

Methoden zur Erzeugung von Kernen

Stefan Frommer

03.07.2007

1. Theoretische Konstruktion von Kernen

1.1 Wiederholung: Kerne

1.2 Auswahl eines Kerns

1.3 Kern-Kompositionen

2. Rechenbeispiele für Kerne

2.1 Bildvergleiche

2.2 Vergleich von Texten

2.3 Untersuchung von DNA-Sequenzen

2.4 Kerne mit bekannter Wahrscheinlichkeitsverteilung

1. Theoretische Konstruktion von Kernen

1.1 **Wiederholung: Kerne**

1.2 Auswahl eines Kerns

1.3 Kern-Kompositionen

2. Rechenbeispiele für Kerne

2.1 Bildvergleiche

2.2 Vergleich von Texten

2.3 Untersuchung von DNA-Sequenzen

2.4 Kerne mit bekannter Wahrscheinlichkeitsverteilung

Wiederholung: Kerne

Man wählt sich Kerne aus, um ein Ähnlichkeitsmaß zu haben, um damit verschiedene Objekte miteinander vergleichen zu können.

Zunächst: repräsentiere die Daten als Vektoren in einem Raum mit Skalarprodukt:

- $\Phi : X \rightarrow H$
- $x \mapsto \Phi(x) = \mathbf{x}$

Ein Kern ist ein Ähnlichkeitsmaß in diesem Merkmal-Raum.

- $k : X \times X \rightarrow \mathbb{K}$
üblicherweise mit $X \subset \mathbb{R}^N$, $X \neq \emptyset$ und $\mathbb{K} = \mathbb{R}$ oder $\mathbb{C}$
- $\dim(X) \ll \dim(H)$
- $(x, x') \mapsto k(x, x') := \langle \Phi(x), \Phi(x') \rangle = \langle \mathbf{x}, \mathbf{x}' \rangle$

Wiederholung: Kerne

mit dem kanonischen Skalarprodukt:

$$\langle \mathbf{x}, \mathbf{x}' \rangle := \sum_{i=1}^N [\mathbf{x}]_i [\mathbf{x}']_i, \quad \mathbf{x}, \mathbf{x}' \in \mathbb{R}^N$$

und den Eigenschaften

- Symmetrie: $k(x, x') = \overline{k(x', x)}$
- positiver Definitheit $\sum_{i,j=1}^m c_i \overline{c_j} k(x_i, x_j) \geq 0 \quad \forall c_i, \overline{c_j} \in \mathbb{K}$
- $\|x\| = \sqrt{\langle x, x \rangle}$

Wiederholung: Kerne

Beispiele für Kerne:

- Gauß'scher Kern: $k(x, x') = e^{-\frac{\|x-x'\|^2}{2\sigma^2}}$ mit $\sigma > 0$
- Polynomialer Kern der Ordnung d: $k(x, x') = \langle x, x' \rangle^d$

Im Folgenden betrachten wir nur noch positiv definite Kerne ("Kerne").

1. Theoretische Konstruktion von Kernen

1.1 Wiederholung: Kerne

1.2 Auswahl eines Kerns

1.3 Kern-Kompositionen

2. Rechenbeispiele für Kerne

2.1 Bildvergleiche

2.2 Vergleich von Texten

2.3 Untersuchung von DNA-Sequenzen

2.4 Kerne mit bekannter Wahrscheinlichkeitsverteilung

Auswahl eines Kerns

- suche ein **Ähnlichkeitsmaß**
z.B. Skalarprodukt
- wähle **lineare Repräsentation** im Funktionenraum H
z. B. Ellipse: $X = \mathbb{R}^2$, $H = \mathbb{R}^3$, $\Phi : (x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}x_1x_2, x_2^2)$
Ellipse ist linear in (z_1, z_2, z_3)
- wähle **passende Funktion** $\rightarrow$ KERN
z.B. polynomialer Kern $\langle x, x' \rangle^d$

1. Theoretische Konstruktion von Kernen

1.1 Wiederholung: Kerne

1.2 Auswahl eines Kerns

1.3 Kern-Kompositionen

2. Rechenbeispiele für Kerne

2.1 Bildvergleiche

2.2 Vergleich von Texten

2.3 Untersuchung von DNA-Sequenzen

2.4 Kerne mit bekannter Wahrscheinlichkeitsverteilung

- **Summen und Grenzwerte von Kernen**

Die Menge aller Kerne bildet einen konvexen Kegel, abgeschlossen unter punktwiser Konvergenz:

– k_1, k_2 Kerne, $\alpha_1, \alpha_2 \geq 0 \Rightarrow \alpha_1 k_1 + \alpha_2 k_2$ ist Kern

– k_i Kerne $\forall i \in \mathbb{N}$, $\lim_{n \rightarrow \infty} k_n(x, x') = k(x, x') \quad \forall x, x' \in X$

$\Rightarrow k$ ist wieder ein Kern

- **Punktweises Produkt**

k_1, k_2 Kerne $\Rightarrow k(x, x') := k_1(x, x')k_2(x, x')$ ist Kern.

- **Beispiel polynomiale Kerne:** $\langle x, x' \rangle^d \langle x, x' \rangle^n = \langle x, x' \rangle^{d+n}$

- Beispiel Konforme Transformationen:

gegeben: Kern k . Konstruiere k_f

$k_f(x, x') := f(x)k(x, x')f(x')$ mit einer positiven Funktion f

dabei ist $k_1 = k$ und $k_2 = f(x) f(x')$

Diese Transformation ist winkeltreu:

$$\begin{aligned} \cos(\angle(\Phi_f(x), \Phi_f(x'))) &= \frac{f(x)k(x, x')f(x')}{\sqrt{f(x)k(x, x)f(x)}\sqrt{f(x')k(x', x')f(x')}} \\ &= \frac{k(x, x')}{\sqrt{k(x, x)}\sqrt{k(x', x')}} = \cos(\angle(\Phi(x), \Phi(x'))) \end{aligned}$$

- **Polynomiale Kerne**

Das sind Kerne der Form $k(x, x') = \langle \Phi(x), \Phi(x') \rangle^d$

- betrachte dazu die Abbildung $\Phi : X \rightarrow H$ so dass $\langle \Phi(x), \Phi(x') \rangle$ effizient berechnet werden kann.
- Betrachte Abbildungen $x \mapsto f_x$, mit reellwertigen f_x
oft mit $f_x : [0, 1] \rightarrow \mathbb{R}$
- falls Φ zu komplizierte Gestalt hat, soll $\langle f_x, f_{x'} \rangle \approx \langle \Phi(x), \Phi(x') \rangle$ als Approximation erreicht werden.
- statte den Raum H aus mit dem Skalarprodukt $\langle f_x, f_{x'} \rangle = \int f_x(u) f_{x'}(u) du$.
- eine gute Approximation dazu ist dann $k(x, x') = \langle f_x, f_{x'} \rangle^d$

Beispiel: Die Eingabe-Muster $x_i \in X$ sind $q \times q$ -Bilder

f_{x_i} : 2-dimensionale Bilddichte-Funktionen, z. B. auf $[0, 1]^2$.

Mit den f_{x_i} ergibt sich folgende Annäherung an das Skalarprodukt:

$$k(x, x') := \int_0^1 \int_0^1 f_x(u) f_{x'}(u) \, du \, du$$

Dies wiederum wird angenähert durch:

$$\frac{1}{q^2} \sum_{i=1}^q \sum_{j=1}^q f_x\left(\frac{i-\frac{1}{2}}{q}, \frac{j-\frac{1}{2}}{q}\right) f_{x'}\left(\frac{i-\frac{1}{2}}{q}, \frac{j-\frac{1}{2}}{q}\right)$$

- **iterierte Kerne**

Gegeben sei ein Kern k . Dann definiert man

$$k^{(2)}(x, x') := \int k(x, x'')k(x', x'')dx''$$

$$k^{(n)}(x, x') := \int k^{(n-1)}(x, x^{(n)})k(x', x^{(n)})dx^{(n)}$$

Vorteil von iterierten Kernen: falls k nicht positiv definit ist, ist $k^{(2)}$ trotzdem positiv definit

- **Tensor-Produkte**

k_1, k_2 Kerne mit $k_1 \in X_1 \times X_1$, $k_2 \in X_2 \times X_2$

Das Tensor-Produkt $(k_1 \otimes k_2)(x_1, x_2, x'_1, x'_2) = k_1(x_1, x'_1)k_2(x_2, x'_2)$ mit $x_1, x'_1 \in X_1$ und $x_2, x'_2 \in X_2$

ist wieder Kern auf $(X_1 \times X_2) \times (X_1 \times X_2)$ da Produkt von Kernen.

- **Direkte Summen**

k_1, k_2 Kerne mit $k_1 \in X_1 \times X_1$, $k_2 \in X_2 \times X_2$

Die direkte Summe $(k_1 \oplus k_2)(x_1, x_2, x'_1, x'_2) = k_1(x_1, x'_1) + k_2(x_2, x'_2)$ mit $x_1, x'_1 \in X_1$ und

$x_2, x'_2 \in X_2$ ist wieder Kern auf $(X_1 \times X_2) \times (X_1 \times X_2)$

- **Interpolation zwischen Produkten und Summen von Kernen: Faltungskerne**

- $R(x_1, \dots, x_D, x)$ ist Relation: $x_1, \dots, x_D$ erzeugen das Objekt x , z. B. Teilstrings x_i vom String x

- $k_d \in X_d \times X_d$ ist Kern bezüglich X_d

Dann bezeichnet man $(k_1 * \dots * k_D)(x, x') := \sum_R \prod_{d=1}^D k_d(x_d, x'_d)$

als R-Faltung von $k_1, \dots, k_d$. Falls die Summe endlich ist, ist die Faltung ein gültiger Kern.

Vorteil von solchen Kompositionen:

Falls man den Input in zwei verschiedene Teile zerlegen kann, für die es gültige Kerne gibt, können diese dann zu einem Kern für den kompletten Input zusammengelegt werden.

Beispiel: **Stringkerne**

- **Spezialfall der R-Faltung: ANOVA-Kerne**

sei $X = S^N$ mit S beliebig.

mit gegebenen Kernen $k^{(i)}$ auf $S \times S$, $i=1,\dots,N$ und $D=1,\dots,N$ ist dann

$$k_D(x, x') := \sum_{1 \leq i_1 < \dots < i_D \leq N} \prod_{d=1}^D k^{(i_d)}(x_{i_d}, x'_{i_d})$$

ANOVA-Kern der Ordnung D .

Kern-Kompositionen

ANOVA-Kern der Ordnung $D=N$:

Die Summe besteht aus einem Term mit $(i_1, \dots, i_D) = (1, \dots, N)$

$\Rightarrow k = k^{(1)} \otimes \dots \otimes k^{(N)}$ ist das Tensor-Produkt

ANOVA-Kern der Ordnung $D=1$:

Das Produkt besteht aus einem Faktor

$\Rightarrow k = k^{(1)} \oplus \dots \oplus k^{(N)}$ ist die direkte Summe

für $1 < D < N$ erhält man Kerne zwischen Tensor-Produkt und direkter Summe

1. Theoretische Konstruktion von Kernen

1.1 Wiederholung: Kerne

1.2 Auswahl eines Kerns

1.3 verschiedene Kern-Kompositionen

2. Rechenbeispiele für Kerne

2.1 Bildvergleiche

2.2 Vergleich von Texten

2.3 Untersuchung von DNA-Sequenzen

2.4 Kerne mit bekannter Wahrscheinlichkeitsverteilung

Vergleich von zwei Bildern in Pixeldarstellung:

- betrachte die Bilder x und x' in Pixeldarstellung
- berechne ein drittes Bild $(x * x')$ als pixelweises Produkt von x und x'
- berechne daraus mit "pyramidischer" Gewichtung ein viertes Bild mit den Pixeln

$$z_{ij} := \sum_{i'j'} \omega(\max\{|i - i'|, |j - j'|\}) (x * x')_{i'j'}$$

- wähle dabei den Durchmesser der Pyramide $p = 2q + 1$; daraus ergibt sich der Parameter $q \in \mathbb{N}_0$
- dann ist die Gewichtungsfunktion gegeben durch:

$$\begin{aligned} \omega : \mathbb{N}_0 &\rightarrow \mathbb{R} \\ \omega(n) &= \max\{q - n, 0\} \end{aligned}$$

Bildvergleiche

Beispiel: $q=3 \Rightarrow \omega(n) = \max\{3 - n, 0\} \in \{0, 1, 2, 3\}$

Daraus ergeben sich für die Berechnung von z_{ij} folgende Gewichte $\omega_{z_{ij}}$, zentriert um $(x * x')_{ij}$:

$$\begin{array}{cccccccc}
 0 & = & 0 & = & 0 & = & 0 & = & 0 & = & 0 & = & 0 \\
 \parallel & & & & & & & & & & & & \parallel \\
 0 & & 1 & = & 1 & = & 1 & = & 1 & = & 1 & & 0 \\
 \parallel & & \parallel & & & & & & & & \parallel & & \parallel \\
 0 & & 1 & & 2 & = & 2 & = & 2 & & 1 & & 0 \\
 \parallel & & \parallel & & \parallel & & & & \parallel & & \parallel & & \parallel \\
 0 & & 1 & & 2 & & 3 & & 2 & & 1 & & 0 \\
 \parallel & & \parallel & & \parallel & & & & \parallel & & \parallel & & \parallel \\
 0 & & 1 & & 2 & = & 2 & = & 2 & & 1 & & 0 \\
 \parallel & & \parallel & & & & & & & & \parallel & & \parallel \\
 0 & & 1 & = & 1 & = & 1 & = & 1 & = & 1 & & 0 \\
 \parallel & & & & & & & & & & & & \parallel \\
 0 & = & 0 & = & 0 & = & 0 & = & 0 & = & 0 & = & 0
 \end{array}$$

Interpretiere die Gewichte als Abbildungen $P_{ij} : (x * x') \rightarrow \mathbb{N}_0$
 $\Rightarrow$ Pyramide mit Durchmesser p und Radius q .

Der zugehörige Kern ist dann definiert als

$$k_p^{d_1, d_2}(x, x') := \left(\sum_{ij} z_{ij}^{d_1} \right)^{d_2}$$

- d_1 berücksichtigt die lokalen Korrelationen innerhalb der Pyramide
- d_2 erlaubt weit entfernte Korrelationen von Ordnung d_2 .
- dieser Kern entspricht einem Skalarprodukt in einem Raum, in dem das Problem in polynomieller (in der Bildgröße) Zeit berechnet wird. Dieser Raum wird von lokal korrelierten Pixeln z_{ij} aufgespannt.
- dieser Kern hat die Ordnung $d_1 * d_2$.

Vorteile dieses Verfahrens:

- Ein solcher Kern kann in polynomieller Zeit berechnet werden
- Dieser Kern hat einen kleineren Fehler als ein polynomialer Kern gleicher Ordnung

1. Theoretische Konstruktion von Kernen

1.1 Wiederholung: Kerne

1.2 Auswahl eines Kerns

1.3 Kern-Kompositionen

2. Rechenbeispiele für Kerne

2.1 Bildvergleiche

2.2 Vergleich von Texten

2.3 Untersuchung von DNA-Sequenzen

2.4 Kerne mit bekannter Wahrscheinlichkeitsverteilung

Vergleich von Texten

Ziel: Vergleiche zwei Texte

Gesucht: Merkmal-Abbildung Φ : Text $\rightarrow$ Raum mit Skalarprodukt

Erste Ansätze:

- (i) Φ : Text $\rightarrow (0, \dots, 0, 1, 0, \dots, 0)^T$: i-te Komponente ist 1, falls Wort i im Text vorkommt, 0 sonst.
Skalarprodukt zwischen 2 solche Vektoren ist einfach zu berechnen $\rightarrow$ gültiger Kern.
- (ii) Φ : Text $\rightarrow$ Menge der Wortpaare innerhalb eines Satzes
- (iii) Φ : Text $\rightarrow$ Wortpaare, deren Worte innerhalb einer bestimmten Nähe zueinander sind

Zentrale Ideen:

- Vergleiche die Strings durch die Teilstrings, die sie enthalten.
- Je mehr Teilstrings zwei Strings haben, desto ähnlicher sind sie.
- Teilstrings dürfen von anderen Elementen oder Leerzeichen unterbrochen sein.

Vergleich von Texten

Notation zu Konstruktion von Stringkernen:

$\Sigma :$	endliches Alphabet
$\Sigma^n :$	Menge aller Strings der Länge n
$\Sigma^* = \bigcup_{n=0}^{\infty} \Sigma^n$	Menge aller endlichen Strings
$s :$	String
$ s :$	Länge von s
$s(1), \dots, s(s) :$	Elemente von s
$st :$	Verknüpfung von $s \in \Sigma^*$ und $t \in \Sigma^*$
$i := (i_1, \dots, i_{ u }) :$	Index-Sequenz
$1 \leq i_1 < \dots < i_{ u } \leq s :$	Index-Menge
$u := s(i) := s(i_1), \dots, s(i_{ u }) :$	Teilsequenz von String s
$l(i) := i_{ u } - i_1 + 1 :$	Länge von Teilsequenz; $l(i)$ länger als u, falls i nicht zusammenhängend
$\Phi : X \rightarrow H_n := \mathbb{R}^{(\Sigma^n)} :$	H_n aufgespannt von Strings der Länge n; H_n hat eine Dimension für jedes Element von Σ^n , bezeichnet durch dieses Element

Vergleich von Texten

Für $0 < \lambda \leq 1$ und $u \in \Sigma^n$ ist die Merkmal-Abbildung gegeben durch:

$$[\Phi_n(s)]_u := \sum_{i:s(i)=u} \lambda^{l(i)}$$

also: suche im String s Teilstrings u der Länge n

u lang $\Rightarrow [\Phi_n(s)]_u$ klein

Beispiel: $s_1 = \text{"bestes"}, s_2 = \text{"bester"}, u = \text{"bes"}, n=3$

$$\begin{aligned} [\Phi_3(\text{bestes})]_{\text{bes}} &= \lambda^3 + 2\lambda^6 \\ [\Phi_3(\text{bester})]_{\text{bes}} &= \lambda^3 \end{aligned}$$

Vergleich von Texten

Φ_n erzeugt den Kern (ANOVA-Kern der Ordnung 2) zum Vergleich von zwei Texten s und t durch Teilstrings der Länge n

$$k_n(s, t) = \sum_{u \in \Sigma^n} [\Phi_n(s)]_u [\Phi_n(t)]_u = \sum_{u \in \Sigma^n} \sum_{(i,j): s(i)=t(j)=u} \lambda^{l(i)} \lambda^{l(j)}$$

Für $c_n \geq 0$, c_n monoton wachsend, setzt man

$$\tilde{k} = \sum_n c_n k_n$$

Mit dieser Definition erhält man den normierten String-Kern

$$k(s, t) := \frac{\tilde{k}(s, t)}{\sqrt{\tilde{k}(s, s) \tilde{k}(t, t)}}$$

1. Theoretische Konstruktion von Kernen

1.1 Wiederholung: Kerne

1.2 Auswahl eines Kerns

1.3 Kern-Kompositionen

2. Rechenbeispiele für Kerne

2.1 Bildvergleiche

2.2 Vergleich von Texten

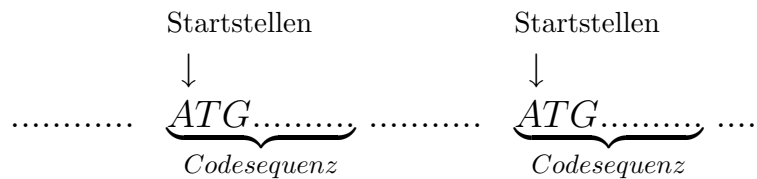
2.3 Untersuchung von DNA-Sequenzen

2.4 Kerne mit bekannter Wahrscheinlichkeitsverteilung

Untersuchung von DNA-Sequenzen

- Im Genom gibt es Codesequenzen, die Proteine verschlüsseln.
- Solche Codesequenzen beginnen mit Nukleotiden-Tripeln
- Man bestimmt die Startstellen solcher Codesequenzen

Ziel: Finde diese Startstellen in einem DNA-String.



die Codesequenzen werden charakterisiert durch

- Abgleichmethode von gleichwertigen Proteinen
- wesentliche Eigenschaften der Nukleotiden-Sequenz

Untersuchung von DNA-Sequenzen

Modelliere die Suche nach Startstellen als Klassifikationsproblem:

- der String "ATG" ist ein potentieller Kandidat für eine Startstelle
- konstruiere symmetrisches Intervall um ATG
- entscheide: deutet dieses Intervall auf wahre Startstelle hin oder nicht?
- jedes Nukleotid im Intervall wird repräsentiert durch:
 - 1, falls Nukleotid bekannt(lesbar)
 - Verteilung bezüglich der 4 Nukleotide gemäß deren Häufigkeiten in der Sequenz, sonst
- SVM bekommt Trainingsmenge (X_i, Y_i)
 - X_i : Nukleotidenstrings der Länge 200, um "ATG" zentriert
 - Y_i : TRUE/FALSE Startstelle
- nützlich für Startstellen-Erkennung: füge biologisches Wissen hinzu.

Untersuchung von DNA-Sequenzen

1. Ansatz:

Abhängigkeiten zwischen entfernten Positionen sind unwichtig / nicht existent.

- Bei jeder Sequenz-Position vergleichen wir beide Sequenzen lokal innerhalb eines Intervalls der Länge $2l + 1$ um die jeweilige Position:

$$match_{p+j}(x, x') = \begin{cases} 1, & \text{falls } x_{p+j} = x'_{p+j} \\ 0, & \text{sonst} \end{cases}$$

- summiere "passende" Nukleotide auf, gewichtet mit ν_j , die von den Grenzen des Intervalls zum Zentrum hin wachsen (wie im Bildvergleich).

$$win_p(x, x') = \sum_{j=-l}^l \nu_j \cdot match_{p+j}(x, x')$$

$$\nu_i := l - |x_i - x_p|$$

Untersuchung von DNA-Sequenzen

- potenziere die Summe mit d_1 , um lokale Korrelation widerzugeben.

$$win_p(x, x') = \left(\sum_{j=-l}^l \nu_i \text{match}_{p+j}(x, x') \right)^{d_1}$$

- Dies führt zum Lokalitäts-verbessertem Kern:

$$k(x, x') = \left(\sum_{p \in \text{Sequenz}} win_p(x, x') \right)^{d_2} = \left(\sum_{p \in \text{Sequenz}} \left(\sum_{j=-l}^l \nu_i \text{match}_{p+j}(x, x') \right)^{d_1} \right)^{d_2}$$

d_2 berücksichtigt dabei Korrelationen zwischen Intervallen bis zur Ordnung d_2 .

Untersuchung von DNA-Sequenzen

2. Ansatz:

- Code-Sequenz hat Codon-Struktur. Ein Codon ist ein Tripel von benachbarten Nukleotiden.
- Codesequenz um drei Nukleotide verschoben sieht immer noch aus wie Codesequenz.

⇒ zusätzlich zum Zählen von "passenden" Nukleotiden an korrespondierenden Stellen zählt man auch "Treffer", die um drei Positionen verschoben sind.

$$match_{p+j}(x, x') = \begin{cases} 1, & x_{p+j} = x'_{p+j} \text{ oder } x_{p+j} = x'_{p+j\pm 3} \\ 0, & \text{sonst} \end{cases}$$

Der dadurch wie oben erzeugte Kern heisst Codon-verbesserter Kern.

Untersuchung von DNA-Sequenzen

Vorteil dieses Verfahrens:

Dieser Kern hat einen kleineren Fehler als ein entsprechendes neuronales Netz und einfache polynomiale Kerne

1. Theoretische Konstruktion von Kernen

1.1 Wiederholung: Kerne

1.2 Auswahl eines Kerns

1.3 Kern-Kompositionen

2. Rechenbeispiele für Kerne

2.1 Bildvergleiche

2.2 Vergleich von Texten

2.3 Untersuchung von DNA-Sequenzen

2.4 Kerne mit bekannter Wahrscheinlichkeitsverteilung

Kerne mit bekannter Wahrscheinlichkeitsverteilung

Ziel: Verwende eine gegebene Wahrscheinlichkeitsfunktion $p(x|\theta)$ der Inputdaten

- Kerne mit solchen Zusatzinformationen nennt man **natürliche Kerne**.
- Familie der Dichtefunktionen $p(x|\theta), \theta = (\theta_1, \dots, \theta_r)$
- Score-Funktion $V_\theta : X \rightarrow \mathbb{R}^r; V_\theta(x) := \nabla_\theta \ln(p(x|\theta))$
- Fisher-Informations-Matrix

$$- I := E_p[V_\theta(x) V_\theta^T]$$

$$- I_{ij} := E_p \left[\frac{\partial \ln(p(x|\theta))}{\partial \theta_i} \frac{\partial \ln(p(x|\theta))}{\partial \theta_j} \right]$$

Kerne mit bekannter Wahrscheinlichkeitsverteilung

Mit einer pos. def. Matrix M (natürliche Matrix) ist ein natürlicher Kern gegeben durch:

$$\begin{aligned} k_M^{nat}(x, x') &:= V_\theta(x)^T M^{-1} V_\theta(x') \\ &= (\nabla_\theta \ln(p(x|\theta)))^T M^{-1} \nabla_\theta \ln(p(x'|\theta)) \end{aligned}$$

falls $M = I$, so spricht man vom Fisher-Kern.

Kerne mit bekannter Wahrscheinlichkeitsverteilung

Berechnung eines natürlichen Kerns:

- allgemein gilt: Es existieren Funktionen ψ_i , so dass gilt

$$k(x, x') = \sum_i \frac{d_i}{\lambda_i} \psi_i(x) \psi_i(x')$$

mit Parametern λ_i und d_i , für die gilt

$$d_n \in \{0, 1\} \quad \forall n$$

$$\sum_i \frac{d_i}{\lambda_i} < \infty$$

Kerne mit bekannter Wahrscheinlichkeitsverteilung

- Berechnung von ψ_i und λ_i :

- berechne $\left(M^{-\frac{1}{2}}IM^{-\frac{1}{2}}\right)$

- berechne Eigenwerte Λ_i und Eigenvektoren s_i von $\left(M^{-\frac{1}{2}}IM^{-\frac{1}{2}}\right)$

- setze dann

$$\psi_i(x) := \frac{1}{\sqrt{\Lambda_i}} s_i^T \nabla_{\theta} \ln(p(x|\theta))$$

$$\lambda_i := \Lambda_i$$

Damit ergibt sich der natürliche Kern

$$k_M^{nat}(x, x') = \sum_i \frac{d_i}{\lambda_i} \frac{1}{\sqrt{\Lambda_i}} s_i^T \nabla_{\theta} \ln(p(x|\theta)) \frac{1}{\sqrt{\Lambda_i}} s_i^T \nabla_{\theta} \ln(p(x'|\theta))$$

Danke für eure Aufmerksamkeit !

Quellen:

- B. Schölkopf, A. Smola. *Learning with kernels*. MIT Press, 2002
Kapitel 1, 2 und 13
- www.wikipedia.de