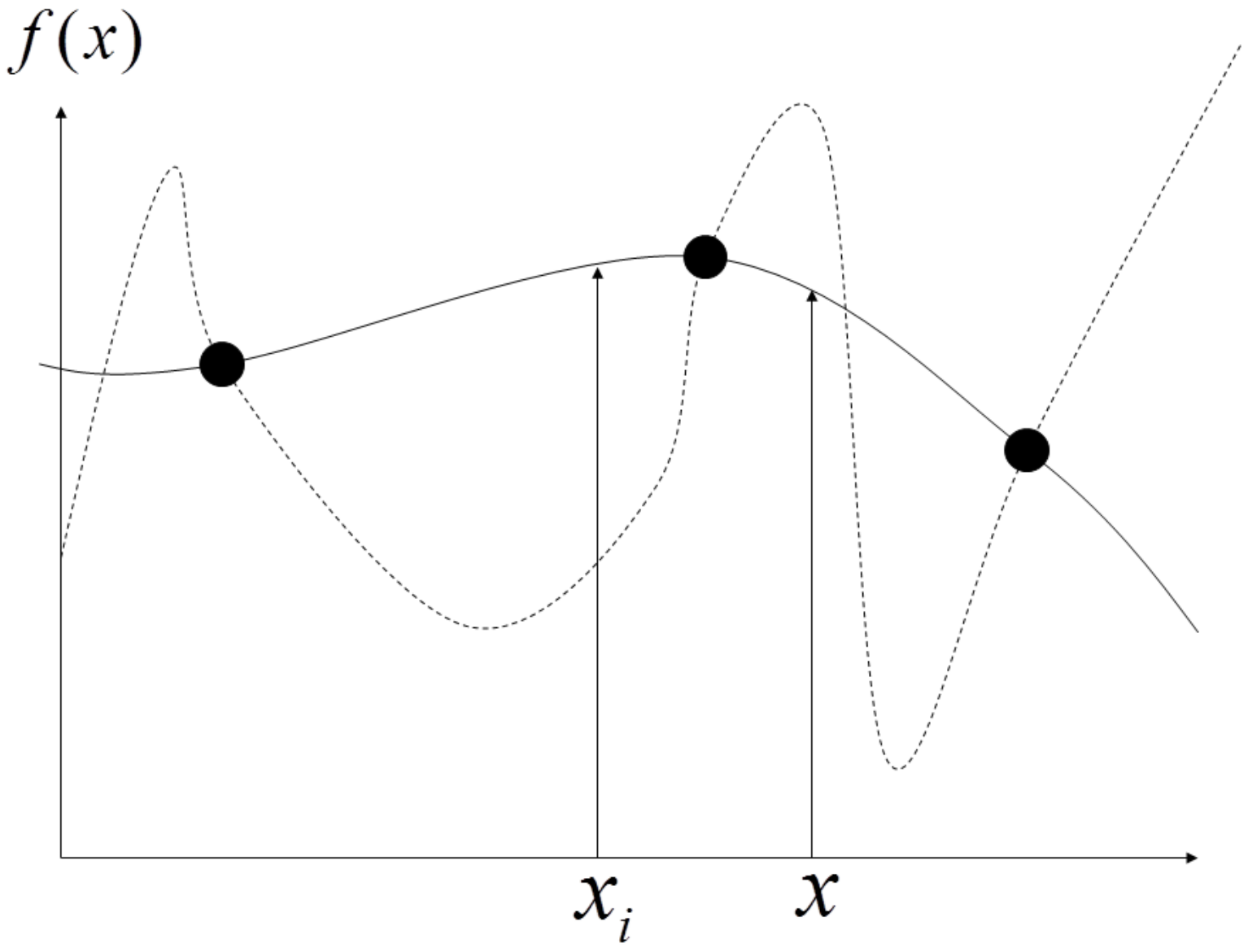
Kerne

Volker Tresp

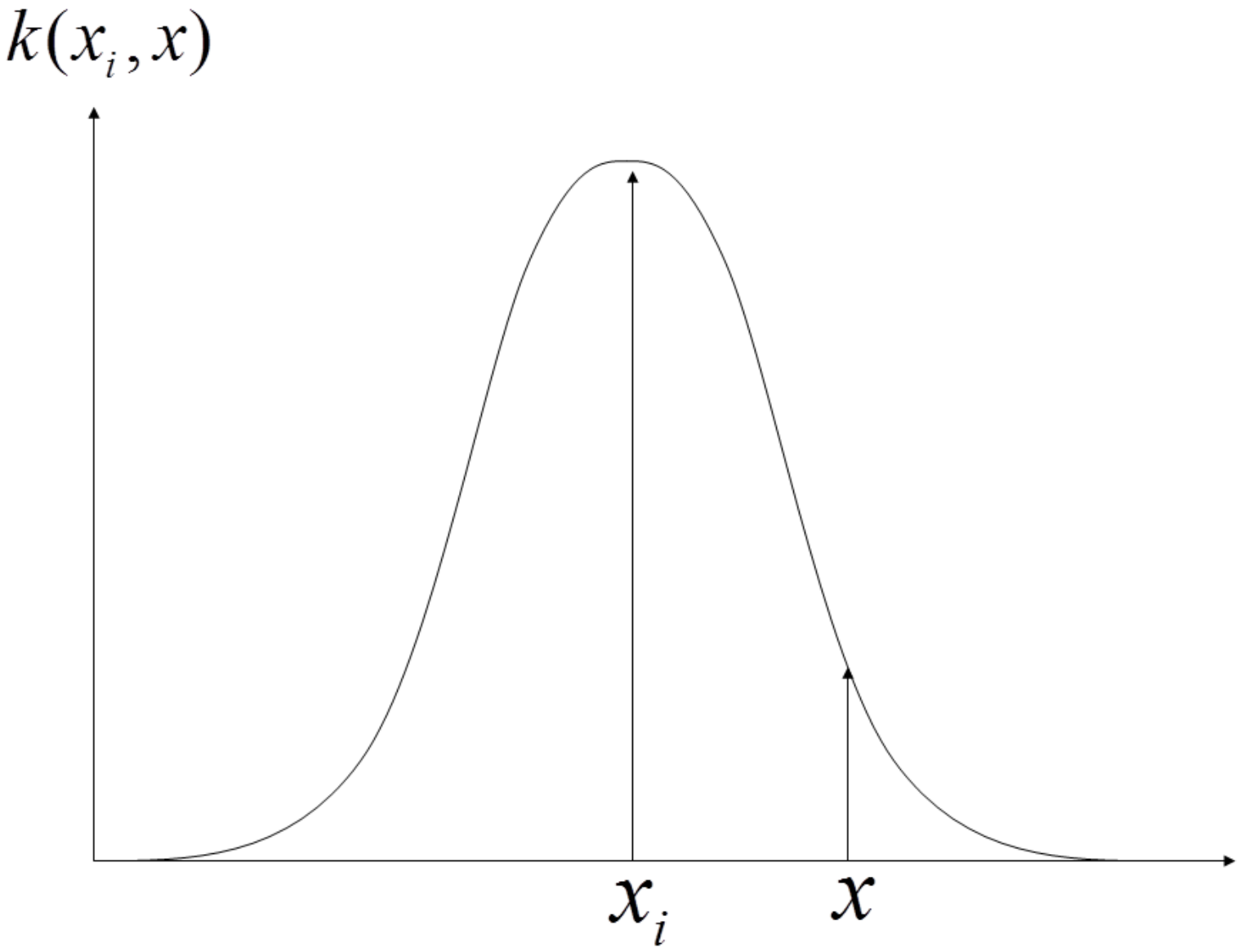
Glattheitsannahme

- Bisher haben wir Vorwissen einbringen können, indem wir geeignete Basisfunktionen definiert haben
- Alternativ mag es sinnvoll sein, glatte Funktionen zu bevorzugen: Funktionswerte an benachbarte Eingangswerten sollen ähnlich sein
- In der Abbildung mag es Sinn machen, anzunehmen, dass die Funktionswerte bei $\mathbf{x}_i$ und $\mathbf{x}$ ähnlich sind (Glattheitsannahme)
- Daher würde man die kontinuierliche Funktion der gestrichelten vorziehen


$f(x)$
x_i
x

Einführung Kerne

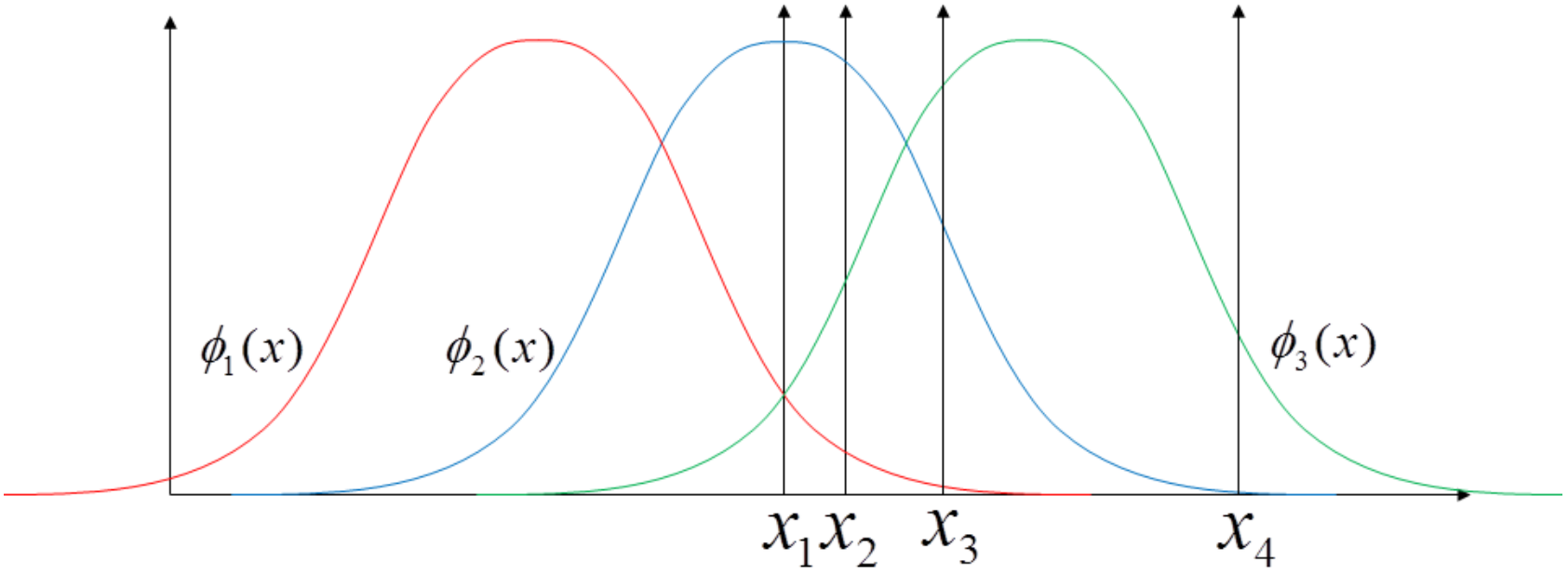
- Man kann diese Glattheitsannahme über Kernfunktionen implementieren
- Eine Kernfunktion $k(\mathbf{x}_i, \mathbf{x})$ gibt an, wie benachbarte Funktionswerte $f(x)$ sich verhalten, wenn $f(\mathbf{x}_i)$ gegeben ist
- Die Abbildung zeigt als Beispiel einen Gauß-schen Kerns


$k(x_i, x)$
x_i
x

Kerne und Basisfunktionen

- Es stellt sich heraus, dass es eine enge Beziehung zwischen Modellen mit festen Basisfunktionen und Kernmodellen gibt:

$$k(\mathbf{x}_i, \mathbf{x}) = \sum_{j=1}^{M_\phi} \phi_j(\mathbf{x}_i)\phi_j(\mathbf{x})$$



$$\phi(x_1) = (0.25, 1.00, 0.25)^T \qquad k(x_1, x_1) = \phi_1^T \phi_1 = 1.12$$
$$\phi(x_2) = (0.10, 0.90, 0.50)^T \qquad k(x_1, x_2) = \phi_1^T \phi_2 = 1.05$$
$$\phi(x_3) = (0.02, 0.60, 0.90)^T \qquad k(x_1, x_3) = \phi_1^T \phi_3 = 0.83$$
$$\phi(x_4) = (0.00, 0.01, 0.30)^T \qquad k(x_1, x_4) = \phi_1^T \phi_4 = 0.08$$

Kern Vorhersage

- Regression

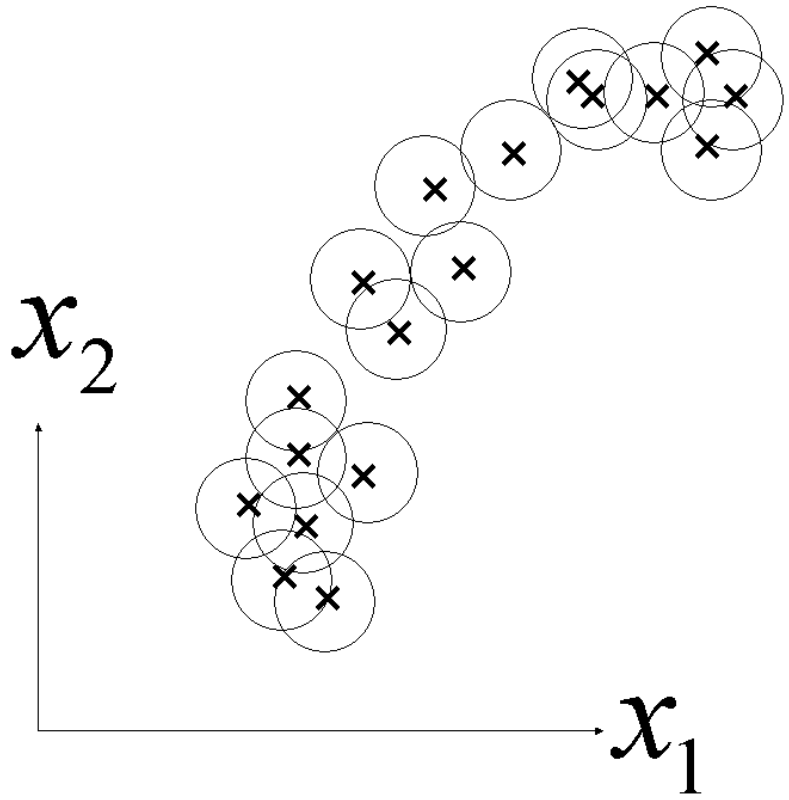
$$\widehat{y}(\mathbf{z}) = \sum_{i=1}^{N} v_i k(\mathbf{z}, \mathbf{x}_i)$$

- Klassifikation

$$\widehat{y}(\mathbf{z}) = \text{sign}\left(\sum_{i=1}^{N} v_i k(\mathbf{z}, \mathbf{x}_i)\right)$$

- Die Lösung hat soviele Kernfunktionen wir Datenpunkte im Training N

Eine Kernfunktion für jeden Datenpunkt



Umformungen der Kostenfunktion

- Wir beginnen mit der RLS Kostenfunktion für Modelle mit Basisfunktionen
- Regularisierte Kostenfunktion

$$\text{cost}^{pen}(\mathbf{w}) = \sum_{i=1}^{N}(y_i - \sum_j w_j \phi_j(x_i))^2 + \lambda \sum_{i=0}^{M} w_i^2$$

$$= (\mathbf{y} - \mathbf{\Phi w})^T(\mathbf{y} - \mathbf{\Phi w}) + \lambda \mathbf{w}^T\mathbf{w}$$

wobei Φ die design matrix ist mit $(\mathbf{\Phi})_{i,j} = \phi_j(\mathbf{x}_i)$.

Implizite Lösung

- Wir setzen die erste Ableitung nach den Parametern gleich Null

$$\frac{\partial \text{cost}^{pen}(\mathbf{w})}{\partial \mathbf{w}} = -2\mathbf{\Phi}^T(\mathbf{y} - \mathbf{\Phi}\mathbf{w}) + 2\lambda\mathbf{w} = 0$$

Daraus folgt, dass man schreiben kann

$$\mathbf{w}_{pen} = \frac{1}{\lambda}\mathbf{\Phi}^T(\mathbf{y} - \mathbf{\Phi}\mathbf{w}_{pen})$$

Ansatz

- Dies ist nicht die Lösung (w erscheint auf beiden Seiten der Gleichung). Aber wir wissen nun, dass man die Lösung als lineare Kombination der Eingangsvektoren schreiben kann

$$\mathbf{w}_{pen} = \mathbf{\Phi}^T \mathbf{v} = \sum_{i=1}^{N} v_i \phi(\mathbf{x_i})$$

- Beachte, dass die Summe über die N Datenpunkte geht!

Kern Modell

- Wir erhalten sofort

$$f(\mathbf{x}) = \sum_{j=1}^{M_\phi} w_{j,pen}\phi_j(\mathbf{x}_i) = \phi(\mathbf{x})^T\mathbf{w}_{pen}$$

$$= \phi(\mathbf{x})^T\mathbf{\Phi}^T\mathbf{v} = \sum_{i=1}^{N} v_i k(\mathbf{x}, \mathbf{x}_i)$$

mit $\mathbf{v} = (v_1, \ldots, v_N)^T$ und

$$k(\mathbf{x}, \mathbf{x}_i) = \phi(\mathbf{x})^T\phi(\mathbf{x}_i) = \sum_{k=1}^{M_\phi} \phi_k(\mathbf{x})\phi_k(\mathbf{x}_i)$$

Eine neue Kostenfunktion

- Wir können die Nebenbedingung in die Kostenfunktion einsetzten und erhalten

$$\text{cost}^{pen}(\mathbf{v}) = (\mathbf{y} - \mathbf{\Phi}\mathbf{\Phi}^T\mathbf{v})^T(\mathbf{y} - \mathbf{\Phi}\mathbf{\Phi}^T\mathbf{v}) + \lambda\mathbf{v}^T\mathbf{\Phi}\mathbf{\Phi}^T\mathbf{v}$$

$$= (\mathbf{y} - K\mathbf{v})^T(\mathbf{y} - K\mathbf{v}) + \lambda\mathbf{v}^T K\mathbf{v}$$

  wobei K eine $N \times N$ Matrix ist mit Elementen

$$k_{i,j} = \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j) = \sum_{k=1}^{M_\phi} \phi_k(\mathbf{x}_i)\phi_k(\mathbf{x}_j)$$

- Ein wichtiges Ergebnis: **Das Optimierungsproblem können wir so schreiben, dass nur die inneren Produkte der Basisvektoren auftauchen, aber nicht die Basisvektoren selber!**

Kern Gewichte

- Wir können nun die Kostenfunktion nach $\mathbf{v}$ ableiten (beachte, dass $K = K^T$)

$$\frac{\partial \mathbf{cost}^{pen}(\mathbf{v})}{\partial \mathbf{v}} = 2K(\mathbf{y} - K\mathbf{v}) + 2\lambda K\mathbf{v}$$

So dass

$$\mathbf{v}_{pen} = (K + \lambda I)^{-1}\mathbf{y}$$

Kern Vorhersage

- Eine Vorhersage lässt sich somit schreiben als

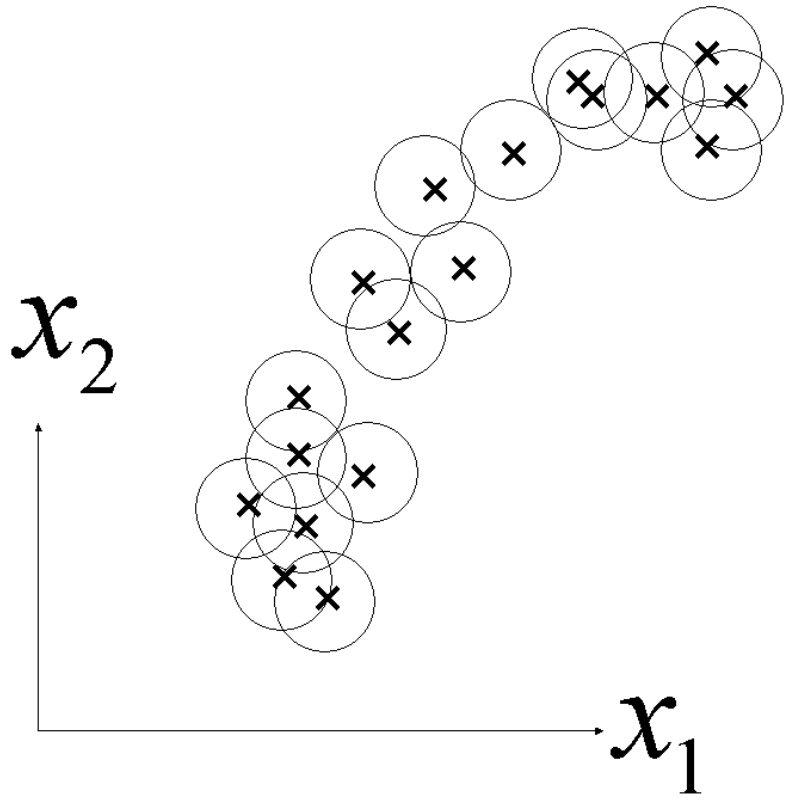
$$\widehat{f}(\mathbf{z}) = \phi(\mathbf{z})^T \mathbf{w} = \phi(\mathbf{z})^T \mathbf{\Phi}^T \mathbf{v}_{pen} = \sum_{i=1}^{N} v_i k(\mathbf{z}, \mathbf{x}_i)$$

  Mit

$$k(\mathbf{z}, \mathbf{x}_i) = \phi(\mathbf{z})^T \phi(\mathbf{x_i})$$

- Wieder ein wichtiges Resultat: auch die Vorhersage lässt sich so schreiben, dass nur innere Produkte verwendet werden; **die Lösung läßt sich als gewichtete Summe von N Kernfunktionen schreiben**.

Eine Kernfunktion für jeden Datenpunkt

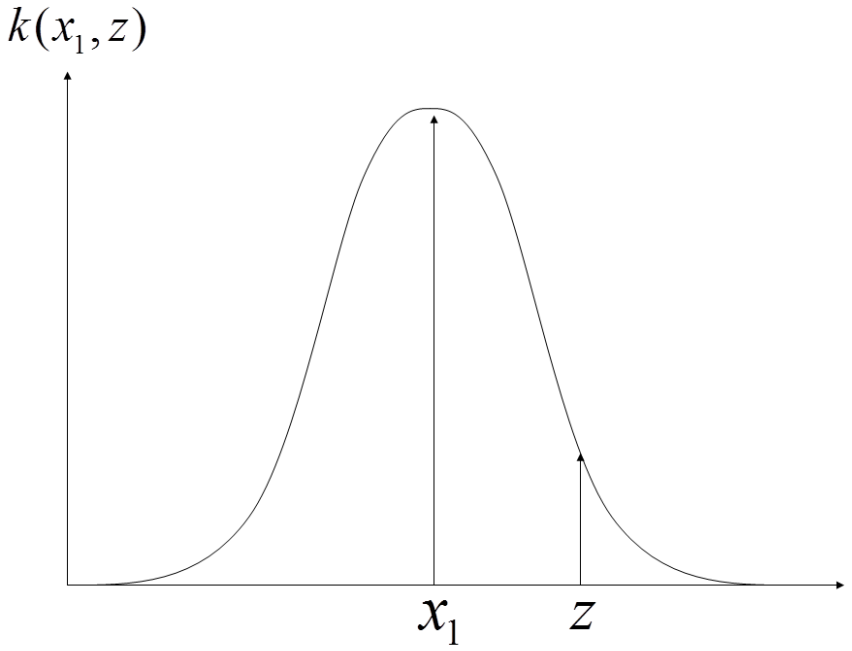


Mit einem Trainingsdatenpunkt

- Mit nur einem Datenpunkt ergibt sich

$$f(\mathbf{z}) = v_1 k(\mathbf{z}, \mathbf{x}_1)$$

- Wie vorher diskutiert:



Bemerkungen und Interpretation des Kerns

- Dies ist nun schon interessanter, da es durchaus mehr Basisfunktionen als Eingangsdimensionen geben kann; insbesondere gilt das Resultat auch, wenn man mit **unendlich vielen Basisfunktionen** arbeitet
- Man kann sogar direkt mit den Kernfunktionen arbeiten, ohne sich besondere Gedanken über die Basisfunktionen zu machen, aus denen sie hergeleitet wurden
- Interpretation des Kerns
  - Als inneres Produkt $k(\mathbf{x}_i, \mathbf{z}) = \phi^T(\mathbf{x})\phi(\mathbf{z})$
  - Als Kovarianz: wie stark sind Funktionswerte an verschiedenen Eingangspunkten korreliert $k(\mathbf{x}_i, \mathbf{z}) = cov(f(\mathbf{x}_i), f(\mathbf{z}))$
- Wenn $N >> M$ ist die originale Formulierung im Merkmalsraum effizienter; wenn $M >> N$, ist die Kern-Version effizienter; genauer: im Merkmalsraum benötigt man $M^3 + M^2N$ Operationen und mit Kernen benötigt man $N^3 + N^2M$ Operationen. Wenn die Kerne a prior bekannt sind benötigt man N^3 Operationen

- In speziell strukturierten Problemen lassen sich Kerne berechnen, ohne explizit die NM^2 Operationen ausführen zu müssen. Beispiel: String Kerne. Dies ist eines der wichtigsten aktiven Forschungsaufgaben!

- Dennoch sind nicht alle Funktionen geeignete Kernfunktionen; dies stellt das folgende Theorem dar ...

Mercer Theorem

- Nach Vapnik: The nature of statistical learning theory
- *Mercer Theorem:* Um zu garantieren, dass die symmetrische Funktion $k(\mathbf{x}, \mathbf{z})$ aus L_2 eine Entwicklung der Art

$$k(\mathbf{x}, \mathbf{z}) = \sum_{h=1}^{\infty} \lambda_h \phi_h^T(\mathbf{x}) \phi_h(\mathbf{z})$$

besitzt, mit positiven Koeffizienten $\lambda_h > 0$, so ist es notwendig und ausreichend, dass

$$\int \int k(\mathbf{x}, \mathbf{z}) g(\mathbf{x}) g(\mathbf{z}) d\mathbf{x} d\mathbf{z} > 0$$

gültig ist für alle $g \neq 0$ für welche

$$\int g^2(\mathbf{x}) d\mathbf{x} < \infty$$

- Das Theorem sagt aus, dass für sogenannte positiv-definite Kerne, eine Zerlegung in Basisfunktionen möglich ist!

- Jede Kern-Matrix K ist dann ebenfalls positiv definit

Kern Design

- Linearer Kern

$$k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$$

Hier sind sowohl Basisfunktionen als auch die entsprechenden Kerne lineare Funktionen

- Polynomialer Kern

$$k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j)^d$$

Hier sind die entsprechenden Basisfunktionen alle geordneten Polynome des Grades d

- Polynomialer Kern (2)

$$k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j + R)^d$$

Hier sind die entsprechenden Basisfunktionen alle geordneten Polynome vom Grad d oder kleiner

- Gauß-Kern (RBF-Kern)

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{1}{2s^2}\|\mathbf{x}_i - \mathbf{x}_j\|^2\right)$$

Diese Kerne entsprechen *unendlich vielen* Gauß-förmigen Basisfunktionen

- Sigmoider Kern

$$k(\mathbf{x}_i, \mathbf{x}_j) = \text{sig}\left(\mathbf{x}_i^T \mathbf{x}_j\right)$$

Interessant im Zusammenhang mit Neuronalen Netze

RKHS

- Wir kennen das innere Produkt von Vektoren
- Hilbert hat gezeigt, dass man Funktionen ebenfalls als (eventuell unendlich dimensionale) Vektoren beschreiben kann und ein inneres Produkt definieren kann
- Für zwei Funktionen, die sich durch die Überlagerung von Basisfunktionen darstellen lassen ($f(.) = \phi^T(.)w_f$ und $g(.) = \phi^T(.)w_g$), kann man ein inneres Produkt definieren als

$$< f, g >_{\mathcal{H}} = w_f^T w_g = \mathbf{v}_f^T K \mathbf{v}_g$$

$< f, g >_{\mathcal{H}}$ steht für ein inneres Produkt in einem *reproducing kernel Hilbert space*

Umschreiben der Kostenfunktion

- Entsprechend kann man die Kostenfunktion schreiben als

$$\text{cost}^{pen} = \sum_{i=1}^{N} (y_i - f(\mathbf{x}_i))^2 + \lambda < f, f >_{\mathcal{H}}$$

- Strafterm als inneres Produkt von Funktionen (ohne explizite Referenz zu Basisfunktionen)
- Man kann nun fragen, welche Kostenfunktionen zu Kern-Lösungen führen. Die Antwort gibt das Representer Theorem

Representer Theorem

- *Representer Theorem:* Sei Ω eine strikt monoton wachsende Funktion, loss() eine beliebige Verlustfunktion, dann erlaubt der Minimierer des regularisierten Risikos

$$\sum_{i=1}^{N} \text{loss}(y_i, f(\mathbf{x}_i)) + \Omega(\|f\|_{\mathcal{H}})$$

  eine Darstellung der Form

$$f(\mathbf{x}) = \sum_{i=1}^{N} v_i k(\mathbf{x}_i, \mathbf{x})$$

- $\|f\|_{\mathcal{H}} = \sqrt{< f, f >}$ ist eine Norm in einem *reproducing kernel Hilbert space*
- Beispiel: Ansätze mit Basisfunktionen mit beliebigen Kostenfunktionen und Strafterm $\mathbf{w}^T\mathbf{w}$
- Das innere Produkt kann auch anders definiert werden (Beispiel: Differentialoperatorn, die höhere Ableitungen bestrafen)

Bemerkungen

- Der "Kern-Trick" erlaubt es, in unendlich hohen Dimensionen zu arbeiten
- Dennoch können durch die Regularisierung sehr gute Ergebnisse erzielt werden;
- Die Regularisierung entspricht einer Glattheitsannahme der Funktion; es gibt eine gewisse Äquivalenz zwischen Kern-Trick, Gauss Prozessen, smoothing Splines, Regularisierungs-Ansätzen, ...
- Der Kern-Trick kann immer dann angewandt werden, wenn sich die Lösung so schreiben lässt, dass die Eingangsvektoren als innere Produkte auftreten; ein bekanntes Beispiel hierfür ist die Kern-PCA

Sinnvolle Kerne sind manchmal leichter zu definieren als sinnvolle Merkmale

Beispiel: Klassifikation von Graphen in der Chemie

- Moleküle werden als Graphen darstellen (Strukturformel) (chemische Graphentheorie)
- Aufgabe: ich weiß von N Molekülen, ob diese medizinisch wirksam sind. Kann ich medizinische Wirksamkeit für neue Moleküle vorhersagen?
- Merkmale, die eine chemische Strukturformelbeschreiben, sind schierig zu definieren; einen sinnvollen Kern kann man leichter definieren

Beispiel: Klassifikation einer Person in einem Sozialen Netzwerk

- Kerne spiegeln die Ähnlichkeit der Personen in Bezug auf die Netzwerk-Topologie wieder. Zum Beispiel kann man einen Kern definieren, basierend auf die Anzahl der Unterstrukturen in der Überlappung von zwei Graphen, die durch die Nachbarschaften der beiden Personen definiert werden

Regression: Kernlösung versus Lösung mit einer Basisfunktion pro Datentpunkt

- Die Vorhersage mit festen Basisfunktionen ist identisch zu der Lösung mit den entsprechenden Kernen!!!
- Als Gedankenspiel: Was ist, wenn ich Kerne auf die Datenpunkte setze und dann diese als feste Basisfuntionen interpretiere?

	Kerne	Basisfunktion
Datenterm	$(\mathbf{y} - K\mathbf{v})^T(\mathbf{y} - K\mathbf{v})$	$(\mathbf{y} - K\mathbf{w})^T(\mathbf{y} - K\mathbf{w})$
Strafterm	$\lambda \mathbf{v}^T K \mathbf{v}$	$\lambda \mathbf{w}^T \mathbf{w}$
Lösung:	$\mathbf{v}_{pen} = (K + \lambda I)^{-1}\mathbf{y}$	$\mathbf{w}_{pen} = (K + \lambda K^{-1})^{-1}\mathbf{y}$
	$\mathbf{v}_{pen} = (KK + \lambda K)^{-1}\mathbf{y}$	$\mathbf{w}_{pen} = (KK + \lambda I)^{-1}K\mathbf{y}$

Wenn K glättet (zB. Gauss Kern), dann ist die Vorhersage mit den Basisfunktionen noch höher geglättet als die Kernlösung.