Rheinisch-Westfälische Technische Hochschule Aachen
Lehrstuhl für Informatik VI

MUSTERERKENNUNG UND NEURONALE NETZE

von
Prof. Dr.-Ing. H. Ney

Sommersemester 1995

Vorlesungsmitschrift
ausgearbeitet von A.Eiden
9. August 1995

Inhaltsverzeichnis

1 Einführung **3**
1.1 Gebiete der Mustererkennung 4
1.2 Aufbau eines Erkennungssystems 5
1.2.1 Ansatz mit Diskriminanten 5
1.2.2 Statistischer Ansatz 7
1.2.3 Typische Aufgaben der Mustererkennung 10
1.3 Zufallsvariablen und Verteilungen 11
1.4 Gauß-Verteilung: univariate und multivariate 17
1.5 Weitere Verteilungen im $\mathbb{R}^D$ 22

2 Bayessche Entscheidungsregel **24**
2.1 Überblick über die verschiedenen Verteilungen 24
2.2 Bayesscher Ansatz und Entscheidungsregel 25
2.3 Diskriminanten und Grenzflächen 27
2.4 Multivariate Gauß-Verteilung 30
2.5 Abstands- oder geometrische Klassifikatoren 33
2.6 Binäre Merkmale 38
2.6.1 Unabhängige binäre Merkmale 39
2.6.2 Abhängige binäre Merkmale 40
2.6.3 Entscheidungsregel und Fehlerrate für Spezialfall . . 42

3 Training und Lernen **45**
3.1 Aufgabenstellung 45
3.2 Verteilung der Abstände im $\mathbb{R}^D$ mit $D \gg 1$ 46
3.3 Momentenmethode 49
3.4 Maximum-Likelihood-Methode 50
3.5 Praktische Aspekte 54
3.6 Bewertungskriterium: empirische Fehlerrate 56
3.7 Bayessches Lernverfahren 57

4 Diskriminanten und Neuronale Netze **63**
4.1 Fehlerquadratkriterium 63
4.2 Strukturen und Multilayer-Perceptron 67
4.2.1 Nichtlinearität 69
4.2.2 Bemerkungen zum Multilayer-Perceptron (MLP) . . 71
4.2.3 Multilayer-Perceptron-ähnliche Strukturen für statistische Klassifikatoren 76

4.3 Error Back Propagation 76
4.4 Diskriminatives Training für statistische Klassifikatoren . . 79

5 Modellfreie Verfahren 80
5.1 Einführung 80
5.2 Kernel Densities 81
5.3 Nächste-Nachbar-Regel (NN-Regel) 84
5.4 Fehlerrate der Nächste-Nachbar-Regel 87
5.5 Ausblick 91

6 Mischverteilungen und Clusteranalyse 92
6.1 Mischverteilungen 92
6.1.1 Einführung 92
6.1.2 EM-Algorithmus 94
6.1.3 Mischverteilungen für K Klassen 100
6.1.4 Maximum-Approximation 101
6.2 Clusteranalyse 102
6.2.1 Übersicht 102
6.2.2 Fehlerquadratkriterium 103
6.2.3 Hierarchische Clusteranalyse 106

7 Stochastische endliche Automaten 112
7.1 Motivation und Modell 112
7.2 Mathematischer Formalismus 115
7.3 Einbau in die Bayessche Entscheidungsregel 118
7.4 Training 119
7.5 Stochastische Grammatiken (Ausblick) 122

8 Merkmalsextraktion und lineare Abbildungen 125
8.1 Ideale Merkmalsextraktion 125
8.2 Lineare Abbildungen 126
8.2.1 Effekt bei gegebener Gauß-Verteilung 126
8.2.2 Minimierung des Repräsentationsfehlers: Karhunen-Loève-Transformation 127
8.2.3 Lineare Diskriminantenanalyse (LDA; Fisher's LDA) 130
8.3 Lineare Klassifikatoren/Diskriminanten 135

9 Literatur 137

1 Einführung

Drei Beispiele für einfache Muster: Links handgeschriebene Ziffern, in der Mitte standardisierte Ziffern und rechts Spannungsverlauf am Mikrofonausgang für das Wort "mit".

Zwei Beispiele für komplexe Muster, links ein Teil eines Stadtgebietes, rechts ein Ausschnitt aus einem Schaltplan.

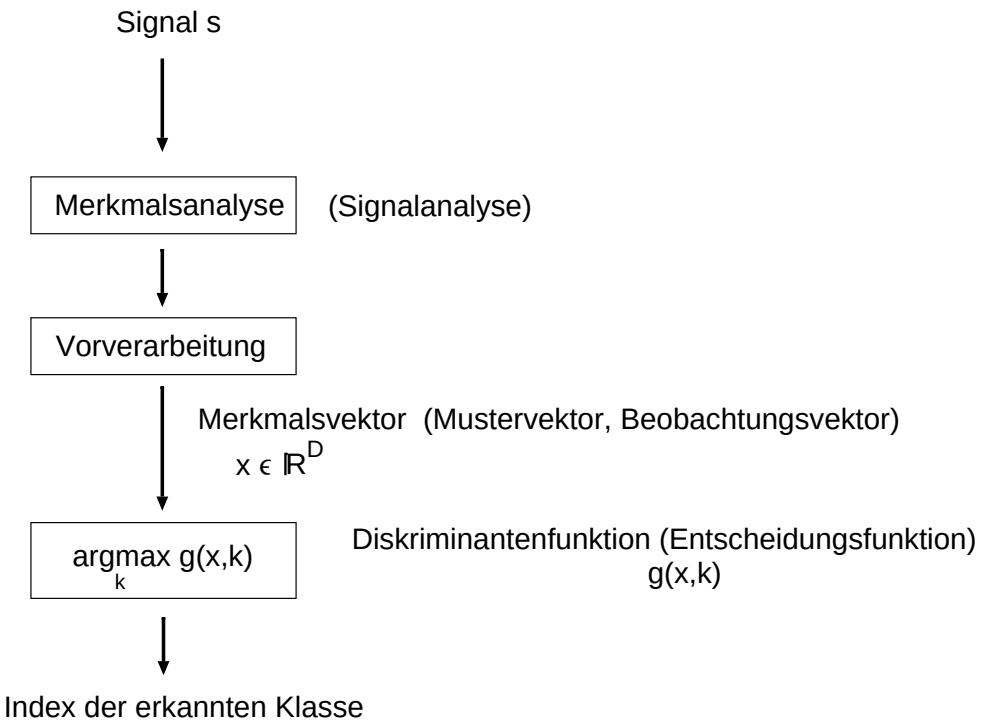
1.1 Gebiete der Mustererkennung

- Signale ("eindimensional")
  - akustische Signale (Geräusche, Motoren, ...)
  - Sprache
  - EKG, EEG, Phonokardiogramm
  - Sonar, Radar
- Bilder ("zweidimensional")
  - medizinische Bilder (Röntgen-, Zell-, Tomographie-, Spin-Tomographie-Bilder)
  - Elementarteilchenphysik
  - Fertigung: Schaltungen, Bauteile, ...
  - Satellitenbilder
- Schriftzeichen
  - OCR (optical character recognition)
  - handgeschriebene Schrift

1.2 Aufbau eines Erkennungssystems

1.2.1 Ansatz mit Diskriminanten

Klassen $k = 1, \ldots, K$ (z.B. Alphabet)

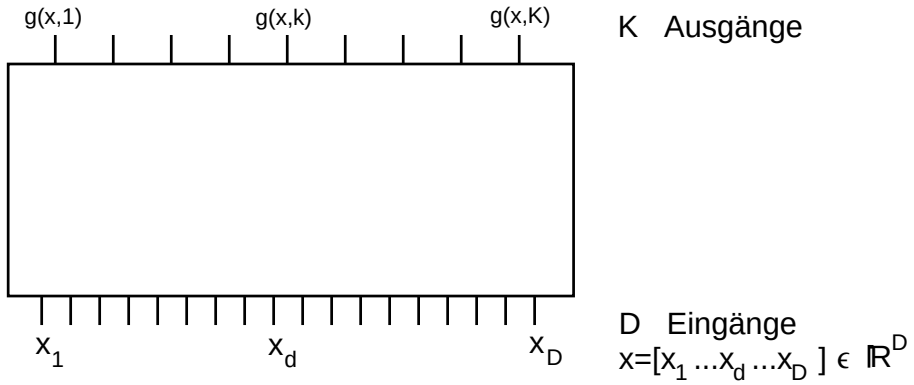


Aufgrund eines beobachteten Signals s und des daraus berechneten Merkmalsvektors $x \in \mathbb{R}^D$ soll auf die zugehörige Klasse geschlossen werden. Dazu benötigen wir eine *Entscheidungsregel* $x \mapsto r(x)$:

$$x \longmapsto r(x) = \underset{k}{\operatorname{argmax}} \{g(x,k)\}$$

$g(x,k)$ heißt *Diskriminantenfunktion.*

Darstellung der Funktion $x \mapsto g(x,k), \quad k = 1, \ldots, K$:



Im Prinzip ist die funktionale Abhängigkeit $x \mapsto g(x,k)$ keinen Einschränkungen unterworfen. In der Praxis wählt man oft:

- lineare Funktionen in x : $\quad g(x,k) = \sum_{d=1}^{D} w_{kd} x_d + w_{k0}$

  quadratische Funktionen, d.h. linear in den Komponenten $x_d\ x_c$:

  $\downarrow$

  Polynomial-Klassifikatoren

- neuronales Netz (ANN = artificial neural net), insbesondere das *Multilayer-Perceptron* (mit einer oder zwei "verborgenen" Schichten)

Entwurfskriterien für die Diskriminante $g(x,k)$:

$g(x,k) \longmapsto 1$ für die "richtige" Klasse
$g(x,k) \longmapsto 0$ für die "falsche" Klasse

Diese idealen Werte lassen sich in der Regel nur approximativ erreichen, so daß (wie wir später sehen werden) das *Fehlerquadratkriterium* in der Lernphase verwendet wird.

1.2.2 Statistischer Ansatz

Klassen $k = 1, \ldots, K$ (z.B. Alphabet)

- a-priori-Wahrscheinlichkeiten: $p(k), \quad \sum_{k=1}^{K} p(k) = 1.$

  In der Regel werden die a-priori-Wahrscheinlichkeiten den relativen Häufigkeiten gleichgesetzt.

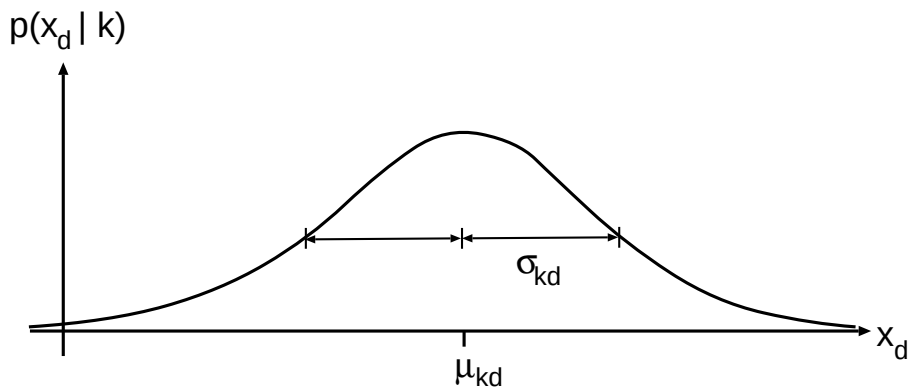
  Beispiel Buchstabenerkennung: Die Buchstaben haben unterschiedliche Häufigkeiten.

- klassenbedingte Wahrscheinlichkeitsdichten: $p(x|k), \qquad x \in \mathbb{R}^D.$

  Beispiel:

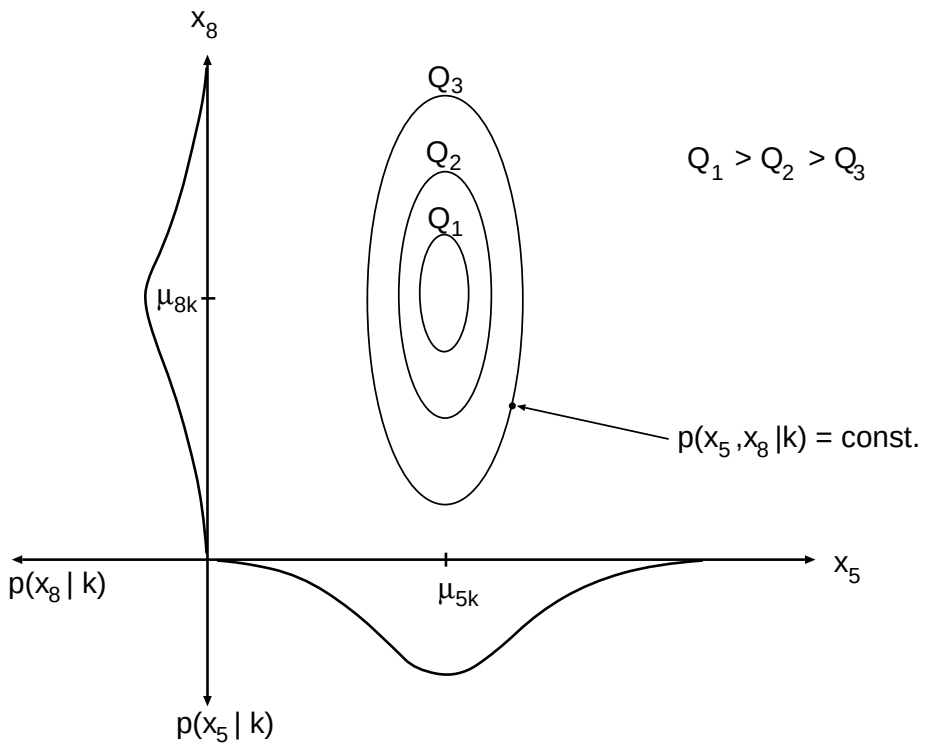
$$p(x|k) = \prod_{d=1}^{D} p(x_d|k)$$

$$p(x_d|k) : \text{ univariate Gauß-Verteilung}$$



μ_{kd} : Mittelwert der Komponente x_d für die Klasse k
σ_{kd}^2 : Varianz der Komponente x_d für die Klasse k

Beispiel (zweidimensional):



Zur Verwendung dieser Verteilungen $p(k)$ und $p(x|k)$ müssen ihre Parameter bekannt sein:

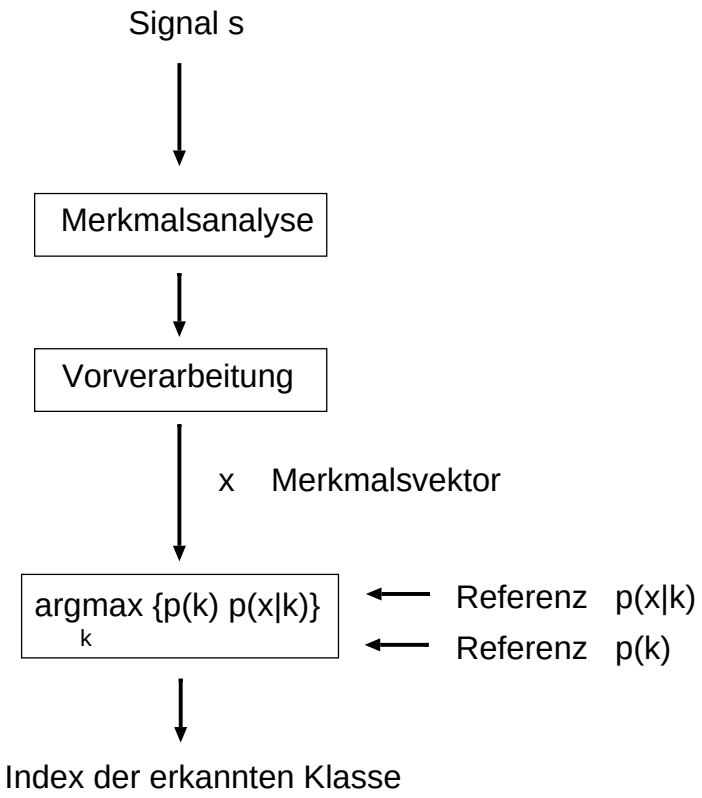
$p(k)$: relative Häufigkeiten

$p(x|k)$: Gauß-Verteilung $p(x|k) = \prod_{d=1}^{D} p(x_d|k)$

$$\left.\begin{array}{l}\text{Mittelwert } \mu_{kd} \\ \text{Varianz } \sigma_{kd}^2\end{array}\right\} \text{ für jede Komponente } x_d \text{ und Klasse } k$$

Für die Anzahl der Parameter ergibt sich:

$K-1$	Werte für $p(k)$	(Normierung: $\sum_{k=1}^{K} p(k) = 1$)
$2 \cdot K \cdot D$	Werte für $p(x\|k)$	$\left[\mu_{kd}, \sigma_{kd}^2\right]$



Zur Bestimmung der zugehörigen Klasse k einer gegebenen Beobachtung x benutzt der statistische Ansatz die Bayessche Entscheidungsregel

$$x \quad \longmapsto \quad r(x) = \underset{k}{\operatorname{argmax}} \{p(k) \cdot p(x|k)\}.$$

Im Verlauf der Vorlesung wird folgender Zusammenhang zwischen den beiden Ansätzen gezeigt:

Im Idealfall approximiert die Diskriminantenfunktion $g(x,k)$ die *a-posteriori-Wahrscheinlichkeit* $p(k|x)$.

$$g(x,k) \quad \rightarrow \quad p(k|x) := \frac{p(k)\ p(x|k)}{\sum_{k'=1}^{K} p(k')\ p(x|k')}$$

1.2.3 Typische Aufgaben der Mustererkennung

In beiden Ansätzen finden wir folgende Aufgaben:

1. Definieren und Berechnen geeigneter Merkmale; in der Regel erfolgt dies problemspezifisch:
   - akustische Signale: Spektralanalyse, Fourier-Transformation, ...
   - Schrift: Form-Merkmale, ...
   - Bilder: Textur, Form-Merkmale, Fourier-Transformation, ...
2. Finden geeigneter Modelle und Strukturen für
   - $p(k)$ und $p(x|k)$ beim statistischen Ansatz
   - $g(x, k)$ beim Ansatz mit Diskriminanten
3. Definition geeigneter Trainingskriterien und Trainingsalgorithmen, um die freien Parameter aus einer Lernstichprobe zu schätzen.
4. Suchproblem: Maximierung
   - unkritisch bei 10 gesprochenen Ziffern oder 36 geschriebenen Buchstaben / Ziffern
   - kritisch bei 10000 Wörtern oder erst recht bei $10000^{10} = 10^{40}$ Sätzen.

Die Aufgabenstellungen 1 und 4 werden in der Vorlesung *Algorithmen der Spracherkennung* behandelt.

1.3 Zufallsvariablen und Verteilungen

Zur Vereinfachung der Notation wird der Klassenindex k weggelassen.

- Diskrete Zufallsvariable:

  Beispiele:

  - Münzwurf: $X = \{0, 1\}$
  - Würfelwurf: $X = \{1, 2, 3, 4, 5, 6\}$

  Wahrscheinlichkeitsverteilung:

$$p(x) \geq 0, \qquad \sum_{x \in X} p(x) = 1$$

  Erwartungswert einer Funktion $x \longmapsto g(x)$:

$$E\{g(x)\} = \sum_{x \in X} p(x) \cdot g(x)$$

  Mittelwert ("gewichtete Mittelung") von x:

$$E\{x\} = \sum_{x \in X} x \cdot p(x)$$

  Varianz von x ("Streuung um den Mittelwert"):

$$\begin{aligned} Var\{x\} &= E\{(x - E\{x\})^2\} \\ &= E\{x^2 - 2x\ E\{x\} + E^2\{x\}\} \\ &= E\{x^2\} - 2\ E^2\{x\} + E^2\{x\} \\ &= E\{x^2\} - E^2\{x\} \end{aligned}$$

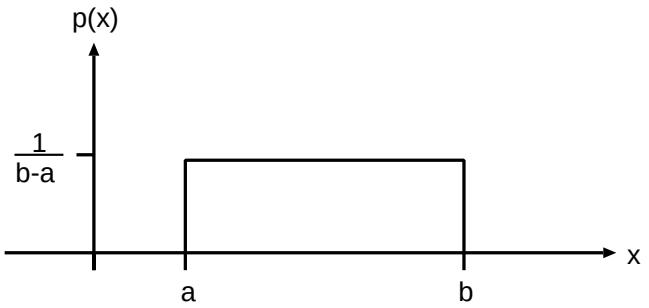
- eindimensionale, kontinuierliche Zufallsvariable: $x \in \mathbb{R}$

  Eine eindimensionale, kontinuierliche Zufallsvariable wird auch *univariate* Zufallsvariable genannt.

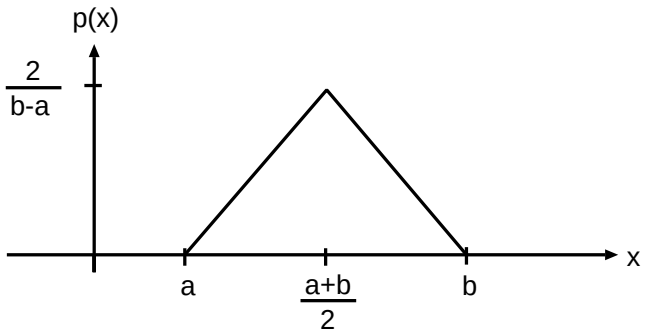
Verteilungsdichten: $p(x) \geq 0, \qquad \int_{-\infty}^{\infty} p(x)\, dx = 1$

Beispiele:

- gleichförmige Verteilung:



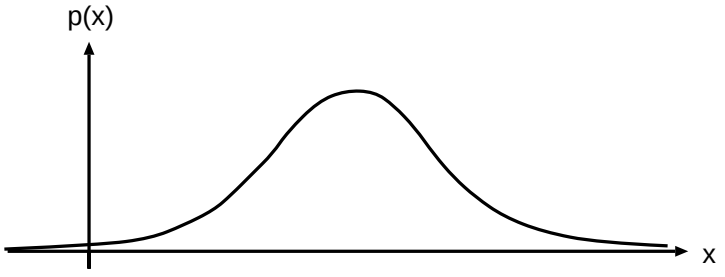
- Dreiecksverteilung:



Diese beiden Beispiele zeigen eine Wahrscheinlichkeitsdichte $p(x)$, die außerhalb eines endlichen Intervalls identisch null ist.

Ein Beispiel für eine Verteilung, deren Wahrscheinlichkeitsdichte überall größer null ist:

- Gauß-Verteilung, Normalverteilung (Glockenkurve):



Gauß-Verteilung:

$$\forall x : p(x) > 0, \qquad p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right).$$

Die Größen $a, b, \mu, \sigma^2, \ldots$ sind zur vollständigen Definition der Dichten erforderlich und heißen Parameter der Verteilungsdichten.

Für kontinuierliche Zufallsvariablen werden Erwartungswert, Mittelwert und Varianz mittels eines Integrals anstatt einer Summe definiert:

Erwartungswert einer Funktion $x \longmapsto g(x)$:

$$E\{g(x)\} = \int_{-\infty}^{\infty} p(x) \cdot g(x)\, dx$$

Mittelwert:

$$E\{x\} = \int_{-\infty}^{\infty} x \cdot p(x)\, dx$$

Varianz:

$$Var\{x\} = \int_{-\infty}^{\infty} (x - E\{x\})^2 \cdot p(x)\, dx$$

Übungsaufgabe: Berechne Mittelwert und Varianz für die drei Beispiele (gleichförmige Verteilung, Dreiecksverteilung, Gauß-Verteilung).

- mehrdimensionale, kontinuierliche Zufallsvariable: $x \in \mathbb{R}^D$

  Mehrdimensionale, kontinuierliche Zufallsvariablen werden auch als *multivariate* Zufallsvariablen bezeichnet.

  $x \in \mathbb{R}^D : \quad x = [x_1, \ldots, x_d, \ldots, x_D]$

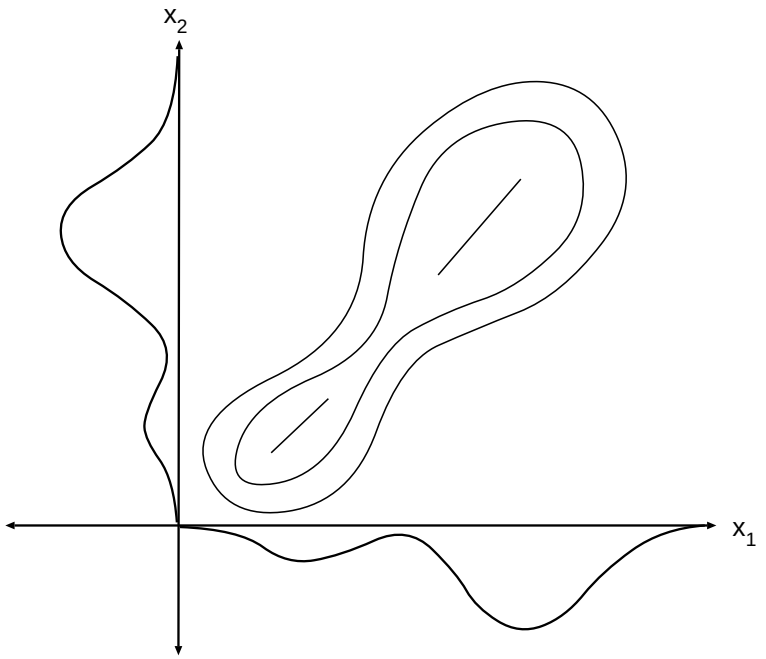
$$p(x) = p(x_1, \ldots, x_d, \ldots, x_D) \geq 0$$

$$\int_{\mathbb{R}^D} p(x)\, dx = \int_{-\infty}^{\infty} \ldots \int_{-\infty}^{\infty} p(x_1, \ldots, x_d, \ldots, x_D)\, dx_1 \ldots dx_d \ldots dx_D = 1$$

Zur Veranschaulichung erfolgen die folgenden Betrachtungen im zweidimensionalen Raum $(D = 2)$: $\quad x = [x_1, x_2]$

$p(x_1, x_2)$ bezeichnet die Verbundverteilung / Verteilungsdichte des Verbundereignisses $[x_1, x_2]$.

Darstellung der Dichte $p(x_1, x_2)$ durch Höhenlinien, d.h. Linien konstanter Wahrscheinlichkeitsdichte:

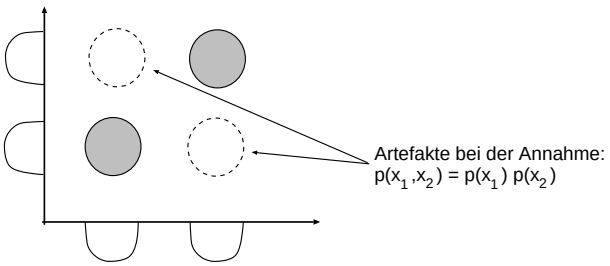


Die Randverteilungen (Marginal-Verteilungen) $p(x_1)$ und $p(x_2)$ ergeben sich durch *Projektionen* auf x_1 bzw. x_2:

$$p(x_1) = \int_{-\infty}^{\infty} p(x_1, x_2)\, dx_2 \ ,$$

$$p(x_2) = \int_{-\infty}^{\infty} p(x_1, x_2)\, dx_1 \ .$$

In der Regel ist $p(x_1, x_2) \neq p(x_1) \cdot p(x_2)$.



Bedingte Verteilungsdichten:

$$\begin{aligned} p(x_1|x_2) &= \frac{p(x_1, x_2)}{p(x_2)} \qquad \text{Bayesscher Satz} \\ &= \frac{p(x_1, x_2)}{\int_{-\infty}^{\infty} p(x_1, x_2)\, dx_1} \end{aligned}$$

Bedeutung: Die Kenntnis von x_2 beinhaltet *Informationen* über x_1, d.h. x_1 kann besser "vorhergesagt" werden.

Sprechweise: x_2 und x_1 sind korreliert (oder abhängig).

(Stochastische) Unabhängigkeit von x_1 und x_2:

$$p(x_1, x_2) = p(x_1) \cdot p(x_2) \qquad \forall\, x_1, x_2.$$

Falls x_1 und x_2 stochastisch unabhängig sind, gilt:

$$\begin{aligned} p(x_1|x_2) &= p(x_1), \\ p(x_2|x_1) &= p(x_2). \end{aligned}$$

Die stochastische Unabhängigkeit von Variablen bedeutet, daß die Kenntnis einer der beiden Variablen keine Information über die andere Variable liefert. Analoges gilt für $D > 2$.

Erwartungswert einer Funktion:

$$\begin{aligned} g: \ \mathbb{R}^D &\longrightarrow \mathbb{R} \\ x &\longmapsto g(x) \end{aligned}$$

$$E\{g(x)\} = \int_{\mathbb{R}^D} g(x) \cdot p(x)\ dx, \qquad E\{g(x)\} \in \mathbb{R}$$

Mittelwert einer Zufallsvariablen $x \in \mathbb{R}^D$:

$$\begin{aligned} E\{x_d\} &= \int_{\mathbb{R}^D} x_d \cdot p(x)\ dx \\ &= \int_{-\infty}^{\infty} x_d \cdot p(x_d)\ dx_d \end{aligned}$$

$E\{x_d\}$ definiert die d-te Komponente des Mittelwertsvektors $E\{x\}$.

Kovarianzmatrix Σ: das Element Σ_{cd} wird definiert durch:

$$\Sigma_{cd} = E\{(x_d - E\{x_d\}) \cdot (x_c - E\{x_c\})\}$$

Das Diagonalelement σ_d^2 der Kovarianzmatrix ist die Varianz der d-ten Komponente.

$$\begin{aligned} \sigma_d^2 &= \Sigma_{dd} = E\{(x_d - E\{x_d\})^2\} \\ &= \int_{-\infty}^{\infty} p(x_d) \cdot (x_d - E\{x_d\})^2\ dx_d \end{aligned}$$

1.4 Gauß-Verteilung: univariate und multivariate

Wir betrachten für eine feste Klasse k die bedingte Wahrscheinlichkeit(sdichte) $p(x|k)$ und die Beobachtungsvektoren $x \in \mathbb{R}^D$.

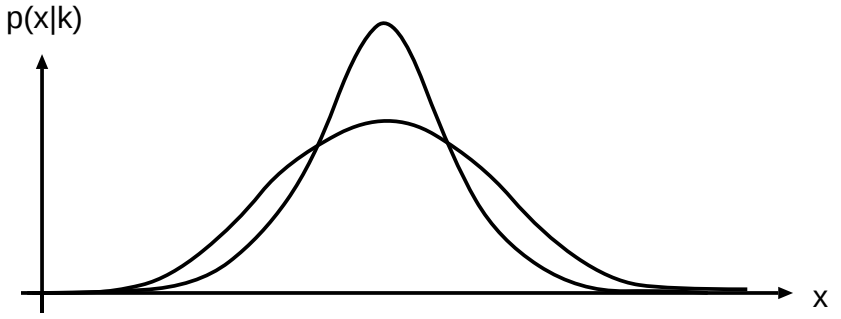
Häufig schwanken die Beobachtungsvektoren um einen "typischen" Prototypvektor. Diese Abweichungen werden oft durch eine Gauß-Verteilung beschrieben.

- Univariate Gauß-Verteilung:

  Wir fixieren eine Vektorkomponente d. Dann lautet die Verteilung der x_d:

$$p(x_d|k) = \frac{1}{\sqrt{2\pi\sigma_{kd}^2}} \exp\left[-\frac{1}{2}\left(\frac{x_d - \mu_{kd}}{\sigma_{kd}}\right)^2\right]$$

  mit den Parametern μ_{kd} und σ_{kd}^2.



  es gilt:

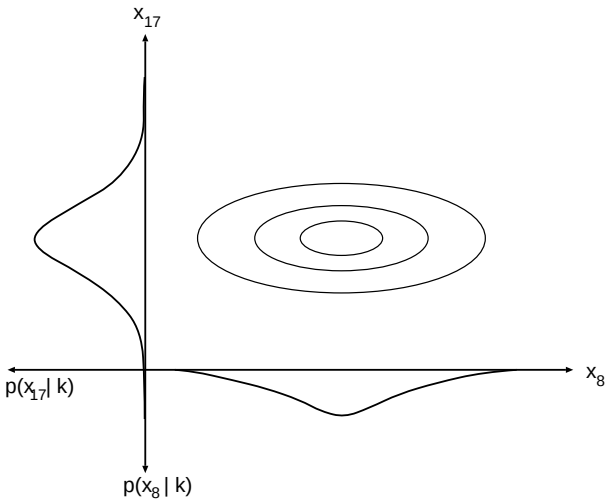
$$\int_{-\infty}^{+\infty} x_d \, p(x_d|k) \, dx_d = \mu_{kd} \qquad \text{Mittelwert}$$

$$\int_{-\infty}^{+\infty} (x_d - \mu_{kd})^2 \, p(x_d|k) \, dx_d = \sigma_{kd}^2 \qquad \text{Varianz}$$

- Multivariate Gauß-Verteilung bei unabhängigen Komponenten:

$$\begin{aligned} p(x|k) &= p(x_1, \ldots, x_d, \ldots, x_D|k) = \prod_{d=1}^{D} p(x_d|k) \\ &= \frac{1}{\prod_{d=1}^{D} \sqrt{2\pi\sigma_{kd}^2}} \exp\left[-\frac{1}{2}\sum_{d=1}^{D}\left(\frac{x_d - \mu_{kd}}{\sigma_{kd}}\right)^2\right] \end{aligned}$$

Höhenlinien $p(x|k) = const$ sind Hyperellipsoide, deren Hauptachsen parallel zu den Koordinatenachsen verlaufen:



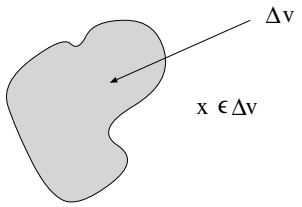
Der negative Logarithmus von $p(x|k)$ kann als Abstand zwischen x und μ_k im $\mathbb{R}^D$ interpretiert werden:

$$-\log p(x|k) = \frac{1}{2}\sum_{d=1}^{D}\left(\frac{x_d - \mu_{kd}}{\sigma_{kd}}\right)^2 + \frac{1}{2}\sum_{d=1}^{D}\log\left(2\pi\sigma_{kd}^2\right)$$

Bemerkungen zur Dimension:

1. Die Argumente in mathematischen Funktionen (wie z.B. exp, log, sin) müssen (eigentlich) dimensionslose Größen sein. Dies wird durch die Festlegung der Dimension der Varianz σ_{kd}^2 erreicht.
2. Wegen 1. können verschiedene Dimensionen (Länge, Zeit, Kraft, ...) in den Komponenten x_d mit $d = 1, \ldots, D$ ohne dimensionsbedingte Probleme kombiniert werden.

3. Dimension der Wahrscheinlichkeitsdichte $p(x|k)$:
   Um eine echte Wahrscheinlichkeit zu berechnen, muß über ein Volumenelement integriert werden.



$$Pr(x \in \Delta \mathrm{v}) \;=\; \int_{\Delta \mathrm{V}} p(x|k)\; dx \cong |\Delta \mathrm{v}| \cdot p(x|k)$$

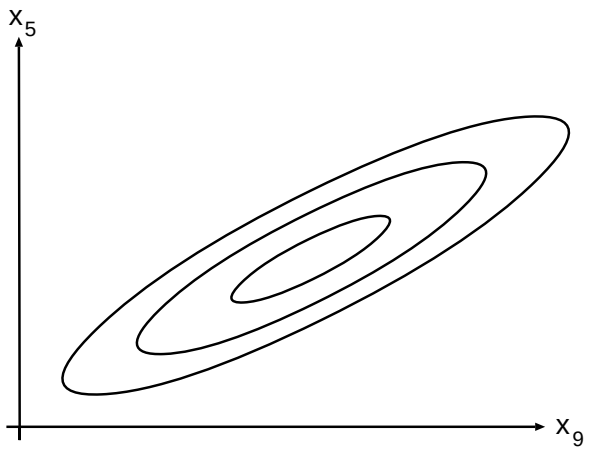
- multivariate Gauß-Verteilung:

  Die allgemeine multivariate Gauß-Verteilung erhält man, wenn im Exponenten eine beliebige quadratische, positiv definite Form verwendet wird:

$$(x - \mu_k)^T \; \Sigma_k^{-1} \; (x - \mu_k)$$

  mit μ_k Mittelwert
  Σ_k^{-1} Inverse der Kovarianz-Matrix

  Die Darstellung der Höhenlinien $p(x|k) = const$ ergibt "gedrehte Hyperellipsoide".



Mit Berücksichtigung der Normierung $\int_{\mathbb{R}^D} p(x|k)\ dx = 1$ gilt dann:

$$p(x|k) \;=\; \frac{1}{\sqrt{(2\pi)^D\ \det \Sigma_k}}\ \exp\left[-\frac{1}{2}(x-\mu_k)^T\ \Sigma_k^{-1}\ (x-\mu_k)\right].$$

Es ergibt sich für den Mittelwert der Komponente d

$$\int_{\mathbb{R}^D} x_d\ p(x|k)\ dx \;=\; \mu_{kd}$$

und für die Kovarianz der Komponenten c und d

$$\int_{\mathbb{R}^D} (x_d - \mu_{kd})\ (x_c - \mu_{kc})\ p(x|k)\ dx \;=\; \Sigma_{k,cd}\ .$$

Übungsaufgabe: Verifiziere, daß im Fall einer diagonalen Kovarianz-Matrix:

$$\Sigma_k = \begin{bmatrix} \sigma_{k1}^2 & & 0 \\ & \ddots & \\ 0 & & \sigma_{kD}^2 \end{bmatrix}$$

sich die Gauß-Verteilung mit unabhängigen Komponenten ergibt.

Vorgriff:

Methode zur empirischen Ermittlung der Modellparameter μ_k und Σ_k:

Wir halten eine Klasse k fest und sammeln alle Beobachtungsvektoren $x_1, \ldots, x_N \in \mathbb{R}^D$, die in die Klasse k fallen. Dann schätzt man den Mittelwertsvektor und die Kovarianzmatrix wie folgt:

$$\begin{aligned} \hat{\mu}_d &:= \frac{1}{N} \sum_{n=1}^{N} x_{nd} \\ \hat{\Sigma}_{cd} &:= \frac{1}{N} \sum_{n=1}^{N} (x_{nd} - \hat{\mu}_d)\ (x_{nc} - \hat{\mu}_c) \end{aligned}$$

Matrix-Schreibweise:

$$\hat{\Sigma} := \frac{1}{N} \sum_{n=1}^{N} (x_n - \hat{\mu})\ (x_n - \hat{\mu})^T$$

Beachte den Unterschied zwischen Skalarprodukt und Matrixmultiplikation: $y \in \mathbb{R}^D$

$$\begin{aligned} y^T \cdot y = \sum_{d=1}^{D} y_d^2 \qquad & [1, D] \times [D, 1] = [1, 1] \\ \left[y \cdot y^T \right]_{cd} = y_c \cdot y_d \qquad & [D, 1] \times [1, D] = [D, D] \end{aligned}$$

1.5 Weitere Verteilungen im $\mathbb{R}^D$

Es gibt (mindestens) zwei Methoden, mit denen man Verteilungen im $\mathbb{R}^D$ "konstruieren" kann.

1. Bilde eine multivariate Verteilung aus dem Produkt univariater Verteilungen. Hierbei wird die paarweise Unabhängigkeit der univariaten Verteilungen vorausgesetzt.

   Sei $p(x_d|k)$ die univariate Verteilung der Vektorkomponente x_d, dann setzt sich die multivariate Verteilung wie folgt zusammen:

   $$p(x_1, \ldots, x_D|k) = \prod_{d=1}^{D} p(x_d|k)$$

   Beispiel: Laplace-Verteilung

   $$p(x_d|k) = \frac{1}{2v_{kd}} \exp\left[-\frac{|x_d - \mu_{kd}|}{v_{kd}}\right]$$

   $$p(x_1, \ldots, x_D|k) = \prod_{d=1}^{D} p(x_d|k) = \left[\prod_{d=1}^{D} \frac{1}{2v_{kd}}\right] \exp\left[-\sum_{d=1}^{D} \left|\frac{x_d - \mu_{kd}}{v_{kd}}\right|\right]$$

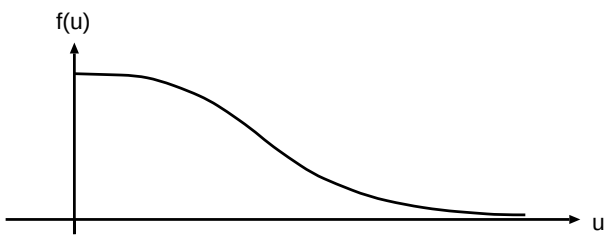
   Normierung : nachrechnen

2. Quadratische Form (oder Abstand):

   $$(x - \mu_k)^T \, W_k \, (x - \mu_k) > 0 \qquad \forall x \in \mathbb{R}^D, x \neq \mu_k$$

   mit einer geeigneten Matrix $W_k \in \mathbb{R}^D \times \mathbb{R}^D$.

   Wähle eine Funktion $f : [0, \infty[\longrightarrow [0, \infty[$, typischerweise monoton fallend



und definiere die Wahrscheinlichkeitsdichte

$$p(x|k) \;=\; \frac{1}{C_{NORM}} f[(x-\mu_k)^T\, W_k\, (x-\mu_k)],$$

wobei sich der Normierungsfaktor C_{NORM} aus der Normierung

$$\int\limits_{\mathbb{R}^D} p(x|k)\, dx \;=\; 1$$

in Abhängigkeit von der Matrix W_k ergibt.

Beispiel: t-Verteilung

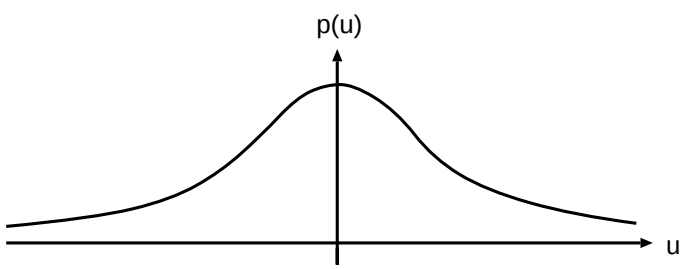
$$p(x|k) \;=\; \frac{1}{C_{NORM}}[1+\frac{1}{M_k}(x-\mu_k)^T\, W_k\, (x-\mu_k)]^{-\left(\frac{M_k+1}{2}\right)},$$

$$\text{mit} \qquad M_k \in \mathbb{N},\ \mu_k \in \mathbb{R}^D,\ W_k \in \mathbb{R}^{D\times D}$$

Ein Unterschied zur Gauß-Verteilung ist der polynomiale Abfall der t-Verteilung für $(x-\mu_k)^T\, W_k\, (x-\mu_k) \to +\infty$.

Ein Spezialfall der t-Verteilung ist die Cauchy-Verteilung. Diese ergibt sich für $M_k = 1$ und $D = 1$:

$$p(u) \;=\; \frac{1}{C_{NORM}} \frac{1}{1+u^2}$$



2 Bayessche Entscheidungsregel

2.1 Überblick über die verschiedenen Verteilungen

Beobachtungen oder Merkmalsvektoren: $x \in \mathbb{R}^D$
Klassen: $k = 1, \ldots, K$

Annahme: Die folgenden Verteilungen seien vollständig bekannt, d.h. bekannt sind die funktionale Form und die Parameter der Verteilungen.

- a-priori-Verteilung der Klassen: $p(k), \quad k = 1, \ldots, K$
- klassenbedingte Verteilungen (oder Modelle) der Merkmalsvektoren x: $p(x|k)$

Daraus lassen sich die folgenden Verteilungen ableiten:

- Verbundverteilung über Paare (x, k):

$$p(x, k) = p(k) \cdot p(x|k)$$

- Verteilung über x (unabhängig von der Klasse k):

$$p(x) = \sum_{k=1}^{K} p(x, k) = \sum_{k=1}^{K} p(k) \cdot p(x|k)$$

- a-Posteriori-Verteilung für jede Klasse k:

$$p(k|x) = \frac{p(x, k)}{p(x)} = \frac{p(k) \cdot p(x|k)}{\sum_{c=1}^{K} p(c) \cdot p(x|c)}$$

  Hierbei ist unmittelbar ersichtlich, daß die Normierungsbedingung

$$\sum_{k=1}^{K} p(k|x) = 1$$

  erfüllt ist.

Beispiel: eindimensional, $D = 1$
Klassen, $K = 2$
a-priori-Wahrscheinlichkeiten $p(k = 1) = \frac{2}{3}$ und $p(k = 2) = \frac{1}{3}$

2.2 Bayesscher Ansatz und Entscheidungsregel

Das Ziel ist es, eine Entscheidungsregel festzulegen, die jede Beobachtung x auf eine Klasse k abbildet.

$$\begin{array}{rcl} r : \mathbb{R}^D & \longrightarrow & \{1, \ldots, K\} \\ x & \mapsto & r(x) \end{array}$$

Die Bewertung der Regel $x \mapsto r(x)$ erfolgt durch eine Kostenfunktion (wie sie bei Bayesschen Ansätzen üblich ist):

- lokale Kosten, falls eine Beobachtung der Klasse c auf eine Klasse k abgebildet wird:

$$L[c,k] \;=\; \begin{cases} (0) & c = k \quad \text{"korrekt"} \\ \ldots & c \neq k \quad \text{"falsch"} \end{cases}$$

- Gesamtkosten ergeben sich durch Integration über den gesamten Raum:

$$\begin{aligned} R &= \int_x \sum_{c=1}^{K} p(x,c) \cdot L[c, r(x)]\; dx \\ &= \int_x p(x) \sum_{c=1}^{K} p(c|x) \cdot L[c, r(x)]\; dx \end{aligned}$$

Für einen gegebenen Merkmalsvektor $x \in \mathbb{R}^D$ werden die Kosten minimiert, indem die folgende Entscheidungsregel gewählt wird:

$$x \mapsto r(x) \;:=\; \underset{k=1\ldots,K}{\operatorname{argmin}} \left\{ \sum_{c=1}^{K} p(c|x) \cdot L[c,k] \right\}$$

("Bayessche Entscheidungsregel in der Mustererkennung")

Oft ist man nur an den Erkennungsfehlern interessiert. Diese ergeben sich als Kosten durch die folgende Definition:

$$L[c,k] \;=\; \begin{cases} 0 & c = k \quad \text{"korrekt"} \\ 1 & c \neq k \quad \text{"falsch"} \end{cases}$$

Damit ergibt sich für die Entscheidungsregel:

$$\begin{aligned} r(x) \;&:=\; \underset{k}{\operatorname{argmin}} \left\{ \sum_{c=1}^{K} p(c|x) \cdot L[c,k] \right\} \\ &=\; \underset{k}{\operatorname{argmin}} \left\{ \sum_{c=1}^{K} p(c|x) - p(k|x) \right\} \\ &=\; \underset{k}{\operatorname{argmin}} \left\{ 1 - p(k|x) \right\} \end{aligned}$$

Es folgt:

$$\begin{aligned} r(x) &= \operatorname*{argmax}_k \{p(k|x)\} \\ &= \operatorname*{argmax}_k \{p(x,k)\} \end{aligned}$$

("Bayessche Entscheidungsregel für minimale Fehlerrate")

Hinweise:

- Beachte die garantierte Optimalität der Bayesschen Entscheidungsregel !
- Beachte die Voraussetzungen !

2.3 Diskriminanten und Grenzflächen

Ausgangspunkt ist die Bayessche Entscheidungsregel

$$r(x) = \operatorname*{argmax}_k \{p(k|x)\} \quad \text{mit} \quad p(k|x) = \frac{p(k) \cdot p(x|k)}{\sum_{c=1}^{K} p(c) \cdot p(x|c)}$$

Oft wird statt $p(k|x)$ eine *Diskriminanten-* oder *Entscheidungsfunktion* $g(x,k)$ verwendet:

$$r(x) = \operatorname*{argmax}_k \{g(x,k)\},$$

wobei sich $g(x,k)$ aus $p(k|x)$ durch geeignete Umformungen ergibt.

Die folgenden Diskriminanten $g(x,k)$ verändern die Entscheidung für eine Klasse k zu gegebenem Merkmalsvektor x nicht. Deshalb ist die Fehlerrate invariant bzgl. der Wahl einer dieser Diskriminanten.

1. $g(x,k) := p(x) \cdot p(k|x) = p(x,k) = p(k) \cdot p(x|k)$
2. $g(x,k) := \log p(x,k) = \log p(k) + \log p(x|k)$
3. $\begin{aligned} g(x,k) &:= \log p(k|x) \\ &= \log\left[p(k) \cdot p(x|k)\right] - \log\left[\sum_{c=1}^{K} p(c) \cdot p(x|c)\right] \end{aligned}$

Aus den Diskriminanten ergeben sich die Grenzflächen. Die Grenzfläche zwischen den Klassen k und c wird definiert durch:

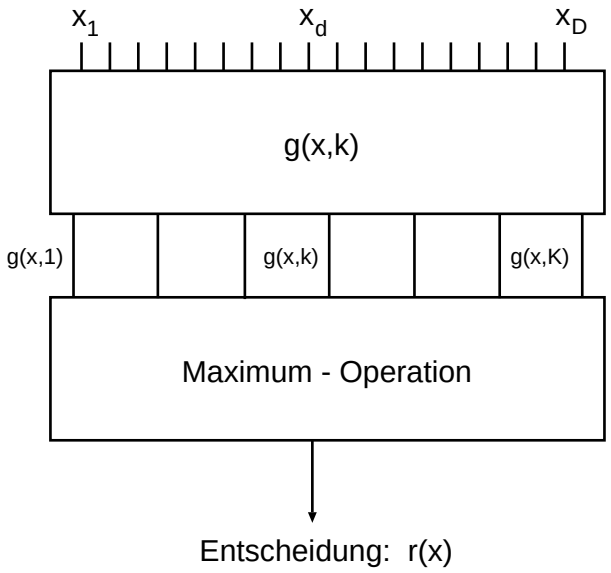
$$\{x \in \mathbb{R}^D \ : \ g(x,k) = g(x,c)\}.$$

Beachte: Bei K Klassen gibt es $\frac{K \cdot (K-1)}{2}$ mögliche Klassenpaare und entsprechend viele Grenzflächen (im maximalen Fall).

Beispiel: $D = 1, K = 3$
Betrachte die Verbundwahrscheinlichkeiten $p(x,k) = p(k) \cdot p(x|k)$:

Veranschaulichung des Diskriminantenansatzes:

$$r(x) \;=\; \underset{k}{\operatorname{argmax}} \{g(x,k)\}$$



Dieses Konzept besteht aus

- Diskriminanten- oder Bewertungsfunktionen $g(x,k)$,
- Maximum-Operation.

Man findet es auch in anderen ("nicht-statistischen") Ansätzen:

- neuronale Netze,
- geometrische Klassifikatoren.

2.4 Multivariate Gauß-Verteilung

Für jede Klasse k und jeden Merkmalsvektor $x \in \mathbb{R}^D$ existiert die klassenbedingte Verteilung

$$p(x|k) = \left[(2\pi)^D \det \Sigma_k\right]^{-\frac{1}{2}} \cdot \exp\left[-\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1} (x-\mu_k)\right]$$

mit μ_k Mittelwertsvektor,
Σ_k Kovarianzmatrix.

Es gilt $(x-\mu_k)^T \Sigma_k^{-1} (x-\mu_k) \geq 0 \qquad \forall x \in \mathbb{R}^D$.
Daher kann dieser Term als Abstand interpretiert werden.

Wir wählen die folgende Diskriminante:

$$\begin{aligned} g(x,k) &:= \log p(k) + \log p(x|k) \\ &= \log p(k) - \frac{1}{2}\log\left[(2\pi)^D \det \Sigma_k\right] - \frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1} (x-\mu_k) \end{aligned}$$

$g(x,k)$ ist quadratisch in x (*quadratische Form*).

Fallunterscheidung zur Bestimmung der Grenzflächen:

1. Σ_k beliebig

   Allgemeinster Fall: Die Grenzfläche $\{x \in \mathbb{R}^D : g(x,k) = g(x,c)\}$ für die Klassengrenze zwischen zwei Klassen k und c ist ebenfalls eine quadratische Form in x. (Hyperquadratiken: -Ebenen, -Kugeln, -Ellipsoide, -Hyperboloide, -Paraboloide)

   Beispiele: $x \in \mathbb{R}^2, K = 2$ Klassen

2. $\Sigma_k = \Sigma$, klassenunabhängige Kovarianzmatrix (*pooled covariance matrix*)

   Die Grenzflächen sind linear in x, d.h. Hyperebenen, weil sich in der Gleichung $g(x,c) = g(x,k)$ die in x quadratischen Terme aufheben.

   Hinweis: Durch Hauptachsentransformation (= Diagonalisierung der Kovarianzmatrix Σ) und Skalierung der Koordinatenachsen kann dieser Fall zurückgeführt werden auf:

$$\Sigma_k = \sigma^2 I$$

(Klassifikation mit *euklidischem Abstand*)

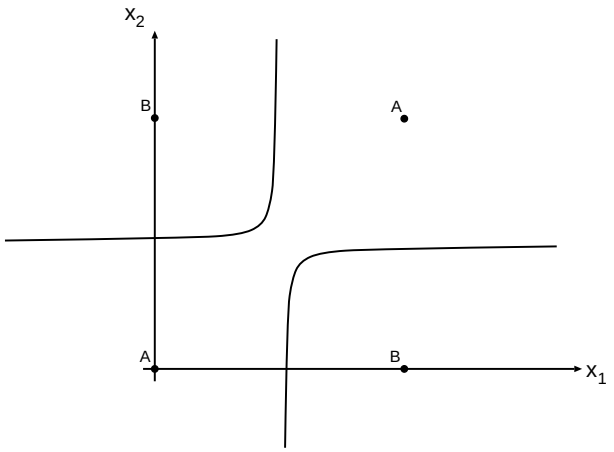
Übungsaufgaben:

1. Berechne explizit die Klassengrenze im Fall:

$$\Sigma_k = \Sigma \qquad \forall\, k$$

2. Rechne die "Bilder" nach.

3. Betrachte das *XOR*-Problem, das oft im Kontext von neuronalen Netzen behandelt wird:



Kann man das *XOR*-Problem mit einem Gauß-Ansatz lösen?

Falls ja, wie?

2.5 Abstands- oder geometrische Klassifikatoren

Ausgangspunkt: Gauß-Verteilung mit $\Sigma_k = \sigma^2 \, I$

Für die Diskriminante $g(x,k) = \log\left[p(k) \cdot p(x|k)\right]$ ergibt sich durch Weglassen der bzgl. k konstanten Terme folgender Ausdruck:

$$\begin{aligned} g(x,k) &= -\frac{1}{2\sigma^2}(x-\mu_k)^T(x-\mu_k) + \log p(k) \\ &= -\frac{1}{2\sigma^2}\sum_{d=1}^{D}(x_d - \mu_{kd})^2 + \log p(k) \end{aligned}$$

$\sum\limits_{d=1}^{D}(x_d - \mu_{kd})^2$ ist der quadrierte euklidische Abstand zwischen x und μ_k.

Variante: Der *Korrelationsklassifikator* ergibt sich durch Weglassen der in x quadratischen Terme:

$$\tilde{g}(x,k) = -\frac{1}{2\sigma^2}\left[\sum_{d=1}^{D}\mu_{kd}^2 - 2\sum_{d=1}^{D}\mu_{kd}\, x_d\right] + \log p(k)$$

Der einzige von x abhängige Term ist die *Korrelation* zwischen x und μ_k:

$$\mu_{kd}^T \, x = \sum_{d=1}^{D}\mu_{kd}\, x_d$$

(Früher war diese Methode wichtig für "schnelle" Hardware-Implementierungen, z.B. Radar.)

Verallgemeinerung: Gauß-Verteilung mit $\Sigma_k = \Sigma$ klassenunabhängig

Durch Diagonalisierung von Σ und Skalierung der Koordinatenachsen kann dieser Fall zurückgeführt werden auf:

$$\Sigma_k = \sigma^2 \, I$$

(Die Transformation heißt auch *Whitening-Transformation*)

Abstandsklassifikatoren:

Abstandsklassifikatoren basieren auf einer Abstandsfunktion zwischen x und μ_k:

$$d(x, \mu_k) \geq 0$$

Man ignoriert die a-priori-Wahrscheinlichkeit $p(k)$ und definiert

$$g(x, k) = -d(x, \mu_k),$$

so daß die Entscheidungsregel lautet:

$$r(x) = \operatorname*{argmin}_k \{d(x, \mu_k)\}$$

Die Terminologie ist nicht eindeutig; es existieren die Bezeichnungen

- minimum distance,
- nearest neighbour,
- nearest prototype (centre, mean).

Oft verwendet man die l_p-*Normen* (mit $p \in \mathbb{N}$):

$$\begin{aligned} d(x, \mu_k) &:= ||x - \mu_k||_p \\ &= \left[\sum_{d=1}^{D} |x_d - \mu_{kd}|^p \right]^{1/p} \end{aligned}$$

Die l_p-Normen erfüllen die Norm-Kriterien, insbesondere die Dreiecksungleichung. Sie definieren somit indirekt ein Abstandsmaß.

Spezialfälle

- $p = 1$: Betragsabstand (City-Block-Abstand, chessboard distance, vgl. Laplace-Verteilung)

$$d(x, \mu_k) = \sum_{d=1}^{D} |x_d - \mu_{kd}|$$

- $p = 2$: Euklidischer Abstand (vgl. Gauß-Verteilung, aber beachte die Wurzel)

$$d(x, \mu_k) = \sqrt{\sum_{d=1}^{D} (x_d - \mu_{kd})^2}$$

- $p = \infty$: "Maximum-Abstand" (Tschebyschew-Abstand)

$$d(x, \mu_k) = \max_d |x_d - \mu_{kd}|$$

Übungsaufgabe: Berechne für $x \in \mathbb{R}^2$ und $K = 3$ Klassen die Grenzflächen für die Fälle $p = 1, p = 2$ und $p = \infty$.

Verfeinerungen: Zuweilen werden die Abstandsklassifikatoren durch klassen- und achsenabhängige Skalierungen v_{kd} (*Varianzen*) verfeinert. Letztlich erhält man so wieder ein statistisches Verfahren.

Bestimmung der Fehlerrate für den Spezialfall:

- Gauß-Verteilung mit $\Sigma_k = \Sigma$
- $K = 2$ Klassen

Für diesen Spezialfall existiert eine geschlossene Lösung.

Wie aus Kapitel 2.4 bekannt ist, liegt wegen der Voraussetzung $\Sigma_1 = \Sigma_2$ eine lineare Grenzfläche vor.

Differenz der Diskriminanten $g(x, 1)$ und $g(x, 2)$:

$$g(x,1) - g(x,2) = \log \frac{p(k=1)}{p(k=2)} + \underbrace{(\mu_1 - \mu_2)^T \Sigma^{-1} x + \frac{1}{2}(\mu_2^T \Sigma^{-1} \mu_2 - \mu_1^T \Sigma^{-1} \mu_1)}_{h(x)}$$

Für eine Beobachtung $x \in \mathbb{R}^D$ reicht es aus, den Abstand $h(x)$ von x zu der linearen Grenzfläche zu berechnen und über diesen Abstand - anstatt über den gesamten Raum $\mathbb{R}^D$ - zu integrieren. Da $h(x)$ linear in x ist, ist auch $h(x)$ gaußverteilt (nach einem allgemeinen Satz über die Gauß-Verteilung).

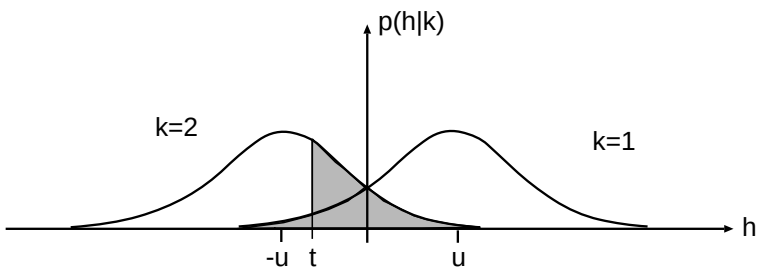
Wir brauchen den Mittelwert u_k und die Varianz v_k^2 von $h(x)$ für jede Klasse $k = 1, 2$. Es ergibt sich ([Fuk90], p.85/86):

$$\begin{aligned} u_1 &= -u_2 = u \\ v_1^2 &= v_2^2 = v^2 \\ \text{mit } v^2 &= 2\,u = (\mu_2 - \mu_1)^T \Sigma^{-1} (\mu_2 - \mu_1) \end{aligned}$$

Die Größe v^2 spielt eine wichtige Rolle und wird als *Mahalanobis-Abstand* (zwischen μ_1 und μ_2) bezeichnet.
Die Gesamtfehlerrate $P(E)$ ergibt sich zu:

$$\begin{aligned} P(E) &= p(k=1)\; P(E_{1\to 2}) + P(p=2)\; P(E_{2\to 1}) \\ &= p(k=1)\frac{1}{\sqrt{2\pi}} \int_{\frac{u+t}{v}}^{\infty} \exp\left(-\frac{h^2}{2}\right) dh + p(k=2)\frac{1}{\sqrt{2\pi}} \int_{\frac{u-t}{v}}^{\infty} \exp\left(-\frac{h^2}{2}\right) dh \end{aligned}$$

$$\text{mit } t := \log \frac{p(k=1)}{p(k=2)}$$



Diese Gleichung für $P(E)$ zeigt, daß die Fehlerrate $P(E)$ nur abhängig ist von

- dem Verhältnis der a-priori-Wahrscheinlichkeiten $\frac{p(k=1)}{p(k=2)}$
- und dem Mahalanobis-Abstand $v^2 = (\mu_2 - \mu_1)^T \Sigma^{-1} (\mu_2 - \mu_1)$.

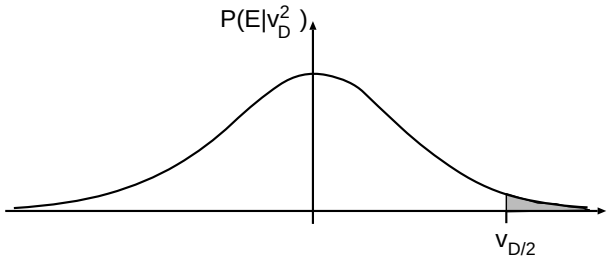
Für den Spezialfall $p(k=1) = p(k=2)$ ergibt sich für den Gesamtfehler $P(E)$ folgende vereinfachte Darstellung:

$$P(E) = \frac{1}{\sqrt{2\pi}} \int_{\frac{v}{2}}^{\infty} \exp\left(-\frac{h^2}{2}\right) dh$$

d.h. der Gesamtfehler $P(E)$ hängt nur von dem Mahalanobis-Abstand ab.

Wie ändert sich die Fehlerrate $P(E)$, wenn die Dimension D der Merkmalsvektoren erhöht wird ?
Um die Gesamtfehlerrate als Funktion der Dimension D auszudrücken, schreiben wir:

$$P(E|v_D^2) = \frac{1}{\sqrt{2\pi}} \int_{v_D/2}^{\infty} \exp\left(-\frac{h^2}{2}\right) dh$$



An der Abbildung ist ersichtlich, daß gilt:

$$\lim_{D \to \infty} P(E|v_D^2) = 0,$$

sofern bei wachsender Dimension D

$$\lim_{D \to \infty} v_D^2 = \infty.$$

In diesem Fall kann die Fehlerrate $P(E|v_D^2)$ beliebig gesenkt werden.

Zur Veranschaulichung nehmen wir an, daß

$\Sigma \in \mathbb{R}^{D \times D}$: eine Diagonalmatrix mit Diagonalelementen $\sigma_d^2 > 0$ ist

Dann ist der Mahalanobis-Abstand v_D^2 für die Dimension D:

$$v_D^2 = \sum_{d=1}^{D} \left(\frac{u_{1d} - u_{2d}}{\sigma_d} \right)^2$$

D.h., wenn jede neue Dimension genügend viel zu diesem Abstand beiträgt, kann die Fehlerrate beliebig gesenkt werden.

2.6 Binäre Merkmale

Bisher haben wir Zufallsvektoren $x \in \mathbb{R}^D$ betrachtet. Der Bayessche Ansatz gilt aber auch bei diskreten Beobachtungen, wie sie durch Diskretisierung von Meßwerten entstehen können.
Strenggenommen ist es jetzt besser, von Merkmalssätzen als von Merkmalsvektoren zu reden:

$$x = [x_1, \ldots, x_d, \ldots, x_D]$$

Besonders einfach ist der Fall binärer Merkmale:

$$x_d \in \{0, 1\}$$

Dann gibt es insgesamt 2^D mögliche Konfigurationen von Merkmalssätzen.

2.6.1 Unabhängige binäre Merkmale

Sei $x = [x_1, \ldots, x_d, \ldots, x_D]$ mit $x_d \in \{0, 1\}$ ein binärer Merkmalssatz, dann bedeutet die (klassenbedingte) Unabhängigkeit der Merkmale:

$$p(x|k) = \prod_{d=1}^{D} p(x_d|k) \qquad \text{für jede Klasse } k = 1, \ldots, K$$

Für ein einzelnes binäres Merkmal x_d gilt:

$$\begin{aligned} p(x_d|k) &= \begin{cases} \vartheta_{kd} & x_d = 1 \\ 1 - \vartheta_{kd} & x_d = 0 \end{cases} \\ &= (\vartheta_{kd})^{x_d} \cdot (1 - \vartheta_{kd})^{1-x_d} \end{aligned}$$

mit der klassenbedingten *Trefferquote* ϑ_{kd}.
Für den ganzen Merkmalssatz x folgt daraus:

$$\begin{aligned} p(x|k) &= \prod_{d=1}^{D} \left[(\vartheta_{kd})^{x_d} (1 - \vartheta_{kd})^{1-x_d} \right] \\ &= \prod_{d=1}^{D} \left[\left(\frac{\vartheta_{kd}}{1 - \vartheta_{kd}} \right)^{x_d} (1 - \vartheta_{kd}) \right] \end{aligned}$$

Wir verwenden die Diskriminante

$$g(x, k) := \log \left[p(k) \cdot p(x|k) \right]$$

und erhalten durch Einsetzen der klassenbedingten Verteilung $p(x|k)$:

$$g(x, k) = \sum_{d=1}^{D} x_d \cdot \log \frac{\vartheta_{kd}}{1 - \vartheta_{kd}} + \sum_{d=1}^{D} \log (1 - \vartheta_{kd}) + \log p(k)$$

Diese Diskriminante ist linear in $[x_1, \ldots, x_D]$.

Übungsaufgabe: Für ternäre Merkmale $x_d \in \{-1, 0, 1\}$ ergibt sich eine quadratische Diskriminante.

2.6.2 Abhängige binäre Merkmale

Rademacher-Walsh-Expansion

Wir betrachten wieder den binären Merkmalssatz $x = [x_1, \ldots, x_d, \ldots, x_D]$ mit $x_d \in \{0, 1\}$.

Zur Darstellung des Modells $p(x|k)$ werden 2^D mögliche Wahrscheinlichkeiten (genauer: $2^D - 1$ wegen der Normierung) benötigt.

Ansatz von Rademacher-Walsh:

$$p(x|k) \;=\; \sum_{i=0}^{2^D-1} a_i(k) \cdot \varphi_i(x)$$

Die Basisfunktionen $\varphi_i(x)$ erfassen mit zunehmendem i komplexere Abhängigkeiten zwischen den Merkmalskomponenten x_d und sind definiert durch:

$$\varphi_i(x) \;=\; \begin{cases} 1 & i = 0 \\ 2x_1 - 1 & i = 1 \\ \vdots & \vdots \\ 2x_D - 1 & i = D \\ (2x_1 - 1)(2x_2 - 1) & i = D + 1 \\ \vdots & \vdots \\ (2x_{D-1} - 1)(2x_D - 1) & i = D + 1 + \dfrac{D(D-1)}{2} \\ (2x_1 - 1)(2x_2 - 1)(2x_3 - 1) & \\ \vdots & \vdots \\ (2x_1 - 1)(2x_2 - 1) \ldots (2x_D - 1) & i = 2^D - 1 \end{cases}$$

Die Basisfunktionen sind orthogonal, d.h.

$$\sum_x \varphi_i(x) \cdot \varphi_j(x) \;=\; \begin{cases} 2^D & , i = j \\ 0 & , i \neq j \end{cases} ,$$

wobei die Summe über alle 2^D Konfigurationen geht. Wegen der Orthogonalität folgt für die Koeffizienten $a_i(k)$:

$$\begin{aligned} a_i(k) &= \frac{1}{2^D} \sum_x \varphi_i(x) \cdot p(x|k) \\ &= \frac{1}{2^D} \, E\left[\varphi_i(x)\right] \end{aligned}$$

Vorgriff: Das Training der $a_i(k)$ erfolgt mittels *empirischer Mittelung* der Trainingsdaten $x_1, \ldots, x_N$ der Klasse k:

$$\hat{a}_i(k) \;=\; \frac{1}{N} \sum_{n=1}^{N} \frac{1}{2^D} \varphi_i(x_n)$$

Bemerkungen zur orthogonalen Reihenentwicklung:

- Man kann die Reihe abschneiden, damit nur die wichtigsten Abhängigkeiten zwischen den x_{kd} erfaßt werden.
- Aber: Die resultierende Funktion $\tilde{p}(x|k)$ kann negativ werden!
- Alternativer Ansatz, der erzwingt, daß $\tilde{p}(x|k)$ positiv ist:

$$\log p(x|k) \;=\; \sum_{i=0}^{2^D-1} b_i(k) \varphi_i(x)$$

verwandte Methoden ([DH73], p.111-114):

- Entwicklung nach Bahadur - Lazarsfeld
- *Dependence Tree* nach Chow

2.6.3 Entscheidungsregel und Fehlerrate für Spezialfall

Annahmen:

- 2 Klassen: $k = 1, 2$ mit $p(1) = p(2) = \frac{1}{2}$
- unabhängige binäre Merkmale mit $\lambda > \frac{1}{2}$ und

$$p(x_d|k=1) = \begin{cases} \lambda & x_d = 1 \\ (1-\lambda) & x_d = 0 \end{cases}$$
$$p(x_d|k=2) = \begin{cases} (1-\lambda) & x_d = 1 \\ \lambda & x_d = 0 \end{cases}$$

Für die beiden Diskriminanten $g(x,1)$ und $g(x,2)$ gilt dann:

$$\begin{aligned} g(x,1) &= \sum_{d=1}^{D} x_d \cdot \log \frac{\lambda}{1-\lambda} + \sum_{d=1}^{D} \log(1-\lambda) + \log(\frac{1}{2}) \\ &= \sum_{d=1}^{D} x_d \cdot \log \frac{\lambda}{1-\lambda} + D \cdot \log(1-\lambda) + \log(\frac{1}{2}) \\ g(x,2) &= \sum_{d=1}^{D} x_d \cdot \log \frac{1-\lambda}{\lambda} + D \cdot \log \lambda + \log(\frac{1}{2}) \end{aligned}$$

Die Differenz der Diskriminanten lautet:

$$g(x,1) - g(x,2) = \sum_{d=1}^{D} (2x_d - 1) \cdot \log \frac{\lambda}{1-\lambda}$$

Wegen $\lambda > \frac{1}{2}$ gilt $\log \frac{\lambda}{1-\lambda} > 0$, und wir erhalten die Entscheidungsregel $r(x)$:

$$r(x) = \begin{cases} k = 1 & \text{falls } \sum_{d=1}^{D} x_d > \frac{D}{2} \\ k = 2 & \text{falls } \sum_{d=1}^{D} x_d < \frac{D}{2} \\ \text{beliebig} & \text{falls } \sum_{d=1}^{D} x_d = \frac{D}{2} \quad \text{(tritt nur auf bei geradem } D\text{)} \end{cases}$$

Bestimmung der Fehlerrate für ungerade Dimension D der Merkmalsvektoren:

Fallunterscheidung:

1. Eine Beobachtung x der Klasse 1 wird irrtümlich der Klasse 2 zugeordnet.

   d (mit $0 \leq d \leq \frac{D-1}{2}$) von D Komponenten sind gleich 1:

   Wahrscheinlichkeit dafür

   $$\lambda^d \cdot (1-\lambda)^{D-d}$$

   Anzahl der verschiedenen Konfigurationen (vgl. Binomial-Verteilung):

   $$\binom{D}{d} = \frac{D!}{d! \cdot (D-d)!}$$

   Damit erhalten wir:

   $$P(E_{1\to 2}|D,\lambda) = \sum_{d=0}^{\frac{D-1}{2}} \binom{D}{d} \cdot \lambda^d \cdot (1-\lambda)^{D-d}$$

2. Eine Beobachtung x der Klasse 2 wird irrtümlich der Klasse 1 zugeordnet.

   d (mit $0 \leq d \leq \frac{D-1}{2}$) von D Komponenten sind gleich 0:

   $$P(E_{2\to 1}|D,\lambda) = \sum_{d=0}^{\frac{D-1}{2}} \binom{D}{d} \cdot \lambda^d \cdot (1-\lambda)^{D-d}$$

Als Gesamtfehlerrate $P(E|D,\lambda)$ ergibt sich:

$$\begin{aligned} P(E|D,\lambda) &= p(k=1) \cdot P(E_{1\to 2}|D,\lambda) + p(k=2) \cdot P(E_{2\to 1}|D,\lambda) \\ &= \sum_{d=0}^{\frac{D-1}{2}} \binom{D}{d} \cdot \lambda^d \cdot (1-\lambda)^{D-d} \end{aligned}$$

Es ergeben sich die Grenzfälle:

1. $\lim_{\lambda \to \frac{1}{2}} P(E|D,\lambda) = \frac{1}{2}$ (Beweis: Übungsaufgabe)
2. $\lim_{D \to \infty} P(E|D,\lambda) = 0$

Das bedeutet, daß zusätzliche Merkmale mit $p(x_d|k) = \lambda^{x_d} \cdot (1-\lambda)^{1-x_d}$ garantiert die Fehlerrate senken.

Beweis zum Grenzfall $\lim_{D \to \infty} P(E|D,\lambda) = 0$:

$$\begin{aligned}
P(E|D,\lambda) &= \sum_{d=0}^{\frac{D-1}{2}} \binom{D}{d} \cdot \lambda^d \cdot (1-\lambda)^{D-d} \qquad \text{mit } \lambda > \frac{1}{2} \\
&= \sum_{d=0}^{\frac{D-1}{2}} \binom{D}{d} \cdot \lambda^{\frac{D}{2}} \cdot (1-\lambda)^{\frac{D}{2}} \cdot \underbrace{\left(\frac{1-\lambda}{\lambda}\right)^{\frac{D}{2}-d}}_{\leq 1 \quad \forall\, 0 \leq d \leq \frac{D-1}{2}} \\
&\leq \lambda^{\frac{D}{2}} \cdot (1-\lambda)^{\frac{D}{2}} \cdot \sum_{d=0}^{\frac{D-1}{2}} \binom{D}{d} \\
&< \lambda^{\frac{D}{2}} \cdot (1-\lambda)^{\frac{D}{2}} \cdot \underbrace{\sum_{d=0}^{D} \binom{D}{d}}_{=2^D = 4^{D/2}} \\
&< \left[4\,\lambda(1-\lambda)\right]^{\frac{D}{2}}
\end{aligned}$$

Wegen

$$4\,\lambda(1-\lambda) < 1 \quad \text{folgt} \quad \lim_{D \to \infty} \left[4\,\lambda(1-\lambda)\right]^{\frac{D}{2}} = 0$$

und somit die Behauptung

$$\lim_{D \to \infty} P(E|D,\lambda) \;=\; 0.$$

3 Training und Lernen

Speziell: klassenindividuell und nicht diskriminativ

3.1 Aufgabenstellung

Notation:	$k = 1, \ldots, K$	Klassen
	$x \in \mathbb{R}^D$	Merkmalsvektor

Innerhalb des statistischen Ansatzes sind wir ausgegangen von:

$p(k)$	a-priori-Wahrscheinlichkeit
$p(x\|k)$	klassenbedingte Verteilungen oder Modelle

In den ersten beiden Kapiteln wurde für die Modelle vorausgesetzt, daß

- die funktionale Form, z.B. Gauß $x \mapsto p(x|k)$,
- sowie die Parameter der Modelle, z.B. bei Gauß-Verteilung

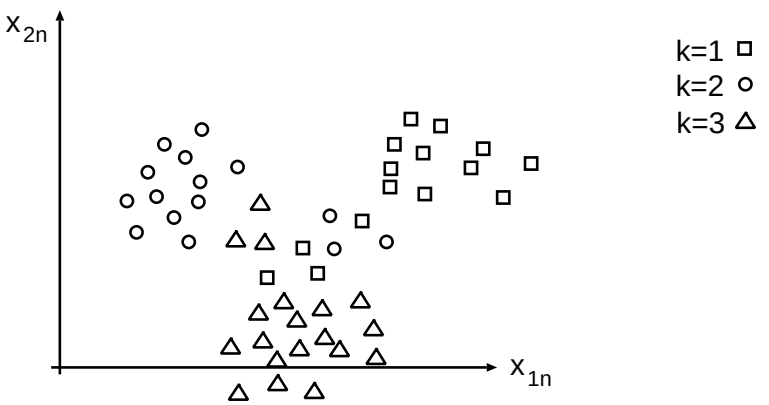
  Mittelwertsvektor μ_k

  Kovarianzmatrix Σ_k

bekannt sind.

In der Praxis sind die Parameter der Modelle $p(x|k)$ nicht bekannt, sondern müssen aus Trainingsdaten gelernt oder geschätzt werden.
Dabei sind die Trainingsdaten Paare (x_n, k_n) mit $n = 1, \ldots, N$, wobei x_n der n-te Merkmalsvektor und $k_n \in \{1, \ldots, K\}$ die zugehörige Klasse ist.

Beispiel: $x_n \in \mathbb{R}^2$



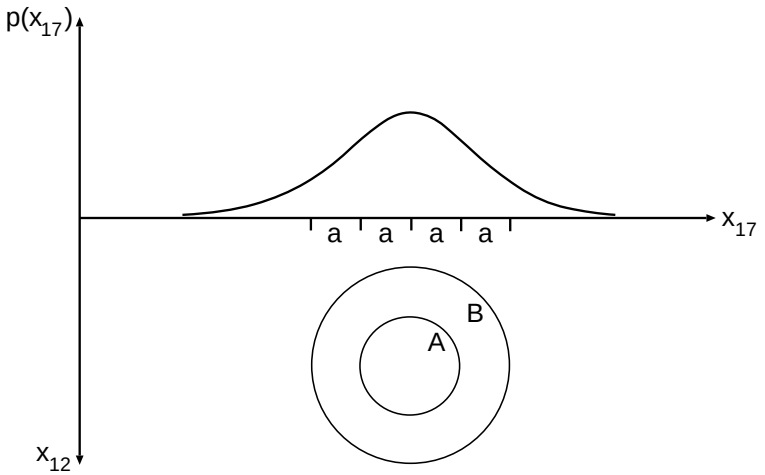
Beispiel: Gauß-Verteilung

Wenn für jede Klasse k die Schätzwerte für den Mittelwertsvektor μ_k und die Kovarianzmatrix Σ_k aus den Trainingsdaten berechnet werden können, dann sind auch die (geschätzten) Klassengrenzen bekannt.

Der Überlapp zwischen den Klassen (oder den entsprechenden Trainingsdaten) ist die Ursache der Erkennungsfehler.

3.2 Verteilung der Abstände im $\mathbb{R}^D$ mit $D \gg 1$

Im Gegensatz zu dem zwei- und dreidimensionalen Fall ist der Raum für $D \gg 1$ (etwa ab $D = 10$) nur sehr dünn mit Trainingsdaten gefüllt.
Sei die Klasse k fest und die Verteilung $p(x|k)$ (z.B. Gauß) für $x \in \mathbb{R}^D$ gegeben.



Verglichen werden sollen die Wahrscheinlichkeiten

$$Pr(x \in A) \text{ und } Pr(x \in B).$$

Dazu berechnen wir die Wahrscheinlichkeit näherungsweise als Produkt aus Volumeninhalt und mittlerer Verteilungsdichte:

$$Pr(x \in A) \;=\; c_D \cdot a^D \cdot \langle p(x|k) \rangle_A$$

mit	c_D	Konstante, abhängig von der Dimension D
	$\langle p(x\|k) \rangle_A$	mittlere Verteilungsdichte in A

Entsprechend ergibt sich für B:

$$\begin{aligned} Pr(x \in B) &= c_D \cdot \left[(2a)^D - a^D\right] \cdot \langle p(x|k)\rangle_B \\ &= c_D \cdot \left[2^D - 1\right] \cdot a^D \cdot \langle p(x|k)\rangle_B \end{aligned}$$

Daraus ergibt sich als Verhältnis der Wahrscheinlichkeiten:

$$\frac{Pr(x \in B)}{Pr(x \in A)} = \left[2^D - 1\right] \cdot \frac{\langle p(x|k)\rangle_B}{\langle p(x|k)\rangle_A}$$

Beispiel: $D = 16$ und $\dfrac{\langle p(x|k)\rangle_B}{\langle p(x|k)\rangle_A} = \dfrac{1}{10}$

$$\frac{Pr(x \in B)}{Pr(x \in A)} = \left[2^{16} - 1\right] \cdot \frac{1}{10} \cong 6400$$

Aus diesem Beispiel ist ersichtlich, daß die Anzahl der Beobachtungen im Bereich A, d.h *im Zentrum der Verteilung*, vernachlässigbar klein im Vergleich zu dem Bereich B, d.h *weit entfernt vom Zentrum der Verteilung*, ist.

Stichwort: *Leere des hochdimensionalen Raumes*

Die exakte Verteilung der Abstände eines Vektors $x \in \mathbb{R}^D$ zu einem Klassenzentrum μ_k der Klasse k

$$\|x - \mu_k\|^2 := \sum_{d=1}^{D} (x_d - \mu_{kd})^2$$

läßt sich für gaußverteilte Vektoren x berechnen. Die Herleitung des Ergebnisses findet man in Statistikbüchern (z.B. [Kre88], p.170-173).

Wir betrachten eine Klasse k mit $p(x|k)$ gaußverteilt und Kovarianzmatrix $\Sigma_k = I$. Der allgemeine Fall mit beliebiger Kovarianzmatrix kann durch Diagonalisierung und Skalierung der Koordinatenachsen auf diesen Spezialfall zurückgeführt werden.

Das Ergebnis ist die χ^2-*Verteilung* ("Chi-Quadrat") für D Freiheitsgrade:

$$\chi^2 = \sum_{d=1}^{D} (x_d - \mu_{kd})^2$$

$$p(\chi^2) = \frac{1}{\Gamma(\frac{D}{2})\, 2^{\frac{D}{2}}} \cdot \left(\chi^2\right)^{\frac{D}{2}-1} \cdot \exp\left(-\frac{1}{2}\chi^2\right)$$

Graphische Darstellung der χ^2-Verteilung (hier $D\hat{=}N$):

Eigenschaften (ohne Beweis):

- Mittelwert: $E\{\chi^2\} = D$
- Varianz: $Var\{\chi^2\} = 2D$
- Grenzübergang $D \to \infty$:

  $p\left(\frac{\chi^2 - D}{\sqrt{2D}}\right) \longrightarrow$ Gauß-Verteilung mit $(\mu_{\chi^2} = 0, \sigma^2_{\chi^2} = 1)$

An der graphischen Darstellung der χ^2-Verteilung für $D = 8, 16, 32, 64$ ist zu erkennen, daß mit wachsender Dimension D die Konzentration um den Mittelwert abnimmt.

3.3 Momentenmethode

Gegeben sind N_k Trainingsvektoren für jede Klasse k:

$$x_{1k}, \ldots, x_{nk}, \ldots, x_{N_k k} \qquad \in \mathbb{R}^D$$

Ansatz: Der Erwartungswert einer Funktion $f(x)$

$$\vartheta_k \;=\; \int\limits_{\mathbb{R}^D} f(x) \cdot p(x|k)\; dx$$

wird ersetzt durch die *empirische Mittelung* der Trainingsdaten:

$$\hat{\vartheta}_k \;=\; \frac{1}{N_k} \sum_{n=1}^{N_k} f(x_{nk})$$

Bei der Momentenmethode werden für $f(x)$ die Momente oder zentrierten Momente gewählt:

- Momente: $f(x) = x^\alpha$ für $x \in \mathbb{R}, \alpha \in \mathbb{N} > 0$
- zentrierte Momente: $f(x) = (x-\mu)^\alpha$ mit $\mu = E\{x\}$

Dann ergibt sich für den Mittelwertsvektor μ_k

$$\mu_k \;=\; \int\limits_{\mathbb{R}^D} x \cdot p(x|k)\; dx$$

der Schätzwert $\hat{\mu}_k$

$$\hat{\mu}_k \;=\; \frac{1}{N_k} \sum_{n=1}^{N_k} x_{nk}$$

und als Schätzwert für die Elemente $\hat{\sigma}_{cd}^{(k)} \quad (c, d = 1, \ldots, D)$
der Kovarianzmatrix $\hat{\Sigma}_k$

$$\hat{\sigma}_{cd}^{(k)} \;=\; \frac{1}{N_k} \sum_{n=1}^{N_k} (x_{nk,c} - \hat{\mu}_{k,c}) \cdot (x_{nk,d} - \hat{\mu}_{k,d}).$$

Für die Varianzen $(\sigma_d^{(k)})^2$ (Diagonalelemente von Σ_k) gilt:

$$(\hat{\sigma}_d^{(k)})^2 \;=\; \frac{1}{N_k}\sum_{n=1}^{N_k}(x_{nk,d}-\hat{\mu}_{k,d})^2.$$

Höhere Momente können nur unzuverlässig geschätzt werden und werden deshalb selten eingesetzt.

3.4 Maximum-Likelihood-Methode

Bei komplexen Modellen versagt die Momentenmethode. Ein allgemeines, weit verbreitetes Verfahren ist die *Maximum-Likelihood-Methode.*
Um die Abhängigkeit des Modells $p(x|k)$ von einem Parameter ϑ auszudrücken, schreibt man

$$p_\vartheta(x|k) \quad \text{oder} \quad p(x|k,\vartheta).$$

Oft ist der Parameter ϑ klassenabhängig, so daß man $p(x|k,\vartheta)$ ersetzt durch

$$p(x|k,\vartheta_k) \quad .$$

Für jede Klasse k sind N_k Trainingsvektoren gegeben:

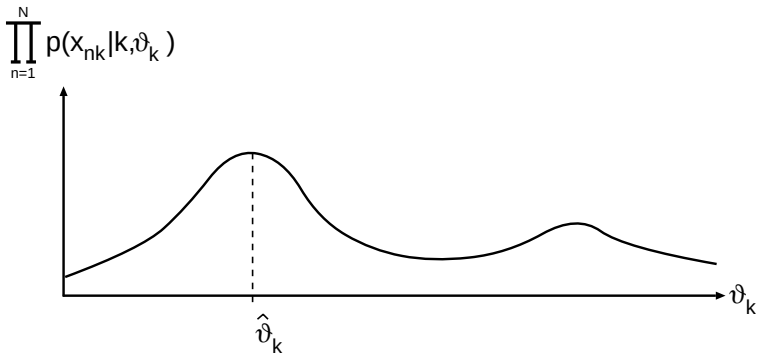
$$x_{1k},\ldots,x_{nk},\ldots,x_{N_kk} \qquad \in \mathbb{R}^D$$

Dann wird die Likelihood-Funktion wie folgt definiert:

$$\vartheta_k \;\longrightarrow\; \prod_{n=1}^{N_k} p(x_{nk}|k,\vartheta_k)$$

oder auch die Log-Likelihood-Funktion:

$$\vartheta_k \;\longrightarrow\; \sum_{n=1}^{N_k} \log p(x_{nk}|k,\vartheta_k).$$



Der Maximum-Likelihood-Schätzwert $\hat{\vartheta}_k$ wird definiert:

$$\hat{\vartheta}_k \;:=\; \underset{\vartheta_k}{\operatorname{argmax}} \left\{ \prod_{n=1}^{N_k} p(x_{nk}|k,\vartheta_k) \right\}.$$

Sprechweisen:
"$\hat{\vartheta}_k$ maximiert die Wahrscheinlichkeit des Modells $p(x|k,\vartheta_k)$ für die vorliegenden Trainingsdaten."
"$\hat{\vartheta}_k$ liefert die *beste* Erklärung der Trainingsdaten für das Modell $p(x|k,\vartheta_k)$."

Oft wird $\hat{\vartheta}_k$ durch Nullsetzen der Ableitung bestimmt:

$$\sum_{n=1}^{N_k} \frac{\partial}{\partial \vartheta_k} \log p(x_{nk}|k,\vartheta_k) \;\stackrel{!}{=}\; 0$$

oder auch

$$\sum_{n=1}^{N_k} \frac{1}{p(x_{nk}|k,\vartheta_k)} \cdot \frac{\partial p(x_{nk}|k,\vartheta_k)}{\partial \vartheta_k} \;\stackrel{!}{=}\; 0$$

Beispiel: Gauß-Verteilung mit diagonaler Kovarianzmatrix

$x \in \mathbb{R}^D$ mit Mittelwertsvektor $\mu_k \in \mathbb{R}^D$
Varianzvektor $\sigma_k^2 \in \mathbb{R}^D$

$$p(x|k,\mu_k,\sigma_k^2) = \frac{1}{\prod_{d=1}^{D}\sqrt{2\pi\sigma_{kd}^2}} \exp\left[-\frac{1}{2}\sum_{d=1}^{D}\left(\frac{x_d-\mu_{kd}}{\sigma_{kd}}\right)^2\right]$$

Trainingsdaten: $x_{1k},\ldots,x_{nk},\ldots,x_{N_k k}$

Log-Likelihood für Klasse k:

$$\sum_{n=1}^{N_k}\log p(x_{nk}|k,\mu_k,\sigma_k^2) = -\frac{1}{2}\sum_{n=1}^{N_k}\left[\sum_{d=1}^{D}\left(\frac{x_{nk,d}-\mu_{kd}}{\sigma_{kd}}\right)^2 + \sum_{d=1}^{D}\log\left(2\pi\sigma_{kd}^2\right)\right]$$

Ableiten nach μ_{kd}:

$$\frac{\partial}{\partial\mu_{kd}} = \sum_{n=1}^{N_k}\left(\frac{x_{nk,d}-\mu_{kd}}{\sigma_{kd}}\right) \stackrel{!}{=} 0$$

woraus folgt

$$\hat{\mu}_{kd} = \frac{1}{N_k}\sum_{n=1}^{N_k} x_{nk,d}$$

Ableiten nach σ_{kd}^2:

$$\frac{\partial}{\partial\sigma_{kd}^2} = \frac{1}{2\sigma_{kd}^4}\sum_{n=1}^{N_k}(x_{nk,d}-\mu_{kd})^2 - \sum_{n=1}^{N_k}\frac{1}{2\sigma_{kd}^2} \stackrel{!}{=} 0$$

nach Einsetzen des Schätzers $\hat{\mu}_{kd}$ für μ_{kd} folgt:

$$\hat{\sigma}_{kd}^2 = \frac{1}{N_k}\sum_{n=1}^{N_k}(x_{nk,d}-\hat{\mu}_{kd})^2$$

Es ergeben sich die üblichen Schätzwerte für Mittelwert und Varianz, diese sind identisch mit den Schätzern der Momentenmethode (oft wird allerdings $\frac{1}{N_k-1}$ anstatt $\frac{1}{N_k}$ für die Varianz verwendet).

Bei einer Gauß-Verteilung mit beliebiger Kovarianzmatrix Σ_k ergeben sich dieselben Schätzwerte wie bei der Momentenmethode:

$$\begin{aligned}\hat{\mu}_{kd} &= \frac{1}{N_k}\sum_{n=1}^{N_k} x_{nk,d} \\ \hat{\Sigma}_{kc,d} &= \frac{1}{N_k}\sum_{n=1}^{N_k}(x_{nk,d}-\hat{\mu}_{kd})(x_{nk,c}-\hat{\mu}_{kc})\end{aligned}$$

Herleitung unter Verwendung der Formel

$$\frac{\partial}{\partial A_{ij}}\, detA \;=\; (A^{-1})_{ij}\cdot detA$$

für eine invertierbare Matrix A.

Wie wir noch sehen werden, kann die Schätzung von Σ_k zu einer singulären Matrix $\hat{\Sigma}_k$ führen. Als Abhilfe kann man eine klassenunabhängige Kovarianzmatrix Σ annehmen.

$$p(x|k,\mu_k,\Sigma) \;:\quad \text{Gauß-Verteilung mit } \mu_k \text{ und } \Sigma$$

Maximum-Likelihood-Schätzung ergibt:

$$\begin{aligned}\hat{\mu}_{kd} &= \frac{1}{N_k}\sum_{n=1}^{N_k} x_{nk,d} \\ \hat{\Sigma}_{cd} &= \frac{1}{N}\sum_{k=1}^{K}\left[\sum_{n=1}^{N_k}(x_{nk,d}-\hat{\mu}_{kd})(x_{nk,c}-\hat{\mu}_{kc})\right] \\ &\quad \text{mit } N=\sum_{k=1}^{K} N_k\end{aligned}$$

Hinweise:

1. Jede Beobachtung x_{nk} geht genau einmal ein und zwar nach Subtraktion des zugehörigen Mittelwertes.
2. Interpretation: $\hat{\Sigma}$ ergibt sich als gewichtete Mittelung von klassenindividuellen Kovarianzmatrizen $\hat{\Sigma}_k$.

Andere Verteilungen, für die die Maximum-Likelihood-Methode zu einfachen Schätzwerten führt:

- Laplace-Verteilung,
- Binomial-Verteilung,
- Multinomial-Verteilung,
- Poisson-Verteilung.

Übungsaufgabe: Berechne die entsprechenden Schätzwerte.

3.5 Praktische Aspekte

1. Typische Probleme

   Am Beispiel eines Gauß-Modells sollen die typischen Probleme aufgezeigt werden:

   - Die Variabilität der Lernstichprobe ist zu gering, so daß die Varianz zu klein geschäzt wird.
   - Diese Unterschätzung der Varianz und der exponentielle Abfall der Gauß-Modelle führen zu einer zu starken Konzentration der Wahrscheinlichkeitsmasse im Zentrum.
   - Die Kovarianzmatrix $\hat{\Sigma}_k$ kann leicht singulär werden:

     Bei N_k Beobachtungsvektoren ist $\hat{\Sigma}_k$ aus N_k Vektoren $(x_{nk} - \hat{\mu}_k)$ aufgebaut, von denen wegen $\hat{\mu}_k$ nur $(N_k - 1)$ unabhängig sein können. D.h., $\hat{\Sigma}_k$ hat den Rang $(N_k - 1)$ und ist garantiert singulär falls $N_k \leq D$.

     In der Praxis fordert man:

     $$N_k \geq (10, \ldots, 100) \cdot D$$

     Diese Singularität von $\hat{\Sigma}_k$ ist der wichtigste Grund, warum oft nur mit diagonalen Kovarianzmatrizen gearbeitet wird.

2. Gegenmaßnahmen gegen singuläre $\hat{\Sigma}_k$

   - klassenunabhängige Kovarianzmatrix $\hat{\Sigma}$:

     In diesem Fall kann eine Koordinatentransformation zweckmäßig sein.

- Glätten von $\hat{\Sigma}_k$ mit $\hat{\Sigma}$:

$$\tilde{\Sigma}_k := (1 - \lambda_k)\hat{\Sigma}_k + \lambda_k \hat{\Sigma} \ ,$$

  wobei λ_k ein klassenabhängiger Wichtungsparameter ist.

- Betonen der Diagonalelemente von $\hat{\Sigma}_k$:

$$\tilde{\Sigma}_k := (1 - \lambda_k)\hat{\Sigma}_k + \lambda_k \cdot diag(\hat{\Sigma}_k) \ ,$$

  mit der Annahme, daß alle Diagonalelemente größer Null sind.

3. Aufbau eines Systems (mit Gauß-Modellen)

   - Berechnung der Mittelwerte $\hat{\mu}_k$ und Kovarianzmatrizen $\hat{\Sigma}_k$ für jede Klasse k.
   - Prüfen, ob die Kovarianzmatrix $\hat{\Sigma}_k$ singulär ist oder dazu tendiert (anhand der Eigenwerte).
   - Glätten der Kovarianzmatrix $\hat{\Sigma}_k$ oder Berechnen einer klassenunabhängigen Kovarianzmatrix $\hat{\Sigma}$ (u.U. mit anschließender Koordinatentransformation).
   - gegebenenfalls:
     Betonen der Diagonalelemente der Kovarianzmarizen.
   - noch offen:
     Wahl der Wichtungsparameter λ_k
     letztlich: Analysiere die Auswirkung auf die Fehlerrate (entweder auf den Trainingsdaten oder auf einer weiteren Stichprobe).
   - mögliche Verfeinerung (oder Testen auf Unimodalität):
     Nearest-Neighbour-Klassifikator:
     Behalte alle Trainingsdaten bei und bestimme den nächsten Trainingsvektor für einen zu klassifizierenden Testvektor x. Dies erfordert ein Abstandsmaß, in das die Kovarianzmatrix oder deren Diagonalelemente eingehen sollten.

3.6 Bewertungskriterium: empirische Fehlerrate

Das wichtigste Bewertungskriterium ist die Fehlerrate, mit der das System im praktischen Betrieb arbeitet. Man unterscheidet 2 Arten von Stichproben:

- die Trainings- oder Lernstichprobe, anhand derer der Klassifikator entworfen wird, d.h., die Wahl des Modells $p(x|k,\vartheta_k)$ und die Schätzung der Parameter ϑ_k erfolgt;
- die Teststichprobe, an der die Zuverlässigkeit des Klassifikators ermittelt wird, und zwar anhand der

$$\text{(empirischen) Fehlerrate} = \frac{\text{Anzahl der Erkennungsfehler}}{\text{Anzahl der Erkennungstests}}$$

Eine theoretische Berechnung der Fehlerrate ohne Rückgriff auf eine Teststichprobe ist schwierig, da im allgemeinen die Form der Modelle $p(x|k,\vartheta_k)$ nicht bekannt ist.

Hinweise:

- Die strenge Trennung von Trainings- und Teststichprobe ist für eine objektive Ermittlung der Fehlerrate absolut notwendig.

  Beispiel: Der Nearest-Neighbour-Klassifikator hat auf der Trainingsstichprobe immer null Fehler.

- Wiederholte Tests auf denselben Testdaten können u.U. irreführend sein, weil das System nach und nach auf die speziellen Testdaten optimiert wird. Dieser Effekt heißt

  training on the testing data.

- Im Sinne einer wissenschaftlichen Objektivität muß sich die Fehlerrate auf eine bisher ungesehene Stichprobe beziehen, da die Fehlerrate eine Prädiktion für neue Tests ermöglichen soll. In der Statistik werden ähnliche Verfahren als *Cross-Validation* bezeichnet.

3.7 Bayessches Lernverfahren

Ausgangspunkt dieses Verfahrens sind die beiden Annahmen:

1. ϑ_k in $p(x|k, \vartheta_k)$ wird selbst als Zufallsvariable behandelt.
2. Es gibt eine a-priori-Verteilung $p(\vartheta_k)$ für den Parameter bzw. die Zufallsvariable ϑ_k.

   a-priori $\hat{=}$ vor den eigentlichen Messungen

Diese a-priori-Verteilung $p(\vartheta_k)$ beruht auf

- früheren Messungen oder Experimenten,
- oder auf Vorwissen über die Aufgabe, das natürlich auch aus früheren Experimenten gewonnen worden ist.

Beispiel:
Viele Systeme zur Schrift- oder Spracherkennung arbeiten benutzerabhängig, d.h., jeder Benutzer muß das System persönlich trainieren.
Ein derartiges System kann mit einer Stichprobe von vielen Sprechern vor der eigentlichen Benutzung trainiert werden, so daß man daraus die Verteilung $p(\vartheta_k)$ der personenabhängigen Parameter ϑ_k ermitteln kann.

Vorteil: Ein derartiges System kann schon mit einer geringen Menge von personenabhängigem Trainingsmaterial in Betrieb gehen.

Die Bayessche Parameterschätzung selbst erfolgt anhand der a-posteriori-Verteilung $p(\vartheta_k|x_{1k}, \ldots, x_{N_k k})$ für die Trainingsdaten x_{nk} mit $n = 1, \ldots, N_k$ der Klasse k.

Verteilung der Verbundereignisse $(\vartheta_k; x_{1k}, \ldots, x_{N_k k})$:

$$\begin{aligned} p(\vartheta_k; x_{1k}, \ldots, x_{N_k k}) &= p(\vartheta_k) \cdot p(x_{1k}, \ldots, x_{N_k k}|\vartheta_k) \\ &= p(\vartheta_k) \cdot \prod_{n=1}^{N_k} p(x_{nk}|\vartheta_k) \\ &\quad \text{mit dem Modell} \quad p(x_{nk}|\vartheta_k) \end{aligned}$$

Definition und Berechnung der a-posteriori Verteilung:

$$\begin{aligned} p(\vartheta_k|x_{1k},\ldots,x_{N_k k}) &= \frac{p(\vartheta_k)\prod_{n=1}^{N_k} p(x_{nk}|\vartheta_k)}{\int p(\vartheta'_k)\prod_{n=1}^{N_k} p(x_{nk}|\vartheta'_k)d\vartheta'_k} \\ &= \frac{p(\vartheta_k)\prod_{n=1}^{N_k} p(x_{nk}|\vartheta_k)}{const(\vartheta_k)} \end{aligned}$$

Nur der Zähler ist direkt von ϑ_k abhängig, und zwar ist er das Produkt aus a-priori-Verteilung und Likelihood.
Oft wird die a-posteriori-Verteilung zu einem einzigen Wert komprimiert:

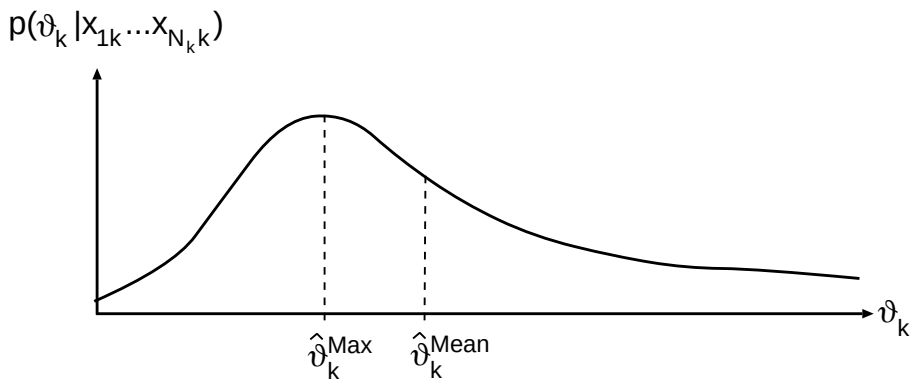
- Posterior Mean:

$$\hat{\vartheta}_k^{Mean} = \int \vartheta_k \; p(\vartheta_k|x_{1k},\ldots,x_{N_k k})d\vartheta_k$$

- Posterior Maximum:

$$\hat{\vartheta}_k^{Max} = \underset{\vartheta_k}{\text{argmax}} \; \{p(\vartheta_k|x_{1k},\ldots,x_{N_k k})\}$$

Veranschaulichung:



Speziell: Falls $p(\vartheta_k) = const(\vartheta_k)$, dann gilt

$$\hat{\vartheta}_k^{Max} \;=\; \hat{\vartheta}_k^{Max.Lik.}.$$

Beispiel: univariate Gauß-Verteilung
Zur Vereinfachung lassen wir den Klassenindex k weg.
Annahmen:

- Gauß-Verteilung: $p(x|\mu,\sigma^2), \quad x \in \mathbb{R}$
  mit μ unbekannt und σ^2 bekannt.
- Die a-priori-Verteilung $p(\mu)$ für den Parameter μ ist selbst eine Gauß-Verteilung:
  $p(\mu|\mu_0,\sigma_0^2)$ mit *Hyperparametern* μ_0 und σ_0^2.

Wir berechnen die a-posteriori-Verteilung für die Trainingsdaten $x_1,\ldots,x_N$:

$$\begin{aligned}
p(\mu|x_1,\ldots,x_N) &= const(\mu)\cdot p(\mu|\mu_0,\sigma_0^2)\cdot\prod_{n=1}^{N} p(x_n|\mu,\sigma^2)\\
&= const'(\mu)\cdot\exp\left[-\frac{1}{2}\left(\frac{\mu-\mu_0}{\sigma_0}\right)^2-\frac{1}{2}\sum_{n=1}^{N}\left(\frac{x_n-\mu}{\sigma}\right)^2\right] \quad (*)\\
&= const'(\mu)\cdot\\
&\quad \exp\left[-\frac{1}{2}\left\{\frac{\mu}{\sigma_0^2}-\frac{2\mu\mu_0}{\sigma_0^2}+\frac{\mu_0^2}{\sigma_0^2}+\frac{N\mu^2}{\sigma^2}-\frac{2\mu\sum_{n=1}^{N}x_n}{\sigma^2}+\frac{\sum_{n=1}^{N}x_n^2}{\sigma^2}\right\}\right]
\end{aligned}$$

An der Gleichung $(*)$ ist zu erkennen, daß die Hyperparameter (μ_0,σ_0^2) ähnlich wie die Paare (x_n,σ^2) aus Messungen x_n und Varianz σ^2 die a-posteriori-Verteilung bestimmen.

Umformen des Exponenten zu einem quadratischen Ausdruck in μ ergibt:

$$p(\mu|x_1,\ldots,x_N) \;=\; \frac{1}{\sqrt{2\pi\sigma_N^2}}\exp\left[-\frac{1}{2}\left(\frac{\mu-\mu_N}{\sigma_N}\right)^2\right]$$

mit geeigneten Ausdrücken

$$\begin{aligned} \sigma_N^2 &\equiv \sigma_N^2(\mu_0, \sigma_0^2; x_1, \ldots, x_N; N), \\ \mu_N &\equiv \mu_N(\mu_0; \sigma_0^2, x_1, \ldots, x_N; N). \end{aligned}$$

Koeffizientenvergleich für μ^2:

$$\frac{1}{\sigma_N^2} = \frac{1}{\sigma_0^2} + \frac{N}{\sigma^2}$$

Koeffizientenvergleich für μ:

$$\frac{\mu_N}{\sigma_N^2} = \frac{\mu_0}{\sigma_0^2} + \frac{1}{\sigma^2} \sum_n x_n$$

Umformen und Einsetzen von σ_N^2 ergibt:

$$\mu_N = \left(1 - \frac{\sigma^2}{\sigma^2 + N\sigma_0^2}\right) \frac{1}{N} \sum_{n=1}^{N} x_n + \frac{\sigma^2}{\sigma^2 + N\sigma_0^2} \cdot \mu_0$$

Dieser Ausdruck beschreibt eine Mittelung zwischen

- empirischem Mittelwert $\frac{1}{N}\sum_{n=1}^{N} x_n$ mit Gewicht $\left(1 - \frac{\sigma^2}{\sigma^2 + N\sigma_0^2}\right)$
- a-priori Mittelwert μ_0 mit Gewicht $\frac{\sigma^2}{\sigma^2 + N\sigma_0^2}$.

Für $N \longrightarrow \infty$ geht der Einfluß der a-priori-Verteilung $p(\mu|\mu_0, \sigma_0^2)$ gegen null, und es ergibt sich für die a-posteriori-Verteilung $p(\mu|\mu_N, \sigma_N^2)$ die Delta-Funktion:

$$\begin{array}{lll} & p(\mu|\mu_N, \sigma_N^2) & \\ & \Big| & \mu_N \to \frac{1}{N}\sum_{n=1}^{N} x_n \\ N \longrightarrow \infty & \Big| & \\ & \downarrow & \sigma_N^2 \to 0 \\ & \text{Delta-Funktion} & \end{array}$$

Graphische Darstellung: Mit wachsendem N (im Bild: n statt N) wird die a-posteriori-Verteilung schärfer konzentriert um μ_N.

Der allgemeine Fall ist schwieriger:

- multivariate (statt univariate) Gauß-Verteilung;
- μ und Σ sind gleichzeitig unbekannt;
- i.a. ist die Kovarianzmatrix nicht diagonal.

Einzelheiten findet man in :

- Fukunaga, 1972, p.219
- [Kee65]
- Vorlesungsskript Mustererkennung WS 1993/94

In diesen und ähnlichen Fällen ergibt sich typischerweise (exakt oder als praktische Näherung), daß der neue Schätzwert sich durch *lineare Interpolation* ergibt aus:

- einem Schätzwert, der auf den echten Messungen beruht,
- und einem geeigneten Parameter der a-priori-Verteilung.

Wie wird das Bayessche Lernverfahren in der Erkennung eingesetzt?
Wir nennen zwei Methoden:

1. Bisher (stillschweigend) vorausgesetzt wurde die *plug-in-Methode*:

   verwende $p(x|k,\hat{\vartheta}_k)$ mit $\hat{\vartheta}_k = \hat{\vartheta}_k(x_{1k},\ldots,x_{N_k k})$,

   d.h. einem Schätzwert $\hat{\vartheta}_k$, der mittels Bayes-Schätzung aus den Trainingsdaten $x_{1k},\ldots,x_{N_k k}$ der Klasse k gewonnen wurde.

2. Eine andere Methode (allerdings selten eingesetzt) beruht auf der *prädiktiven Verteilung* $p(x|k;x_{1k},\ldots,x_{N_k k})$:

$$p(x|k;x_{1k},\ldots,x_{N_k k}) := \int_{\vartheta_k} p(x|k,\vartheta_k)\cdot p(\vartheta_k|k;x_{1k},\ldots,x_{N_k k})\, d\vartheta_k,$$

   wobei $p(\vartheta_k|k;x_{1k},\ldots,x_{N_k k})$ die a-posteriori-Verteilung der Parameter ϑ_k ist, jetzt mit zusätzlichem Klassenindex k.

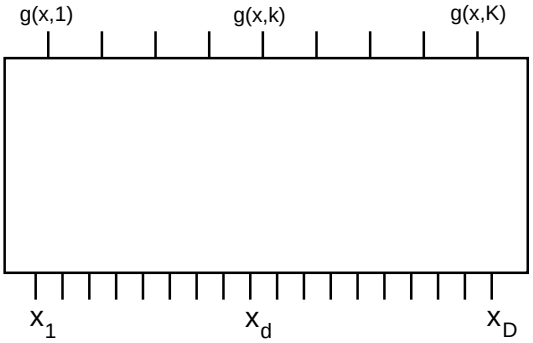
   In der Praxis sind für die auftretenden Integrale nur in Ausnahmefällen geschlossene Lösungen bekannt. Näheres findet man an den schon genannten Stellen: [DH73], p.55-59 und [Kee65].

4 Diskriminanten und Neuronale Netze

4.1 Fehlerquadratkriterium

Die in Kapitel 3 behandelten Lernverfahren wurden für Verteilungsmodelle $p(x|k, \vartheta_k)$ entwickelt. Bei allgemeinen Diskriminanten $g(x, k)$ liegt zunächst kein statistisches Modell zugrunde.

Zur Erinnerung: Diskriminantenansatz



Entscheidungsregel: $r(x) = \underset{k}{\operatorname{argmax}} \{g(x, k)\}$

Die Funktion $g(x, k)$ hängt von Parametern ϑ ab:

$$g(x, k) = g_\vartheta(x, k).$$

Diese Parameter werden in der Lernphase ermittelt.

Dazu gibt man in der Lernphase die idealen Ausgangswerte vor:

$$\begin{aligned} g(x, k) &\stackrel{!}{=} 1, && \text{falls } x \text{ zur Klasse } k \text{ gehört,} \\ g(x, k) &\stackrel{!}{=} 0, && \text{falls } x \text{ nicht zur Klasse } k \text{ gehört.} \end{aligned}$$

Diese idealen Werte sind in der Regel nur approximativ erreichbar. In der Lernphase verwendet man (in der Regel) das Fehlerquadratkriterium für einen Trainingsvektor x_n mit Klassenzugehörigkeit k_n:

$$\sum_{c=1}^{K} [g(x_n, c) - t(k_n, c)]^2$$

mit der Kronecker-Funktion (t steht für *target*):

$$t(k,c) = \begin{cases} 1\,, & c = k \quad \text{(richtig)} \\ 0\,, & c \neq k \quad \text{(falsch)} \end{cases} \quad .$$

Statt eines einzelnen Trainingsvektors liegt ein ganzer Satz vor:

$$\text{Trainingsvektoren:} \qquad (x_1, k_1), \ldots, (x_n, k_n), \ldots, (x_N, k_N)$$

(Beachte die im Vergleich zu Kapitel 3 geänderte Notation.)

Wir summieren die Fehler über alle Trainingsvektoren und definieren:

$$F_N(g) \;:=\; \frac{1}{N} \sum_{n=1}^{N} \sum_{c=1}^{K} \left[g(x_n, c) - t(k_n, c) \right]^2$$

Die Schreibweise $F_N(g)$ soll ausdrücken:

- die Abhängigkeit von der Funktion $(x,k) \longrightarrow g(x,k)$, d.h. deren funktionaler Form und deren Parameter;
- die Abhängigkeit von den Trainingsdaten (x_n, k_n) mit $n = 1, \ldots, N$.

Im Grenzfall $N \longrightarrow \infty$ ergibt sich:

$$\begin{aligned} F(g) &= \lim_{N \longrightarrow \infty} \frac{1}{N} \sum_{n=1}^{N} \sum_{c=1}^{K} \left[g(x_n, c) - t(k_n, c) \right]^2 \\ &= \int_x \sum_{k=1}^{K} p(x,k) \cdot \sum_{c=1}^{K} \left[g(x,c) - t(k,c) \right]^2 \, dx \\ &= \int_x p(x) \cdot \underbrace{\sum_{k=1}^{K} p(k|x) \cdot \sum_{c=1}^{K} \left[g(x,c) - t(k,c) \right]^2}_{=:\, e(x)} \, dx \end{aligned}$$

$e(x)$ ist der lokale Fehler an der Stelle x.

Umformen des lokalen Fehlers $e(x)$:

$$\begin{aligned} e(x) &= \sum_{k=1}^{K} p(k|x) \cdot \sum_{c=1}^{K} \left[g(x,c) - t(k,c)\right]^2 \\ &= \sum_{k=1}^{K} p(k|x) \cdot \left(\left[g(x,k) - 1\right]^2 + \sum_{c \neq k} g^2(x,c) \right) \\ &= \sum_{k=1}^{K} p(k|x) \cdot \left(1 - 2\ g(x,k) + \sum_{c=1}^{K} g^2(x,c) \right) \\ &= 1 - 2 \sum_{c=1}^{K} p(c|x) \cdot g(x,c) + \sum_{c=1}^{K} g^2(x,c) \\ &= 1 - \sum_{c=1}^{K} p^2(c|x) + \sum_{c=1}^{K} \left[p(c|x) - g(x,c)\right]^2 \end{aligned}$$

Nach dieser Gleichung besteht $e(x)$ aus zwei Teilen:

- $1 - \sum_{c=1}^{K} p^2(c|x)$: unabhängig von der Diskriminantenfunktion $g(x,k)$ (tritt auch sonst auf; im Kontext von Gini-Kriterium, Nearest-Neighbour-Fehlerrate).
- $\sum_{c=1}^{K} \left[p(c|x) - g(x,c)\right]^2$: Fehlerquadratkriterium zwischen $g(x,c)$ und der a-posteriori-Klassenverteilung $p(c|x)$ an der Stelle $x \in \mathbb{R}^D$.

Der Ausdruck $F(g)$ lautet also:

$$F(g) = \int_x p(x) \cdot \left(1 - \sum_{c=1}^{K} p^2(c|x) + \sum_{c=1}^{K} \left[p(c|x) - g(x,c)\right]^2 \right) dx$$

Anmerkungen zu $F(g)$:

- Globale Optimalität wird erreicht, falls $g(x,c) = p(c|x) \quad \forall (x,c)$, d.h. falls die Werte der Diskriminantenfunktion *exakt* die a-posteriori-Klassenwahrscheinlichkeiten sind.

  Im globalen Optimum g_0 gilt:

$$F(g_0) = \int_x p(x) \cdot \left(1 - \sum_{c=1}^{K} p^2(c|x)\right) \, dx \ ,$$

  d.h. dieser Wert (als Fehlerkriterium) kann nie unterschritten werden.

- Um das globale Optimum (approximativ) zu erreichen, sollte die Diskriminantenfunktion in Hinblick auf

  - funktionale Form (*Struktur*) und
  - Anzahl der freien Parameter ϑ

  geeignet gewählt sein.

- Falls die Gleichheit $g(x,c) = p(c|x)$ für alle (x,c) nicht erreichbar ist, wird der Fehler vorzugsweise an Stellen x minimiert, an denen $p(x)$ groß ist.

- Das Finden des (globalen) Optimums ist ein schwieriges mathematisches Problem, für das iterative Verfahren benutzt werden:

  - Gradientenverfahren (numerische Mathematik)
  - Error Back Propagation (*stochastischer Gradient*)

4.2 Strukturen und Multilayer-Perceptron

Einfache Strukturen für die Diskriminantenfunktion $(x, k) \longrightarrow g(x, k)$ sind (vgl. Kapitel 1):

$$\text{linear} \quad : \quad g(x,k) = \alpha_{k0} + \sum_{d=1}^{D} \alpha_{kd} \cdot x_d$$

$$\text{quadratisch} : \quad g(x,k) = \alpha_{k0} + \sum_{d=1}^{D} \alpha_{kd} \cdot x_d + \sum_{d=1}^{D} \sum_{e=1}^{D} \beta_{kde} \cdot x_d x_e$$

mit den Parametern α_{kd} und β_{kde}.

Diese Strukturen können entsprechend für höhere Polynome in x definiert werden, und man erhält die *Polynom- oder Regressionsklassifikatoren* (*Regression* wegen des Fehlerquadratkriteriums).

Vorteil: Es gibt einen effizienten mathematischen Formalismus zur Bestimmung der Parameter.

Nachteil: Polynome haben keine guten Interpolationseigenschaften (bekannt aus der Approximation von Funktionen). Sie sind nicht genügend glatt und versagen daher leicht bei neuen, im Training nicht gesehenen Daten.

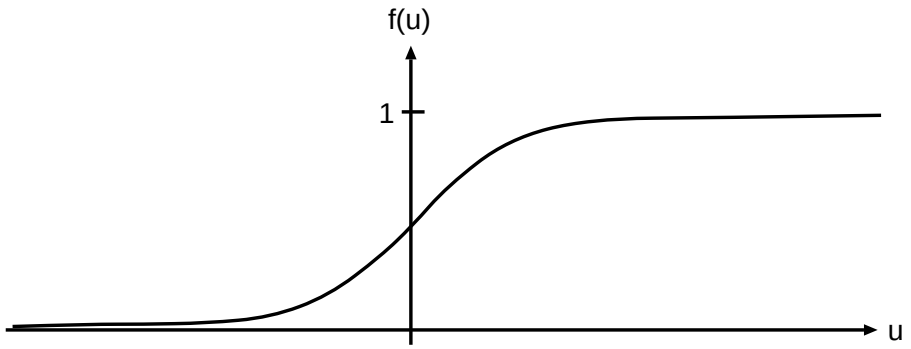
Die Nachteile werden (weitgehend) vermieden durch das *Multilayer-Perceptron.*

Struktur des Multilayer-Perceptrons:

- Es gibt mehrere Schichten von Knoten (*multilayer*):
  - Ausgangsschicht (Ausgänge $g(x,k), \quad k = 1, \ldots, K$);
  - verborgene Schichten;
  - Eingangsschicht (Eingänge $x_d, \quad d = 1, \ldots, D$).
- In jedem Knoten (*Neuron*) einer Schicht werden die entsprechenden Eingänge y_l linear kombiniert (*Skalarprodukt*):

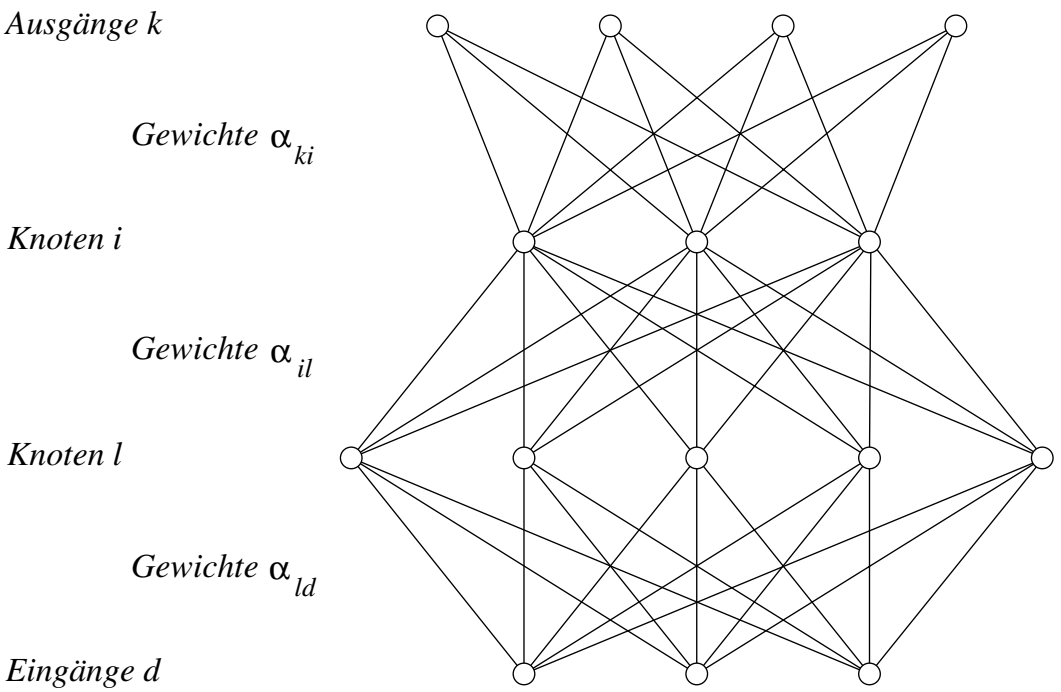
$$\sum_l \alpha_{il} \cdot y_l \qquad \text{für Knoten der Schicht } i$$

Jede derartige Linearkombination durchläuft eine nichtlineare Funktion $u \longrightarrow f(u)$:



Eine lineare Diskriminante $x \longrightarrow g(x,k)$ wird auch (lineares) Perceptron genannt.

Darstellung eines Multilayer-Perceptrons (Beispiel mit 2 verborgenen Schichten):



$$\begin{aligned}
\text{Ausgänge } k &: y_k = f\left(\sum_i \alpha_{ki} \cdot y_i\right) \\
\text{Knoten } i &: y_i = f\left(\sum_l \alpha_{il} \cdot y_l\right) \\
\text{Knoten } l &: y_l = f\left(\sum_d \alpha_{ld} \cdot y_d\right) \\
\text{Eingänge } d &: y_d = x_d
\end{aligned}$$

Also realisiert das Multilayer-Perceptron eine Diskriminantenfunktion:

$$\begin{aligned}
g(\cdot, k): \; & \mathbb{R}^D \longrightarrow \mathbb{R} \\
& x \longmapsto g(x,k) = y_k(x)
\end{aligned}$$

Um auch eine Konstante (*Bias, Offset*) in die Linearkombination einzubeziehen, fügt man in jeder Schicht einen fiktiven, konstanten Knoteneingangswert $y_0 \equiv 1$ hinzu, z.B.:

$$\begin{aligned}
\sum_l \alpha_{il} \cdot y_l &= \sum_{l=0}^{L} \alpha_{il} \cdot y_l \\
&= \alpha_{i0} + \sum_{l=1}^{L} \alpha_{il} \cdot y_l.
\end{aligned}$$

4.2.1 Nichtlinearität

Typischerweise wählt man für f die *Sigmoid-Funktion*:

$$f(u) = \frac{1}{1+e^{-u}} \quad .$$

Für die Ableitung berechnet man:

$$f'(u) = f(u) \cdot (1 - f(u)).$$

Diese Funktion $f(u)$ ist in der Wahrscheinlichkeitsrechnung/Statistik als *logistische* Verteilung bekannt.

Alternativ zur Sigmoid-Funktion kann auch der Arcustangens verwendet werden (quadratischer Abfall):

$$\begin{aligned} f'(u) &= \frac{1}{\pi} \cdot \frac{1}{1+u^2} \\ f(u) &= \frac{1}{\pi} \cdot \arctan(u) + \frac{1}{2} \,. \end{aligned}$$

Andere Varianten sind denkbar, werden allerdings selten benutzt:

- Gauß-Modell:

$$\begin{aligned} f'(u) &= \frac{1}{\sqrt{2\pi}} \cdot \exp\left(-\frac{u^2}{2}\right) \\ f(u) &= \int_{-\infty}^{u} f'(z)\, dz \,. \end{aligned}$$

- *elementare* Funktionen:

$$\begin{aligned} f(u) &= \frac{u}{1+|u|} \\ f'(u) &= \frac{1}{(1+|u|)^2} \qquad \text{für } u \neq 0 \,. \end{aligned}$$

Beachte: Wegen des Offsets $y_0 \equiv 1$ können die Ausgangswerte auch von $[0,1]$ nach $[-1,1]$ umdefiniert werden.

Bei der Sigmoid-Funktion läßt sich die Nichtlinearität leicht veranschaulichen. Die folgende Reihenentwicklung ist bekannt:

$$\frac{1}{1+z} = 1 - z + z^2 - z^3 + \ldots \qquad |z| < 1 \,.$$

Anwendung auf die Sigmoid-Funktion (Entwicklung um den Punkt $u_0 = 0$):

$$\begin{aligned} f(u) &= \frac{1}{1+e^{-u}} \\ &= \frac{1}{2} \cdot \frac{1}{1+\frac{1}{2}(e^{-u}-1)} \\ &\tilde{=} \frac{1}{2}\left[1 - \left(\frac{e^{-u}-1}{2}\right) + \left(\frac{e^{-u}-1}{2}\right)^2 - \left(\frac{e^{-u}-1}{2}\right)^3 + \dots\right] \end{aligned}$$

Interpretation: Die Sigmoid-Funktion erzeugt die nichtlinearen Glieder in einer *bestimmten* Art.

4.2.2 Bemerkungen zum Multilayer-Perceptron (MLP)

- Das Multilayer-Perceptron ist der Prototyp eines *künstlichen* neuronalen Netzes. Das Attribut *künstlich* wird zur Unterscheidung von biologischen neuronalen Netzen verwendet.
- Die Vorzüge der (künstlichen) neuronalen Netze werden oft wie folgt zusammengefaßt:
  - Orientierung am biologischen Vorbild,
  - massiver Parallelismus ("pdp" $\hat{=}$ parallel distributed processing),
  - diskriminatives Training,
  - Approximation beliebiger Input-Output-Zusammenhänge.
- Dieselben Vorzüge gelten auch für viele nichtneuronale Ansätze. Allerdings wurde durch die neuronalen Netze die Idee des diskriminativen Trainings wiederbelebt, insbesondere in der Spracherkennung.

Wahl der Struktur des Multilayer-Perceptrons:
Für die Wahl der

- Anzahl der verborgenen Schichten
- Anzahl der Knoten pro Schicht

gibt es im wesentlichen nur empirische Aussagen.

Fig.

Darüber hinaus ist es auch zweckmäßig, Symmetriebeziehungen zwischen den Gewichten vorzuschreiben, falls sich diese aus der Problemstellung ergeben.

Beispiel: Multilayer-Perceptron für die Schrifterkennung nach Yan le Cun.

Fig

Aufbau des Multilayer-Perceptrons:

Eingangsschicht: 16×16 Pixel (Grauwerte)

Schicht H1: Jeder Knoten sieht ein Teilbild aus 5×5 Pixeln. Mit Verschiebung von 2 Pixeln ergeben sich 8×8 Zellen. Es werden 12 Ebenen mit je 8×8 Zellen angenommen.
Parameter pro Ebene:
25 Gewichte (unabhängig von der Zelle)
64 Biases (Offsets)
insgesamt: $12 \times (25 + 64) = 1068$ freie Parameter

Schicht H2: besteht ebenfalls aus 12 Ebenen, hier mit je 4×4 Zellen. Jede H2-Zelle sieht 8 der 12 H1-Ebenen, und zwar 5×5 Zellen in den 8 Ebenen.
Parameter pro Ebene:
$8 \cdot 5 \cdot 5 = 200$ Gewichte (unabhängig von der Zelle)
$4 \cdot 4 = 16$ Biases (Offsets)
insgesamt: $12 \times (200 + 16) = 2592$ freie Parameter

Idee:
H2 kann aus den Merkmalen von H1 komplexere Merkmale aufbauen.

Schicht H3: besteht aus 30 Zellen, von denen jede alle $12 \cdot 16 = 192$ Zellen von H2 sieht.

Ausgangsschicht: besteht aus 10 Zellen, von denen jede alle 30 Zellen von H3 sieht.

verborgene Knoten:

H1:	768	Knoten
H2:	192	Knoten
H3:	30	Knoten
insgesamt:	990	Knoten

Gewichte (*links*):

H1:	$768 \cdot 25 =$	19200
H2:	$192 \cdot 200 =$	38400
H3:	$30 \cdot 192 =$	5760
Ausgangsschicht:	$10 \cdot 30 =$	300
insgesamt:		63660

unabhängige Parameter:

H1:	1068
H2:	2592
H3:	5790
Ausgangsschicht:	310
insgesamt:	9760

Dieses Beispiel zeigt, wie Strukturen in das Multilayer-Perceptron eingebaut werden können.

4.2.3 Multilayer-Perceptron-ähnliche Strukturen für statistische Klassifikatoren

Unabhängig von neuronalen Netzen gibt es Ansätze, die für die a-posteriori-Wahrscheinlichkeit $p(k|x)$ zu ähnlichen Strukturen führen.
Aus Kapitel 2 ist bekannt:

$$\begin{aligned} p(k|x) &= \frac{p(k) \cdot p(x|k)}{\sum_{c=1}^{K} p(c) \cdot p(x|c)} \\ &= \frac{1}{1 + \sum_{c \neq k} \frac{p(c) \cdot p(x|c)}{p(k) \cdot p(x|k)}} \\ &= \frac{1}{1 + \sum_{c \neq k} \frac{p(c)}{p(k)} \cdot \exp\left[\log p(x|c) - \log p(x|k)\right]} \end{aligned}$$

Wenn die Differenz $[\log p(x|c) - \log p(x|k)]$ linear in x ist, ergeben sich Operationen, die ähnlich sind zu denen innerhalb eines Multilayer-Perceptron-Knotens. Dies läßt sich erreichen durch

1. log-lineare-Modelle: $\log p(x|k) = \sum_d \alpha_{kd} \cdot x_d$

   (Beachte: x ist meistens diskret, da sonst in der Regel Normierungsprobleme für $\int_{\mathbb{R}^D} p(x|k)\, dx = 1$ bestehen.)

2. Gauß-Modelle mit $\Sigma_k = \Sigma$ (klassenunabhängig)

Übungsaufgabe: Gib für 1. und 2. die jeweilige neuronale Struktur an.

4.3 Error Back Propagation

Error Back Propagation ist das am meisten verbreitete Verfahren zum Training eines Multilayer-Perceptrons und wurde von Rumelhart, Hinton und Williams in den 80er Jahren propagiert. Es gab allerdings schon frühere Arbeiten: Amiri 1966, Werbos 1974, Parker 1982.

Kurz zusammengefaßt gilt:

Error Back Propagation = stochastischer Gradient und Kettenregel

Ausgangspunkt ist das Fehlerquadratkriterium für die Trainingsdaten (x_n, c_n) mit $n = 1, \ldots, N$:

$$\begin{aligned} E &= \sum_{n=1}^{N} E_n \\ E_n &= \frac{1}{2} \sum_{k=1}^{K} [y_k(x_n) - t(k, c_n)]^2 \end{aligned}$$

Fallunterscheidung:

a) Ausgangsschicht: $y_k = f(z_k)$ mit $z_k = \sum_i \alpha_{ki} \cdot y_i = z_k(x_n)$

$$\begin{aligned} \frac{\partial E_n}{\partial \alpha_{ki}} &= \frac{\partial E_n}{\partial y_k} \cdot \frac{\partial y_k}{\partial \alpha_{ki}} \\ &= \frac{\partial E_n}{\partial y_k} \cdot \frac{\partial y_k}{\partial z_k} \cdot \frac{\partial z_k}{\partial \alpha_{ki}} \end{aligned}$$

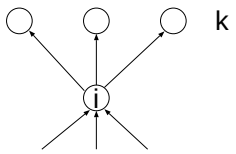
$$\frac{\partial E_n}{\partial \alpha_{ki}} = [y_k(x_n) - t(k, c_n)] \cdot f'(z_k) \cdot y_i(x_n)$$

b) verborgene Schicht: $y_i = f(z_i)$ mit $z_i = \sum_l \alpha_{il} \cdot y_l$

$$\frac{\partial E_n}{\partial \alpha_{il}} = \frac{\partial E_n}{\partial y_i} \cdot \frac{\partial y_i}{\partial z_i} \cdot \frac{\partial z_i}{\partial \alpha_{il}}$$

$$\frac{\partial E_n}{\partial \alpha_{il}} = \frac{\partial E_n}{\partial y_i} \cdot f'(z_i) \cdot y_l$$

$\frac{\partial E_n}{\partial y_i}$ ist jetzt komplizierter.
Alle Ausgänge y_k ändern sich, falls bei y_i eine Änderung auftritt.



$$\begin{aligned}\frac{\partial E_n}{\partial y_i} &= \sum_k \frac{\partial E_n}{\partial y_k} \cdot \frac{\partial y_k}{\partial z_k} \cdot \frac{\partial z_k}{\partial y_i} \\ &= \sum_k \frac{\partial E_n}{\partial y_k} \cdot f'(z_k) \cdot \frac{\partial}{\partial y_i} \sum_{i'} \alpha_{ki'} \cdot y_{i'}\end{aligned}$$

$$\frac{\partial E_n}{\partial y_i} = \sum_k \frac{\partial E_n}{\partial y_k} \cdot f'(z_k) \cdot \alpha_{ki}$$

Anmerkung: Die Ableitungen $\dfrac{\partial E_n}{\partial y_i}$ werden *rekursiv* berechnet, indem man mit der Ausgangsschicht anfängt und Schicht für Schicht absteigt (Error Back Propagation).

Veränderung (*Update*) der Parameter α_{il} (oder auch α_{ki}):

$$\bar{\alpha}_{il} = \alpha_{il} - \gamma \cdot \frac{\partial E_n}{\partial \alpha_{il}} \qquad \text{mit Schrittweite } \gamma \, .$$

Die praktische Implementierung des Error Back Propagation erfordert geeignete Heuristiken bzgl.:

- Wahl der Startwerte,
- Wahl der Schrittweite (in der Regel abnehmend mit der Zahl der Iterationen),
- Vermeidung von schlechten lokalen Optima,
- Steuerung der Konvergenzgeschwindigkeit.

Insgesamt erfordert die Realisierung und Implementierung erhebliche Sorgfalt.

4.4 Diskriminatives Training für statistische Klassifikatoren

Damit bezeichnet man Kriterien oder Verfahren, die versuchen, die a-posteriori-Wahrscheinlichkeit als Kriterium für die Lernphase zu verwenden; z.B.

$$\vartheta \longrightarrow \prod_{n=1}^{N} p(k_n|x_n, \vartheta)$$

oder mit Logarithmierung

$$\vartheta \longrightarrow \sum_{n=1}^{N} \log p(k_n|x_n, \vartheta),$$

wobei ϑ für die unbekannten Parameter steht.
(Beachte die Analogie, aber auch den Unterschied zu Maximum-Likelihood.)

Für $p(k_n|x_n, \vartheta)$ gilt:

$$p(k_n|x_n, \vartheta) = \frac{p(k_n) \cdot p(x_n|k_n, \vartheta)}{\sum_{c=1}^{K} p(c) \cdot p(x_n|c, \vartheta)}$$

Terminologie:
Diskriminativ, weil $p(k_n|x_n, \vartheta)$ den Klassenüberlapp erfaßt.

Zur Bestimmung von ϑ muß man Näherungsverfahren einsetzen, z.B. die entsprechende Modifikation des Error Back Propagation.

Übungsaufgabe: Formuliere das Error Back Propagation für das obige Kriterium, wobei $p(x|k, \mu_k, \Sigma_k)$ Gauß-Verteilungen mit diagonalen Kovarianzmatrizen Σ_k sind.

5 Modellfreie Verfahren

5.1 Einführung

Terminologie: Modellfreie Verfahren heißen auch "nichtparametrisch".

Für Zufallszahlen $x \in \mathbb{R}^D$ haben wir in Kapitel 2 und 3 spezielle Modelle $p(x|k, \vartheta_k)$ verwendet,

z.B. Gauß- oder ähnliche Verteilungen.

Charakteristisch dabei war die starke Datenreduktion der Trainingsdaten in wenige Parameter mit

$$\text{Schätzwerten} \qquad \hat{\vartheta}_k = \hat{\vartheta}_k(x_{1k}, \ldots, x_{N_k k}).$$

Das andere Extrem wird durch die modellfreien Verfahren erreicht, deren Eigenschaften sind:

- es werden (fast) keine speziellen Annahmen über die Verteilungen gemacht;
- die wahren Verteilungen können u.U. beliebig genau approximiert werden.

Ein möglicher Nachteil ist der erhöhte Bedarf an CPU-Zeit und Speicherplatz.

Verteilungen, die mittels modellfreier Verfahren beschrieben werden sollen, können sein:

- die klassenbedingten Verteilungen $p(x|k)$,
- die a-posteriori-Verteilungen $p(k|x)$,
- ...

Typische modellfreie Verfahren sind:

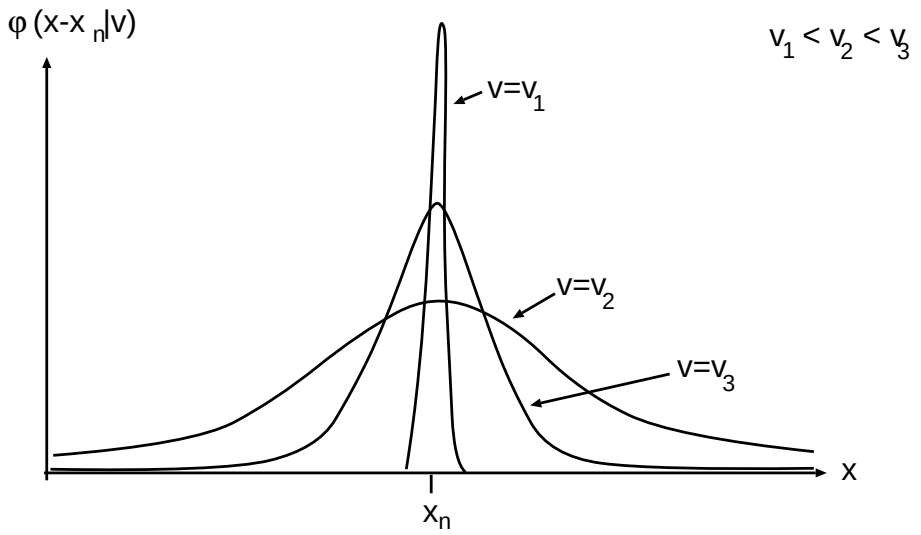
- *Kernel Densities* (Kernschätzverfahren) oder *Parzen Densities*;
- Nächste-Nachbar-Regel für die Klassifikation;
- Diskriminantenverfahren, z.B. Polynomklassifikatoren oder Multilayer-Perceptron (Kapitel 4);
- Mischverteilungen (Kapitel 6).

Der Übergang zwischen *modellfreien* und *modell-basierten* Verfahren ist natürlich fließend. Falls ein Multilayer-Perceptron zu wenige Knoten (Parameter) hat, kann man strenggenommen nicht mehr von einem modellfreien Verfahren sprechen.

5.2 Kernel Densities

Der Ausgangspunkt ist eine Art von Histogramm-Approximation für die Trainingsdaten x_n mit $n = 1, \ldots, N$.

Zur Vereinfachung sei $x_n \in \mathbb{R}$.



Der Beitrag eines Trainingsvektors x_n wird über seine Umgebung *verteilt*. Diese Verteilung wird durch eine Funktion φ beschrieben, die *Kernel Density, Parzen Density* oder *Parzen Window* genannt wird:

$$\begin{aligned} \varphi : \mathbb{R}^D &\longrightarrow [0, \infty[\\ x &\longmapsto \varphi(x) \end{aligned}$$

mit

- $\varphi(x) \geq 0$,
- $\int\limits_{\mathbb{R}^D} \varphi(x)\, dx = 1$.

$\varphi(x)$ ist selbst eine Verteilung. Man wählt diese "glatt" und unimodal.

Die Verteilung φ kann im allgemeinen von der Klasse k abhängen, daher schreiben wir $\varphi_k(x)$. Typische Wahl für $\varphi_k(x)$ ist die Gauß-Verteilung:

$$\varphi_k(x) \;=\; \prod_{d=1}^{D} \frac{1}{\sqrt{2\pi v_{kd}^2}} \exp\left[-\frac{1}{2}\left(\frac{x_d}{v_{kd}}\right)^2\right] = \varphi(x|v_k)$$

mit geeigneten klassenabhängigen Varianzen v_{kd}.

Als Schätzung für die Verteilung $p(x|k)$ wählt man

$$\hat{p}(x|k) \;=\; \frac{1}{N_k} \sum_{n=1}^{N_k} \varphi_k(x - x_{nk}) \;,$$

wobei $x_{1k}, \ldots, x_{N_k k}$ die Trainingsdaten der Klasse k sind.

Noch zu bestimmen sind die Varianzen v_{kd}. Die Maximum-Likelihood-Schätzung funktioniert nicht ohne weiteres, weil das Maximum für $v_{kd} \to 0$ angenommen wird.

Abhilfe:
Kombination mit *leaving-one-out*, wobei sich allerdings sehr unhandliche Gleichungen ergeben.

Als Ausweg kann man versuchen, einen klassenabhängigen Faktor α_k einzuführen und die üblichen Schätzwerte für die Varianzen mit diesem Faktor α_k zu multiplizieren:

$$v_{kd}(\alpha_k) \;=\; \alpha_k \cdot \frac{1}{N_k} \sum_{n=1}^{N_k} (x_{nk,d} - \hat{\mu}_{kd})^2 \;.$$

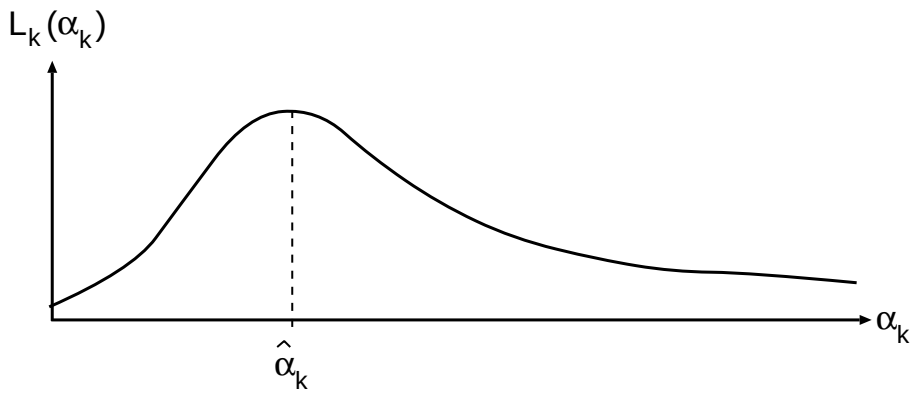
Der optimale Wert für α_k kann jetzt durch einfaches "Durchfahren" gefunden werden (in Verbindung mit leaving-one-out):

$$\alpha_k \;\longrightarrow\; L_k(\alpha_k) := \sum_{m=1}^{N_k} \log f_m(x_m)$$

$$\text{mit} \quad f_m(x) := \frac{1}{N_k - 1} \sum_{\substack{n=1 \\ n \neq m}}^{N_k} \varphi(x - x_n | v_k(\alpha_k)) \;.$$

Eine Näherungslösung erhält man mittels der Maximum-Approximation, indem man die Summe in $f_m(x)$ durch das Maximum ersetzt:

$$f_m(x) \;\tilde{=}\; \frac{1}{N_k - 1} \max_{\substack{n=1,\ldots,N_k \\ n \neq m}} \{\varphi(x - x_n | v_k(\alpha_k))\}$$



Übungsaufgabe:
Berechne die Faktoren α_k für den Fall, daß die Kernel Density eine Gauß-Verteilung mit diagonaler Kovarianzmatrix ist.

Verfeinerungen:
Wie schon bei der Gauß-Verteilung in Kapitel 2 kann man die Abhängigkeit $u \longrightarrow \varphi_k(u)$ über Abstände definieren ($u \in \mathbb{R}^D$):

- radialsymmetrisch:

$$\varphi_k(u) \;:=\; \varphi_k(u^T u) = \varphi_k(||u||^2)$$

- elliptischsymmetrisch mit Gewichtsmatrix A_k:

$$\varphi_k(u) \;:=\; \varphi_k((A_k u)^T (A_k u)) = \varphi_k(u^T A_k^T A_k u)$$

Beispiele für Kernel Densities in radialsymmetrischer Formalisierung (zur Vereinfachung ohne Varianzen):

- Gauß: $\varphi(u) = \frac{1}{\sqrt{2\pi}} \cdot \exp\left(-\frac{1}{2}\|u\|^2\right)$;
- t-Verteilung: $\varphi(u) = \frac{A}{(B + ||u||^2)^M}$, $A, B, M > 0$.

Eng verwandt ist die Methode der Potentialfunktion, bei der oft das Coulomb-Potential (aus der Elektrotechnik) verwendet wird:

$$\varphi(u) = \frac{1}{\epsilon + \|u\|} , \qquad A, \epsilon > 0 ,$$

(problematisch wegen der Normierung).

5.3 Nächste-Nachbar-Regel (NN-Regel)

genauer: *nearest neighbor decision rule*, und zwar 1-NN im Unterschied zu m-NN.

Wir leiten die Nächste-Nachbar-Regel aus den Kernel Densities ab (in der Literatur findet man eine andere Herleitung).

Trainingsdaten für jede Klasse k: $x_{1k}, \ldots, x_{N_k k}$

Als Schätzwert für $p(x|k)$ verwenden wir die Kernel Density:

$$\hat{p}(x|k) = \frac{1}{N_k} \sum_{n=1}^{N_k} \varphi(||x - x_{nk}||),$$

mit $\varphi : \mathbb{R}^+ \longrightarrow \mathbb{R}^+$ und geeignetem Abstand $||\ ||$.
Weiterhin nehmen wir zur Vereinfachung an, daß die Kernel Density φ klassenunabhängig ist.

Schätzwert für $p(k)$:

$$\hat{p}(k) \;=\; \frac{N_k}{N} \qquad \text{mit} \;\; N = \sum_{k=1}^{K} N_k \; .$$

Wir setzen $\hat{p}(x|k)$ und $\hat{p}(k)$ in die Bayessche Entscheidungsregel ein:

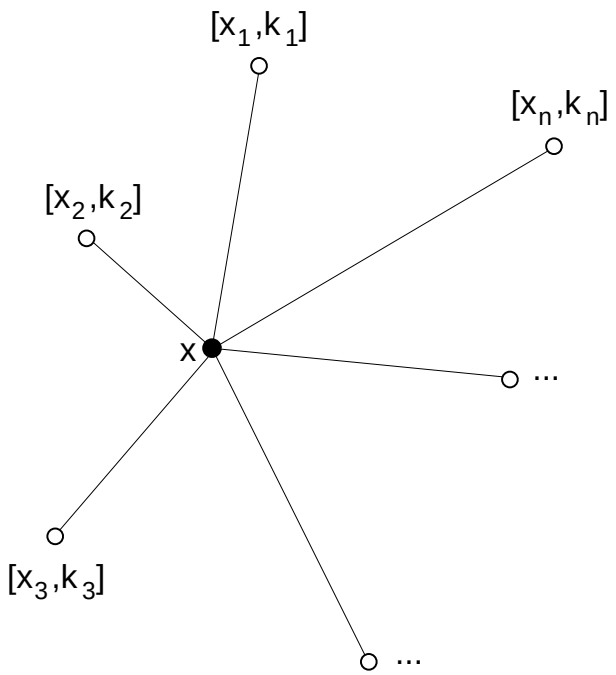
$$\begin{aligned}
r(x) \;&=\; \underset{k}{\text{argmax}} \, \{\hat{p}(k) \cdot \hat{p}(x|k)\} \\
&=\; \underset{k}{\text{argmax}} \, \{\frac{N_k}{N} \cdot \frac{1}{N_k} \sum_{n=1}^{N_k} \varphi(||x - x_{nk}||)\} \\
&=\; \underset{k}{\text{argmax}} \, \{\sum_{n=1}^{N_k} \varphi(||x - x_{nk}||)\} \\
&\qquad \text{(Maximum-Approximation: ersetze } \textstyle\sum \text{ durch max)} \\
&\tilde{=}\; \underset{k}{\text{argmax}} \, \{\max_{n=1,\ldots,N_k} \varphi(||x - x_{nk}||)\} \\
&\qquad (z \longrightarrow \varphi(z) \text{ ist monoton fallend}) \\
&=\; \underset{k}{\text{argmin}} \, \{\min_{n=1,\ldots,N_k} ||x - x_{nk}||\}
\end{aligned}$$

Diese Approximation heißt Nächste-Nachbar-Regel, weil für den Vektor x diejenige Klasse gewählt wird, zu der der nächste Nachbar unter den Trainingsdaten x_{nk} gehört:

$$r(x) \;=\; \underset{k}{\text{argmin}} \, \{\min_{n=1,\ldots,N_k} ||x - x_{nk}||\}$$

Beachte die Abhängigkeit der NN-Regel vom gewählten Abstandsmaß $||x - y||$.

Veranschaulichung der NN-Regel für Trainingsvektoren $[x_n, k_n]$
mit $n = 1, \ldots, N$ und einem Testvektor x:



Es gibt Ansätze (erfolgreiche und erfolglose), den Rechenaufwand, d.h. die Anzahl der Abstandsberechnungen, zu reduzieren:

1. vorzeitiger Abbruch (*partial distance*);
2. Dreiecksungleichung;
3. hierarchische Strukturierung der Trainingsvektoren;
4. *k-d trees*;
5. Reduktion der Anzahl der Trainingsvektoren.

5.4 Fehlerrate der Nächste-Nachbar-Regel

genauer: Fehlerrate im asymptotischen Grenzfall, wenn die Anzahl N der Trainingsdaten unendlich groß wird (Cover, 1967).

Zunächst betrachten wir die Fehlerrate für die Bayessche Entscheidungsregel (Kapitel 2):

$$r_B(x) \;=\; \underset{k}{\operatorname{argmax}} \{p(k|x)\} \qquad \text{für } x \in \mathbb{R}^D.$$

Wir erhalten für den lokalen Fehler $p_B(e|x)$:

$$\begin{aligned} p_B(e|x) &= "p_{Bayes}(error|x)" \\ &= 1 - p_B(correct|x) \\ &= 1 - \max_k p(k|x) \\ &= 1 - p(r_B(x)|x) \end{aligned}$$

Die Gesamtfehlerrate $p_B(e)$ ergibt sich durch Integration über den gesamten Raum:

$$\begin{aligned} p_B(e) &= \int_x p(x) \cdot p_B(e|x)\; dx \\ &= \int_x \Big[1 - \max_k p(k|x)\Big] \cdot p(x)\; dx \\ &= \int_x 1 \cdot p(x)\; dx - \int_x \max_k p(k|x) \cdot p(x)\; dx \\ &= 1 - \int_x \max_k p(k,x)\; dx \end{aligned}$$

Für die Nächste-Nachbar-Regel definieren wir:

x_{NN} : nächster Nachbar des beobachteten Testvektors x, d.h.

$$x_{NN} \;:=\; \underset{x_n : n=1,\ldots,N}{\operatorname{argmin}} \{||x - x_n||\},$$

mit den Trainingsvektoren $[x_n, k_n], \quad n = 1, \ldots, N$.

Für die lokale Fehlerrate $p_{NN}(e|x, x_{NN})$ der NN-Regel bei gegebenem x und daraus berechnetem x_{NN} ergibt sich

$$\begin{aligned} p_{NN}(e|x, x_{NN}) &= \sum_k \left[p(k|x) \cdot \sum_{c \neq k} p(c|x_{NN}) \right] \\ &= \sum_k p(k|x) \cdot [1 - p(k|x_{NN})] \\ &= 1 - \sum_k p(k|x) \cdot p(k|x_{NN}) \; . \end{aligned}$$

Im Grenzfall $N \longrightarrow \infty$, d.h. bei beliebig vielen Trainingsdaten, gilt

$$p(k|x_{NN}) = p(k|x) \; .$$

Damit vereinfacht sich die lokale Fehlerrate zu

$$p_{NN}(e|x, x_{NN}) = 1 - \sum_k p^2(k|x) \; .$$

Wir schätzen die lokale Fehlerrate der NN-Regel durch die Bayessche Fehlerrate wie folgt ab:

- Abschätzung nach unten:

$$\begin{aligned} p_B(e|x) &= 1 - \max_k p(k|x) \\ &= 1 - \sum_c p(c|x) \cdot \max_k p(k|x) \\ &\leq 1 - \sum_k p^2(k|x) \\ &= p_{NN}(e|x, x_{NN}) \; . \end{aligned}$$

- Abschätzung nach oben:

$$\begin{aligned} p_{NN}(e|x, x_{NN}) &= 1 - \sum_k p^2(k|x) \\ &= 1 - p^2(r_B(x)|x) - \sum_{k \neq r_B(x)} p^2(k|x) \end{aligned}$$

Wir verwenden einen Spezialfall der Cauchy-Schwarzschen Ungleichung:

$$\begin{aligned}\left[\sum_{k\neq r_B(x)} p(k|x)\right]^2 &\leq \sum_{k\neq r_B(x)} 1^2 \cdot \sum_{k\neq r_B(x)} p^2(k|x) \\ &= (K-1)\cdot \sum_{k\neq r_B(x)} p^2(k|x)\end{aligned}$$

und damit

$$\begin{aligned}\sum_{k\neq r_B(x)} p^2(k|x) &\geq \frac{1}{K-1}\left[\sum_{k\neq r_B(x)} p(k|x)\right]^2 \\ &= \frac{1}{K-1}\cdot\left[1-p(r_B(x)|x)\right]^2 \ .\end{aligned}$$

Einsetzen der Ungleichung in $p_{NN}(e|x, x_{NN})$ ergibt:

$$\begin{aligned}p_{NN}(e|x,x_{NN}) &\leq 1-p^2(r_B(x)|x)-\frac{1}{K-1}\cdot\left[1-p(r_B(x)|x)\right]^2 \\ &= \ldots \\ &= 2\cdot\left[1-p(r_B(x)|x)\right]-\frac{K}{K-1}\cdot\left[1-p(r_B(x)|x)\right]^2\end{aligned}$$

$$p_{NN}(e|x,x_{NN}) \leq 2\cdot p_B(e|x)-\frac{K}{K-1}\cdot p_B^2(e|x)$$

Für die Gesamtfehlerrate benötigen wir eine weitere Ungleichung:
Sei $f(x)$ eine Funktion der Zufallsvariablen x, dann gilt:

$$Var\{f(x)\} = E\{f^2(x)\}-E^2\{f(x)\}\geq 0$$

und somit

$$E^2\{f(x)\} \leq E\{f^2(x)\} \ .$$

Also ergibt sich bei Integration über den gesamten Raum aus beiden Abschätzungen:

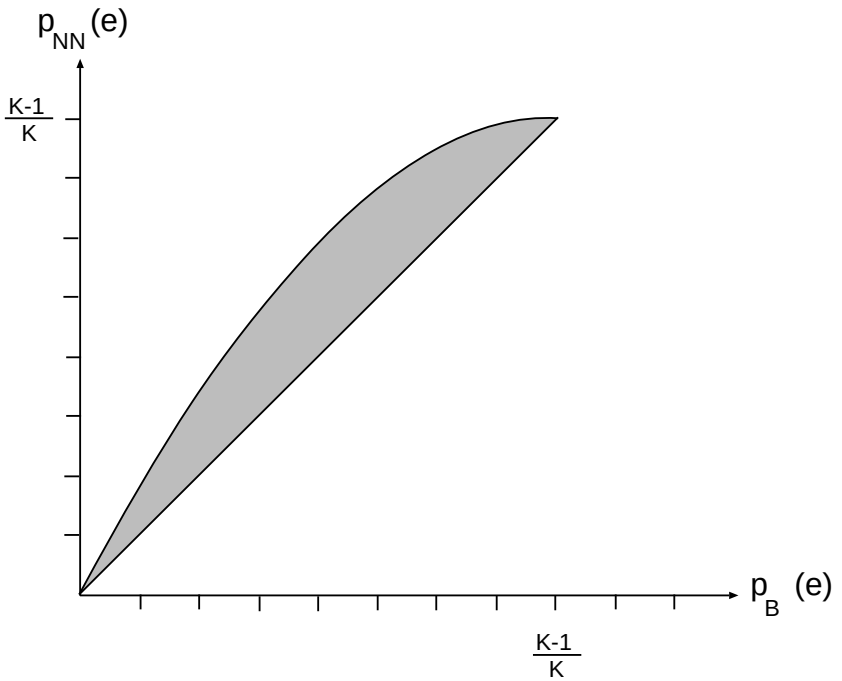
$$p_B(e) \leq \ p_{NN}(e) \ \leq 2 \cdot p_B(e) - \frac{K}{K-1} \cdot p_B^2(e) \leq 2 \cdot p_B(e)$$

Übungsaufgaben:

1. Wann ist die obere Schranke scharf?
2. Wie sieht die obere Schranke für $p_{NN}(e)$ im Fall einer beliebigen (statt der Bayesschen) Entscheidungsregel aus?

Veranschaulichung der Ungleichung für $p_{NN}(e)$:

$$p_B(e) \leq \ p_{NN}(e) \ \leq p_B(e) \cdot \left[2 - \frac{K}{K-1} \cdot p_B(e)\right]$$



Der schraffierte Bereich zeigt die Werte $p_{NN}(e)$, die bei gegebenem $p_B(e)$ möglich sind.

5.5 Ausblick

Die in Kapitel 5.4 hergeleitete Ungleichung für $p_{NN}(e)$ gilt unabhängig von der Anwendung der Nächsten-Nachbar-Regel. Tatsächlich ist $p_{NN}(e)$ identisch mit dem in Kapitel 4.1 erwähnten Gini-Kriterium. Damit erhält man einen Zusammenhang zwischen Erkennungsfehlerrate und Fehlerquadratkriterium bei Diskriminanten:

Zusammenstellung der Definitionen:

$$\begin{aligned} p_B(e|x) &= 1 - \max_k p(k|x) \\ p_{NN}(e|x, x_{NN}) &= 1 - \sum_{k=1}^{K} p^2(k|x) \\ f(g;x) &= \sum_{k=1}^{K} p(k|x) \cdot \sum_{c=1}^{K} [g(x,c) - t(k,c)]^2 \\ &= \left[1 - \sum_{k=1}^{K} p^2(k|x)\right] + \sum_{c=1}^{K} [p(c|x) - g(k,c)]^2 \end{aligned}$$

(lokales Fehlerquadratkriterium am Ausgang eines neuronalen Netzes bzw. Diskriminante, vgl. 4.1)
Damit gilt für jeden Punkt $x \in \mathbb{R}^D$:

$$p_B(e|x) \leq p_{NN}(e|x, x_{NN}) \leq f(g;x)$$

und falls $g(x,k) \equiv p(k|x)$ für alle k:

$$p_{NN}(e|x, x_{NN}) = f(g;x).$$

Eine entsprechende Ungleichung/Gleichung gilt auch für die globalen Erwartungswerte:

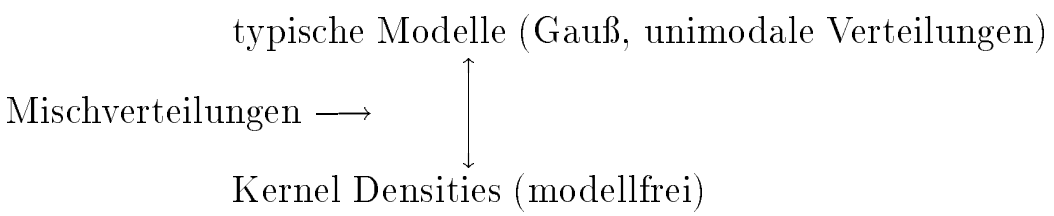
$$\begin{aligned} p_B(e) &:= \int p_B(e|x) \cdot p(x)\, dx \\ p_{NN}(e) &:= \int p_{NN}(e|x) \cdot p(x)\, dx \\ f(g) &:= \int f(g;x) \cdot p(x)\, dx \end{aligned}$$

6 Mischverteilungen und Clusteranalyse

6.1 Mischverteilungen

6.1.1 Einführung

bisher zwei Extreme:



Kompromiß:

- multimodal statt unimodal
- Datenreduktion im Vergleich zu Kernel Densities
- beliebig genaue Approximation für beliebige Verteilungen

Standardmodell: $x \in \mathbb{R}^D$ (Klasse k fest)

$$p(x) = \sum_{i=1}^{I} c_i \cdot p(x|i, \vartheta_i)$$

(gewichtete Summe von elementaren Verteilungen)

mit den Gewichten (*mixture weights*) c_i, für die gilt

$$c_i \geq 0, \quad \sum_i c_i = 1$$

und den Basisverteilungen, Einzelverteilungen (*component densities*)

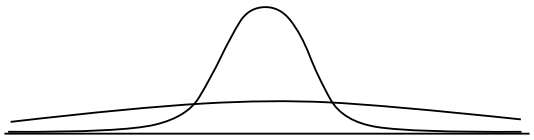
$$p(x|i, \vartheta_i).$$

Typischerweise wählt man die Einzelverteilungen unimodal, meistens Gauß:

$$\vartheta_i = (\mu_i, \Sigma_i)$$

ähnliche Anwendungen sind:

- Ausreißermodell



Zweck: ϵ-Wahrscheinlichkeit für den nichtzentralen Bereich

$$p(x) \;=\; (1-\epsilon)\cdot p_{Gauß}(x|\mu,\Sigma) + \epsilon \cdot p_{Gauß}(x|\mu,\alpha\Sigma)$$

$\alpha > 1$: Vergrößerungsfaktor für die Streuung

- Interpolation bei Language-Modellen, z.B. Bigramm
  $N(v,w)$: Zähler für das Wortpaar v, w der Trainingsdaten
  Ansatz:

$$p(w|v) \;=\; \frac{N(v,w)}{N(v)}$$

Viele Paare werden im Training nicht gesehen, d.h. $N(v,w) = 0$. Dies hat zur Folge, daß die entsprechende bedingte Wahrscheinlichkeit $p(w|v) = 0$ ist.

Verbesserter Ansatz durch Interpolation:

$$p(w|v) \;=\; (1-\mu)\cdot \frac{N(v,w)}{N(v)} + \mu \cdot q(w)$$

mit μ : Interpolationsparameter
$q(w)$: Unigramm-Verteilung

6.1.2 EM-Algorithmus

Wir fassen die unbekannten Parameter c_i und ϑ_i zusammen unter der Bezeichnung λ:

$$\lambda \equiv \{c_i, \vartheta_i\}$$

Modell: $p_\lambda(x)$ oder $p(x|\lambda)$

Trainingsdaten: $x_1, \ldots, x_n, \ldots, x_N$

Maximum-Likelihood-Kriterium:

$$\max_\lambda \prod_{n=1}^{N} p(x_n|\lambda) \quad \text{oder} \quad \max_\lambda \sum_{n=1}^{N} \log p(x_n|\lambda)$$

Damit ist zunächst ein unhandliches Optimierungsproblem gegeben. Für Mischverteilungen und ähnlich komplexe Modellverteilungen existiert keine geschlossene Lösung.
Das Ziel des EM-Algorithmus liegt darin,

- ein einfaches, konsistentes Iterationsverfahren,
- ohne Heuristiken und
- ohne (expliziten) Gradienten

zu entwickeln.

Der EM-Algorithmus beruht auf dem Konzept einer *hidden variable* y (diskrete Zufallsvariable):
Umschreiben:

$$\begin{aligned} p(x_n|\lambda) &= \sum_y p(x_n, y|\lambda) \\ &= \sum_y p(y|\lambda) \cdot p(x_n|y, \lambda) \end{aligned}$$

Mischverteilung: $y \hat{=} i$

$$p(x_n|\lambda) = \sum_i \underbrace{p(i|\lambda)}_{\hat{=} c_i} \cdot p(x_n|i, \vartheta_i)$$

i ist nicht beobachtbar

Anwendungen des EM-Algorithmus:

- Mischverteilungen
- Glättung in der Language-Modellierung
- Hidden Markov Modelle
- IBM-Übersetzungsmethode
- ...

Wir betrachten die Differenz der log-Likelihood-Funktion für zwei Parametersätze λ und $\bar{\lambda}$:

$$\sum_{n=1}^{N} \log \frac{p(x_n|\bar{\lambda})}{p(x_n|\lambda)} = \sum_{n=1}^{N} \underbrace{\sum_y p(y|x_n,\lambda)}_{\equiv 1} \cdot \log \frac{p(x_n|\bar{\lambda})}{p(x_n|\lambda)}$$

umformen: $p(x_n|\bar{\lambda}) \cdot p(y|x_n,\bar{\lambda}) = p(x_n,y|\bar{\lambda})$

$$= \sum_{n=1}^{N} \sum_y p(y|x_n,\lambda) \cdot \log \left[\frac{p(x_n,y|\bar{\lambda})}{p(x_n,y|\lambda)} \cdot \frac{p(y|x_n,\lambda)}{p(y|x_n,\bar{\lambda})} \right]$$

$$= \sum_{n=1}^{N} \sum_y p(y|x_n,\lambda) \cdot \log \frac{p(x_n,y|\bar{\lambda})}{p(x_n,y|\lambda)} + \underbrace{\sum_{n=1}^{N} \sum_y p(y|x_n,\lambda) \cdot \log \frac{p(y|x_n,\lambda)}{p(y|x_n,\bar{\lambda})}}_{(*) \qquad \geq 0 \qquad \forall\, \lambda,\bar{\lambda}}$$

$(*)$ ist die Divergenz-Ungleichung der Informationstheorie.
Definiere:

$$Q(\lambda,\bar{\lambda}) := \sum_{n=1}^{N} \sum_y p(y|x_n,\lambda) \cdot \log p(x_n,y|\bar{\lambda})$$

Dann gilt

$$\sum_{n=1}^{N} \log \frac{p(x_n|\bar{\lambda})}{p(x_n|\lambda)} \geq Q(\lambda,\bar{\lambda}) - Q(\lambda,\lambda)$$

wegen $(*) \geq 0$.

EM-Iterationsverfahren:

Startwert λ
Iterationen:

1. Expectation (over y): berechne $Q(\lambda, \bar{\lambda})$
2. Maximization (over $\bar{\lambda}$): $\underset{\bar{\lambda}}{\text{argmax}} \{Q(\lambda, \bar{\lambda})\}$

   nächste Iteration mit $\lambda := \bar{\lambda}$

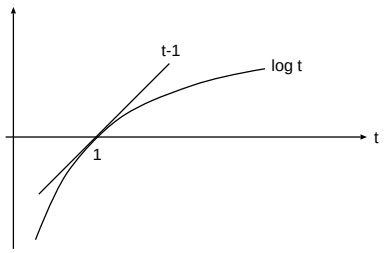
Beweis der Divergenz-Ungleichung:

$$\sum_{n=1}^{N} \sum_{y} p(y|x_n, \lambda) \cdot \log \frac{p(y|x_n, \lambda)}{p(y|x_n, \bar{\lambda})} \;\geq\; 0$$

Es reicht zu zeigen:

$$\sum_{y} p(y) \cdot \log \frac{p(y)}{q(y)} \geq 0 \quad \text{mit} \quad p(y) > 0,\ \sum_{y} p(y) = 1,$$
$$q(y) > 0,\ \sum_{y} q(y) = 1.$$

Für den Logarithmus gilt:



$$\log t \;\leq\; t - 1, \qquad t > 0$$

Damit gilt

$$\begin{aligned} \sum_{y} p(y) \cdot \log \frac{q(y)}{p(y)} \;&\leq\; \sum_{y} p(y) \left[\frac{q(y)}{p(y)} - 1 \right] \\ &=\; \sum_{y} q(y) - \sum_{y} p(y) = 1 - 1 = 0 \end{aligned}$$

und es folgt die Divergenz-Ungleichung.

Anwendung auf Mischverteilungen:

$$\begin{aligned} p(x_n|\lambda) &= \sum_i p(x_n, i|\lambda) \\ &= \sum_i p(i|\lambda) \cdot p(x_n|i, \lambda) \\ \text{Modell} &\hat{=} \sum_i c_i \cdot p(x_n|i, \vartheta_i) \end{aligned}$$

abgekürzte Schreibweise für den Parametersatz: $\lambda \equiv (\{c_i\}, \{\vartheta_i\})$

$$\begin{aligned} Q(\lambda, \bar{\lambda}) &= \sum_n \sum_i p(i|x_n, \lambda) \cdot \log\left[\bar{c}_i \cdot p(x_n, i|\bar{\vartheta}_i)\right] \\ \text{mit } p(i|x_n, \lambda) &= \frac{c_i \cdot p(x_n|i, \vartheta_i)}{\sum_{i'} c_{i'} \cdot p(x_n|i', \vartheta_{i'})} \end{aligned}$$

Ableitungen zur Bestimmung von $\bar{\lambda}$ als Funktion von λ:

a) $\dfrac{\partial Q}{\partial \bar{c}_i}$: Die Nebenbedingung $\sum_i \bar{c}_i = 1$ wird mittels eines Lagrange-Multiplikators ausgedrückt.

Ergebnis:

$$\bar{c}_i = \frac{1}{N} \sum_{n=1}^{N} p(i|x_n, \lambda) = \frac{1}{N} \sum_{n=1}^{N} \frac{c_i \cdot p(x_n|i, \vartheta_i)}{\sum_{i'} c_{i'} \cdot p(x_n|i', \vartheta_{i'})}$$

b)

$$\frac{\partial Q}{\partial \bar{\vartheta}_i} = \sum_{n=1}^{N} p(i|x_n, \lambda) \cdot \frac{\partial}{\partial \bar{\vartheta}_i} \log\left[\bar{c}_i \cdot p(x_n|i, \bar{\vartheta}_i)\right] \stackrel{!}{=} 0$$

speziell Gauß: $p(x_n|i, \mu_i, \Sigma_i)$

mit μ_i = Mittelwertsvektor
Σ_i = Kovarianzmatrix

$$\frac{\partial Q}{\partial \bar{\mu}_{id}} = \sum_n p(i|x_n, \lambda) \cdot [\bar{\mu}_{id} - x_{nd}] \stackrel{!}{=} 0$$

$$\Leftrightarrow \quad \bar{\mu}_{id} \;=\; \frac{\sum\limits_n p(i|x_n,\lambda)\cdot x_{nd}}{\sum\limits_n p(i|x_n,\lambda)}$$

Umschreiben als Vektorgleichung mit Gewichten $\gamma_i(n)$ liefert

$$\bar{\mu}_i = \sum_{n=1}^{N} \gamma_i(n)\cdot x_n \quad \text{mit} \quad \gamma_i(n) = \frac{p(i|x_n,\lambda)}{\sum\limits_n p(i|x_n,\lambda)}.$$

Die Gewichte $\gamma_i(n)$ drücken aus, mit welchem Anteil die Beobachtung x_n in die Schätzung von $\bar{\mu}_i$ eingeht.

Beachte: $\sum\limits_{n=1}^{N} \gamma_i(n) = 1$ gilt nach Definition.

Entsprechend ergibt sich für die Kovarianzmatrix:

$$\bar{\Sigma}_i \;=\; \sum_{n=1}^{N} \gamma_i(n)\cdot [x_n-\bar{\mu}_i][x_n-\bar{\mu}_i]^T.$$

Interpretation:
Gewichtete Maximum-Likelihood-Schätzung, wobei die Gewichte von $p(i|x_n,\lambda)$ abhängen und sich während der Iteration selbst verändern:

- μ_i : gewichteter empirischer Mittelwertsvektor
- Σ_i : gewichtete empirische Kovarianzmatrix
- c_i : gewichtete relative Häufigkeit

beachte:

- nur lokale Konvergenz;
- offen, wie die Startwerte zu wählen sind.

Algorithmus:

wähle geeignete Startwerte μ_i, Σ_i, c_i

Loop:
$$p(i|x_n, \{c_i, \mu_i, \Sigma_i\}) := \frac{c_i \cdot p(x_n|i, \mu_i, \Sigma_i)}{\sum_{i'} c_{i'} \cdot p(x_n|i', \mu_{i'}, \Sigma_{i'})}$$

$$\gamma_i(n) := \frac{p(i|x_n, \{c_i, \mu_i, \Sigma_i\})}{\sum_n p(i|x_n, \{c_i, \mu_i, \Sigma_i\})}$$

$$\bar{c}_i := \frac{1}{N} \sum_n p(i|x_n, \{c_i, \mu_i, \Sigma_i\})$$

$$\bar{\mu}_i := \sum_n \gamma_i(n) \cdot x_n$$

$$\bar{\Sigma}_i := \sum_n \gamma_i(n) \cdot [x_n - \bar{\mu}_i][x_n - \bar{\mu}_i]^T$$

update: $c_i = \bar{c}_i \qquad \mu_i = \bar{\mu}_i \qquad \Sigma_i = \bar{\Sigma}_i$

GOTO Loop

Übungsaufgaben:

1. Schreibe ein C-Programm für obiges Beispiel.
2. Berechne die Formeln für den EM-Algorithmus bei Gauß-Verteilung mit *tied covariance matrix*, d.h. $p(x_n|i, \mu_i, \Sigma)$ mit Σ unabhängig von i.

6.1.3 Mischverteilungen für K Klassen

Fallunterscheidung:

a) keine Kopplung zwischen den Klassen:

für jede Klasse k: Gewichte c_{ik}, $\sum_{i=1}^{I_k} c_{ik} = 1$

Parameter der bedingten Verteilungen ϑ_{ik}

Modell: $$p(x|k) = \sum_{i=1}^{I_k} c_{ik} \cdot p(x|i,k,\vartheta_{ik})$$

Das Training kann separat für jede Klasse k erfolgen.

b) *Semicontinuous mixture densities*:
Kopplung zwischen den Modellen $p(x|k)$ durch klassenunabhängige Parameter ϑ_i.

Modell: $$p(x|k) = \sum_{i=1}^{I_k} c_{ik} \cdot p(x|i,\vartheta_i)$$

d.h. die Einzelverteilungen $p(x|i,\vartheta_i)$ spielen die Rolle von *Basisverteilungen.*

Übungsaufgabe:

Berechne die EM-Iterationsformel für obiges Modell der *semicontinuous mixture densities* für Gauß-Verteilungen.

Ansatz:

$$Q(\lambda,\bar{\lambda}) = \sum_{k=1}^{K}\sum_{n=1}^{N_k}\sum_{i=1}^{I_k} p(i|x_{nk},k,\lambda)\cdot\log\left[\bar{c}_{ik}\cdot p(x_{nk}|i,\bar{\vartheta}_i)\right]$$

mit

$$p(i|x_{nk},k,\lambda) = \frac{c_{ik}\cdot p(x_{nk}|i,\vartheta_i)}{\sum_{i'=1}^{I_k} c_{i'k}\cdot p(x_{nk}|i',\vartheta_{i'})}$$

und den Trainingsvektoren $x_{1k},\ldots,x_{nk},\ldots,x_{N_k k}$ für die Klasse k.

6.1.4 Maximum-Approximation

(ohne Klassenindex k)

Ersetze die Summe durch das Maximum:

$$p(x) \;=\; \max_i \,[c_i \cdot \underbrace{p(x|i,\mu_i,\Sigma_i)}_{\text{Gauß-Verteilung}}\;].$$

Rechtfertigung: Die Gauß-Verteilung fällt exponentiell ab, so daß erwartungsgemäß ein Summand dominiert.

Trainingsverfahren:

wähle geeignete Startwerte μ_i, Σ_i, c_i

Loop: bestimme die *beste* Einzelverteilung für jede Beobachtung x_n

$$\operatorname*{argmax}_i [c_i \cdot p(x_n|i,\mu_i,\Sigma_i)]$$

Die Beobachtung x_n geht nur in das Training einer Einzelverteilung ein, nämlich der besten.

Parameterschätzung:

- c_i = relative Häufigkeit, mit der i als beste Einzelverteilung ausgewählt wird
- μ_i = Mittelwert über die jeder Einzelverteilung i zugeordneten Beobachtungen x_n
- Σ_i = Kovarianzmatrix über die jeder Einzelverteilung i zugeordneten Beobachtungen x_n

GOTO Loop

Interpretation: Die Maximum-Approximation ergibt sich durch "Spezialisierung":

$$p(i|x_n,\{c_i,\mu_i,\Sigma_i\}) = \begin{cases} 1, & i = \textit{beste} \text{ Einzelverteilung}, \\ 0, & \text{sonst}. \end{cases}$$

6.2 Clusteranalyse

6.2.1 Übersicht

Cluster = Ansammlung, Anhäufung (von Beobachtungen, in der Regel Punkte im $\mathbb{R}^D$)

Ziel: Finden (und Interpretation) von Clustern (Untergruppen) für einen gegebenen Satz von Beobachtungen.

Für diese Aufgabe gibt es in vielen Fällen kein klares Optimierungskriterium und keine garantiert beste Optimierungsstrategie. Der im Kapitel 6.1 entwickelte Ansatz hat diese Schwierigkeiten (teilweise) vermieden, indem die Aufgabe durch ein klares statistisches Modell definiert wurde.

Die Literatur zur Clusteranalyse ist praktisch unüberschaubar und bietet verschiedenste Methoden, die sich nach verschiedenen Kriterien ordnen lassen:

- Abstandsmaße (Ähnlichkeitsmaße) $\leftrightarrow$ statistische Verfahren (mixture densities), fuzzy clustering
- diskrete $\leftrightarrow$ kontinuierliche Beobachtungen
- hierarchisch $\leftrightarrow$ nicht hierarchisch
- vollautomatisch $\leftrightarrow$ interaktiv

Kombinatorisches Problem:

- Es gibt exponentiell viele Partitionierungen für eine vorgegebene Menge von N Vektoren (paarweise verschieden).

  Vektoren $x_1, \ldots, x_N$ mit Label 0 oder 1, d.h. 2 Cluster

  $\Longrightarrow \frac{2^N - 2}{2} = 2^{N-1} - 1$ Partitionierungen
- Prinzipiell müssen alle Partitionierungen betrachtet werden, um die beste (z.B. Maximum-Likelihood-Kriterium) auszuwählen.

$S(N, I)$ ist die Anzahl der Partionierungen für I Cluster und N (paarweise verschiedene) Vektoren.

$$S(N,I) \;=\; \frac{1}{I!} \cdot \sum_{i=1}^{I} \binom{I}{i} (-1)^{I-1} \cdot i^N \cong \frac{1}{I!} I^N \;.$$

Bezeichnung: Stirling-Zahlen zweiter Art

Rekursion:

$$\begin{aligned} S(N+1,I) \;&=\; S(N,I-1) && x_{N+1} \text{ definiert das neue } I\text{-te Cluster} \\ &\quad + I \cdot S(N,I) && x_{N+1} \text{ wird einem der } I \text{ Cluster zugefügt} \end{aligned}$$

Beispiele:

- $I = 2 \Longrightarrow S(N,2) = \dfrac{2^N - 2}{2} = 2^{N-1} - 1$
- $N = 5, I = 3 \Longrightarrow S(5,3) = 25$

  Zerlegungen: $\{a,b,c\} \cup \{d\} \cup \{e\}$ $\quad \binom{5}{3} = 10$

  $\{a,b\} \cup \{c,d\} \cup \{e\}$ $\quad 5 \cdot 3 = 15$

In der Praxis:

- heuristische Verfahren;
- iterative Verfahren, die eine Start-Partitionierung schrittweise verbessern und so zumindest ein lokales Optimum garantieren (vgl. EM-Algorithmus).

6.2.2 Fehlerquadratkriterium

(häufig auch Varianzkriterium; minimum distance Verfahren)

Namen für das Iterationsverfahren zur Optimierung des Fehlerquadratkriteriums:

Nearest Mean, K-Means, Isodata
(hier: $K = I \hat{=}$ Zahl der Cluster)

Annahme: Alles wird auf ein (euklidisches) Abstandsmaß reduziert (nach geeigneter Skalierung der Achsen).

Seien A_i Cluster mit $i = 1, \ldots, I$, dann

$$\begin{aligned} G_{TOT} &:= \sum_{i=1}^{I} G_i \\ &\quad \text{mit } G_i := \sum_{x \in A_i} ||x - \mu_i||^2 \\ &\quad \text{und } \mu_i := \frac{1}{N_i} \sum_{x \in A_i} x \\ N_i &:= |A_i| \quad \text{Zahl der Elemente in } A_i \end{aligned}$$

Austauschoperation: Bewege x aus dem Cluster A_i in das Cluster A_j.

Wie ändern sich die Zentren μ_i und μ_j und die Bewertungen G_i und G_j bei einer Austauschoperation (nachrechnen, [DH73], p. 227):

$$\begin{aligned} \mu_j^* &= \mu_j + \frac{x - \mu_j}{N_j + 1} \\ \mu_i^* &= \mu_i - \frac{x - \mu_i}{N_i - 1} \\ G_j^* &= G_j + \frac{N_j}{N_j + 1} ||x - \mu_j||^2 \\ G_i^* &= G_i - \frac{N_i}{N_i - 1} ||x - \mu_i||^2 \end{aligned}$$

Variante: Clusterabhängige Varianzen einführen.

Algorithmen:

NEAREST MEAN (*sequentiell, adaptiv, online*)

wähle Start-Partitionierung $A_1, \ldots, A_I$ und berechne G_i und μ_i

Loop: wähle nächste Beobachtung x; sei $x \in A_i$
berechne G_j^* für alle Cluster A_j und transferiere x zu bestem Cluster

update: μ_j, G_j, μ_i, G_i

falls (G_{TOT} unverändert in N Durchgängen) STOP
sonst GOTO Loop

NEAREST MEAN (*batch, offline*)

wähle Start-Partition $A_1, \ldots, A_I$ und berechne G_i und μ_i

Loop: bestimme für jede Beobachtung x_n mit $n = 1, \ldots, N$ das nächstgelegene Cluster A_i und speichere den Clusterindex i

berechne für jedes Cluster A_i mit $i = 1, \ldots, I$ den neuen Mittelwert μ_i aus den dem Cluster zugeordneten Beobachtungen x_n

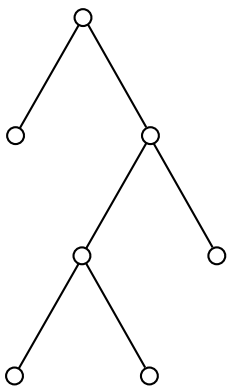
falls (keine Veränderung) STOP
sonst GOTO Loop

Anmerkungen:

1. Es ist klar, daß beide Algorithmen auf andere Modelle als "Mittelwert-Modelle" übertragbar sind.
2. Empirische Erfahrung: Batch-Variante weniger anfällig für schlechte lokale Minima.
3. Immer noch offen: Wie ist die Start-Partitionierung zu wählen?
   $\longrightarrow$ hierarschiches Clustern

6.2.3 Hierarchische Clusteranalyse

Die Hierarchie wird modelliert durch einen binären Baum:



Kriterien:

- Abstandsmaß oder Likelihood
- Strukturierung/Verfeinerung des Baumes

Zwei Varianten:

1. Top-down, divisives Verfahren (splitting, aufspalten)
2. Bottom-up, agglomeratives Verfahren (merging, verschmelzen)

Top-Down-Verfahren: Grundkonzept

Start: alle Daten $x_1, \ldots, x_N$ bilden ein Cluster

Loop: wähle Cluster mit der schlechtesten Bewertung,
spalte dieses Cluster in zwei neue Cluster auf

verbessere die Parameter aller Cluster durch iteratives Training
(NEAREST-MEAN-STYLE), bis Abbruchkriterium erfüllt ist

prüfe die Qualität der Gesamt-Partitionierung

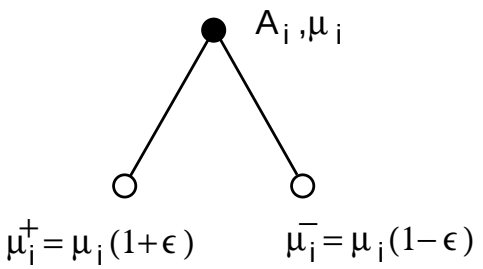
falls "gut" STOP,

sonst GOTO Loop

Problem:
Beim Aufspalten eines Clusters aus m Daten gibt es $2^{m-1}-1$ Möglichkeiten, 2 Cluster zu bilden. Diese können wegen der Rechenzeit in der Regel nicht alle ausgewertet werden.

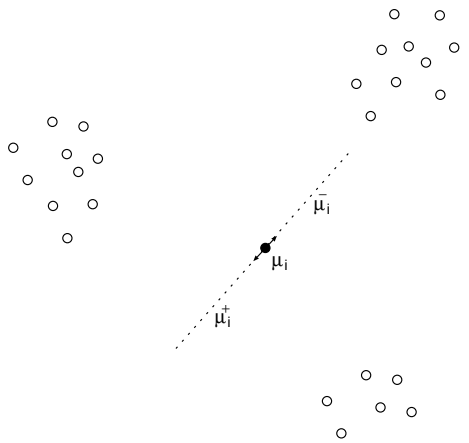
Heuristik für "Abstandsverfahren":

1. *infinitesimale* Störung des Mittelwertes μ_i



2. iteratives Training zur Verbesserung der μ_i^- und μ_i^+

   Beispiel:



   Die Störung erfolgt normalerweise in Standardrichtung, da die Richtung unkritisch ist.

   Sprachverarbeitung: Linde, Buzo, Gray [IEEE COM. 1980] "LBG-Algorithmus"

Beachte:

- Erweiterung auf Mischverteilungen ist offensichtlich
- Für Vektorräume geeignet, kein Äquivalent für die Language-Modellierung bekannt.

Bottom-Up-Verfahren (agglomeratives hierarchisches Clustern)

Start: jede Beobachtung ist ihr eigenes Cluster

Loop: bestimme das Paar von Clustern mit dem geringsten Abstand $(*)$

verschmelze die beiden Cluster

falls (Zahl der Cluster > 1) GOTO Loop

sonst STOP

Komplexität:

- Bei m Clustern sind $\frac{m(m-1)}{2}$ Schritte nötig, um das optimale Paar zu finden.
- Aufsummieren über alle $m = I, I-1, \ldots, 1$

$$\sum_{m=1}^{M} \frac{m(m-1)}{2} \cong \frac{1}{6} N^3$$

kubische Komplexität $\Longrightarrow$ ab $N \cong 10000$ wird der Rechenaufwand kritisch.

verbesserte Variante $(*)$:

Loop: Bestimme das Paar von Clustern, deren Verschmelzung das globale Kriterium am wenigsten verschlechtert.

Standard-Verfahren:

- [Gor81], p.46
- [Spa77], p.170
- [DH73], p.230

Abstandsmaße zwischen Clustern:

gegeben: Cluster A_i mit Vektoren x_i und Cluster A_j mit Vektoren x_j, $||.||$ euklidischer Abstand (nicht quadriert)

- single link

$$d(A_i, A_j) := \min_{i,j} ||x_i - x_j||$$

- complete link

$$d(A_i, A_j) := \max_{i,j} ||x_i - x_j||$$

- group average link

$$d(A_i, A_j) := \frac{1}{N_i \cdot N_j} \sum_i \sum_j ||x_i - x_j||$$

- weighted average link

$$d(A_i, A_j) := \sum_i \sum_j ||x_i - x_j||$$

- centroid link

$$d(A_i, A_j) := ||\mu_i - \mu_j||$$

- median link Mediane $\tilde{\mu}_i, \tilde{\mu}_j$

$$d(A_i, A_j) := ||\tilde{\mu}_i - \tilde{\mu}_j||$$

- sum of squares (Ward-Algorithmus)

$$d(A_i, A_j) := \frac{N_i \cdot N_j}{N_i + N_j} ||\mu_i - \mu_j||^2$$

(konsistent mit Gauß-Modell)

Übungsaufgabe:

Gauß-Modell ansetzen
(aber: im ersten Schritt ist jeder Vektor sein eigenes Cluster, d.h. Varianzen sind nicht berechenbar.
deshalb: im ersten Schritt gepoolte Kovarianzmatrix)

Anmerkung:
In der Literatur sind *effiziente* Umformulierungen zu finden.

Hinweise:

- möglichst vermeiden (oft in Literatur): "Lesen im Kaffeesatz"
- möglichst objektive Kriterien für neue Testdaten definieren:
  - Likelihood
  - Fehlrerate (bei Klassifikationsaufgaben)

Schrittweise optimaler Algorithmus für hierarchisches agglomeratives Clustern

Start: jede Beobachtung ist ihr eigenes Cluster

Loop: bestimme das Paar von Clustern, deren Verschmelzung das globale Kriterium am wenigsten verschlechtert
verschmelze diese beiden Cluster

falls (Zahl der Cluster > 1) GOTO Loop

Anmerkungen:

1. Der Rechenaufwand steigt kaum an (höchstens Faktor 2) im Vergleich zu Standard-Verfahren.
2. Bei jeder Verschmelzung wird das "Optimum" erreicht.
3. Der Algorithmus von Ward ("sum of squares") hat bereits diese Optimalitätseigenschaft (vgl. [DH73], p.260, Aufgabe 19).

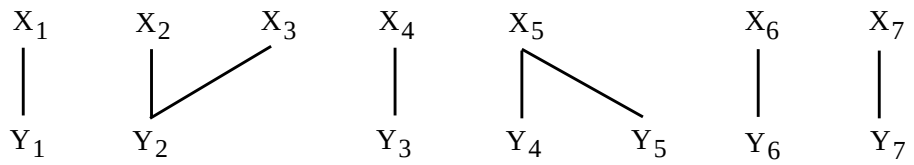
7 Stochastische endliche Automaten

7.1 Motivation und Modell

Wir betrachten eine zeitliche Folge von Beobachtungen:

$$x_1, \ldots, x_t, \ldots, x_T, \qquad x_t \in \mathbb{R}^D, \quad t = \text{Zeit} .$$

Ohne feste Zeitbasis ist die Zuordnung entsprechender Ereignisse (z.B. Laute in der Sprache) schwierig:



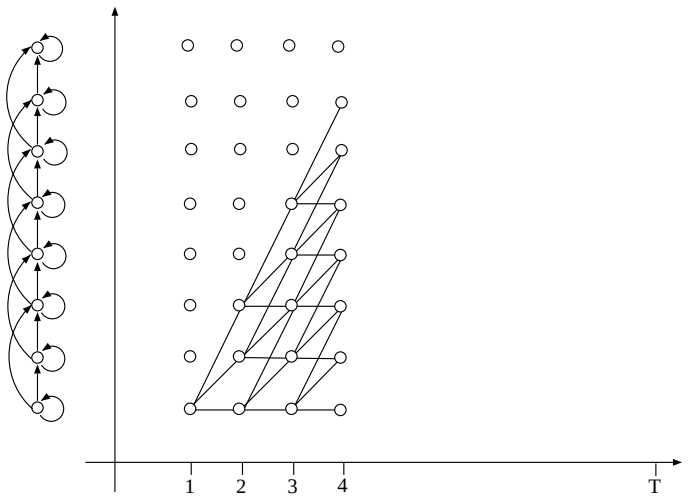
Anders ausgedrückt:
Die Folge als Ganzes läßt sich (praktisch) nicht als Element eines geeigneten Vektorraums beschreiben, da die Zuordnung der Beobachtungen x_t zu den einzelnen Koordinatenachsen schwierig ist.

Eine lineare Skalierung und Normierung der Zeitachse ist hilfreich, aber nicht ausreichend.

Analogie:

- Formale Grammatiken werden eingeführt, um Ketten von diskreten Symbolen kompakt zu beschreiben.
- Stochastische endliche Automaten werden eingeführt, um Folgen von kontinuierlichen Vektoren (oder diskreten *verrauschten* Symbolen) kompakt zu beschreiben.

Stochastischer endlicher Automat
= stochastische reguläre Grammatik
= Hidden Markov Model (HMM)



Abschätzung der Anzahl der möglichen Pfade für das im Bild dargestellte $(0, 1, 2)$-Standard-Modell, das in der Spracherkennung eingesetzt wird:

> Für ein genügend langes Wortmodell, d.h. $S \geq \frac{T}{2}$, ergeben sich bei T Beobachtungsvektoren $x_1, \ldots, x_t, \ldots x_T$ insgesamt 3^{T-1} mögliche Pfade, wenn wir alle erreichbaren $2 \cdot (T-1)$ Endzustände zulassen. Tatsächlich ist jedoch nur ein Endzustand erlaubt. Nimmt man vereinfachend an, daß alle Endzustände über dieselbe Zahl von unterschiedlichen Pfaden erreicht werden können (grobe Approximation!), erhalten wir bei einem erlaubten Endzustand näherungsweise $3^{T-1}/(2 \cdot (T-1))$ mögliche Pfade.

abstrakte (=hidden) Zustände: $s, \sigma = 1, \ldots, S$
Spezifikation eines HMM:

- Transitionen: Angabe von Paaren $\sigma \longrightarrow s$ und deren Wahrscheinlichkeit $p(s|\sigma)$ mit der Bedingung:

$$\sum_s p(s|\sigma) = 1, \qquad \forall \sigma.$$

- Emissionen ($\hat{=}$ Beobachtungen x_t) während einer Transition $\sigma \longrightarrow s$:

  $p(x_t|\sigma, s)$ = "übliches Modell" (z.B. Gauß, Mischverteilung, ...).

- Angabe der Start- und Endzustände.

Beispiele:

1. Sprache:

   x_t = Energieverteilung (über der Frequenzachse), die mittels Fourier-Analyse alle 10 ms berechnet wird.

   HMM's extrem wichtig:

   - extreme nichtlineare Schwankungen der Sprechgeschwindigkeit;
   - Vermeidung lokaler Entscheidungen (z.B. über Identität eines Lautes).

   ähnlich: Signalverläufe über einer Zeitachse
   (z.B. Meßkurven, EKG, EEG, ...)

2. Schrifterkennung:

   Ansatz:

   (a) Zeitachse $\hat{=}$ horizontale Achse

   (b) Zerlegung des Schriftbildes mittels Fensterfunktion

   (c) Merkmalsvektor x_t für jedes Fenster

3. Verarbeitung von "fehlerbehafteten" Symbolketten

   - *Levenshtein*-Distanz: deletions, insertions, substitutions
   - speziell: geschriebene Sprache

====== Abb 1,2,3 ========

7.2 Mathematischer Formalismus

x_1^T zeitliche Folge der Vektoren $x_1, \ldots, x_T$
s_1^T zeitliche Folge der Zustände $s_1, \ldots, s_T$

$$p(x_1^T) = \sum_{[s_1^T]} p(x_1^T, s_1^T) \qquad \text{(Einführung von Zuständen)} \quad (*)$$

$$p(x_1^T, s_1^T) = \prod_{t=1}^{T} p(x_t, s_t | x_1^{t-1}, s_1^{t-1})$$

Modellannahmen:

- Abhängigkeit nur über Zustände (*hidden*):

$$\boxed{p(x_t, s_t | x_1^{t-1}, s_1^{t-1}) = p(x_t, s_t | s_1^{t-1})}$$

- Abhängigkeit nur über den Vorgängerzustand (*first-order Markov*):

$$\boxed{p(x_t, s_t | s_1^{t-1}) = p(x_t, s_t | s_{t-1}) = p(s_t | s_{t-1}) \cdot p(x_t | s_{t-1}, s_t)}$$

($\hat{=}$ Transition Probability $\cdot$ Emission Probability (density))

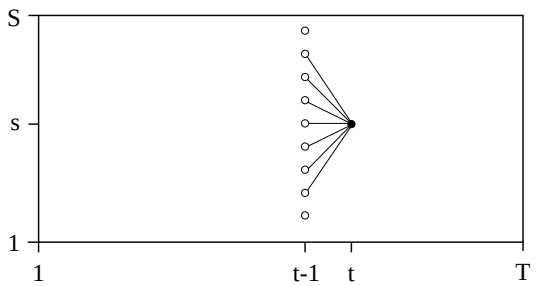
Einsetzen in $(*)$ ergibt:

$$\begin{aligned} p(x_1^T) &= \sum_{[s_1^T]} \prod_{t=1}^{T} [p(s_t | s_{t-1}) \cdot p(x_t | s_{t-1}, s_t)] \\ &= \sum_{[s_1^T]} \prod_{t=1}^{T} p(x_t, s_t | s_{t-1}) \end{aligned}$$

Die direkte Berechnung der Summe in dieser Gleichung ist schwierig, weil wir über ca. $3^{T-1}/(2 \cdot (T-1))$ Pfade zu summieren haben. Es gibt jedoch eine einfache und effiziente Rekursionsformel, die wir im folgenden herleiten (vgl. auch Vorlesung *Algorithmen der Spracherkennung*, WS 94/95):

Definiere die Hilfsgröße $Q(t, s)$ als "Summe über alle Pfade durch (t, s)".

$$Q(t, s) := \sum_{s_1^t : s_t = s} \prod_{\tau=1}^{t} p(x_\tau, s_\tau | s_{\tau-1})$$



Zerlegung: $[\cdots \longrightarrow (s,t)] = [\cdots \longrightarrow (\sigma, t-1)]\ [(\sigma, t-1) \longrightarrow (s,t)]$

$$Q(t,s) \;=\; \sum_{\sigma} [p(x_t, s|\sigma) \cdot \underbrace{\sum_{s_1^{t-1}: s_{t-1}=\sigma} \prod_{\tau=1}^{t-1} p(x_\tau, s_\tau | s_{\tau-1})}_{=Q(t-1,\sigma)}]$$

$$\boxed{Q(t,s) = \sum_{\sigma} p(x_t, s|\sigma) \cdot Q(t-1,\sigma)}$$

Rekursionsgleichung vom Typ *divide and conquer*
(*dynamische Programmierung*)

Der Rechenaufwand ist linear in $T \cdot S$ (statt $3^{T-1}/(2 \cdot (T-1)) \cong$ Anzahl der Pfade im $(0,1,2)$-Standard-Modell).

Die (historische) Bezeichnung dieser Gleichung lautet *Baum(- Welch)-Rekursion.*

• *Viterbi-Algorithmus* (Maximum-Approximation):
Bei dieser Approximation beschränkt man sich auf den besten Pfad durch (t,s):

$$p(x_1^T) \;\cong\; \max_{[s_1^T]} \prod_{t=1}^{T} p(x_t, s_t | s_{t-1}) \ .$$

Dies ist eine gute Approximation der Summe, falls ein Pfad "dominiert".

Definiere in Analogie zur Baum(-Welch)-Rekursion $Q(t,s)$ als:

$$Q(t,s) = \max_{s_1^t: s_t = s} \prod_{\tau=1}^{t} p(x_\tau, s_\tau | s_{\tau-1}) \ .$$

Die Zerlegung ergibt:

$$Q(t,s) \;=\; \max_{\sigma}\,[p(x_t,s|\sigma)\cdot \underbrace{\max_{s_1^{t-1}:s_{t-1}=\sigma}\prod_{\tau=1}^{t-1} p(x_\tau,s_\tau|s_{\tau-1})}_{=Q(\tau-1,\sigma)}]$$

$$\boxed{Q(t,s)=\max_{\sigma}\,[p(x_t,s|\sigma)\cdot Q(t-1,\sigma)]}$$

Rekursionsgleichung der dynamischen Programmierung

Der Rechenaufwand ist wieder linear in $T\cdot S$ (statt $3^{T-1}/(2\cdot(T-1))$).

Für $t=1,\ldots,T$ werden die Einzelentscheidungen abgespeichert. Nach Erreichen des Endzeitpunkts ($t=T$) wird der beste Pfad zurückverfolgt.

Analogie zu *formalen Sprachen*:

Recognizer	$\hat{=}$	Baum-Algorithmus	:	alle parse trees
Parser	$\hat{=}$	Viterbi-Algorithmus	:	bester parse tree

7.3 Einbau in die Bayessche Entscheidungsregel

Jede Klasse k hat einen eigenen Automaten mit Endzustand $S(k)$.
Für eine Vektorfolge x_1^T wird mittels der Rekursionsgleichung (entweder exakt oder DP) berechnet:

$$Q_k(T,S(k)) \qquad \text{für jede Klasse (Automaten) } k=1,\ldots,K$$

Damit gilt nach Definition:

$$p(x_1^T|k) \;=\; Q_k(T,S(k)),$$

so daß die Entscheidungsregel lautet:

$$\begin{aligned} x_1^T \longrightarrow r(x_1^T) \;&=\; \operatorname*{argmax}_k\,\{p(k)\cdot p(x_1^T|k)\} \\ &=\; \operatorname*{argmax}_k\,\{p(k)\cdot Q_k(T,S(k))\} \end{aligned}$$

7.4 Training

Ansatz wie bei Mischverteilungen, d.h. Baum-Algorithmus, EM-Algorithmus

Vektorfolge: x_1^T
$y = s_1^T = \textit{hidden variable}$

$$\begin{aligned} Q(\lambda, \bar{\lambda}) &= \sum_{[s_1^T]} p(s_1^T | x_1^T, \lambda) \cdot \log p(x_1^T, s_1^T | \bar{\lambda}) \\ &= \frac{1}{p(x_1^T|\lambda)} \cdot \sum_{[s_1^T]} p(x_1^T, s_1^T | \lambda) \cdot \log p(x_1^T, s_1^T | \bar{\lambda}) \end{aligned}$$

- Modell: $p(x_1^T, s_1^T | \bar{\lambda}) = \prod_{t=1}^{T} [p(s_t|s_{t-1}) \cdot p(x_t|s_{t-1}, s_t)]$
- Parameter λ: $\alpha(s|\sigma)$, $\sum_s \alpha(s|\sigma) = 1$, $\forall\sigma$ statt $p(s|\sigma)$.

$p(x|\sigma, s; \vartheta) = p(x|s; \vartheta_s)$ z.B. Gauß- oder Mischverteilung mit Parametern ϑ_s

Parametersatz: $\lambda = \{\alpha(s|\sigma), \vartheta_s\}$

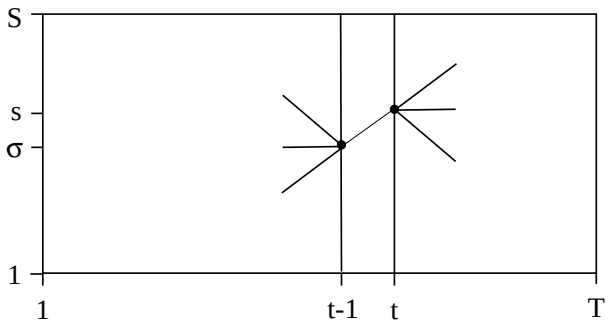
Umgruppieren der Terme ergibt:

$$\begin{aligned} Q(\lambda, \bar{\lambda}) = \frac{1}{p(x_1^T|\lambda)} \cdot \Bigg[& \sum_{s,\sigma} \sum_t p(x_1^T, s_{t-1} = \sigma, s_t = s | \lambda) \cdot \log \bar{\alpha}(s|\sigma) \\ & + \sum_s \sum_t p(x_1^T, s_t = s | \lambda) \cdot \log p(x|s, \bar{\vartheta}_s) \Bigg] \end{aligned}$$

Anmerkung:
Die Startverteilung $\pi(s)$, die man in der Literatur häufig findet, ist weggelassen worden; sie kann einfach durch Hinzufügen eines fiktiven Zustandes s=0 und der Gleichsetzung $\pi(s) := p(s|0)$ modelliert werden.

$$
\begin{aligned}
p(x_1^T|\lambda) &= \sum_{[s_1^T]} p(x_1^T, s_1^T|\lambda) \\
p(x_1^T, s_{t-1}=\sigma, s_t=s|\lambda) &= \sum_{\substack{[s_1^T];\\ s_{t-1}=\sigma; s_t=s}} p(x_1^T, s_1^T|\lambda)
\end{aligned}
$$

"Pfade gehen durch $(t-1,\sigma)$ und (t,s)"



Definiere die Posterior-Hilfsgrößen:

$$
\begin{aligned}
\gamma_t(\sigma, s) &= p(s_{t-1}=\sigma, s_t=s|x_1^T, \lambda) = \frac{p(x_1^T, s_{t-1}=\sigma, s_t=s|\lambda)}{p(x_1^T|\lambda)} \\
\gamma_t(s) &= p(s_t=s|x_1^T, \lambda) = \frac{p(x_1^T, s_t=s|\lambda)}{p(x_1^T|\lambda)}
\end{aligned}
$$

Modell: Gauß-Verteilung $\vartheta_s = [\mu_s, \Sigma_s]$ für jeden Zustand s

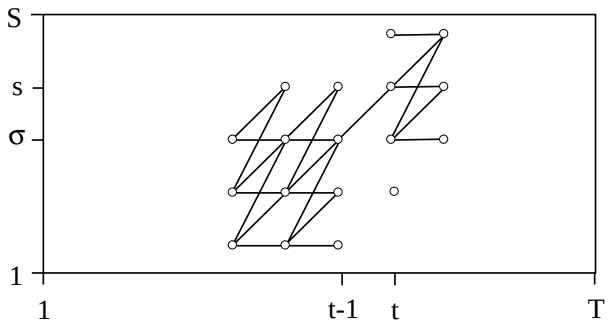
Iterationsformeln (Reestimation):

$$
\begin{aligned}
\bar{\alpha}(s|\sigma) &= \frac{\sum_t \gamma_t(\sigma, s)}{\sum_t \sum_{s'} \gamma_t(\sigma, s')} \\
\bar{\mu}_s &= \frac{\sum_t \gamma_t(s) \cdot x_t}{\sum_t \gamma_t(s)} \\
\bar{\Sigma}_s &= \frac{\sum_t \gamma_t(s) \cdot [x_t - \bar{\mu}_s] \cdot [x_t - \bar{\mu}_s]^T}{\sum_t \gamma_t(s)}
\end{aligned}
$$

Anmerkung:

1. Mischverteilungen: analoges Ergebnis, aber Beweis des EM-Verfahrens aufwendiger (Übungsaufgabe)
2. Notwendig: Erweiterung auf mehrere Vektorfolgen
   $\longrightarrow$ "analoge" Iterationsformeln für eine lange Vektorfolge

Erläuterungen zum Training:



$\gamma_t(\sigma, s)$ "zwingt" Pfade durch σ und s, um deren Beitrag zu isolieren ($\gamma_t(s)$ analog).

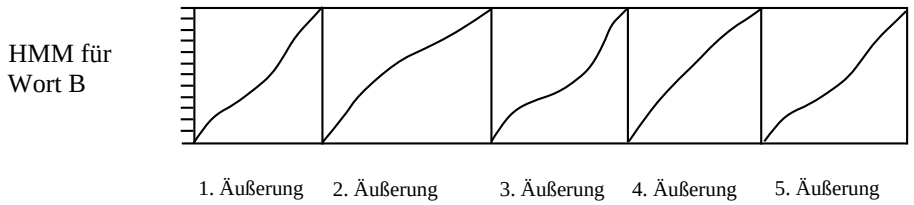
Die Berechnung der $\gamma_t(\sigma, s)$ und $\gamma_t(s)$ erfolgt effizient durch Kombination der Vorwärts- und Rückwärts-Variante der Baum-Rekursion, d.h. den *Forward-Backward-Algorithmus*.
Das Prinzip ist einfach, aber die Implementation muß wegen des CPU- und Speicherbedarfs sowie möglichen Overflows trickreich erfolgen.

Statt dessen werden wir die *Maximum-Approximation* betrachten (heißt auch *Viterbi-Training*), die nur den besten Pfad verwendet:

$$\gamma_t(\sigma, s) = \begin{cases} 1, & \text{bester Pfad durch } \sigma, s \\ 0, & \text{sonst} \end{cases}$$

$$\gamma_t(s) = \begin{cases} 1, & \text{bester Pfad durch s} \\ 0, & \text{sonst} \end{cases}$$

Training mit Maximum-Approximation: (siehe Vorlesung *Algorithmen der Spracherkennung*, WS 94/95)
Trainingssprachproben: Einzelne Äußerung eines Wortes B



Iteratives Trainieren mit den beiden Schritten

1. Zeitanpassung: Bestimmung des besten Pfades
2. Schätzung der Modell-Parameter:
   (a) Sammeln der Beobachtungen für jeden Zustand und Übergang
   (b) Maximum-Likelihood-Schätzung, z.B.
   Gauß: $\hat{\mu}$ = empirischer Mittelwert
   $\hat{\Sigma}$ = empirische Kovarianz

   Transitionswahrscheinlichkeit = relative Häufigkeit

Beachte:

1. Dies ist ein weiteres Beispiel für *decision-directed learning* (wie bei Clusteranalyse).
2. Erweiterung des Trainings auf den Fall von Wortketten anstelle von Einzelwörtern ist einfach möglich.

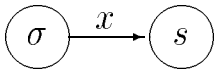
7.5 Stochastische Grammatiken (Ausblick)

Ziel: Übertragung des HMM-Konzepts auf allgemeine kontextfreie Grammatiken.

Notation: $A, B, C, \ldots, Z$ Nonterminals
$x_1, \ldots, x_t, \ldots, x_T$ Terminals (Beobachtungen, diskret oder $\in \mathbb{R}^D$)

- *Stochastische reguläre Grammatik* :

  Die Wahrscheinlichkeit dafür, daß man vom Zustand σ ausgehend das Terminal x beobachtet und in den Zustand s wechselt,

  wird mit $p(x, s|\sigma)$ bezeichnet.

  Dies entspricht dem Produktionsregel-Typ $s \longrightarrow \sigma x$ oder $\sigma \longrightarrow xs$ (je nach Interpretation).

  Aus diesem Regel-Typ kann (bis auf die Anfangszustände) die gesamte reguläre Grammatik aufgebaut werden.

- *Stochastische kontextfreie Grammatik* (Chomsky-Normalform):

Struktur	$A \longrightarrow BC$	$p(BC\|A)$
Beobachtung	$A \longrightarrow x$	$p(x\|A)$

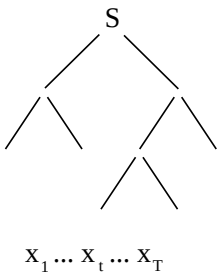
  $p(BC|A)$ entspricht der Wahrscheinlichkeit für die Anwendung der Regel $A \longrightarrow BC$, wenn das Nonterminal A vorgelegen hat.

  Anmerkung:

  Bei deutlich ausgeprägten Strukturen sind nur wenige Wahrscheinlichkeiten $p(BC|A) > 0$.

Betrachte die Kette von Regeln $r_1 \dots r_M$:

$$S \xrightarrow{r_1} \cdots \xrightarrow{r_m} \cdots \xrightarrow{r_M} [x_1 \dots x_t \dots x_T]$$



r: A ⟶ BC
p(BC|A) := q(r)
r: A ⟶ x
p(x|A) := q(r)

Modell: $q(r_m|r_{m-1}) = q(r_m)$ "Reihenfolge beliebig"

$$\boxed{p(x_1^T, r_1^M) = \prod_{m=1}^{M} q(r_m)}$$

$$\begin{aligned} p(x_1^T) &= \sum_{[r_1^M]} p(x_1^T, r_1^M) && \text{"alle Ableitungen" (recognizer)} \\ &\cong \max_{[r_1^M]} p(x_1^T, r_1^M) && \text{"beste Ableitungen" (parser)} \end{aligned}$$

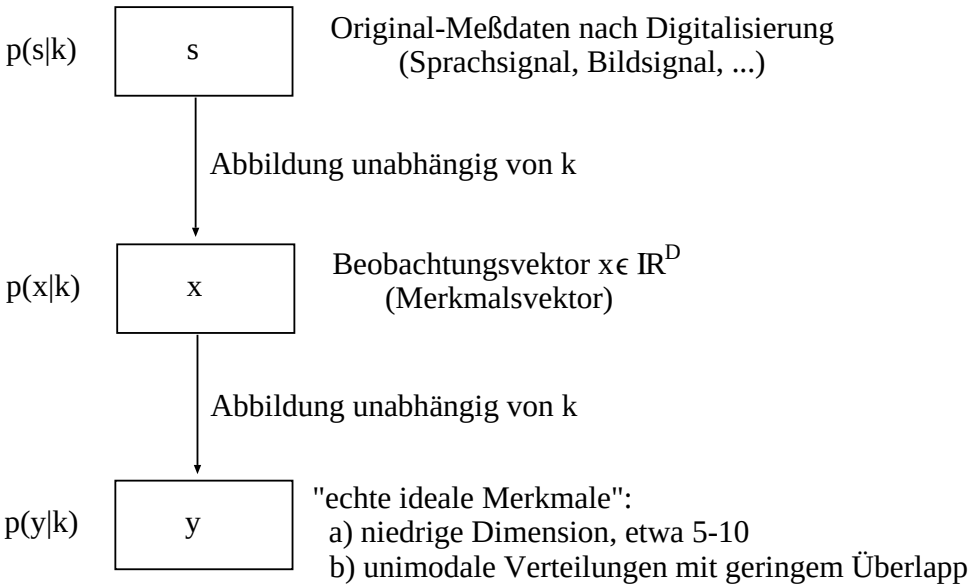
Bemerkungen:

1. Für beide Varianten $(\sum, \max)$ gibt es effiziente Algorithmen, die auf dynamischer Programmierung (oder divide and conquer) beruhen.
2. Auf diese Weise erhält man die stochastische Variante des Cocke-Younger-Kasami- oder des Early-Algorithmus (Bottom-Up bzw. Top-Down-Parser für kontextfreie Grammatiken).
3. Training:
   - Erweiterungen der Reestimations-Gleichungen existieren
   - bisher wenig praktische Erfahrung für reale Probleme
   - offenes Problem: Lernen der Regeln (*grammatische Inferenz*)
4. Hierarchiebildung ist möglich:
   Wortebene: Kombination der Beobachtungen zu Wörtern
   Satzebene: Kombination der Wörter zu Sätzen
   (Beispiel: Einbau von Paritätschecks in die Ziffernfolgenerkennung)

8 Merkmalsextraktion und lineare Abbildungen

8.1 Ideale Merkmalsextraktion

Verschiedene Arten von Daten (k = Klassenindex):



Aussagen:

1. Bei *vollständig und exakt* bekannten Verteilungen

$$p(s|k),\ p(x|k),\ p(y|k)$$

und *invertierbaren* Transformationen $s \longleftrightarrow x \longleftrightarrow y$ gibt es bei Verwendung der Bayesschen Entscheidungsregel keinen Unterschied in der Fehlerrate, höchstens im Rechenaufwand (Originaldaten s sind immer extrem umfangreich).

In der Regel ist mit den Abbildungen $s \longrightarrow x \longrightarrow y$ eine Datenreduktion verbunden, so daß bei gegebenen Verteilungen nichts gewonnen wird hinsichtlich der Reduktion der Fehlerrate.

2. In der Praxis sind die Verteilungen nicht gegeben, so daß die Merkmalsextraktion Vorteile bietet:

   (a) geringere Datenmenge;

   (b) fest dimensionierter Beobachtungsvektor $x, y \in \mathbb{R}^D$;

(c) Abhängigkeiten zwischen den Komponenten von x oder y lassen sich einfacher erfassen (vgl. Problem: Schätzung der Kovarianzmatrix).

3. (Mögliche) Vorgehensweise:

   (a) Vollautomatische Bestimmung der Abbildung $s \longrightarrow x$ (oder $x \longrightarrow y$) erfordert ein gutes Kriterium, z.B. die empirische Fehlerrate (auf den Trainingsdaten).
   $\Longrightarrow$ resultierendes Optimierungsproblem fast immer zu komplex

   (b) Aufgabenabhängige Wahl für die Abbildung $s \longrightarrow x$, die auf problemspezifischem Wissen basiert (z.B. Spektralanalyse für Sprache, "Primitive" für Schrifterkennung, Textur bei Bildern).

   (c) Für den letzten Schritt $x \longrightarrow y$ wird eine lineare Abbildung angesetzt:

$$\begin{aligned} V: \ \mathbb{R}^D &\longrightarrow \mathbb{R}^d, \qquad d < D \\ x &\longmapsto y = Vx \end{aligned}$$

   Die Abbildung V wird mittels Optimalitätskriterien bestimmt, die im folgenden betrachtet werden.

8.2 Lineare Abbildungen

8.2.1 Effekt bei gegebener Gauß-Verteilung

gegeben μ und Σ:

(Beispiel zur Veranschaulichung)
$x \in \mathbb{R}^D$: $\quad p(x|\mu_x, \Sigma_x)$
Sei $V \in \mathbb{R}^{D \times D}$ invertierbar, dann bleiben die "Abstandsverhältnisse" erhalten.

Beweis:

$$\begin{aligned} y = V \cdot x \implies \mu_y &= E\{y\} = V \cdot \mu_x \\ \Sigma_y &= \cdots = V \cdot \Sigma_x \cdot V^T \end{aligned}$$

$$
\begin{aligned}
d_x &\equiv (x-\mu_x)^T\Sigma_x^{-1}(x-\mu_x) \quad \text{(Exponent in } p(x|\mu_x,\Sigma_x)) \\
d_y &\equiv (y-\mu_y)^T\Sigma_y^{-1}(y-\mu_y) \\
&= (x-\mu_x)^T \cdot \underbrace{V^T\Sigma_y^{-1}V}_{\substack{V^T(V^T)^{-1}\Sigma_x^{-1}V^{-1}V \\ =\Sigma_x^{-1}}} \cdot(x-\mu_x)
\end{aligned}
$$

$$\Longrightarrow \quad d_x \equiv d_y$$

8.2.2 Minimierung des Repräsentationsfehlers: Karhunen-Loève-Transformation

([Fuk72], p.226)

andere Namen: KL-Entwicklung, Principal Component Analysis, Hotelling-Transformation [und Hauptachsentransformation, Diagonalisierung]

$x \in \mathbb{R}^D$ [keine Klassenzugehörigkeit angegeben]
Basisvektoren, orthogonal und normiert:
$V = [v_1, \ldots, v_i, \ldots, v_D], \quad v_i \in \mathbb{R}^D, \quad i = 1, \ldots, D$
mit

$$v_i^T v_j = \begin{cases} 1, & \text{i=j} \\ 0, & \text{sonst} \end{cases} \quad .$$

Zerlegung:

$$x = \sum_{i=1}^{D} \alpha_i(x) \cdot v_i, \qquad \text{mit } \alpha_i(x) = v_i^T \cdot x.$$

Ansatz:
Ab $i > d$ ersetze die Koeffizienten $\alpha_i(x)$ durch eine Konstante β_i (unabhängig von x):

$$
\begin{aligned}
x &= \sum_{i=1}^{d} \alpha_i(x) \cdot v_i + \sum_{i=d+1}^{D} \alpha_i(x) \cdot v_i \\
&\Downarrow \\
\hat{x} &= \sum_{i=1}^{d} \alpha_i(x) \cdot v_i + \sum_{i=d+1}^{D} \beta_i \cdot v_i
\end{aligned}
$$

Kriterium: Wähle die Basisvektoren so, daß der Repräsentationsfehler

$$\begin{aligned} F(d) &= E_x\{\|x-\hat{x}\|^2\} \\ &= \int_x ||x-\hat{x}||^2 \cdot p(x)\, dx \\ &= E_x\{\|\sum_{i=d+1}^{D} (\alpha_i(x)-\beta_i)\cdot v_i\|^2\} \end{aligned}$$

minimiert wird.

$$\begin{aligned} F(d) &= E_x\{\sum_{i>d}\sum_{j>d}(\alpha_i(x)-\beta_i)\cdot(\alpha_j(x)-\beta_j)\cdot v_i^T v_j\} \\ &= E_x\{\sum_{i>d}(\alpha_i(x)-\beta_i)^2\} \\ &= \sum_{i>d} E_x\{(\alpha_i(x)-\beta_i)^2\} \end{aligned}$$

Optimale Wahl der β_i (v_i = Mittelwert der $\alpha_i(x)$, $\mu := E_x\{x\}$):

$$\begin{aligned} \beta_i &= v_i^T \cdot E_x\{x\} = v_i^T \cdot \mu \\ \beta_i &= E_x\{\alpha_i(x)\} = E_x\{v_i^T \cdot x\} = v_i^T \cdot E_x\{x\} = v_i^T \cdot \mu \end{aligned}$$

$$\boxed{\beta_i = v_i^T \cdot \mu}$$

Umschreiben von $F(d)$ ergibt:

$$\begin{aligned} F(d) &= \sum_{i>d} E_x\{[v_i^T\cdot(x-\mu)]^2\} \\ &= \sum_{i>d} E_x\{v_i^T\cdot(x-\mu)\cdot(x-\mu)^T\cdot v_i\} \\ &= \sum_{i>d} v_i^T \cdot \underbrace{E_x\{(x-\mu)\cdot(x-\mu)^T\}}_{\Sigma}\cdot v_i \\ &= \sum_{i>d} v_i^T\cdot\Sigma\cdot v_i \end{aligned}$$

Minimierung von $F(d)$ bzgl. $v_i \in \mathbb{R}^D$ mit $\|v_i\|^2 = 1$ ergibt die Eigenwertgleichung:

$$\Sigma\cdot v_i = \lambda_i\cdot v_i, \qquad \lambda_i > 0.$$

(Herleitung: entweder geschickt abschätzen mittels linearer Algebra oder ableiten mit Hilfe des Lagrange-Formalismus)

$$F(d) = \sum_{i>d} \lambda_i$$

Anmerkung:
Dieselbe Herleitung funktioniert auch, wenn der Repräsentationsfehler über eine Stichprobe gemittelt wird (Unterschied: $\lambda_i \geq 0$).

Karhunen-Loève-Transformation:

1. Berechne (empirische) Kovarianzmatrix Σ;
2. Diagonalisiere Σ;
3. Ordne die Eigenvektoren v_i nach den Eigenwerten λ_i;
4. Wähle (*trial-and-error*) die neue Dimension; manchmal ist als Kriterium der relative Fehler hilfreich:

$$E_d = \frac{F(d)}{F(D)} = \frac{\sum\limits_{i>d} \lambda_i}{\sum\limits_{i=1}^{D} \lambda_i} \; .$$

Anmerkungen:

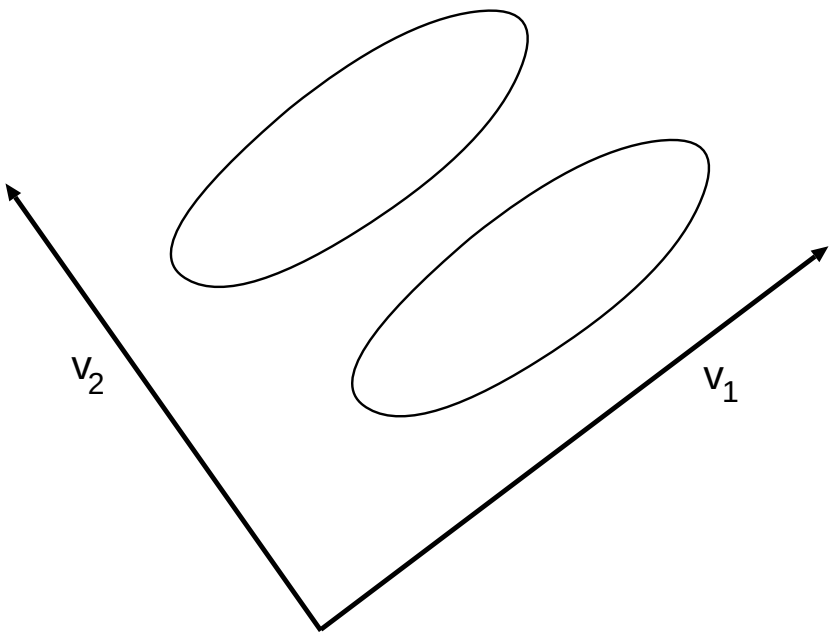
1. Der Begriff "Klasse" oder "Klassentrennbarkeit" geht an keiner Stelle in die Herleitung ein.
2. Daher ist auch keine Optimalität für Klassentrennung zu erwarten (Warnung: Literatur oft unklar bis verkehrt).
3. Die Herleitung erfolgt ohne Annahme über die Verteilungen.
4. Die neuen Merkmale $\alpha_i(x) = v_i^T \cdot x$ haben eine diagonale Kovarianzmatrix, d.h. sie sind "dekorreliert".

5. Falls (im Kontext von Zeitreihenanalyse/Signalanalyse, siehe Vorlesung *Algorithmen der Spracherkennung*, WS 94/95) gilt:

$$\Sigma_{ij} \;=\; R(i-j),$$

d.h. falls Σ_{ij} eine *Toeplitz-Struktur* hat, wird die Diagonalisierung durch Fourier/cos-Transformation erreicht.

Gegenbeispiel (zu Punkt 2):
"Karhunen-Loève-Transformation ist gut für Klassentrennbarkeit."



- v_1 : wichtig für Repräsentationsfehler (Karhunen-Loève-Transformation)
- v_2 : wichtig für Klassentrennung

8.2.3 Lineare Diskriminantenanalyse (LDA; Fisher's LDA)

([DH73], p.118-120)

Gegeben: Trainingsdaten $x \in \mathbb{R}^D$ mit $n = 1, \ldots, N$:

$$\mu \;:=\; \frac{1}{N} \sum_{n=1}^{N} x$$

Für jede Klasse A_k gilt:

$$\mu_k := \frac{1}{n_k} \sum_{x \in A_k} x$$

$$n_k := |A_k| = \text{Anzahl der Beobachtungen der Klasse } k$$

gesucht: lineare Abbildung $x \longrightarrow y = (V^T x)^T = x^T V$, so daß

i) die Abstände innerhalb einer Klasse konstant bleiben;

ii) die Abstände zwischen den Klassenzentren μ_k maximiert werden.

zu i)

$$\begin{aligned} D_W(V) &:= \sum_k \sum_{x \in A_k} \|V^T(x - \mu_k)\|^2 \\ &= \sum_k \sum_{x \in A_k} V^T (x - \mu_k)(x - \mu_k)^T V \\ &= V^T \underbrace{\sum_k \sum_{x \in A_k} (x - \mu_k)(x - \mu_k)^T}_{=:W} V \end{aligned}$$

W : Within-Class Scatter Matrix

zu ii)

$$\begin{aligned} D_B(V) &:= \sum_k n_k \|V^T(\mu_k - \mu)\|^2 \\ &\vdots \\ &= \sum_k n_k \, V^T (\mu_k - \mu)(\mu_k - \mu)^T V \\ &= V^T \underbrace{\sum_k n_k \, (\mu_k - \mu)(\mu_k - \mu)^T}_{=:B} V \end{aligned}$$

B : Between-Class Scatter Matrix

Daraus ergibt sich die *Total Scatter Matrix*

$$T := \sum_x (x-\mu)(x-\mu)^T$$

unabhängig von Klassenzugehörigkeiten.

Dann gilt:

$$\boxed{T = W + B}$$

Optimalitätskriterium:
Wähle eine dimensionsreduzierende Abbildung V, so daß

- $F(V) = \dfrac{det(V^T B V)}{det(V^T W V)} = \ maximum$
- $F(V) = \dfrac{tr(V^T B V)}{tr(V^T W V)} = \ maximum$

Lösung: verallgemeinerte Eigenvektoren

$$B \cdot v_d \ = \ \lambda_d \cdot W \cdot v_d, \quad \lambda_d \in \mathbb{R}, \quad d = 1 \ldots D, \quad v_d \in \mathbb{R}^D.$$

wegen $T = W + B$ folgt:

$$\boxed{T \cdot v_d = (1 + \lambda_d) \cdot W \cdot v_d}$$

a) bestimme die Eigenvektoren v_d
b) sortiere nach Größe der Eigenwerte λ_d

Direkte Herleitung:

Abbildung $V = [v_1, \ldots, v_d, \ldots, v_D]$ mit $v_d \in \mathbb{R}^D$, wähle jeden Spaltenvektor v_d so, daß gilt:

- $v_d^T \cdot B \cdot v_d = \ maximum$
- unter der Nebenbedingung: $v_d^T \cdot W \cdot v_d = \ const$

$$\text{(äquivalent zu: } \frac{v_d^T \cdot B \cdot v_d}{v_d^T \cdot W \cdot v_d} = maximum)$$

Lagrange: $f(v_d, \lambda_d) = v_d^T \cdot B \cdot v_d - \lambda_d(v_d^T \cdot W \cdot v_d - 1)$

Ausrechnen (lineare Algebra, Differenzieren) ergibt:

$$B \cdot v_d = \lambda_d \cdot W \cdot v_d.$$

Bemerkungen:

1. Die Lösung für V ist wegen der Eigenschaften der Funktionen tr und det nicht eindeutig:

   Eine invertierbare Abbildung läßt das Optimum invariant.

2. Ohne Dimensionsreduktion, d.h. $V \in \mathbb{R}^{D \times D}$, gilt:

   $$F(I) = F(V)$$

   In Worten:
   Ohne Dimensionsreduktion ist die LDA sinnlos. Eine reguläre Abbildung im neuen Merkmalsraum läßt das Kriterium $F(V)$ invariant.

3. Die Eigenwerte λ_d sind invariant bzgl. invertierbaren linearen Transformationen. Es gilt ([DH73], p.223):

   $$tr(W^{-1}B) = \sum_d \lambda_d \qquad tr(W^{-1}T) = \sum_d (1 + \lambda_d)$$
   $$det(W^{-1}B) = \prod_d \lambda_d \qquad det(W^{-1}T) = \prod_d (1 + \lambda_d)$$
   $$(= 0 \text{ wegen } 4.)$$

4. Die Between-Class Scatter Matrix B hat den Rang $(K-1)$ gemäß ihrer Definition (vgl. Kovarianzmatrix bei Gauß-Modell), d.h. nur $(K-1)$ Eigenwerte λ_d sind von Null verschieden.

5. Spezialfall: $W = I$ (*Isotropie*)

   Dann gilt: $\{\mu_k - \mu;\ k = 1, \ldots, K\}$

   $\Downarrow$

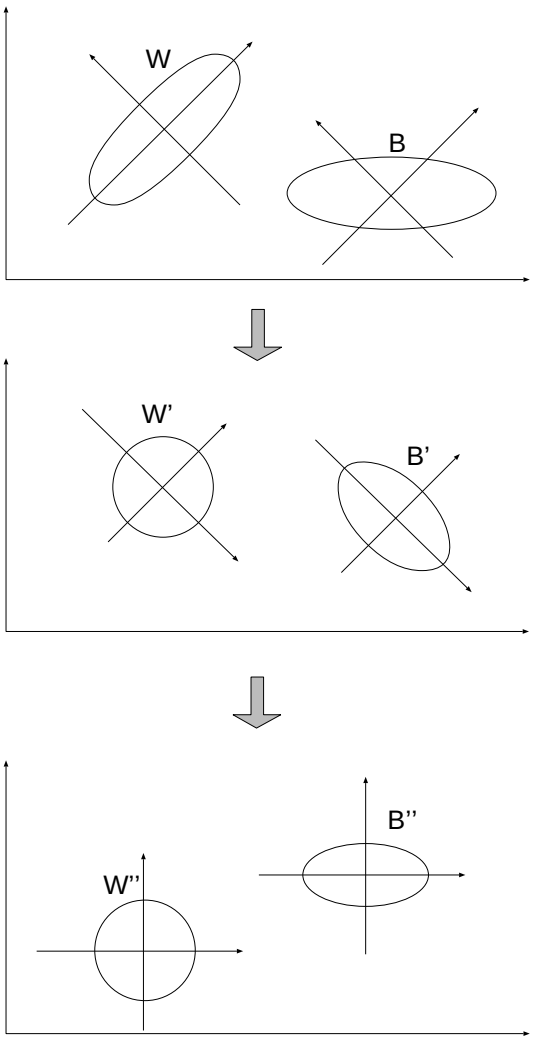
   Gram-Schmidt-Orthogonalisierung

   $\Downarrow$

   Eigenvektoren v_k

6. Die Herleitung der LDA verwendet nur Abstände, kein Gauß-Modell.

Illustration ("simultane Diagonalisierung") ([Fuk72],p.34)



8.3 Lineare Klassifikatoren/Diskriminanten

(vgl. Kapitel 4.2)

Die Anzahl der Klassen K ist beliebig (d.h. nicht nur 2 Klassen).

Erinnerung: Gauß-Modell mit $\Sigma_k = \Sigma$ führt auf lineare Grenzflächen.

Vorgehensweise (anders als typischerweise in der Literatur):

1. allgemeiner Fall $K > 2$;
2. keine Annahmen über Separabilität (Klassentrennbarkeit);
3. statt Betrachtung von Klassenpaaren ein einziges Kriterium ([DH73], p.151,177).

Modell:
Entscheidungsregel:

$$r(x) \;:=\; \operatorname*{argmax}_k \{g(x,k)\}$$

Diskriminante:

$$g(x,k) \;=\; v_k^T \cdot x, \qquad x, v_k \in \mathbb{R}^D.$$

Training: (ideale Werte)

$$\begin{aligned} v_k^T x &\stackrel{!}{=} 1\ , \qquad \text{falls } x \text{ aus Klasse } k, \\ v_k^T x &\stackrel{!}{=} 0\ , \qquad \text{falls } x \text{ nicht aus Klasse } k\ . \end{aligned}$$

Im allgemeinen sind die idealen Zielvorgaben nur näherungsweise zu realisieren. Daher wählt man in der Regel das Fehlerquadratkriterium:

$$F(v_1 \ldots v_k) = \sum_n \sum_c [v_c^T x_n - t(c,k_n)]^2,$$

mit Trainingsdaten (x_n, k_n) und der Kronecker-Funktion t:

$$t(c,k) = \begin{cases} 1, & c = k \\ 0, & c \neq k \end{cases} \quad .$$

Lösung:

F ist quadratisch in v_k; daher ergibt sich ein lineares Gleichungssystem für v_k mit $k = 1, \ldots, K$. Die Struktur dieses Gleichungssystems ähnelt Normalgleichungen und Pseudoinversen.

verallgemeinerte Merkmale:

1. $g(x,k) = b_k + v_k^T x$
2. $g(x,k) = b_k + v_k^T x + x^T W_k x, \qquad W_k \in \mathbb{R}^{D \times D}$
3. $g(x,k) = v_k^T f(x)$

Bezeichnung: *Polynomiale Diskriminantenfunktion*, *Regressionsklassifikator*

9 Literatur

R.O. Duda, P.E. Hart: *Pattern Classification and Scene Analysis.* J. Wiley, New York, 1973.

P.A. Devijver, J. Kittler: *Pattern Recognition: A Statistical Approach.* Prentice-Hall, Englewood Cliffs, NJ, 1982.

K. Fukunaga: *Introduction to Statistical Pattern Recognition.* Academic Press, New York, 1990.

S.Y. Kung: *Digital Neural Networks.* Prentice-Hall, Englewood Cliffs, NJ, 1993.

R. Schalkoff: *Pattern Recognition: Statistical, Structural and Neural Approaches.* J. Wiley, New York, 1992.

H. Niemann: *Klassifikation von Mustern.* Springer, Berlin, 1983.

speziell: HMM

L.R. Rabiner, B.H. Juang: *Fundamentals of Speech Recognition.* Prentice-Hall, Englewood Cliffs, NJ, 1993.

X.D. Huang, Y. Ariki, M.A. Jack: *Hidden Markov Models for Speech Recognition.* Edinburgh Univ. Press, 1990.

speziell: Stochastik

G. Casella, R.L. Berger: *Statistical Inference.* Wadsworth&Brooks/Cole, Pacific Grove, CA, 1990.

U. Krengel: *Einführung in die Wahrscheinlichkeitstheorie und Statistik.* Vieweg, 1988.

Literatur

[CB90] G. Casella and R.L. Berger. *Statistical Inference.* Wadsworth&Brooks/Cole, Pacific Grove, CA, 1990.

[DH73] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis.* J. Wiley, New York, NY, 1973.

[DK82] P.A. Devijver and J. Kittler. *Pattern Recognition: A Statistical Approach.* Prentice-Hall, Englewood Cliffs, NJ, 1982.

[Fuk90] J. Fukunaga. *Introduction to Statistical Pattern Recognition.* Academic Press, New York, NY, 2nd edition, 1990.

[Gor81] A. D. Gordon. *Classification.* Chapman and Hall, London, 1981.

[HAJ90] X.D. Huang, Y. Ariki, and M.A. Jack. *Hidden Markov Models for Speech Recognition.* Edinburgh Univ. Press, 1990.

[Kee65] D. G. Keehn. A note on learning for gaussian properties. In *IEEE Trans. on Information Theory, IT-11*, pages 126–132, 1965.

[Kre88] U. Krengel. *Einführung in die Wahrscheinlichkeitstheorie und Statistik.* Vieweg, 1988.

[Kun93] S.Y. Kung. *Digital Neural Networks.* Prentice-Hall, Englewood Cliffs, NJ, 1993.

[Lip87] R. Lippmann. *An Introduction to Computing with Neural Nets.* IEEE ASSP Magazine, April 1987.

[Nie83] H. Niemann. *Klassifikation von Mustern.* Springer, Berlin, 1983.

[RJ93] L.R. Rabiner and B.H. Juang. *Fundamentals of Speech Recognition.* Prentice-Hall Signal Processing Series. Prentice Hall, Englewood Cliffs, NJ, 1993.

[Sch92] R. Schalkoff. *Pattern Recognition: Statistical, Structural and Neural Approaches.* J.Wiley, New York, NY, 1992.

[Spa77] H. Spath. *Clusteranalyse-Algorithmen.* 2. Auflage, R. Oldenbourg Verlag, München, 1977.