & \cdots & \lambda_J \end{pmatrix} ; \quad \boldsymbol{w} := (w_1, \ldots, w_J)^T$$

Lineare Regression und Regularisierung

Es gilt: $$\boldsymbol{f}(\boldsymbol{x}^m) = \sum_{\boldsymbol{j}=1}^{\boldsymbol{J}} \boldsymbol{w_j h_j}(\boldsymbol{x}^m) = \vec{\boldsymbol{h}}^{\boldsymbol{ze_m}} \cdot \boldsymbol{w}$$

Zusammenfassung für alle $\boldsymbol{m} \in \{\boldsymbol{1}, \ldots, \boldsymbol{M}\}$: $\boldsymbol{F} = \boldsymbol{H} \cdot \boldsymbol{w}$

Verwendung in erster Formel von letzter Folie:

$$\boldsymbol{H^T \cdot R} = \boldsymbol{H^T \cdot F + L \cdot w} = \boldsymbol{H^T \cdot H \cdot w + L \cdot w} = \boldsymbol{(H^T \cdot H + L) \cdot w}$$

Ergibt optimalen Gewichtsvektor unter Zwang:

$$\boldsymbol{w^* := (H^T \cdot H + L)^{-1} \cdot H^T \cdot R}$$

$\rightsquigarrow$ allg. Pseudoinverse

6.6 Statistische Interpretation von RBF-Netzen

Überblick:

- Funktion von RBF-Netz als Bayes-Formel
- Probabilistische Deutung von RBF-Netzen

Annahme: RBF-Netze zur Klassifikation

Funktion von RBF-Netz als Bayes-Formel

$$\boldsymbol{P}(\boldsymbol{c}^k|\boldsymbol{x}) = \frac{\boldsymbol{P}(\boldsymbol{x}|\boldsymbol{c}^k) \cdot \boldsymbol{P}(\boldsymbol{c}^k)}{\boldsymbol{P}(\boldsymbol{x})} \tag{1}$$

$$\boldsymbol{P}(\boldsymbol{x}|\boldsymbol{c}^k) = \sum_{\boldsymbol{j}=\boldsymbol{1}}^{\boldsymbol{J}} \boldsymbol{P}(\boldsymbol{x}|\boldsymbol{j}) \cdot \boldsymbol{P}(\boldsymbol{j}|\boldsymbol{c}^k) \tag{2}$$

$$\boldsymbol{P}(\boldsymbol{x}) \stackrel{!}{=} \sum_{\boldsymbol{j}'} \boldsymbol{P}(\boldsymbol{x}|\boldsymbol{j}') \cdot \boldsymbol{P}(\boldsymbol{j}') \tag{3}$$

Deutung von $\boldsymbol{P}(\boldsymbol{x}|\boldsymbol{j})$: Zugehörigkeit von $\boldsymbol{x}$ zu einer bestimmten Gauß-Verteilung (Nummer $\boldsymbol{j}$).

Deutung von $\boldsymbol{P}(\boldsymbol{j}|\boldsymbol{c}^k)$: Relevanz der Gauß-Funktion $\boldsymbol{j}$ für die Charakterisierung der Klasse $\boldsymbol{c}^k$.

Funktion von RBF-Netz als Bayes-Formel

$$
\begin{aligned}
P(x) &= \sum_{k=1}^{K} P(x|c^k) \cdot P(c^k) \\
&= \sum_{k=1}^{K} \left(\sum_{j=1}^{J} P(x|j) \cdot P(j|c^k) \right) \cdot P(c^k) \\
&= \sum_{j=1}^{J} \sum_{k=1}^{K} \underbrace{P(x|j)}_{\text{unabh. von } \sum_k} \cdot P(j|c^k) \cdot P(c^k) \\
&= \sum_{j=1}^{J} P(x|j) \cdot P(j)
\end{aligned}
$$

Funktion von RBF-Netz als Bayes-Formel

Einsetzen von Gleichung 2 und 3 in Gleichung 1:

$$P(c^k|x) = \frac{\sum_{j=1}^{J} P(x|j) \cdot P(j|c^k) \cdot P(c^k) \cdot \frac{P(j)}{P(j)}}{\sum_{j'} P(x|j') \cdot P(j')}$$

$$= \sum_{j=1}^{J} \underbrace{\frac{P(j|c^k) \cdot P(c^k)}{P(j)}}_{\substack{=P(c^k|j) \\ =w_{jk}}} \cdot \underbrace{\frac{P(x|j) \cdot P(j)}{\sum_{j'} P(x|j') \cdot P(j')}}_{\substack{=P(j|x) \\ =h_j(x)}}$$

Probabilistische Deutung von RBF-Netzen

Das Lernen eines RBF-Netzes wird so realisiert, dass sich für einen Input-Vektor $\boldsymbol{x}$ die a posteriori Wahrscheinlichkeit für eine Klasse $\boldsymbol{c^k}$ ergibt.

Eine Basisfunktion $\boldsymbol{h_j}$ gibt die Wahrscheinlichkeit der Zugehörigkeit des Input-Vektors $\boldsymbol{x}$ zur Gauß-Funktion mit Nummer $\boldsymbol{j}$ an.

Das Gewicht $\boldsymbol{w_{jk}}$ gibt die Relevanz der Gauß-Funktion $\boldsymbol{j}$ für die Charakterisierung der Klasse $\boldsymbol{c^k}$ an, ausgedrückt als bedingte Wahrscheinlichkeit.