Analyse von Steroiddaten mit Methoden des überwachten und unüberwachten Lernens

Diplomarbeit im Fachgebiet *Informatik*
der Technischen Universität Berlin
bei Prof. Manfred Opper

vorgelegt von
Sevil Üretmen
`Matrikelnummer 184069`

27. Juli 2010

Die selbständige und eigenhändige Anfertigung versichere ich an Eides statt.

Berlin, den 27. Juli 2010

(Sevil Üretmen)

Danksagung

Ich möchte mich für die moralische und seelische Unterstützung durch meinen Mann, Ayhan Üretmen bedanken. Ich danke zudem meinen Schwestern Helin und Canan, die oftmals auf meine Tochter Nisa aufpassen mussten. Nicht zuletzt bedanke ich mich bei Prof. Manfred Opper und Andreas Ruttor für die Betreuung bei diesem interessanten Thema.

Inhaltsverzeichnis

Einleitung

Jede Reise beginnt mit einem ersten Schritt
- chin. Sprichwort

Bestimmte menschliche Erkrankungen sind auf hormonelle Störungen zurückzuführen. Sogenannte Steroide (chemische Verbindungen), welche beim Ab- und Umbau von Hormonen im menschlichen Körper beteiligt sind, können mit Hilfe von Massenspektrometern im Urin gemessen werden. Hierbei ist der medizinische Zusammenhang zwischen gemessenen Daten und einer eindeutigen Zuordnung zu entsprechendem Krankheitsbild noch ungenügend erforscht.

In folgender Arbeit soll mit Hilfe von Methoden des maschinellen Lernens eine Datenanalyse von Steroiddaten durchgeführt werden. Bestimmte Methoden aus dem unüberwachten Lernen ermöglichen ggf. die Ermittlung von Strukturen im Datensatz, ohne hierbei die tatsächliche Zugehörigkeit einzelner Datenpunkte zum jeweiligen Krankheitsbild kennen zu müssen. Andere Methoden aus dem Bereich des überwachten Lernens verwenden das Wissen um die tatsächliche Zugehörigkeit von Daten zu einzelnen Klassen zur Bildung eines sog. Klassifikators, welcher anschliessend zur Klassifikation von neu präsentierten Datenpunkten verwendet werden kann.

Neuere Forschungen aus dem Bereich des maschinellen Lernens, insbesondere zu *kernel machines*, ermöglichen die Verwendung von sog. stochastischen Prozessen. Gauß Prozesse (GP), als Spezialfälle dieser, bieten innerhalb des überwachten Lernens die besondere Möglichkeit, die Klassifikationentscheide anhand von Wahrscheinlichkeiten zu bewerten. Ferner ist durch Anwendung spezieller Formen von GP es möglich, in Bezug auf die Messmerkmale eines Datensatzes eine Rangfolge für diese anzugeben und so den Einfluss von bestimmten Merkmalen auf entsprechende Klassen auszudrücken.

Die vorliegende Arbeit ist wie folgt gegliedert: In Kapitel 1 werden zunächst die aus dem Bereich der Stochastik stammenden wichtigsten Grundbegriffe und Kenngrössen vorgestellt. Zudem wird der Wahrscheinlichkeitsbegriff als solches behandelt und diesbezüglich wichtige Rechenregeln vorgestellt. Anschliessend werden in Kapitel 2 Methoden aus dem Bereich des unüberwachten Lernens für die Strukturerkennung behandelt. Hauptsächlich

beinhalten diese Methoden zur *Hauptkomponentenanalyse* und sog. *Mixture Modelle*. Kapitel 3 behandelt Methoden zum überwachten Lernen. Hierbei wird allgemein auf Problematiken zur sog. Modelwahl und folgend auf Regression und Klassifikation mit Hilfe von Gauß Prozessen eingegangen. Im spezielleren werden zu Gauß Prozessen Besonderheiten zur Extraktion von signifikanten Merkmalen aufgezeigt. Die Verwendung von Gaussian Mixture Modelle (GMM) zur Klassifikation wird ebenfalls vorgestellt. Im Anschluss werden gängige aber auch eigens entwickelte Verfahren zur Krossvalidierung erklärt. Im letzten Kapitel 4 dieser Arbeit werden die vorgestellten Methoden an einem Praxisbeispiel zu Daten aus dem medizinischen Bereich angewendet und bewertet.

Kapitel 1

Präliminarien

Körper und Stimme leiht die
Schrift dem stummen Gedanken ...
- Friedrich Schiller

1.1 Grundbegriffe

Das folgende Kapitel gibt einen kurzen Überblick über die in dieser Arbeit verwendeten und aus dem mathematischen Teilgebiet der *Stochastik* stammenden wichtigsten Grundbegriffe und Definitionen. Somit finden sich daher sowohl aus der Statistik als auch aus der Wahrscheinlichkeitsrechnung stammende Definitionen. Da im weiteren Verlauf dieser Arbeit einige Begriffe wiederholt in anderem Kontext auftauchen, ist dieses Kapitel der näheren Betrachtung dieser gewidmet.

1.1.1 Kenngrössen aus der deskriptiven Statistik

Als wichtiger Zweig der Statistik, befasst sich die *deskriptive Statistik* damit, eine vorliegende Datenmenge durch bestimmte charakteristische Kenngrössen zu beschreiben. Diese Datenmengen sind häufig Stichproben aus einer großen Population. Zur Beschreibung dieser Stichproben (oder einfacher zum Vergleich von verschiedenen Stichproben) macht es besonders bei großen Datenmengen Sinn, diese durch wenige charakteristische Merkmale zu beschreiben. Einige dieser wichtigen Kenngrössen sollen im folgenden vorgestellt werden.

Die folgenden Definitionen sollen zunächst beispielhaft an folgender Zahlenmenge

$$X = \{12, 3, 7, 1, 18, 2, 41\}$$

als Stichprobenmenge erläutert werden.

Definition arithmetisches Mittel 1.1. *Das arithmetische Mittel einer endlichen Zahlenmenge X, bezeichnet als $\overline{X}$, ist definiert als*

$$\overline{X} = \frac{\sum_{i=1}^{n} X_i}{n}$$

wobei $n = |X|$ die Kardinalität von X ist[1].

Für die beispielhaft angegebene Stichprobe X ist somit das arithmetische Mittel

$$\overline{X} = \frac{12+3+7+1+18+2+41}{7} = 12$$

Das arithmetische Mittel aus obigem Beispiel umschreibt jedoch auch jede andere beliebige Stichprobe (z.B. $X' = \{4, 20\}$). Daher gibt es eine weitere wichtige Kenngröße, die *Standardabweichung*, welche als Streumaß die "Ausdehnung" der Daten um einen gegebenen Mittelwert beschreibt:

Definition Standardabweichung 1.2. *Die Standardabweichung einer endlichen Zahlenmenge X um ihren Mittelwert $\overline{X}$ ist definiert als*

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n} (X_i - \overline{X})^2}{(n-1)}}$$

Für das obige Beispiel wäre die Standardabweichung von X

$$\sigma = \sqrt{\frac{(12-12)^2+(3-12)^2+(7-12)^2+(1-12)^2+(18-12)^2+(2-12)^2+(41-12)^2}{6}} = \sqrt{200, \overline{66}} \approx 14, 16$$

(Zum Vergleich ist die Standardabweichung für oben angegebenes X' mit $\sigma' = \sqrt{\frac{(4-12)^2+(20-12)^2}{1}} = \sqrt{128} \approx 11, 31$).

Eine weitere Kenngröße für die "Ausdehnung" der Daten um einen Mittelwert ist die *Varianz*. Sie berechnet sich aus dem Quadrat der bereits vorgestellten Standardabweichung:

Definition Varianz 1.3. *Die Varianz einer endlichen Zahlenmenge X um ihren Mittelwert $\overline{X}$ ist definiert als*

$$\sigma^2 = \frac{\sum_{i=1}^{n} (X_i - \overline{X})^2}{(n-1)}$$

[1] An dieser Stelle sei angemerkt dass in der Literatur die Notation $|M|$ sowohl für Determinanten einer Matrix M verwendet wird, aber auch für die Anzahl von Elementen einer endlichen Menge M

X: Körpergröße in Meter	Y: Körpergewicht in Kilo
1.78	85
1.55	50
1.90	90
1.20	18
1.60	70
1.70	55

Tabelle 1.1: Beispielhafte zweidimensionale Stichprobe D in tabellarischer Form

Die bisher beispielhaft vorgestellte Stichprobe X war lediglich eindimensional (in der Praxis könnte X z.B. eine Zahlenreihe von Klausurnoten einer bestimmten Menge von Studenten sein). Jedoch gibt es für den in der Praxis häufigeren Fall von *mehrdimensionalen* Daten eine weitere wichtige Kenngröße, die *Kovarianz*. Die Bedeutung der Kovarianz soll wieder an einem weiteren Beispiel für eine (diesmal mehrdimensionale) Datenmenge D näher erläutert werden:
In diesem Beispiel (siehe Tabelle 1.1) enthält ein Stichproben-Datensatz D zweidimensionale Messdaten: die erste Dimension X sei *Körpergröße* und die Dimension Y das *Körpergewicht* eines Menschen (das Datum des k'ten Menschen aus D ist somit gegeben durch ein Tupel (x_k, y_k)).

Für diese gegebene Datenmenge lassen sich ebenfalls die bisher vorgestellten Kenngrößen Mittelwert, Standardabweichung und Varianz ermitteln, jedoch unabhängig für jede einzelne Dimension (jeweils Körpergröße und Gewicht). Als ein weiteres Mass für die Streuung um einen gegebenen Mittelwert kann nun die *Kovarianz* verwendet werden, um die Streuungs-Eigenschaften der einzelnen Datendimensionen *in Relation zueinander* zu ermitteln. Diese Kenngröße enthält dann Informationen inwiefern sich Daten einer bestimmten Dimension in Relation zu einer anderen Dimension um einen gegebenen Mittelwert ausdehnen. Die Kovarianz drückt somit Abhängigkeiten von Datendimensionen untereinander aus.
Ein Kovarianz-Wert ist immer für zwei Dimensionen angegeben. Handelt es sich bei den beiden betrachteten Dimensionen um ein und die gleiche Dimension, sind Kovarianz- und Varianzwert identisch. Die Varianz (diesmal in funktionaler Notation als Funktion der Daten-Dimension X) kann nämlich auch notiert werden als:

$$var(X) = \frac{\sum_{i=1}^{n} (X_i - \overline{X})(X_i - \overline{X})}{(n-1)}$$

In Anlehnung hierran ist die Definition der Kovarianz wie folgt:

Definition Kovarianz 1.4. *Die Kovarianz zweier Datendimensionen X und Y um die Mittelwerte $\overline{X}$ und $\overline{Y}$ ist definiert als*

$$cov(X, Y) = \frac{\sum_{i=1}^{n} (X_i - \overline{X})(Y_i - \overline{Y})}{(n-1)}$$

An dieser Stelle soll noch folgende kompakte Notierungsform für Kovarianzen in Form einer *Kovarianzmatrix* Σ vorgestellt werden:

Definition Kovarianzmatrix 1.5. *Für eine d-dimensionale Stichprobenmenge D wird die quadratische und symmetrische*[2] *Matrix* Σ *der Größe* $(d \times d)$ *als Kovarianzmatrix bezeichnet mit*

$$\Sigma_{i,j} = cov(i, j)$$

mit $i, j \in \{1, .., d\}$.

Für die in Tabelle 1.1 beispielhafte 2-dimensionale Datenmenge D ist die entsprechende Kovarianzmatrix Σ:

$$\Sigma = \begin{pmatrix} 0.0584 & 6.0253 \\ 6.0253 & 700.6667 \end{pmatrix}$$

Interpretation:

Die Diagonal-Elemente $\Sigma_{i,i}$ enthalten, wie bereits erwähnt, die Varianzen der jeweils i'ten Dimension. Die Kovarianzwerte $\Sigma_{i,j}$ für zwei Dimensionen i, j mit $i \neq j$ drücken aus, inwiefern ein linearer Zusammenhang der Werte der beiden Dimensionen i und j besteht. Jedoch spielt hierbei weniger die Größe eines Kovarianzwertes eine Rolle, als sein Vorzeichen:

- Die Kovarianz ist positiv, wenn die Werte der Dimensionen i und j tendenziell einen gleichsinnigen linearen Zusammenhang besitzen, d.h. hohe Werte von i gehen mit hohen Werten von j einher und niedrige mit niedrigen.
- Die Kovarianz ist hingegen negativ, wenn i und j einen gegensinnigen linearen Zusammenhang aufweisen, d. h. hohe Werte der einen Dimension gehen mit niedrigen Werten der anderen einher.
- Ist das Ergebnis 0, so besteht kein Zusammenhang[3] zwischen den beiden Dimensionen i und j.

D.h. die Kovarianzwerte geben zwar eine Art Richtungsbeziehung der Werte zweier verschiedener Dimensionen an, jedoch kann über die tatsächliche Stärke dieses Zusammenhangs keine wirkliche Aussage getroffen werden. Der Grund hierfür ist die Abhängigkeit des Kovarianzwertes von den Maßeinheiten der jeweiligen betrachteten Dimensionen:

Würde die Körpergröße in obigem Beispiel nicht in Metern sondern in Zentimetern vorliegen, so wäre der Kovarianzwert zwischen Körpergröße und

[2] Eine quadratische Matrix M hat genausoviele Zeilen wie Spalten und ist symmetrisch, wenn gilt $M_{i,j} = M_{j,i}$

[3] zumindest kein linearer

Körpergewicht zehnmal so hoch. Daher ist die Kovarianz in dieser Form als Maßzahl für die Stärke eines Zusammenhangs weniger anschaulich und verwendbar. Um Zusammenhänge vergleichbar zu machen, müsste die Kovarianz noch *normiert* werden. In der Literatur wird dieser normierte Wert als *Korrelationskoeffizient* bezeichnet und errechnet sich aus

$$\varsigma(i,j) = \frac{cov(i,j)}{\sqrt{var(i)}\sqrt{var(j)}}$$

Dieser kann Werte aus dem Intervall $[-1,1]$ annehmen. Bei einem Wert von +1/-1 besteht ein vollständig positiver/negativer linearer Zusammenhang zwischen den beiden betrachteten Merkmalen i und j. Bei einem Wert von 0 bestünde überhaupt kein linearer Zusammenhang zwischen i und j.

1.1.2 Diskrete und kontinuierliche Wahrscheinlichkeiten

Es gibt zwei mögliche Interpretationen des Wahrscheinlichkeitsbegriffs. Die erste ist die klassische Sichtweise (*objectiv/frequentist-view*), in welcher die Wahrscheinlichkeit eines Ereignisses als dessen erwartete Häufigkeit des Eintretens aufgefasst wird [ChrBish, S.21]. Für Zufallsexperimente mit wiederholbaren Ereignissen macht diese Sichtweise Sinn. Jedoch gibt es eine zweite, allgemeinere: die *bayessche* (oder *subjektive*) Sichtweise des Wahrscheinlichkeitsbegriffes. Hier ist Wahrscheinlichkeit ein Mass von Unsicherheit.

Zwei Beispiele für obige Sichtweisen sind z.B. das Ereignis des Münzwurfes, welches zwei Resultate haben kann: Kopf oder Zahl. Die Wahrscheinlichkeit, dass am Ende diesen Jahrhunderts das arktische Eis geschmolzen sein wird, wäre ein weiteres.

Diese Auffassungen von Wahrscheinlichkeit zusammen mit bestimmten Rechenregeln bilden eine wichtige Grundlage für wissenschaftliches Schlussfolgern und kommt in der wissenschaftlichen Praxis bei der Beschreibung von Systemen mit Zufallsprozessen häufig zur Anwendung. Das folgende Kapitel stellt grundlegende Rechenregeln aus der Wahrscheinlichkeitsrechnung vor.

Die einfachste Form von Wahrscheinlichkeiten sind *diskrete Wahrscheinlichkeiten*. Hierbei nimmt X als Zufallsvariable einer diskreten Wahrscheinlichkeitsfunktion P verschiedene Resultate $x_1, x_2, x_3...$ an.

Beispiel(e):

Stellt X also die Aussage dar: *Es wird morgen regnen*, so ist

$$x_i \in \{wahr, falsch\}.$$

Stellt X den Münzwurf dar, so sind die möglichen Resultate

$$x_i \in \{Kopf, Zahl\}.$$

Definition Wahrscheinlichkeit 1.6. *Die Wahrscheinlichkeit, dass X als Zufallsvariable einen bestimmten Wert x_i annimmt, wird als $p(x_i)$ notiert, wobei*

$$p(x_i) \in [0, 1],$$

Für das sichere nicht-Eintreten bwz. Eintreten von $X = x_i$ gilt $p(x_i) = 0$ bzw. $p(x_i) = 1$. Je kleiner/größer der Wert von $P(x_i)$, umso unwahrscheinlicher/wahrscheinlicher ist das Eintreten des Resultats $X = x_i$.

Mit der Annahme, dass X zu einem Zeitpunkt jeweils nur einen Wert x_i annehmen kann (gegenseitiger Ausschluss), ist die Wahrscheinlichkeit für

$$p(X = x_1 \text{ oder } X = x_2) = p(x_1) + p(x_2).$$

Da X stets irgendeinen der möglichen Werte x_i annimmt, gilt ferner:

$$\sum_{\forall x_i} p(x_i) = 1 \tag{1.1}$$

In diesem Fall ist die Wahrscheinlichkeit *normalisiert*.

In der Praxis ist man häufig an kontinuierlichen Wahrscheinlichkeitsvariablen interessiert. In diesem Fall belegt die Zufallsvariable X kontinuierliche Werte x. Deshalb wird die Wahrscheinlichkeitsfunktion p als Funktion einer kontinuierlichen Variable x notiert als $p(x)$.

Beispiel:

Ein Bauer ist an der Wahrscheinlichkeit einer gegebenen Niederschlagsmenge innerhalb einer Saison interessiert

Die Interpretation von $p(x)$ ist nun (anders als im diskreten Fall) *nicht* mehr die Wahrscheinlichkeit, dass X den Wert x annimmt, da die Wahrscheinlichkeit hierfür fast immer 0 ist. Ferner ist es nicht möglich eine unendliche Anzahl von Wahrscheinlichkeiten $\neq 0$ zu haben, da diese sonst nicht normalisiert werden könnten. Die Wahrscheinlichkeit $p(x)$ ist im kontinuierlichen Fall mit $p(x)dx$ definiert als die Wahrscheinlichkeit, dass X zwischen x und $x + dx$ liegt[4]. Somit ändert sich für den kontinuierlichen Fall die Gleichung 1.1 zu:

$$\int_{\forall x} p(x)dx = 1 \tag{1.2}$$

[4]Wobei dx entsprechend klein ist

Bevor nun im nächsten Abschnitt auf die Kombination von mehreren Zufallsvariablen eingegangen wird, sei noch kurz auf den Begriff des Erwartungswertes einer Zufallsgröße eingegangen:

Eine sog. *Wahrscheinlichkeitsverteilung* ist eine Vorschrift, die angibt, wie sich Wahrscheinlichkeiten auf einzelne Zufallsergebnisse verteilen. Eine wichtige Instanz dieser ist die sog. *Gauß-Verteilung*, welche im Unterkapitel 1.1.5 vorgestellt wird. Für ein Zufallsexperiment mit der Zufallsgröße x ist der Erwartungswert $\mathbb{E}(x)$ jener Wert, der sich bei oftmaligem wiederholen des Zufallsexperiments als Mittelwert für x ergibt. Der Erwartungswert bestimmt die Lage von Wahrscheinlichkeitsverteilungen und kann mit dem bereits vorgestellten arithmetischen Mittel einer Häufigkeitsverteilung verglichen werden.

Der Erwartungswert $\mathbb{E}(x)$ einer diskreten Zufallsvariablen x ist gegeben durch die Summe der Produkte über alle n möglichen Werte von x mit ihren zugehörigen Wahrscheinlichkeitswerten $p(x)$:

$$\mathbb{E}(x) = \sum_{i}^{n} x_i p(x_i) = \sum_{i}^{n} x_i P(X = x_i) \tag{1.3}$$

Bei einer stetigen Zufallsvariable x wird der Erwartungswert über eine sog. *Wahrscheinlichkeitsdichtefunktion* $f(x)$ definiert, welche allgemein als Hilfsmittel zur Berechnung einer Wahrscheinlichkeit, dass eine stetige Zufallsgröße x zwischen zwei reellen Werten a und b liegt, verstanden werden kann: Die Modellierung eines einfachen Zufallsprozesses durch die konkrete Angabe von Wahrscheinlichkeitsräumen kann durch eine Menge Ω von Elementarereignissen und einem Wahrscheinlichkeitsmaß $P(A)$ für Teilmengen A von Ω geschehen[5]. Für endliche viele Elemente ω aus Ω mit den Wahrscheinlichkeiten $q(\omega)$ sei nun ein Wahrscheinlichkeitsmaß

$$P(A) = \sum_{\omega \in A} q(\omega) \tag{1.4}$$

mit $\forall A \subseteq \Omega$.

Eine reelle Zufallsvariable X kann für den Fall $|\Omega| = \infty$ als Abbildung $X : \Omega \to \mathbb{R}$ interpretiert werden. Eine Abbildung $f_X : \mathbb{R} \to \mathbb{R}$ wird als *Wahrscheinlichkeitsdichte* von X bezeichnet falls gilt:

$$P(a < X < b) = \int_a^b f_X(t)dt \tag{1.5}$$

Insbesondere gilt $\int_{-\infty}^{\infty} f(x)dx = 1$.

Der Erwartungswert $\mathbb{E}(x)$ einer stetigen Zufallsvariablen x wird nun definiert als

$$\mathbb{E}(x) = \int_{-\infty}^{\infty} x f(x)dx \tag{1.6}$$

[5] $P(X)$ ist hier wieder ein Wahrscheinlichkeitsmaß mit $p(x) \in [0, 1]$

1.1.3 Kombination von mehreren Zufallsvariablen

Seien für zwei Zufallsvariablen X und Y aus einem Zufallsexperiment Z mit N Versuchen folgende Punkte definiert:

- x_i sind die möglichen Werte, die X annehmen kann, wobei $i = 1..M$
- y_j sind die möglichen Werte, die Y annehmen kann, wobei $j = 1..L$
- n_{ij} ist die Gesamtanzahl an Resultaten in denen $X = x_i$ **und** $Y = y_j$
- c_i ist die Anzahl an Resultaten in denen $X = x_i$ (unabhängig von Y)
- r_j ist die Anzahl an Resultaten in denen $Y = y_i$ (unabhängig von X)

Definition vereinigende Wahrscheinlichkeit 1.7. *Die vereinigende Wahrscheinlichkeit $p(X = x_i, Y = y_j)$ für die Zufallsvariablen X und Y ist die Wahrscheinlichkeit dass $X = x_i$ **und** $Y = y_j$ und entspricht*

$$p(X = x_i, Y = y_j) = \frac{n_{ij}}{N} \tag{1.7}$$

Hierbei ist die Wahrscheinlichkeit $p(X = x_i) = \frac{c_i}{N}$ und $p(Y = y_j) = \frac{r_i}{N}$. Ausserdem folgt aus der Defintion für c_i und Defintion 1.7 der Zusammenhang

$$c_i = \sum_j n_{ij} \tag{1.8}$$

Hieraus leitet sich nun die folgende **Summenregel** der Wahrscheinlichkeitsrechnung ab

$$p(X = x_i) = \sum_j^L p(X = x_i, Y = y_j) \tag{1.9}$$

Die Wahrscheinlichkeit $p(X = x_i)$ wird in diesem Fall auch häufig als *marginale Wahrscheinlichkeit* bezeichnet.

Definition bedingte Wahrscheinlichkeit 1.8. *Die bedingte Wahrscheinlichkeit $p(X = x_i|Y = y_j)$ ist die Wahrscheinlichkeit für $X = x_i$ bei gegebenem $Y = y_j$ (egal wie unwahrscheinlich $Y = y_j$ ist) und ist gegeben durch*

$$p(X = x_i|Y = y_j) = \frac{n_{ij}}{r_j} \tag{1.10}$$

Aus den Definitionen für *vereinigende* und *bedingte Wahrscheinlichkeit*, folgt die weitere wichtige **Produktregel** der Wahrscheinlichkeitsrechnung

$$p(X = x_i, Y = y_j)$$

$$= \frac{n_{ij}}{N} = \frac{n_{ij}}{r_j} \cdot \frac{r_j}{n_{ij}}$$

$$= p(X = x_i|Y = y_j)p(Y = y_j) \tag{1.11}$$

Im folgenden wird für eine Wahrscheinlichkeit $p(X = x_i)$ kurz $p(x)$ notiert, entsprechend kurz $p(X = x_i, Y = y_j) = p(x, y)$ und $p(X = x_i|Y = y_i) = p(x|y)$.

Für den kontinuierlichen Fall von Zufallsvariablen X und Y wird die *Summenregel* aus Gleichung 1.9 zu:

$$p(x) = \int_{\forall y} p(x, y) \tag{1.12}$$

1.1.4 Die Bayes Regel

Durch die *Produktregel* und die Symmetrie-Eigenschaft

$$p(x, y) = p(y, x) = p(x|y)p(y) = p(y|x)p(x)$$

kann die **Bayes Regel** abgeleitet werden:

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)} \tag{1.13}$$

Aus der Summenregel folgt für den Nenner $p(x)$ aus Gleichung 1.13

$$p(x) = \sum_y p(x|y)p(y) \tag{1.14}$$

Der Nenner kann hierbei als Normalisierungskonstante angesehen werden, durch den die bedingte Wahrscheinlichkeit auf der linke Seite der Gleichung 1.13 über alle x zu 1 aufsummiert.

1.1.4.1 Beispiel zur Bayes Regel

Folgendes Beispiel zur Bayes Regel ist aus [ChrBish] entnommen:

Seien zwei Kisten ((g)rün/(b)lau) gegeben mit folgenden Anteilen an Äpfeln (a) und Birnen (b):

Abbildung 1.1: Kistenbeispiel

Ferner sei ein Zufallsexperiment mit zwei Zufallsvariablen ((**K**)iste und (**F**)rucht) definiert, innerhalb welchem zehn Mal zufällig eine Frucht aus einer der beiden Kiste gewählt wird mit Häufigkeiten der Ereignisse

$$p(K = g) = 6/10 \text{ und } p(K = b) = 4/10$$

somit (siehe Abbildung 1.1):

- $p(F = a|K = g) = \frac{3}{4}$
- $p(F = b|K = g) = \frac{1}{4}$
- $p(F = a|K = b) = \frac{1}{4}$
- $p(F = b|K = b) = \frac{3}{4}$

Zu beachten ist, dass die Wahrscheinlichkeiten normalisiert sind:

$$p(F = a|K = g) + p(F = b|K = g) = 1$$
$$\text{und}$$
$$p(F = a|K = b) + p(F = b|K = b) = 1$$

Durch Anwendung der Produkt- und Summenregel kann nun beantwortet werden:[6]

[6] Aus der Summenregel folgt für $p(F = b) = 1 - p(F = a) = \frac{9}{20}$

$$p(F = a) =$$
$$p(F = a|K = g)p(K = g) + p(F = a|K = b)p(K = b) =$$
$$\frac{3}{4} \times \frac{6}{10} + \frac{1}{4} \times \frac{4}{10} = \frac{11}{20}$$

Einleitend zum Kistenbeispiel wurden Wahrscheinlichkeiten für $p(F|K)$ angegeben. Die Umkehrung, also die bedingte Wahrscheinlichkeiten $p(K|F)$ können durch das Bayes Theorem ausgerechnet werden:

$$p(K = g|F = a) = \frac{p(F=a|K=g)p(K=g)}{p(F=a)} = \frac{3}{4} \times \frac{4}{10} \times \frac{20}{9} = \frac{2}{3}$$

Entsprechend für $p(K = b|F = a)$:

$$p(K = b|F = a) = 1 - p(K = g|F = a) = \frac{1}{3}$$

Interpretation:
Folgende Interpretation der Bayes Regel ist an dieser Stelle möglich: Würde man gefragt werden, welche Kiste gewählt wurde *bevor* bekannt wäre welche Frucht vorliegt, dann wäre die einzige zur Verfügung stehende und vollständige Information $p(K)$. Die Wahrscheinlichkeit $p(K)$ wird in diesem Kontext deshalb auch als *prior*- oder *apriori* Wahrscheinlichkeit bezeichnet. Nachdem bekannt ist, welche Frucht vorliegt, kann *danach* durch die Bayes Regel die *posterior* (oder posteriori) Wahrscheinlichkeit $p(K|F)$ ausgerechnet werden.
In obigem Zufallsexperiment ist $p(K = b) = \frac{4}{10}$ und demnach würde man intuitiv eher zur grünen Kiste tendieren[7] mit $p(K = g) = \frac{6}{10}$. Durch die zusätzliche Information, dass ein Apfel vorliegt, erhöht sich jedoch die Wahrscheinlichkeit, dass es sich um die grüne Kiste handelt, da der Anteil an Äpfeln zu Birnen in der grünen Kiste höher ist.

1.1.5 Die Gauß-Wahrscheinlichkeitsverteilung

Die Gauß-Verteilung[8], auch als *Normalverteilung* bezeichnet, ist ein häufig verwendetes Modell für die Verteilung von kontinuierlichen Zufallsvariablen. Der sog. zentrale Grenzwertsatz[9] beschreibt eine besondere Eigenschaft der Gauß-Verteilung, die diese für die Modellbildung besonders qualifiziert: Bei Messungen von statistisch unabhängigen Zufallsgrößen und der Betrachtung der Summe dieser (welche wiederrum eine einzige Zufallsgröße ergibt), ist diese Summe bei ausreichend vielen Summanden Gauß verteilt[ChrBish, S.78].

Für den Fall einer einzigen Zufallsvariablen x ist die Gauß-Verteilung definiert als

$$\mathcal{N}(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{1}{2\sigma^2}(x - \mu)^2) \tag{1.15}$$

[7] unabhängig davon welche Frucht vorliegt
[8] benannt nach dem deutschen Mathematiker Johann Carl Friedrich Gauß
[9] im Englischen bezeichnet als (central limit theorem CLT)

Hierbei sind die beiden Parameter μ der *Mittelwert* und σ^2 die *Varianz* der Normalverteilung. Für eine Zufallsvariabe x, die einer Gauß-Verteilung mit den Parametern μ und σ^2 unterliegt, wird statt der Schreibweise $\mathcal{N}(x|\mu, \sigma^2)$ auch notiert $x \sim \mathcal{N}(\mu, \sigma^2)$.

In Abbildung 1.2 ist der typische Glockenkurvenverlauf der Normalverteilung gezeigt (in diesem Fall für $x \sim \mathcal{N}(0, 1)$).

Abbildung 1.2: Glockenkurve einer Gauß-Normalverteilung

Für einen D-dimensionalen Vektor x ist die *multivariate* Gauß-Verteilung definiert als

$$\mathcal{N}(x|\mu, \Sigma) = \frac{1}{2\pi^{D/2}|\Sigma|^{1/2}} \exp(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)) \tag{1.16}$$

Wobei Σ die *Kovarianzmatrix* und $|\Sigma|$ ihre Determinante ist. Abbildung 1.3 zeigt beispielhaft eine multivariate (in diesem Fall zweidimensionale) Gauß-Verteilung: Einmal im dreidimensionalen Raum als Funktion der beiden Dimensionen der Zufallsvariablen x (diese spannen die untere Ebene im Plott auf) und einmal als sog. zweidimensional aufgetragenen *Konturplot*, in welchem unterschiedliche Funktionswerte der Gauß-Verteilung durch unterschiedliche farbliche Darstellung gezeigt werden (dies kann auch als senkrechter Blick auf den linken Plott verstanden werden).

Folgend sei auf bestimmte Besonderheiten der Gauß-Verteilung als Wahrscheinlichkeitsmodell hingewiesen, die sich in bestimmten Situationen als nachteilhaft erweisen können: Die Größe der Kovarianzmatrix Σ einer multivariaten Gauß-Verteilung hängt offensichtlich von der Dimension des jeweiligen Datenraumes der Zufallsvariable x ab. Aufgrund der Symmetrie-Eigenschaft einer Kovarianzmatrix (siehe auch Definition 1.5), kann die

Abbildung 1.3: 2D Gauß-Normalverteilung und Konturplot

Anzahl der eineindeutigen Elemente einer Kovarianzmatrix für einen Datenraum mit D-Dimensionen also mit $\frac{D(D+1)}{2}$ angegeben werden. Nimmt man nun noch die D-Elemente des Mittelwert-Parameters μ hinzu, dann ist die Gesamtanzahl der freien Parameter einer multivariaten Gauß-Verteilung $\frac{D(D+3)}{2}$. Diese Anzahl der freien Parameter wächst somit quadratisch mit der Datendimension D, was bei der Invertierung der Kovarianzmatrix (siehe Gleichung 1.16) einen erheblichen Rechenaufwand bedeuten kann [ChrBish] (S.84). Um diesem Problem entgegenzuwirken, reicht es manchmal in praktischen Anwendungen aus, nur bestimmte Formen der Kovarianzmatrizen zuzulassen. Allgemein können folgende Formen einer Kovarianzmatrix angegeben werden:

- Eine Kovarianzmatrix, die keinen Einschränkungen unterliegt wird als *volle Kovarianzmatrix* bezeichnet
- Eine Kovarianzmatrix, die lediglich Diagonalelemente enthält, wird als *diagonale Kovarianzmatrix* bezeichnet
- Ist eine Kovarianzmatrix $\Sigma = \sigma^2 I$, also ein vielfaches der Einheitsmatrix, wird sie als *isotropische Kovarianzmatrix* bezeichnet

Die geometrischen Auswirkungen der verschiedenen Formen einer Kovarianzmatrix sind beispielhaft in Konturplots verschiedener Gauß-Verteilung in Abbildung 1.4 gezeigt. Hierbei zeigen Abbildung 1.4(a), (b) und (c) die Auswirkungen der Verwendung einer vollen bzw. diagonalen oder aber isotropischen Kovarianzmatrix. Eine Einschränkung auf bestimmte Formen der Kovarianzmatrix bedeutet jedoch neben dem Gewinn weniger freie Parameter zu haben, gleichzeitig aber auch eine weniger flexible Modelliermöglichkeit.

Abbildung 1.4: Verschiedene Formen der Kovarianzmatrix und ihre geometrischen Auswirkungen

1.1.5.1 Bedingte und marginale Gauß-Verteilungen

Im Folgenden werden zwei wichtige Rechenregeln bezüglich multivariater Gauß-Verteilungen vorgestellt. Hierzu seien zunächst die Zufallsvariable x und die Parameter μ und Σ einer multivariaten Gauß-Verteilung $x \sim \mathcal{N}(\mu, \Sigma)$ in sog. *Partitionen* notiert mit

$$x = \begin{pmatrix} x_a \\ x_b \end{pmatrix}, \mu = \begin{pmatrix} \mu_a \\ \mu_b \end{pmatrix} \text{ und } \Sigma = \begin{pmatrix} \Sigma_{aa}\Sigma_{ab} \\ \Sigma_{ba}\Sigma_{bb} \end{pmatrix}$$

Interpretiert man nun x als eine Vereinigung einzelner univariater Gauß-Verteilungen mit

$$x_a \sim \mathcal{N}(\mu_a, \Sigma_{aa}) \text{ und } x_b \sim \mathcal{N}(\mu_b, \Sigma_{bb}),$$

so gibt es die besondere Eigenschaft, dass die *bedingte Wahrscheinlichkeitsverteilung* $p(x_a|x_b)$ ebenfalls eine Gauß-Verteilung ergibt.
Die Parameter der bedingten Gauß-Verteilung $x_a|x_b \sim \mathcal{N}(\mu_{a|b}, \Sigma_{a|b})$ sind gegeben durch

$$\mu_{a|b} = \mu_a + \Sigma_{ab}\Sigma_{bb}^{-1}(x_b - \mu_b) \tag{1.17}$$

$$\Sigma_{a|b} = \Sigma_{aa} - \Sigma_{ab}\Sigma_{bb}^{-1}\Sigma_{ba} \tag{1.18}$$

Eine *marginale* Wahrscheinlichkeitsverteilung $p(x_a) = \int p(x_a, x_b)dx_b$ für die bereits angegebenen Partitionen ist ebenfalls Gauß verteilt und parametrisiert durch

$$\mu_a = \mu_a \tag{1.19}$$

$$\Sigma_a = \Sigma_a \tag{1.20}$$

Beispiel:

Zu obigen Ausführungen ist in Abbildung 1.5 links beispielhaft ein Konturplot einer vereinigenden Gauß-Verteilung $p(x_a, x_b) = \mathcal{N}(\mu, \Sigma)$ und rechts die marginale $p(x_a) = \mathcal{N}(\mu_a, \Sigma_{aa})$ bzw. bedingte Gauß-Verteilung $p(x_a|x_b = 0.7) = \mathcal{N}(\mu_{a|b}, \Sigma_{a|b})$ zu sehen mit

$$\mu = \begin{pmatrix} 0.25 \\ 0.3 \end{pmatrix}, \Sigma = \begin{pmatrix} 10.4 & 10.1 \\ 10.1 & 10.6 \end{pmatrix},$$
$$\mu_{a|b} = 10.3, \Sigma_{a|b} = 0.77,$$
$$\mu_a = 0.25 \text{ und } \Sigma_{aa} = 10.4,$$

Abbildung 1.5: Beispiel zur vereinigenden, marginalen und bedingten Gauß-Verteilung

1.1.5.2 Bayes Theorem für Gaußsche Zufallsgrößen

Folgend soll noch eine weitere wichtige Regel bezüglich Gauß-Verteilungen aufgelistet werden. Der Mittelwert einer bedingten Gauß-Verteilung $p(x_a|x_b)$ ist eine lineare Funktion von x_b (siehe Gleichung 1.17). Bezüglich des Bayes Theorems zu Gaußschen Zufallsgrößen wird angenommen, dass eine marginale Gauß-Verteilung $p(x)$ und eine bedingte Gauß-Verteilung $p(y|x)$, deren Mittelwert eine lineare Funktion von x ist (und dessen Kovarianz unabhängig von x ist), gegeben sind[10]. Es wird nun die marginale bwz. bedingte Gauß-Verteilung $p(y)$ bzw. $p(x|y)$ gesucht, wobei

$$p(x) = \mathcal{N}(x|\mu, \Lambda^{-1}) \tag{1.21}$$

$$p(y|x) = \mathcal{N}(y|Ax + b, L^{-1}) \tag{1.22}$$

μ, A und b sind Mittelwerts-Parameter während die inversen Kovarianz-Matrizen Λ^{-1} und L^{-1} allgemein im Englischen auch als *precision matrices*

[10] Dies wird auch als ein lineares Gauß-Modell bezeichnet [ChrBish]

bezeichnet werden. Die gesuchten Verteilungen $p(y)$ und $p(x|y)$ sind dann gegeben durch[11]:

$$p(y) = \mathcal{N}(y|A\mu + b, L^{-1} + A\Lambda^{-1}A^T) \tag{1.23}$$

$$p(x|y) = \mathcal{N}(x|\Sigma\{A^T L(y-b) + \Lambda\mu\}, \Sigma) \tag{1.24}$$

mit $\Sigma = (\Lambda + A^T LA)^{-1}$.

1.1.6 Maximum Likelihood Methode

In diesem Abschnitt soll die Maximum-Likelihood (ML) Methode zur Parameterschätzung von diskreten/kontinuierlichen Verteilungsfunktionen vorgestellt werden. Hintergrund hierzu ist folgender: In der Praxis kommt es häufig zu Situationen, in denen Kennwerte wie Mittel-/Erwartungswert oder Standardabweichung einer Wahrscheinlichkeitsverteilung zu einer Population gefragt sind. Die Population ist hierbei meistens als solche nicht gegeben (bzw. aufgrund ihrer Größe nicht gänzlich untersuchbar), sondern wird lediglich durch eine Stichproben-Teilmenge repräsentiert. Aufgabe ist es dann, die unbekannten Kennwerte der Gesamtpopulation durch die gegebene Stichprobenmenge zu schätzen.
Die allgemeine Vorgehensweise der ML Parameterschätzung kann wie folgt beschrieben werden: Für eine Zufallsvariable X mit vorliegenden n Stichprobenwerten $x_1, x_2, .., x_n$, die einer Wahrscheinlichkeitsfunktion $f(X; q)$ mit gesuchtem Parameter q unterliegen, können die Funktionswerte der Stichproben faktorisiert werden durch:

$$f(x_1, x_2, .., x_n; q) = \prod_{i=1}^{n} f(x_i; q) \tag{1.25}$$

Wird nun die Wahrscheinlichkeitsfunktion als Funktion des unbekannten Parameters q interpretiert, so ergibt sich hierraus die sog. Likelihood-Funktion

$$L(q) = \prod_{i=1}^{n} f(x_i; q) \tag{1.26}$$

Maximiert man jetzt die Funktion $L(q)$ in Abhängigkeit ihres Parameters q, so ergibt sich ein Maximum-Likelihood Schätzer für den unbekannten Parameter q. Dieser Schätzer kann in diesem Sinne als plausibelster Parameterwert für die Realisierungen der Stichprobenwerte $x_1, x_2, .., x_n$ gesehen werden. Die besagte Maximierung der Funktion $L(q)$ geschieht durch Ableiten von $L(x)$ nach q und anschliessendes Null-setzen. Generell sind in

[11] Zur Herleitung hierzu sei verwiesen auf [ChrBish, S.91]

der Praxis Ausdrücke für $L(q)$ durch kompliziertere Formen gegeben. Daher wird stattdessen das Maximum der logarithmierten Likelihood gesucht[12]:

$$ln(L(q)) = ln(\prod_{i=1}^{n} f(x_i; q)) \tag{1.27}$$

Der Einfachheit der Maximum-Likelihood Methode steht entgegen, dass in der Praxis meist sehr komplexe Modelle von Zufallsprozessen zum Einsatz kommen, für die eine eindeutige analytische Vorgehensweise mit der ML-Methode oft nicht möglich ist (siehe hierzu auch Abschnitt 2.3 des folgenden Kapitels).

[12] die logarithmierte Funktion besitzt an der gleichen Stelle ein Maximum, kann aber leichter errechnet werden

Kapitel 2

Auswertungsmethoden unüberwachtes Lernen

Raffiniert ist der Herrgott,
aber boshaft ist er nicht
- Albert Einstein

2.1 Principal Component Analysis

Die Principal Component Analyse (PCA) ist eine häufig verwendete Methodik zur Feature Extraktion und Datenvisualierung in der Praxis. Gründe hierfür sind die einfache Umsetzung und schnelle Anwendbarkeit [ChrBish, S.226]. Die PCA dient zur Identifikation von Strukturen in Datensätzen. Hierbei wird ein oftmals aus einem hochdimensionierten Raum stammender Datensatz in einer Art dargestellt, in der Ähnlichkeiten und Unterschiede dieser Strukturen besser zu erkennen sind. Die Projektion des Ursprungsdatensatzes auf einen niedriger dimensionierten Raum ohne Verlust von strukturellen Informationen ist dabei wesentlicher Bestandteil der PCA. Hierzu ist ein Anwendungsgebiet der PCA z.B. die Bildkompression.
Es seien n Datenvektoren x^n mit $n = 1, ..., N$ aus dem Raum V = $\mathbb{R}^d$ auf die Vektoren z^n aus U = $\mathbb{R}^M$, einem Unterraum von V, linear abzubilden. Ferner seien $u_1, u_2, ..., u_d$ eine orthonormale Basis in V mit

$$u_i^T u_j = \delta_{ij}, \tag{2.1}$$

mit δ_{ij} dem sog. Kronecker delta [ChrBish, S.227]:

$$\delta_{ij} = \begin{cases} 1 & \text{falls i=j} \\ 0 & fallsi \neq j \end{cases} \tag{2.2}$$

und $i, j \in \mathbb{N}$.

Jeder Vektor x lässt sich nun darstellen als folgende Linearkombination:

$$x = \sum_{i=1}^{M} x_i u_i + \sum_{i=M+1}^{d} x_i u_i \tag{2.3}$$

Entsprechend sei die lineare Abbildung der Datenvektoren x auf z definiert als

$$z = \sum_{i=1}^{M} x_i u_i + \sum_{i=M+1}^{d} b_i u_i \tag{2.4}$$

Die Koeffizienten b_i in obiger Gleichung und die orthonormalen Basisvektoren u_i werden so gewählt, dass die Projektion z^n die x^n am besten approximieren. Die Qualität dieser Abbildung kann durch folgenden Fehler gemessen werden:

$$\begin{aligned} E &= \frac{1}{2}\sum_{n=1}^{N} \|x^n - z^n\|^2 \\ &= \frac{1}{2}\sum_{n=1}^{N}\sum_{i=M+1}^{d}\sum_{j=M+1}^{d} (x_i^n - b_j) u_i^T u_j \\ &= \frac{1}{2}\sum_{n=1}^{N}\sum_{i=M+1}^{d} (x_i^n - b_i)^2 \end{aligned} \tag{2.5}$$

Die Nullstellen der Ableitung von E nach b_i ergeben

$$b_i = \frac{1}{N}\sum_{n=1}^{N} x_i^n \tag{2.6}$$

was der i'ten Komponente des Mittelwert-Vektors $\bar{x}$ bezogen auf das Koordinatensystem $u_1, .., u_d$ entspricht oder entsprechend $u_i^T \bar{x}$. Der Fehler aus 2.5 kann durch Umformung nun auch notiert werden als [ChrBish, S.228]:

$$\begin{aligned} E &= \frac{1}{2}\sum_{i=M+1}^{d} \left\{u_i^T (x^n - \bar{x})^T\right\} \left\{(x^n - \bar{x}) u_i\right\} \\ &= \frac{1}{2}\sum_{i=M+1}^{d} u_i^T \Sigma u_i, \end{aligned} \tag{2.7}$$

Σ ist hierbei die Kovarianzmatrix des Datensatzes.

Es kann durch Benutzung von Lagrange Multiplikatoren gezeigt werden, dass die Minima von E in Abhängigkeit zu den gewählten Basisvektoren u_i

den Eigenvektoren von Σ entsprechen mit $\Sigma u_i = \lambda_i u_i$ eingesetzt in Gleichung 2.7 [ChrBish, S.228]:

$$E = \frac{1}{2} \sum_{n=M+1}^{d} \lambda_i \tag{2.8}$$

Aus den d Eigenvektoren von Σ werden diejenigen zueinander orthonormalen Eigenvektoren genommen, die den größten M Eigenwerten entsprechen. Diese als *Hauptkomponenten (Principal Components)* bezeichneten Basisvektoren dienen dazu den Datensatz auf den durch sie aufgespannten Raum zu projizieren.

Für die Festlegung der Größe M gibt es in der Praxis keine generelle Vorgabe [1]. Es ist jedoch möglich durch folgenden Term für verschiedene M einen Wert zu ermitteln, bei dem die Projektion der Daten einen bestimmten Varianzwert beibehält (dieser könnte z.B. 0.9 oder 0.95 sein):

$$\frac{\sum_{i=1}^{M} \lambda_i}{\sum_{i=1}^{d} \lambda_i} \tag{2.9}$$

Im folgenden vereinfachten Praxisbeispiel soll die Anwendungsmöglichkeit der PCA Analyse graphisch dargestellt werden.

Kleines Beispiel zur PCA-Analyse:
Es werden Messdaten von afrikanischen Tierarten durch zwei Sensoren erfasst und liegen für die Tiere *Löwe, Leopard, Zebra* und *Wasserbüffel* vor. In Abbildung 2.1 sind diese Messdaten (im Sensorraum) aufgezeigt. Die graphische Ausdehnung der Messdaten ist durch die grüne Ellipse gezeigt. In Abbildung 2.2 wird nun in diese Ellipse die erste Hauptkomponente *Comp1* so gelegt, dass sie den größten Anteil der Varianz beschreibt (graphisch entspricht das der Hauptachse der Ellipse). Anschliessend werden die Messdaten orthogonal auf die erste Hauptkomponente projiziert (siehe rote Pfeile in Abbildung 2.2 links), sodass die Lage der Messdaten in Bezug auf die erste Hauptkomponente zu sehen ist (Abbildung 2.2 rechts). Entsprechend die Vorgehenseweise für die zweite Hauptkomponente *Comp2*, welche orthogonal zur ersten Hauptkomponente liegt. Es werden wiederrum die Messdaten orthogonal auf die Hauptkomponente *Comp2* projiziert (rote Pfeile) bzw. die Lage der Messdaten auf der zweiten Hauptkomponente dargestellt (Abbildung 2.3 links bzw. rechts). In Abbildung 2.4 sind die Messdaten in den durch die beiden Hauptkomponenten *Comp1* und *Comp2* aufgespannten Hauptkomponentenraum projiziert worden (die durch die Hauptkomponenten anteilig erklärte Varianz in Prozent ist ebenfalls an den Achsen vermerkt).
Es sind folgende Interpretationen möglich: Offensichtlich reicht bereits die erste Hauptkomponente *Comp1* aus, um zwischen einzelnen Tierarten zu

[1] Zu diesem Punkt siehe auch probabilistische Variante der PCA in Kapitel 2.2

unterscheiden. Die Hauptkomponente *Comp2* kann zwar nicht die Tierarten unterscheiden, jedoch scheint diese Huftiere von huflosen Tieren zu unterscheiden[2].

Abbildung 2.1: Einfaches Beispiel für Messdaten von afrikanischen Tieren zur PCA Analyse

Abbildung 2.2: Erste Hauptkomponente im Sensorraum der Messdaten

[2]Diese Interpretationen sind rein spekulativ und dienen nur der Veranschaulichung

Abbildung 2.3: Zweite Hauptkomponente im Sensorraum der Messdaten

Abbildung 2.4: Projektion der Messdaten in den Hauptkomponentenraum

2.2 Propabilistic Principal Component Analysis

PPCA leitet sich aus der Standard Faktor Analyse ab, in welcher die Daten x sich durch eine Linearkombination von einer Anzahl latenter (versteckter) Variablen z bilden lassen [HarHot, S.2]:

$$x = W * z + \mu + \epsilon \tag{2.10}$$

wobei W eine $d \times M$ Matrix ist (d und M äquivalent zu PCA s.o.) und von einer Gauss-Verteilung $z \sim N(0, I)$ der voneinander unabhängigen Variablen z mit Null-Mittelwert und einer Einheits-Varianz ausgegangen wird. Durch zusätzliche Annahme von $\epsilon \sim N(0, \Psi)$ induziert Gleichung 2.10 die Gauss-Verteilung $x \sim N(\mu, WW^T + \Psi)$ [ChrBish, S.226]. Es gibt keine analytische Vorgehensweise zur Ermittlung der Parameter W und Ψ dieses Modells. Sie müssen durch einen iterativen Prozess wie einer Variante des Expectation Maximization Algorithmuses errechnet werden [HarHot, S.3].

2.3 Modellbildung mit Gaussian Mixture Modells (GMM)

In Kapitel 1.1.5 wurde bereits die Gauß-Normalverteilung vorgestellt und positive Eigenschaften dieser zur Modellbildung von Zufallsprozessen angesprochen. In der Praxis besitzen diese Zufallsprozesse meist eine Komplexität, die durch einzelne einfache Gauß-Verteilungen nicht nachzubilden ist. Abbildung 2.5 soll das Problem verdeutlichen: Im linken Plot sind Datenpunkte aus einem zweidimensionalen Datenraum gezeigt (blaue Sterne) und ebenfalls ein Konturplot einer multivariaten Gauß-Verteilung, die den Datenpunkten zugrundeliegenden Zufallsprozess modellieren soll. Im rechten Plot sind wieder die gleichen Datenpunkte gezeigt (rote und blaue Sterne). Allerdings sind es diesmal zwei verschiedene Gauß-Verteilungen, die die charakteristische Lage der Datenpunkte besser erfassen. Die Kombination mehrerer Gauß-Verteilung führt zur Bildung eines sog. GMM Mixture Modells, bestehend aus Kombinationen von verschiedenen einfachen Gauß-Verteilungen. Die Modellbildung einer Wahrscheinlichkeitsverteilung $p(x)$ einer Zufallsvariable X geschieht hierbei durch Bildung von Linearkombinationen über K einfache Gauß-Verteilungen mit jeweils verschiedenen sog. *Mixtur-Koeffizienten* π_k:

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k) \tag{2.11}$$

Damit $p(x)$ als Wahrscheinlichkeiten aufgefasst werden können, muss für die einzelnen Mixtur-Koeffizienten π_k mit $0 \leq \pi_k \leq 1$ gelten:

$$\sum_{k=1}^{K} \pi_k = 1 \tag{2.12}$$

Abbildung 2.5: Problematik zu einfachen Gauß-Verteilungen

Abbildung 2.6: Mixtur von Gauß-Verteilungen

Nach der Produkt- und Summenregel (siehe auch Abschnitt 1.1.3) ist die marginale Wahrscheinlichkeitsverteilung $p(x)$ hier gegeben durch:

$$p(x) = \sum_{k=1}^{K} p(k)p(x|k), \tag{2.13}$$

wobei die Mixtur-Koeffizienten $\pi_k = p(k)$ als apriori Wahrscheinlichkeiten der k'ten Komponente und $\mathcal{N}(x|\mu_k, \Sigma_k) = p(x|k)$ die Likelihood Wahrscheinlichkeiten von x bei gegebenem k gesehen werden. Die posterior Wahrscheinlichkeit $p(k|x)$ ist nach der Bayes-Regel gegeben durch

$$\begin{aligned} p(k|x) =& \frac{p(k)p(x|k)}{\sum_l p(j)p(x|j)} \\ =& \frac{\pi_k \mathcal{N}(x|\mu_k, \Sigma_k)}{\sum_l \pi_j \mathcal{N}(x|\mu_j, \Sigma_j)} \end{aligned} \tag{2.14}$$

Die Parameter eines GMM sind somit insgesamt gegeben durch

$$\pi \equiv \{\pi_1, \pi_2, .., \pi_K\},\ \mu \equiv \{\mu_1, \mu_2, .., \mu_K\} \text{ und } \Sigma \equiv \{\Sigma_1, \Sigma_2, .., \Sigma_K\}.$$

Die Angabe einer geschlossenen, analytischen Form der Maximum Likelihood Methode aus Abschnitt 1.1.6 ist hier für die Ermittlung dieser Parameter für folgende logarithmierte Likelihood Wahrscheinlichkeitsfunktion

$$ln(p(X|\pi, \mu\Sigma)) = \sum_{i=1}^{N} \sum_{k=1}^{K} \pi_k \mathcal{N}(x_i|\mu_k, \Sigma_k), \tag{2.15}$$

für vorliegende Daten $X = \{x_1, x_2, .., x_n\}$ aufgrund der Summe über die variablen k-Komponenten leider nicht mehr möglich. Die Ermittlung der Parameter erfolgt stattdessen durch Anwendung der sog. Expectation Maximization Methode, welche eine iterative numerische Optimierungstechnik ist und in Kapitel 2.3.2 vorgestellt wird. Zur Initialisierung der k-Komponenten eines GMM wird häufig ein sog. k-means Clusterverfahren angewendet. Deshalb soll folgend auf die Clusteranalyse allgemein und das *k-means-clustering* im speziellen kurz eingegangen werden.

2.3.1 Verfahren zur Clusteranalyse

Eine sog. Clusteranalyse[3] ist ein multivariates Analyseverfahren zum Entdecken von Strukturen bwz. Untergruppen von Objekten mit bestimmten Ähnlichkeiten. Allgemein sind die zu untersuchenden Objekte als Zufallsvariablen aufgefasst, wobei die Objektmerkmale als Dimensionen eines Vektorraums und die Objekte als Punkte in diesem intepretiert werden. Ein Cluster kann somit als eine Anhäufung von Punkten mit bestimmten Ähnlichkeitseigenschaften aufgefasst werden. Die Ähnlichkeit wird hierbei durch ein vorher festgelegtes Abstandsmaß beeinflusst, wobei eine hohe Ähnlichkeit zweier Objekte einen kürzeren Abstand im Merkmalsraum bedeutet. Cluster sind auch durch ihre sog. *Clusterzentren* charakterisiert, welche eine Art Objekt-Prototypen für das jeweilige Cluster darstellen und die Lage eines Clusters im Merkmalsraum bestimmen.

Allgemein lassen sich Verfahren zur Clusteranalyse grob in *harte* bzw. *weiche* Verfahren unterteilen: Bei harten Methoden wird jedes Objekt exakt einem Cluster zugeordnet. Weiche Methoden dagegen ordnen für die Zugehörigkeit eines Objektes zu jedem Cluster eine Wahrscheinlichkeit zu. Weiche Methoden sind besonders bei homogeneren Verteilungen Objekten im Merkmalsraum und größeren Überschneidungen von Clustergrenzen nützlich. Eine Instanz für weiche Methoden sind z.B. die bereits vorgestellten Gaussian Mixture Models (GMM). Ein Beispiel für eine harte Methode dagegen ist das sog. *k-Means-Clustering* Verfahren, welches nun seiner Einfachheit und der Verwendung als Initialisierungsschritt für andere Verfahren (wie z.B. GMM) wegen kurz vorgestellt wird.

[3] selten auch als *Ballungsanalyse* bezeichnet

2.3.1.1 K-Means-Clustering (KMC)

Beim KMC werden ähnliche Objekte zu k Gruppen (Cluster) zusammengefasst. Das KMC zeichnet sich durch seine Einfachheit und Schnelligkeit aus. Diesem steht allerdings eine niedrigere Genauigkeit der Ergebnisse entgegen. Die Ähnlichkeit von Objekten wird beim KMC durch eine Abstandsfunktion modelliert[4]. Bevor das KMC durchgeführt werden kann, muss die Anzahl k der zu findenden Cluster bestimmt werden. Folgend sind die nötigen Schritte zur Ausführung des KMC aufgeführt:

1. Ermittlung von k Clusterschwerpunkten (z.B. zufällig oder durch Auswahl von k bestimmten Objekten)
2. Zuordnung jedes Objektes zu dem Cluster, zu dessen Clusterzentrum es den kleinsten Abstand hat
3. Korrektur der k Clusterzentren anhand der in Schritt 2. durchgeführten Zuordnungen
4. Wiederholung von Schritt 2 und 3 bis keine Zugehörigkeitsänderungen der Objekte mehr auftauchen

Zu beachten ist an dieser Stelle, dass der Algorithmus höchstens, jedoch nicht immer exakt k Cluster ermittelt.

Im nächsten Kapitel soll nun die Expectation Maximization (EM) Methode vorgestellt werden, welche als Parameterschätzung für GMM verwendet wird.

2.3.2 Parameterschätzung durch die Expectation Maximization (EM) Methode

Wie in Kapitel 2.3 bereits vorgestellt, basiert die Ermittlung der Parameter für ein GMM auf Maximierung der Likelihood Wahrscheinlichkeitsfunktion. Ein gleichwertiger Ansatz ist es, die negative log-Likelihood Funktion als Fehlerfunktion zu sehen und diese zu minimieren:

$$E = -\sum_{n=1}^{N} logp(x_n) \tag{2.16}$$

Es gibt folgende Proble bei dieser Vorgehensweise:

1. Sobald eine Komponente j eines GMM auf einen einzigen Datenpunkt kollabiert mit $\mu_j = x$ und die zugehörige Varianz $\sigma_j \to 0$, wird das globale Minimum zu $E = -\infty$

[4]Ein Beispiel solch einer Abstandsfunktion ist der euklidsche Abstand zweier Objekte im Merkmalsraum

2. Unter den vielen lokalen Minima, gilt es genau diejenigen zu finden, die auch ein gutes Modell liefern

Neben der Möglichkeit globale Optimierungsverfahren zum Finden der Minima zu verwenden, gibt es eine alternative Methode, die als *Expectation Minimization* (EM) bekannt ist. Neben der einfachen Implementierung und schnellerer Konvergierung ist ein weiterer Vorteil der EM-Methode gegenüber klassischen Optimierungsverfahren die Fähigkeit mit fehlenden Messwerten umgehen zu können [NabNey, S.90].

Allgemein modifiziert die EM-Methode die Parameter eines GMM iterativ und minimiert hierbei die Fehlerfunktion E. Hierzu wird zunächst angenommen, dass die Datenpunkte x_n einem unbekannten Mixture Model entstammen, wobei die Zugehörigkeiten der Datenpunkte zu den einzelnen Komponenten ebenfalls unbekannt sind. Es wird vereinbart, dass I_j die Indexmenge der Datenpunkte der j ten Komponente ist und N die Gesamtanzahl aller vorliegenden Datenpunkte. Die apriori Wahrscheinlichkeit $P(j)$ ist nun gegeben durch

$$P(j) = \frac{|I_j|}{N} \tag{2.17}$$

Der Mittelwert der j'ten Komponente ist

$$\mu_j = \frac{1}{|I_j|} \sum_{i \in I_j} x_i \tag{2.18}$$

Die Varianz ist an dieser Stelle abhängig von der Form der Kovarianz-Matrix. Für eine isotropische Kovarianzmatrix wäre dies[5]:

$$\sigma_j^2 = \frac{1}{d|I_j|} \sum_{i \in I_j} \|x_i - \mu_j\|^2 \tag{2.19}$$

An dieser Stelle wird nun angenommen, dass für jeden Datenpunkt x_n eine korrespondierende Zufallsvariable z_n existiert, die einen ganzzahligen Wert von 1 bis K annehmen kann, wobei M die Gesamtanzahl der Komponenten ist. Im Folgenden wird kurz notiert für das Tupel $(x_n, z_n) = y_n$ und w für die Parameter des Mixture Modells. Der EM Algorithmus generiert sequentiell fortlaufende Parameter-Schätzungen w_s, wobei w_0 einen initialen Startzustand der Schätzungen darstellt. Die Likelihood für einen Datenpunkt bei $z = j$ ist gegeben durch:

$$\begin{aligned} p((x, z = j)|w) =& p(x|z = j, w)P(z = j|w) \\ =& p(x|\theta_j)P(z = j|w), \end{aligned} \tag{2.20}$$

wobei θ_j die Parameter der j ten Komponente sind (in diesem Fall Mittelwert μ_j und Varianz σ_j). Die Marginalisierung der Likelihood aus Gleichung 2.20

[5]Für den absoluten Betrag einer Zahl a wird notiert: $\|a\|$

über z ist einfach durch das Aufsummieren über alle möglichen Werte für z gegeben:

$$p(x|w) = \sum_{j=1}^{K} P(z = j|w)p(x|\theta_j) \tag{2.21}$$

Auffällig ist an dieser Stelle, dass die $P(z = j|w)$ die gleiche Bedeutung wie die Mixturkoeffizienten aus Gleichung 2.11 haben.

Bei gegebenen Schätzwerten w_s gilt es nun, die nächsten Schätzungen w_{s+1} mit Hilfe der Gleichungen 2.17, 2.18 und 2.19 auszurechnen. Da die einzelnen Zuordnungen z_n von Datenpunkten x_n zu den jeweiligen Komponenten unbekannt sind, kann zumindest das apriori Vorwissen der bereits vorliegenden Verteilungen genutzt werden (siehe Gleichung 2.17) und ein Erwartungswert der Zuordnungen z_n bei gegebenen Parametern w_s durch folgende Q-Funktion angegeben werden:

$$\begin{aligned} Q(w|w_s) =& E(logp(y|w))p(z_n|x_n, w_s) \\ =& \sum_{j=1}^{K}\sum_{i=1}^{N} [log(p(x_n, z_n|w))]P(z_n = j|x_n, w_s) \\ =& \sum_{j=1}^{K}\sum_{i=1}^{N} [logP(j) + logp(x_n|\theta_j)]P_{(s)}(j|x_n), \end{aligned} \tag{2.22}$$

wobei

$$\begin{aligned} P_{(s)}(j|x_n) :=& P(z_n = j|x_n, w_s) \\ =& \frac{P_{(s)}(j)p(x_n|\theta_{(s)j}}{\sum_{j=1}^{M} P_{(s)}(j)p(x_n|\theta_{(s)j})}. \end{aligned} \tag{2.23}$$

$P_{(s)}(j|x_n)$ ist die erwartete posterior Wahrscheinlichkeit der Komponentenzuordnungen bei gegebenen Daten. Die Q-Funktion aus Gleichung 2.22 ist eine Funktion der Parameter $P(j)$ und θ_j, wobei $P_{(s)j}$ und $\theta_{(s)j}$ feste Werte darstellen. Die Berechnung der Q-Funktion wird im allgemeinen als *E-Schritt* bezeichnet. Zur Neuberechnung der Parameterwerte w_{s+1} wird die Q-Funktion optimiert durch

$$w_{s+1} = argmaxQ(w|w_s). \tag{2.24}$$

Dieser Schritt wird als *M-Schritt* bezeichnet. Neue Parameterwerte $\mu_{(s+1)j}$ werden durch Differenzieren der Q-Funktion nach dem j'ten Mittelwert μ_j und anschliessender Nullstellenermittlung gefunden und sind gegeben durch:

$$\mu_{(s+1)j} = \frac{\sum_{n=1}^{N} P_{(s)}(j|x_n)x_n}{\sum_{n=1}^{N} P_{(s)}(j|x_n)} \tag{2.25}$$

Die Gleichungen für neue Varianzparameter hängen von den verwendeten Arten von Kovarianzmatrizen ab. Diese sind gegeben durch:

- **Isotropische Kovarianzmatrix**

$$(\sigma_{(s+1)j})^2 = \frac{1}{d}\frac{\sum_{n=1}^{N} P_{(s)}(j|x_n)\|x_n - \mu_{(s+1)j}\|^2}{\sum_{n=1}^{N} P_{(s)}(j|x_n)} \tag{2.26}$$

- **Diagonale Kovarianzmatrix**

$$(\sigma_{(s+1)i,j})^2 = \frac{\sum_{n=1}^{N} P_{(s)}(j|x_n)(x_{n,i} - \mu_{(s+1)i,j})^2}{\sum_{n=1}^{N} P_{(s)}(j|x_n)} \tag{2.27}$$

- **Volle Kovarianzmatrix**

$$\Sigma_j = \frac{\sum_{n=1}^{N} P_{(s)}(j|x_n)(x_n - \mu_{(s+1)j})(x_n - \mu_{(s+1)j})^T}{\sum_{n=1}^{N} P_{(s)}(j|x_n)} \tag{2.28}$$

Der Algorithmus terminiert bei einem geeigneten Abbruchkriterium, z.B. sobald die Änderungen der neu berechneten Parameterwerte hinreichend klein sind oder eine bestimmte Anzahl von Iterationen durchgeführt wurden. Die Konvergenz ist für den Fall, dass die Likelihood Funktion beschränkt ist, gesichert [Dempster]. Falls die Likelihood Funktion mehrere Optima besitzt, so ist die letztendliche Lösung stark von den Startwerten abhängig.

Kapitel 3

Auswertungsmethoden überwachtes Lernen

Der Mensch soll lernen,
nur die Ochsen büffeln.
- Erich Kästner

Das folgende Kapitel stellt sog. *Gauß Prozesse* als einen möglichen Ansatz zur Behandlung des überwachten Lernens vor. Der in der Fachliteratur oft vorzufindenden Chronologie zu diesem Thema folgend [Rasmus][ChrBish], wird zunächst allgemein auf das Themengebiet des *überwachten Lernens* und den dort vorhandenen Problemen eingegangen. Danach wird einleitend *Regression* als eine mögliche Form des überwachten Lernens vorgestellt. Im Anschluss werden *Gauß Prozesse* und ihre Anwendung zur Klassifikation im spezielleren behandelt.

3.1 Standpunkt und Problematiken

Das Teilgebiet *überwachtes Lernen* des maschinellen Lernens behandelt das Problem, eine Funktion über Eingabemuster auf Ausgabemustern zu erlernen. Abhängig vom Typ der Ausgabemuster ist dieses Problem im kontinuierlichen bzw. diskreten Fall als *Regressions-* bzw. *Klassifikationsproblem* bekannt. Praxisbeispiele für den diskreten Fall wären z.B. die Kreditwürdigkeit bestimmter Bankkunden vorrauszusagen[1] oder aber ein Spracherkennungsproblem bei gegebenen Audiodaten, mit dem Ziel eine Zuordnung zu einer Menge von bestimmten Personen zu lernen. Ein Beispiel für den kontinuierlichen Fall von Ausgabemustern ist, anhand von gegebenen Messdaten eines Landes das zukünftige Bevölkerungswachstum vorrauszusagen.

[1] Die Ausgabemuster (Kreditwürdigkeit) der gelernten Funktion könnten in diesem Fall z.B. beschränkt sein auf die Menge $\{ja, nein\}$

Prinzipiell wird beim überwachten Lernen einem Lernsystem während eines Lernprozesses eine endliche *Trainingsmenge*

$$D = \{(x_0, y_0), (x_1, y_1), .., (x_n, y_n)\}$$

von korrekten[2] Abbildungspaaren durch einen Lehrer[3] präsentiert. Die Eigenschaft eines Lernsystems nach einem Lernprozess ungesehene Eingabemuster x_* korrekten Ausgabemustern zuordnen zu können, wird *Generalisierungsfähigkeit* genannt. Eine hohe Generalisierungsfähigkeit steht also in direktem Zusammenhang zu einem niedrigen Fehler in Bezug auf die Abbildung von ungesehenen Eingabemustern und ist somit eine erwünschte Eigenschaft. Ein Problem hierbei ist das der sog. *Überanpassung* (overfitting) eines Lernsystems: Das aus einer Trainingsmenge extrahierte Wissen ist nicht ausreichend auf ungesehene Beispiele anwendbar (ggf. lediglich auf die Trainingsmenge selbst).

3.1.1 Bayessche Modellwahl

Die *Modellwahl* für die zu erlerndende Funktion erfolgt im überwachten Lernen, indem zunächst eine aus einer sog. Hypothesenklasse stammende *Hypothese* gewählt wird. Diese Hypothesenklasse ist eine Klasse von parametrisierten Funktionen und der Lernprozess ein Vorgang zur Findung optimaler Parameter für die Hypothese[4]. Die Qualität eines Modells im Hinblick auf die zu modellierende Funktion ist oft stark beeinflusst von der Anzahl dieser Parameter: je mehr Parameter, desto flexibler die Anpassbarkeit der Hypothese an gegebenes Vorwissen aus der Trainingsmenge und mehr Möglichkeiten Charakteristika der zu erlernenden Funktion nachzubilden. Gleichzeitig ist die Gefahr einer Überanpassung an eine gegebene Trainingsmenge bei zu vielen Parametern (und evtl. zu kleiner Trainingsmenge) umso größer.

Im überwachten Lernen existieren zwei grundlegende Ansätze zur Modellwahl und Behandlung der obig angesprochenen Probleme[Rasmus, S.2]: Der erste Ansatz schränkt die Hypothesenklasse ein, indem z.B. lediglich lineare Funktionen der Eingabemuster als Hypothese gewählt werden. Ein weiterer sog. *Bayessche* Ansatz vergibt jeder potenziellen Hypothese eine apriori (prior) Wahrscheinlichkeit, wobei Hypothesen mit angenommener höherer Modellierungsqualität höhere Wahrscheinlichkeiten vergeben werden. Beide Vorgehen haben Nachteile: Der Nachteil des ersten Ansatzes ist das Dilemma, die Gefahr der Überanpassung durch ein Beschränken auf bestimmte Hypothesenklassen umgehen zu wollen, was sich negativ auf Generalisierungsfähigkeit des Modells auswirken kann. Der zweite Ansatz sagt nichts über die betrachteten Hypothesenklassen aus und somit besteht das

[2] Dieses Vorwissen stammt z.B. aus einem Expertenwissen

[3] daher der Begriff *überwachtes Lernen*

[4] Das Trainieren eines neuronalen Netzes mit Neuronen-Aktivierungen und den Gewichten ihrer Verbindungen ist ein Beispiel hierfür

Problem aus dem Überangebot (es gibt nämlich unendlich viele [Rasmus, S.2]) von potentiellen Möglichkeiten zur Modellierung der zu erlernenden Funktion geeignete auszuwählen.
Die *Bayessche Modellwahl* soll nun folgend anhand eines kleinen aus [Rasmus] entnommenen Regressionsproblems näher gebracht werden:

Regressionsbeispiel zur Bayesschen Modellwahl

Eindimensionale Eingabedaten x sollen auf Ausgabewerte $2f(x)$ abgebildet werden. In Abbildung 3.1(a) sind hierzu zunächst vier zufällige aus einer apriori Verteilung über Funktionen stammende Beispiel-Funktionen geplottet. Diese Funktionen repräsentieren eine Art Roh-Hypothese noch bevor echte Abbildungspaare der zu modellierenden Funktion vorliegen. An dieser Stelle sei auf zwei spezielle Eigenschaften der vorliegenden Funktionen hingewiesen:

1. Es lässt sich eine charakteristische Mittelwertsfunktion, die den mittleren Funktionswert $f(x)$ über x darstellt, durch vorliegende Funktionen insgesamt errechnen

2. Zu jedem Eingabemuster x lässt sich eine Varianz unter den an dem Punkt x betrachteten Funktionswerten ausrechnen

Abbildung 3.1: Regressions-Beispiel zur Bayesschen Modellwahl

Es wird nun eine Menge $D = \{(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)\}$ von zu erlernenden Abbildungspaaren präsentiert. Das Ziel ist es, aus den vorliegen-

den Funktionen nur diejenigen zuzulassen, die die Eingabemuster x_1, x_2, x_3 bzw. x_4 korrekt auf y_1, y_2, y_3 und y_4 abbilden[5]. Diese Einschränkung, kombiniert mit den prior-Funktionen aus Abb.3.1(a), ergibt eine *posterior* Verteilung über Funktionen. In Abb.3.1(b) sind beispielhaft (dünne Linien) einige dieser posterior Funktionen gezeigt (die dicker gezeichnete Funktion entspricht hierbei der Mittelwertsfunktion der posterior Funktionen).

Klassifikationsbeispiel zur Bayesschen Modellwahl

Ein weiteres Beispiel zur Bayesschen Modellwahl verdeutlicht die Anwendung auf ein *binäres Klassifikationsproblem*: Hierbei sei angenommen, es existierten Messdaten krebskranker und gesunder Patienten. Die betrachteten Ausgabewerte (Klassenlabels) der zu erlernenden Funktion seien z.B. -1 bzw. 1 für *krebskrank* bzw. *gesund*. Die Aufgabe ist nun, für einen neuen Patienten x^* die Wahrscheinlichkeit $\pi(x^*)$ einer Erkrankung vorrauszusagen. Bei Gaußschen Prozessen wird keinerlei Einschränkungen bezüglich des Wertebereiches der prior-Funktionen gemacht (siehe hierzu auch Abb.3.1). Daher werden betrachtete Funktionswerte noch durch eine sog. *response-function* nachbearbeitet, um sie als Wahrscheinlichkeit interpretieren zu können. Eine hierzu häufig verwendete Funktion ist die in Abb.3.2 abgebildetete logistische Sigmoid-Funktion $\lambda(x)$.

Abbildung 3.2: Logistische Sigmoide

[5]Solche Einschränkung können evtl. abgeschwächt werden, indem auch Funktionsausgaben zugelassen werden, die nahe an den gegebenen Ausgabewerten liegen

Abbildung 3.3: Klassifikations-Beispiel zur Bayesschen Modellwahl

In Abb.3.3 ist folgendes zu sehen:

- oben wird eine Beispiel Verteilung über prior-Funktionen für Daten aus einem zweidimensionalem Eingaberaum gezeigt
- unten links ist die Lage der betrachteten Datenpunkte im zweidimensionalen Eingaberaum gezeigt, wobei rote Punkte Individuen der Klasse 1 sind und blaue Punkte der Klasse -1 angehören.
- unten rechts ist ein Konturplot der vorraussagenden Wahrscheinlichkeiten (als Funktion über die Eingaben x) zu sehen mit den in der Farbskala angegebenen Wahrscheinlichkeiten. Hierbei geben Wahrscheinlichkeitswerte nahe bei 0 bzw. 1 eine Zugehörigkeit zur Klasse -1 bzw. 1. Wahrscheinlichkeitswerte um 0.5 können als Klassen-Trennlinienbereiche intepretiert werden.

Obwohl in dieser Arbeit die *Klassifikation* mit Gauß-Prozessen das Hauptaugenmerk sein wird, soll dennoch *Regression* als hinleitendes Thema behandelt werden (zumindest entspricht dies auch der in der Literatur vorzufindenden Behandlungsreihenfolge, z.B. in [ChrBish] oder [Rasmus]).

3.2 Regression

Grundsätzlich gibt es zwei verschiedene Sichtweisen zu Regressions-Modellen mit Gauß-Prozessen[Rasmus, S.7]:

- *function-space* Sichtweise
- *weight-space* Sichtweise

Die *function-space* Sichtweise behandelt das Regressionsproblem durch Definition von Verteilungen über Funktionen und spielt sich daher direkt im Funktionsraum ab. Die *weight-space* Sichtweise dagegen modifiziert Eingabedaten in bestimmter Form direkt und spielt sich daher im Eingabedatenraum ab. Beide Sichtweisen sind in den von ihnen gelieferten Ergebnissen her äquivalent. Jedoch ist die *function-space* Sichtweise vom Verständnis her etwas anspruchsvoller und schwieriger. Daher wird zunächst die *weight-space* Sichtweise vorgestellt.

Vorab sei an dieser Stelle für die weiteren Ausführungen auf folgende Besonderheit der Notierung zu einem gegebenen Datensatz $D = \{(x_i, y_i)|i = 1..n\}$ hingewiesen: Die Eingabedaten x_i werden durch eine Matrix X zusammengefasst[6]. Diese besteht aus den n einzelnen Spaltenvektoren x_i. Die einzelnen Targetwerte y_i werden ebenfalls in einem Vektor y zusammengefasst, sodass auch notiert werden kann: $D = (X, y)$.

3.2.1 Die *weight-space* Sichtweise

Das einfachste Regressionsmodell in der *weight-space* Sichtweise ist das lineare Modell, in welchem die Ausgabedaten $f(x)$ lediglich Linearkombinationen der Eingabedaten x sind:

$$f(x) = x^T w \tag{3.1}$$

Wobei der Parameter w ein reelwertiger Gewichtsvektor ist. Während eines Lernprozesses beobachtete Ausgabewerte (Targets) y entsprechen in diesem Ansatz den Funktionswerten $f(x)$ zusätzlich einem Rauschwert ϵ:

$$y = f(x) + \epsilon, \tag{3.2}$$

Mit der Annahme $\epsilon \sim \mathcal{N}(x|0, \sigma^2)$.

Der Bayessche Ansatz zur Inferenz mit diesem linearen Modell richtet sich grundsätzlich nach der *posterior Wahrscheinlichkeit* $p(w|X, y)$, also der Wahrscheinlichkeit der Gewichte w bei gegebenen Eingabedaten X und Targets y und ist definiert als:

$$p(w|X, y) = \frac{\text{likelihood} \times \text{prior}}{\text{marginal likelihood}} = \frac{p(y|X, w)p(w)}{p(y|X)} \tag{3.3}$$

[6] In der Literatur wird diese häufig als *Designmatrix* bezeichnet

Die drei Teilterme auf der rechten Seite der Gleichung 3.3 werden nun näher betrachtet:

- Die **Likelihood** Wahrscheinlichkeitsverteilung $p(y|X,w)$ der Targets y bei gegebenen Gewichten w und Daten X ist wie folgt definiert:

$$\begin{aligned} p(y|X,w) =& \prod_{i=1}^{n} p(y_i|x_i,w) \\ =& \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma_n^2}} \exp(-\frac{(y_i-(x_i^T w))^2}{2\sigma_n^2}) \\ =& \frac{1}{(2\pi\sigma_n^2)^{\frac{n}{2}}} \exp(-\frac{1}{2\pi\sigma_n^2}|(y-X^T w)|^2) \\ =& \mathcal{N}(X^T w, \sigma_n^2 I) \end{aligned} \tag{3.4}$$

  mit der $n \times n$ Einheitsmatrix I und $|z|$ als euklidscher Länge eines Vektors z.

- Die **prior** Wahrscheinlichkeit $p(w)$ steht für die Hypothese bezüglich der Parameter w noch bevor Eingabedaten X vorliegen und ist unabhängig von diesen. Sie wird hier festgelegt durch eine Gauß-Verteilung mit 0-Mittelwert und Kovarianzmatrix Σ_p:

$$w \sim \mathcal{N}(0, \Sigma_p) \tag{3.5}$$

- Der als **marginal-Likelihood** bezeichnete Nenner der Gleichung 3.3 dient als Normalisierungskonstante und ist definiert als:

$$p(y|X) = \int p(y|X,w)p(w)d\mathbf{w} \tag{3.6}$$

Durch Weglassen der Ausdrücke in den Teiltermen, die unabhängig von w sind und durch Umformung lässt sich Gleichung 3.3 auch wie folgt darstellen:

$$\begin{aligned} p(w|X,y) \propto& \exp(-\frac{1}{2\sigma_n^2}(y-X^T w)^2) \exp(-\frac{1}{2} w^T \Sigma_p^{-1} w) \\ \propto& \exp(-\frac{1}{2}(w-w')^T (\frac{1}{\sigma_n^2} X X^T + \Sigma_p^{-1})(w-w')) \end{aligned} \tag{3.7}$$

mit $w' = \sigma_n^{-2}(\sigma_n^{-2} X X^T + \Sigma_p^{-1})^{-1} X y$.

Somit lässt sich die posterior Wahrscheinlichkeit also auch angeben als

$$p(w|X,y) \sim \mathcal{N}(w', A^{-1}), \tag{3.8}$$

wobei zusammenfassend $A = \sigma_n^{-2} X X^T + \Sigma_p^{-1}$

Die Wahrscheinlichkeit $p(f_*|x_*, X, y)$ von Funktionswerten f_* für ungesehene Eingabedaten x_* wird durch eine Mittelung über die Ausgaben aller möglichen linearen Modelle gewichtet nach entsprechender posterior Wahrscheinlichkeit der Parameter w errechnet:

$$\begin{aligned} p(f_*|x_*, X, y) &= \int p(f_*|x_*, w) p(w, X, y) dw \\ &= \mathcal{N}(\frac{1}{\sigma_n^2} x_*^T A^{-1} X y, x_*^T A^{-1} x_*) \end{aligned} \tag{3.9}$$

3.2.1.1 Alternative Vorgehensweise bezüglich der Nachteile von linearen Regressionsmodellen

Die Einfachheit linearer Modelle ist gleichzeitig auch ein Nachteil: die lineare Modifikation der Eingabedaten stellt eine Einschränkung im Hinblick auf die Flexibilität der Modellerierung einer Ausgabe-Funktion $f(x)$ dar. Eine einfache alternative Vorgehensweise hierzu ist es, die Eingabedaten zunächst zu transformieren und erst danach das lineare Modell in dem transformierten Raum anzuwenden. Hierzu wird eine feste gegebene Funktion $\phi(x)$ verwendet, die die Eingaben x aus dem D-dimensionalen Raum auf einen N-dimensionalen sog. *Feature-Raum* abbildet. Das lineare Modell aus Gleichung 3.1 wird dann also zu:

$$f(x) = \phi(x)^T w \tag{3.10}$$

An dieser Stelle sei zusammenfassend die $\phi(x)$ spaltenweise enthaltende Matrix Φ eingeführt. Die Länge des Parametervektors w ist N (entsprechend der Länge von $\phi(x)$).
Mit dieser Festlegung ist entsprechend Gleichung 3.9 und durch Ersetzung der Teilterme x durch die Transformation $\phi(x)$ (bzw. Φ statt X):

$$f_*|x_*, X, y \sim \mathcal{N}(\frac{1}{\sigma_n^2} \phi(x_*)^T A^{-1} \Phi y, \phi(x)_*^T A^{-1} \phi(x_*)) \tag{3.11}$$

(wobei $A = \sigma_n^{-2} \Phi \Phi^T + \Sigma_p^{-1}$).

Die Invertierung der Matrix A in Gleichung 3.11 kann für große Dimensionen N aufwendig sein. Daher folgende alternative Schreibweise

$$f_*|x_*, X, y \sim \mathcal{N}(\mu_{f_*|x_*,X,y}, \Sigma_{f_*|x_*,X,y}) \tag{3.12}$$

mit

$\mu_{f_*|x_*,X,y} = \phi_*^T \Sigma_p \Phi (K + \sigma_n^2 I)^{-1} y$
$\Sigma_{f_*|x_*,X,y} = \phi_*^T \Sigma_p \phi_* - \phi_*^T \Sigma_p \Phi (K + \sigma_n^2 I)^{-1} \Phi_*^T \Sigma_p \phi_*$

und der verkürzenden Schreibweise $\phi(x_*) = \phi_*$ und $K = \Phi^T \Sigma_p \Phi$. Für den Beweis der Gleichheit der beiden Gleichungen 3.11 und 3.12 sei verwiesen auf [Rasmus, S.12].

An dieser Stelle fallen folgende positive Besonderheiten zu Gleichung 3.12 auf:

1. Es sind Matrizen der Größe $(n \times n)$ zu invertieren, was für $(n < N)$ weniger Berechnungsaufwand bedeutet.

2. Der Feature-Raum liegt allgemein in den Formen $\Phi^T \Sigma_p \Phi$, $\phi_*^T \Sigma_p \Phi$ oder $\phi_*^T \Sigma_p \phi_*$ vor. D.h. Matrizenwerte sind hier durch $\phi(x)^T \Sigma_p \phi(x')$ gegeben (wobei x bzw. x' aus der Trainings- bzw. Testmenge stammen). Mit der Bezeichnung der Teilausdrücke der Form $\phi(x)^T \Sigma_p \phi(x') = k(x, x')$ als ***Kovarianzfunktion*** oder ***Kernel*** ist diese Ersetzung allgemein als ***Kerneltrick*** bekannt[7] und kann in Situationen angewandt werden, in denen die Berechnung des Kernelwertes ggf. einfacher ist als die entsprechender Feature Vektoren.

3.2.2 Die *function-space* Sichtweise

Eine weitere alternative Sichtweise zu Regressions-Modellen spielt sich, im Gegensatz zum vorhergehenden Abschnitt zur linearen Regression, ausschliesslich im Funktionsraum ab. Hier beschreiben Gauß Prozesse *Verteilungen über Funktionen.* Allgemein kann ein Gauß Prozess (GP) als eine vereinigende Gauß-Verteilung von endlich vielen Zufallsvariablen angesehen werden, mit folgenden Besonderheiten:

Definiton 3.1. *Ein Gauß Prozess ist gegeben, falls für eine vektorielle Zufallsvariable $\vec{X}$ eine beliebige, endliche Untermenge $K \subseteq T$ einer Indexmenge T existiert, für die $\vec{X}_i$ mit $i \in K$ Gauß verteilt ist*[8].

Gauß-Prozesse als Verteilungen über Funktionswerte $f(x)$ sind ferner durch eine *Mittelwertsfunktion* $m(x)$ und eine *Kovarianzfunktion* $k(x, x')$ festgelegt:

$$m(x) = \mathbb{E}(f(x)) \tag{3.13}$$

$$k(x, x') = \mathbb{E}((f(x) - m(x))(f(x') - m(x'))) \tag{3.14}$$

Eine zusammenfassende Schreibweise für einen Gauß-Prozess (GP) ist

$$f(x) \sim \mathcal{GP}(m(x), k(x, x')) \tag{3.15}$$

[7] Diese Idee stützt sich darauf, dass das Produkt $\phi(x)^T \Sigma_p \phi(x')$ als Skalarprodukt (in Bezug auf Σ_p) verstanden werden kann[Rasmus, S.12]

[8] Es sei angemerkt, dass die angesprochene Indexmenge T hier lediglich als Indizierung der einzelnen vektoriellen Ausprägungen $x_i \in \vec{X}$ dient und in keinem Zusammenhang mit den tatsächlichen Dimensionen von x_i steht

Die Sichtweise von GP als Verteilungen über Funktionen mit einem unendlich dimensioniertem Mittelwertsvektor bzw. Kovarianzmatrix ist für die Praxis eher unhandlich wird durch die sogenannte *Marginalisierungs-Eigenschaft* erst praktisch nutzbar. Demnach folgt beispielsweise für zwei Zufallsvariablen[9] z_1 und z_2:

$$\begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \sim \mathcal{N}(\mu, \Sigma) \Rightarrow z_1 \sim \mathcal{N}(\mu_1, \Sigma_{11})$$

$$\text{mit } \mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}$$

Dies ist bei GP durch die Spezifizierung von Einträgen der Kovarianzmatrix durch Funktionswerte der Kovarianzfunktion erfüllt [Rasmus, S.13].

Eine beliebte Wahl der Kovarianfunktion $k(x, x')$ ist die sog. *squared exponential* Kovarianzfunktion[10]:

$$k(x, x') = covSE(x, x') = \sigma_f^2 e^{[\frac{-(x-x')^2}{2l^2}]} \tag{3.16}$$

Hierbei ist σ_f^2 ein Parameter zur Steuerung der maximal möglichen Kovarianz, die sich für $k(x, x')$ ergibt, falls x und x' identisch sind. Der Parameter l wird als *length-scale* Parameter bezeichnet. Bei unterschiedlichen x und x' dient er dazu, den Einfluss der Unterschiedlichkeit von x und x' auf $f(x)$ zu steuern. Vereinfacht formuliert kann durch den *length-scale* Parameter l gesteuert werden, wie weit man im Eingaberaum gehen muss, damit sich Werte von f signifikant ändern. Die Parametermenge einer Kovarianzfunktion $k(x, x')$ wird kurz mit θ (in diesem Fall $\theta = \{\sigma_f^2, l\}$) zusammengefasst angegeben. Diese Parameter einer Kovarianzfunktion werden als *Hyperparameter* bezeichnet.

Die Festlegung einer Kovarianzfunktion für Gleichung 3.15 impliziert eine Verteilung über Funktionen (mit den Hyperparametern als Funktionsparametern). Für Eingaben X_* kann somit folgender Zufallsvektor generiert werden

$$f_* \sim \mathcal{N}(0, K(X_*, X_*)) \tag{3.17}$$

3.2.2.1 Schätzer für f_* (bei rauschfreien Daten)

In der praktischen Anwendung von GP ist man (für den einfachen Fall von rauschfreien Daten) bei gegebenen n Trainingsdaten X bzw. n_* Testdaten

[9] siehe hierzu auch Abschnitt 1.1.5.1 für Rechenregeln zu Gauss-Verteilungen

[10] Oftmals auch als Radial Basis Function (RBF) bezeichnet

X_* an folgender vereinigenden Normalverteilung der Trainigs- und Testfunktionswerte f und f_* interessiert:

$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim \mathcal{N}(0, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix}) \tag{3.18}$$

Hierbei ist $K(X, X_*)$ die $(n \times n_*)$ Kovarianzmatrix mit Kovarianzen zwischen den Trainigsdaten X und den Testdaten X_* (entsprechend $K(X_*, X)$, $K(X, X)$ und $K(X_*, X_*)$).

Um nun bei gegebenen Trainingsdaten X und entsprechenden Funktionswerten f einen geeigneten Schätzer für Funktionswerte f_* der Testdaten X_* zu konstruieren, wird folgende bedingte Wahrscheinlichkeit definiert:

$$\begin{aligned} f_*|X_*, X, f \sim \mathcal{N}(&K(X_*,X)K(X,X)^{-1}f, \\ &K(X_*,X_*) - K(X_*,X)K(X,X)^{-1}K(X,X_*)) \end{aligned} \tag{3.19}$$

3.2.2.2 Schätzer für f_* (bei Daten mit Rauschen)

Für den realistischeren Fall, in welchem (Gleichung 3.2 entsprechend) Funktionswerte $f(x)$ mit Rauschen $\epsilon \sim \mathcal{N}(x|0, \sigma_n^2)$ behaftet sind, ist für Ausgaben $y = f(x) + \epsilon$ die vereinigende Normalverteilung aus Gleichung 3.18 abzuändern in

$$\begin{bmatrix} y \\ f_* \end{bmatrix} \sim \mathcal{N}(0, \begin{bmatrix} K(X,X) + \sigma_n^2 I & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix}) \tag{3.20}$$

(mit I als Einheitsmatrix). Entsprechend der bedingten Wahrscheinlichkeit aus Gleichung 3.19 ist nun stattdessen folgende bedingte Wahrscheinlichkeit für einen Schätzer für f_* definiert

$$\begin{aligned} f_*|X, y, X_* \sim \mathcal{N}(&K(X_*,X)[K(X,X) + \sigma_n^2 I]^{-1}y, \\ &K(X_*,X_*) - K(X_*,X)[K(X,X) + \sigma_n^2 I]^{-1}K(X,X_*)) \end{aligned} \tag{3.21}$$

3.3 Klassifikation mit Gauß Prozessen

Der entscheidene Unterschied zwischen der Verwendung von GP im Regressionsfall und der Verwendung von GP für Klassifikation, liegt in der Relation der Ausgaben y zu Funktionsausgaben f:

Im Klassifikationsfall stehen Ausgaben y zu Funktionswerten $f(x)$ nicht mehr nur durch ein Rauschen $y = f(x)+\epsilon$ zueinander. Stattdessen wird beim Einsatz für binäre Klassifikation z.B. von diskretwertigen Ausgaben $y = 0$ bzw. $y = 1$ für die eine bzw. andere Klasse ausgegangen. Hierbei kommt meist eine sog. *squashing function* $\pi(f)$ (wie z.B. die Sigmoid-Funktion aus

Abbildung 3.2 auf Seite 37) zum Einsatz, welche die Funktionswerte f auf das Intervall $[0, 1]$ abbildet und als Wahrscheinlichkeit für $y = 1$ bei gegebenen f zu verstehen ist. Klassifikation eines Testbeispiels x_* kann durch folgende Schritte beschrieben werden:

1. Auswertung der latenten Funktion f, welche qualitativ die Möglichkeiten der jeweiligen Klassen bei gegebenen Trainingsdaten X modelliert (das ist der GP)
2. Benutzung einer *squashing function* $\pi(f)$ zur Berrechnung der Ausgabe y

In Abbildung 3.4 sind die zwei Schritte zur Klassifikation mit GP schematisch dargestellt.

Abbildung 3.4: Schematische Darstellung der Klassifikation mit Gauss Prozessen

Für die Inferenz bei GP zur Klassifikation wird allgemein folgender Erwartungswert ausgerechnet:

$$p(f_*|X, y, X_*) = \int p(f_*|X, X_*, f)p(f|X, y)df \tag{3.22}$$

Der Term $p(f|X, y)$ stellt hierbei eine posterior Wahrscheinlichkeitsverteilung über f dar (mehr hierzu in Abschnitt 3.3.1). Anschliessend wird mit dem Erwartungswert

$$\overline{\pi_*} = p(y_* = 1|X, y, X_*) = \int \pi(f_*)p(f_*|X, y, X_*)df_* \tag{3.23}$$

eine Wahrscheinlichkeit der Klassenzugehörigkeit von X_* zur Klasse mit $y_* = 1$ errechnet[11].

[11] Die Wahrscheinlichkeit für $p(y_* = 0|X, y, X_*)$ ist durch $1 - p(y_* = 1|X, y, X_*)$ gegeben, womit der Wert 0.5 als sog. Diskriminanzschwelle zur Unterscheidung beider Klassen angesehen werden kann

3.3.1 Trainieren eines GP

Bezüglich Gleichungen 3.19, 3.21 (Seite 44) und Gleichung 3.22 (Seite 45) stellt sich die Frage, wie verschiedene mögliche GP untereinander quantitativ bewertet werden können. Hierzu lässt sich anhand des Bayes Theorems folgende posterior Wahrscheinlichkeit für Ausgaben f eines GP bei gegebenen Trainingsdaten X angeben:

$$p(f|X,y) = \frac{p(y|f)p(f|X)}{p(y|X)} \tag{3.24}$$

Der Term $p(y|f) = \prod_{i=1}^{n} p(y_i|f_i)$ ist als likelihood Wahrscheinlichkeit für y bei gegebenen f zu verstehen und ist in diesem Fall durch die Ausgaben der Sigmoide $\pi(f)$ repräsentiert[12]. Der Term $p(f|X)$ ist eine gaußsche *priori Wahrscheinlichkeit* über f und der Nenner $p(y|X)$ ein Normalisierungsterm, die sog. *marginal likelihood*, mit $p(y|X) = \int p(y|f)p(f|X)$.

Insgesamt lässt sich aber leider zur Errechnung der posterior $p(f|X,y)$ (aufgrund des Produktes einer sigmoiden likelihood mit einer gaußschen priori Wahrscheinlichkeit im Zähler von Gleichung 3.24) keine analytisch errechnete Form angeben. Es wird daher in der Praxis auf Techniken zurückgegriffen, mit denen die posterior $p(f|X,y)$ zumindest approximiert wird. Neben vielen existierenden Approximierungsmethoden hierzu, wie z.B. sog. *mean field* Approximierungen aus der statistischen Physik oder einer Laplace Approximierung (siehe hierzu [OpperWinther] bzw. [Rasmus]), wird meist die in [Minka] vorgestellte *Expectation Propagation* Methode angewandt und so die posterior Wahrscheinlichkeit $p(f|X,y)$ durch

$$q(f|X,y) \cong p(f|X,y)$$

approximativ genähert.

3.3.1.1 Kovarianzfunktionen

Wie bei der Anwendung von GP zur Regression, ist auch für Klassifikationsergebnisse mit GP die Wahl der Kovarianzfunktion sehr bedeutend. Entsprechende Hyperparameter θ einer Kovarianzfunktion werden durch Maximierung der posterior Wahrscheinlichkeit $p(\theta|y,X)$ errechnet. Durch Anwendung des Bayes Theorems kann jedoch die Hyperparametermenge θ stattdessen auch durch Maximierung von $log(p(y|X,\theta))$ mit $p(y|X,\theta) = \int p(y|f)p(f|X)df$ optimiert werden. Die in dieser Arbeit verwendeten Implementierungen zu binärer Klassifikation mit GP aus [Rasmus] benutzen an dieser Stelle zur Optimierung von θ eine spezielle Gradienten Abstiegsmethode.

[12] In [Rasmus] kommt hier z.B. die sog. *cumulative density function* $\Phi(z) = \int_{-\infty}^{z} \mathcal{N}(x|0,1)dx$ als Sigmoide zum Einsatz

Folgend sollen neben der in Gleichung 3.16 (S.43) bereits vorgestellten RBF Kovarianzfunktion noch weitere mögliche Kovarianzfunktionen vorgestellt werden, welche in den Implementierungen zur Klassifikation mit GP aus [Rasmus] zur Verfügung standen.

$$covSEiso(x, x') = \sigma_f^2 e^{(-\frac{1}{2}[(x-x')^T M^{-1}(x-x')]} \tag{3.25}$$

Hierbei ist $covSEiso(x, x')$ eine zur Familie der RBF gehörende Kovarianzfunktion mit isotropischem Abstandsmass, wobei $M = l^2 I$ die Einheitsmatrix multipliziert mit dem (quadrierten) *length-scale* parameter l ist. Die Hyperparametermenge für *covSEiso* ist $\theta = \{l, \sigma_f^2\}$.

Gleichung 3.26 stellt eine lineare Kovarianzfunktion mit nur einem Parameter t vor. Hierbei ist $M = tI$ wieder die Einheitsmatrix multipliziert mit dem Parameter t. Der Term $\frac{1}{t}$ spielt die Rolle eines Biasterms.

$$covLINone(x, x') = [x^T M^{-1} x'] + \frac{1}{t} \tag{3.26}$$

Die Parametermenge für *covLINone* ist $\theta = \{t\}$

3.3.1.2 Automatic Relevance Detection (ARD)

Oftmals ist in der Praxis die Interpretation der Hyperparameterwerte einer Kovarianzfunktion wichtig, falls daraus ein gewisses Verständis für die Eigenschaften der vorliegenden Daten intepretiert werden soll. Mit den beiden folgenden Kovarianzfunktionen *covSEard* und *covLINard* ist eine sog. *automatic relevance detection* möglich:

$$covSEard(x, x') = \sigma_f^2 e^{-\frac{1}{2}[(x-x')^T M^{-1}(x-x')]} \tag{3.27}$$

$$covLINard(x, x') = x^T M^{-1} x' \tag{3.28}$$

In Gleichungen 3.27 und 3.28 ist M eine diagonale $(D \times D)$ Matrix mit den ARD Hyperparametern $l_1^2, l_2^2, ..., l_D^2$ auf der Diagonalen, wobei D die Dimension des Eingaberaums X darstellt. Der Parameter σ_f^2 in Gleichung 3.27 ist, wie gehabt, der Signalvarianzparameter. Die Hyperparametermenge für *covSEard* bzw. *covLINard* ist somit gegeben durch

$$\theta = \{l_1, l_2, .., l_D, \sigma_f^2\}$$

bzw.

$$\theta = \{l_1, l_2, .., l_D\}$$

Die Hyperparameterwerte $l_1, l_2, .., l_D$ spielen hierbei die Rolle von charakteristischen *length-scale* Parametern: Diese treffen (vereinfacht formuliert) eine Aussage darüber, wie groß der Einfluss der jeweiligen Eingabedimensionen auf die Unkorreliertheit der Kovarianzfunktionswerte ist. Durch Invertierung der *length-scale* Parameter wird somit eine Aussage darüber getroffen, wie relevant die entsprechenden Eingabedimensionen sind [Rasmus, S.106].

Es sind neben den hier vorgestellten Kovarianzfunktionen noch etliche andere Kovarianzfunktionen möglich. Bezüglich der Vorraussetzungen, wann eine Funktion als Kovarianzfunktion angesehen werden kann und auf welche Arten sich Kovarianzfunktionen kombinieren bzw. erweitern lassen, sei an dieser Stelle fortführend auf [Rasmus] verwiesen.

3.4 Klassifikation mit GMM

Die in Kapitel 2.3 bereits vorgestellen GMM können ebenfalls als Klassifikatoren eingesetzt werden. Zur Bildung eines Klassifikationsmodells wird hierbei wie folgt vorgegangen: Bei vorliegenden Daten X wird für jede Klasse C_k ein GMM gebildet und dieses auf dessen Daten $X_k \subset X$ trainiert. Der trainierte GMM bildet dann einen Schätzer für die Datenwahrscheinlichkeit $p(X_k|C_k)$. Nun kann zur Schätzung der Wahrscheinlichkeit einer Klasse C_k bei gegebenen Daten X_k das Bayes Theorem angewandt werden durch:

$$p(C_k|X_k) = \frac{p(X_k|C_k)p(C_k)}{p(X)} \tag{3.29}$$

Der Nenner aus Gleichung 3.29 dient hierbei als Normalisierungskonstante und errechnet sich aus[13]

$$p(X) = \sum_{k=1}^{n} p(X_k|C_k)p(C_k) \tag{3.30}$$

Eine Schätzung der apriori Wahrscheinlichkeit $p(C_k)$ ist anteilig gegeben durch die Anzahl von Trainingsbeispielen aus der Gesamttrainingsmenge X, die zur Klasse C_k gehören, d.h. $\frac{\#(X_k)}{\#(X)}$.

Die Vorgehensweise zur binären Klassifikation mit GMM lässt sich in folgenden Punkten zusammenfassen:

1. Für ein binäres Klassifikationsproblem mit zwei Klassen A und B mit ihren Daten X_A bzw. X_B bilde zunächst die beiden Modelle GMM_A und GMM_B und trainiere GMM_A auf X_A bzw. GMM_B auf X_B.

2. Errechne für die zu klassifizierenden Testdaten X_{Test} jeweils die posterior Wahrscheinlichkeiten $p_A = p(A|X_{Test})$ bzw. $p_B = p(B|X_{Test})$

3. Entscheide für Klasse A falls $p_A > p_B$ oder für Klasse B falls $p_B > p_A$ (falls $p_A = p_B$ ist kein Klassenentscheid möglich)

Die Wahl der Anzahl bzw. Form der Komponenten (isotropisch, diagonal oder voll) eines GMM ist in der Praxis anwendungsbezogen und bedarf

[13] der Summenindex k in Gleichung 3.30 läuft über alle vorliegenden n Klassen

häufig an Vorwissen über die Verteilung und Lage der Trainingsdaten. Hierbei ist es z.B. hilfreich für Messdaten mit zwei Messmerkmalen, diese sich vorher entsprechend graphisch zu veranschaulichen, um mögliche Hinweise für eine gute Wahl an Anzahl bzw. Form der Komponenten zu bekommen.

3.5 Bewertungsverfahren für Klassifizierer

Im diesem Abschnitt werden spezielle Techniken vorgestellt, die zur Bewertung verschiedener Klassifizierungsverfahren in dieser Arbeit implementiert wurden. Zum einen bestehen diese aus speziellen Kreuzvalidierungstechniken, zum anderen aber auch durch ein einfaches Verfahren zur Erstellung eines sog. *Reciever Operating Characteristic* (kurz ROC) Graphen. Zum leichteren Verständnis der im Folgekapitel aufgelisteten Ergebnisse zu diversen Experimenten, soll hier auf diese Verfahren kurz eingegangen werden.

3.5.1 Techniken der Kreuzvalidierung (CV)

Allgemein bestehen Verfahren zur Kreuzvalidierung von Klassifizierern darin, eine vorhandene Datenmenge in eine Trainings- und Testmenge mit jeweils bestimmter Größe zu unterteilen und die Klassifikationsmethodiken auf diese anzuwenden. Die Generalisierungsfähigkeit kann, da die Klassenzugehörigkeiten sowohl für Trainings- als auch Testmenge vorher bekannt sind, so gut überprüft werden. Ferner können auch klassen- oder individuenspezifische Untersuchungen vorgenommen werden, z.B. zu Fragestellungen wie: *Welcher Klassifikator erkennt eine bestimmte Klasse besonder gut?* oder *Welche Individuen werden besonders oft fehlklassifiziert?* Für diese Arbeit wurden hierzu drei verschiedene Methodiken umgesetzt, welche nun erläutert werden sollen:

3.5.1.1 Trainingsfehler

Zur Ermittlung eines sogenannten Trainingsfehlers wird die Datenmenge X sowohl als Trainings- als auch Testmenge verwendet. Hiebei kann geprüft werden, ob ein bestimmter Klassifikator fähig ist, die ihm im Training präsentierten Daten allesamt beim Testen *korrekt wiederzuerkennen.*

3.5.1.2 Leave-One-Out Kreuzvalidierung (LOO-CV)

Die Leave-One-Out Kreuzvalidierung ist für eine Datenmenge X mit den Elementen x_i in Pseudo-Code im Algorithmus 1 dargestellt. Die Bedeutung folgender Bezeichner ist hierbei:

1. `DoTraining(TrainSet)`

   → Methode, die auf einem gegebenen Datensatz $TrainSet$ einen binären Klassifikator trainiert

2. `p = Classify(TestSet)`

   → Funktion, die auf einer gegebenen Testmenge $TestSet$ eine binäre Klassifikation durchführt und einen Vektor p mit Wahrscheinlichkeiten für die Klassenzugehörigkeit der einzelnen Elemente aus $TestSet$ liefert

3. `EvaluateClassification(p,Y)`

   → Methode zur Bewertung der prognostizierten Klassenzugehörigkeiten p anhand der tatsächlichen Klassenzugehörigkeiten (Targets) Y der Testmenge $TestSet$.

In der LOO-CV wird über alle n Elemente der Datenmenge $X = \{x_1, .., x_n\}$ iteriert: Im i'ten Schritt wird x_i aus X entfernt und dient als Testmenge. Übrigbleibende $n - 1$ Datenelemente dienen als Trainingsmenge. Nach dem Trainieren und anschliessendem Klassifizieren der Testmenge $\{x_i\}$ wird die Klassenprognose p bezüglich der tatsächlichen Klassenzugehörigkeit y_i bewertet.

```
1  foreach x_i ∈ X do
2  |  X_TrainSet ← X\{x_i}
3  |  DoTraining (X_TrainSet)
4  |  p ← Classify ({x_i})
5  |  EvaluateClassification (p, {y_i})
   end
```

Algorithmus 1 : Leave-One-Out Kreuzvalidierung (LOO-CV)

3.5.1.3 90/10 Kreuzvalidierung (90/10 CV)

Die 90/10 Kreuzvalidierung ist in Algorithmus 2 als Pseudocode aufgeführt. Zu den Bereits bei der Leave-One-Out aufgeführten Bezeichnern im Pseudocode kommen folgende:

- `[XA, XB] = DevideClassSets(X)`

  → Funktion, die eine gegebene Datenmenge X in zwei Untermengen X_A und X_B unterteilt, wobei X_A Elemente der Klasse A und X_B alle Datenelemente der Klasse B enthält.

- `[TrainSet,TestSet] = Shuff9010(X)`

  → Funktion, die die disjunkten Mengen $TrainSet$ und $TestSet$ zurückliefert, wobei $TrainSet$ zufällige 90% und $TestSet$ zufällige 10% der Elemente von X enthält.

Bei der 90/10 Kreuzvalidierung wird eine Datenmenge X zunächst in die beiden Untermengen X_A und X_B unterteilt, wobei X_A alle Stichprobenwerte

der ersten Klasse und X_B die der zweiten Klassen enthält. Hiernach wird in $MaxIter$ Durchläufen folgendes durchgeführt:

1. Zunächst werden beide Stichprobenmengen getrennt durchmischt und aus X_A und X_B jeweils die Trainingsmengen $X_{TrainSetA}$ und $X_{TrainSetB}$ bzw. die Testmengen $X_{TestSetA}$ und $X_{TestSetB}$ gebildet, wobei die Trainingsmengen $X_{TrainSetA}$ und $X_{TrainSetB}$ 90% der Individuen von X_A bzw. X_B enthalten und die Testmengen $X_{TestSetA}$ und $X_{TestSetB}$ die restlichen 10%

2. Die beiden Mengen $X_{TrainSetA}$ und $X_{TrainSetB}$ bzw. $X_{TestSetA}$ und $X_{TestSetB}$ werden zu einer großen Trainingsmenge $X_{TrainSet}$ bzw. Testmenge $X_{TestSet}$ zusammengefasst[14]

3. Training auf der Trainigsmenge $X_{TrainSet}$

4. Klassifikation auf der Testmenge $X_{TestSet}$

5. Bewertung der Klassifikation anhand der Klassenzugehörigkeits-Prognosen p und den durch $Y_{TestSet}$ gegebenen tatsächlichen Zugehörigkeiten der Testmenge $X_{TestSet}$

```
1 [X_A, X_B] ← DevideClassSets (X)
2  for i ← 1 to MaxIter do
3  |  [X_TrainSetA, X_TestSetA] ← Shf9010 (X_A)
4  |  [X_TrainSetB, X_TestSetB] ← Shf9010 (X_B)
5  |  X_TrainSet ← X_TrainSetA ∪ X_TrainSetB
6  |  X_TestSet ← X_TestSetA ∪ X_TestSetB
7  |  DoTraining (X_TrainSet)
8  |  p ← Classify (X_TestSet)
9  |  EvaluateClassification (p, Y_TestSet)
   end
```

Algorithmus 2 : 90/10 Kreuzvalidierung (90/10 CV)

[14]Das zufällige getrennte Mischen der Daten erfolgte hier aufgrund evtl. unterschiedlicher Anzahl von vorliegenden Stichproben jeder Klasse und damit in der letztendlichen Trainings- bzw. Testmenge immer 90 bzw. 10% der Stichprobenwerte einer Klasse vertreten sind

3.5.2 Reciever Operating Characteristics (ROC) Analyse

Eine Reciever Operating Characteristics (ROC) Analyse ist eine Visualisierungstechnik zur Bewertung der Performanz von Klassifizierern. Während dieses Bewertungsverfahren in diversen wissenschaftlichen Bereichen bereits seit 1977 zum Einsatz kommt, wird es im Bereich des Maschinellen Lernens erst seit ein paar Jahren angewendet[Fawcett]. Aufgrund ausgiebiger Anwendung im medizinischen Bereich und des gleichzeitigen medizinischen Charakters des in dieser Arbeit verwendeten Datensatzes zu Experimenten (siehe hierzu auch Kapitel 4), sollen in diesem Abschnitt kurz die Grundbegriffe dieses Verfahrens vorgestellt werden.

3.5.2.1 Klassifikationsperformanz

Zunächst seien mögliche, tatsächliche Klassenzugehörigkeiten von zu klassifizierenden Objekten in einem binären Klassifikationsproblem durch die Menge {p, n} (positive/negative) abstrahiert. Bei Klassifizierern, die zu klassifizierende Objekte auf *Klassenlabels* abbilden, wird zwischen *harten* und *weichen* Klassifizierern unterschieden: Harte Klassifizierer ordnen den zu klassifizierenden Objekten diskretwertige Klassenzugehörigkeiten zu. Dagegen drücken weiche Klassifizierer den Grad einer Klassenzugehörigkeit durch reelwertige Ausgabewerte (wie z.B. einer Wahrscheinlichkeit einer Klassenzugehörigkeit) aus. Die Klassenzuordnung eines Objektes durch einen Klassifizierer sei nun ebenfalls durch eine Menge {Y,N} (Yes/No) abstrahiert[15]. Bei einem vorliegenden Klassifizierer sind also 4 mögliche Szenarien von Klassifikationen zu erwarten:

1. Ein Objekt ist tatsächlich (p)ositive und wird durch den Klassifikator durch die Vorhersage (Y)es auch als solches erkannt. In diesem Fall wird das Objekt dann als sog. ***true positive*** angesehen.

2. Ein Objekt ist tatsächlich (p)ositive und wird durch den Klassifikator durch die Vorhersage (N)o nicht erkannt. In diesem Fall wird das Objekt als sog. ***false negative*** angesehen.

3. Ein Objekt ist tatsächlich (n)egative und wird durch den Klassifikator durch die Vorhersage (Y)es nicht erkannt. In diesem Fall wird das Objekt als sog. ***true negative*** angesehen.

4. Ein Objekt ist tatsächlich (n)egative und wird durch den Klassifikator durch die Vorhersage (N)o erkannt. In diesem Fall wird das Objekt als sog. ***false positive*** angesehen.

Folgende zwei weitere wichtige Messgrößen zu ROC:

[15] wobei dies im Falle von weichen Klassifizierern durch die vorhergehende Festlegung eines sog. Treshholds bzw. einer Diskriminanzschwelle geschieht und bei Über- bzw. Unterschreitung dieser, für die eine bzw. andere Klasse entschieden wird

- Die ***Spezifität*** berechnet sich aus

$$specifity = \frac{\text{Anzahl true negatives}}{\text{Anzahl false positives + Anzahl true negatives}}$$

- Die ***Sensitivität*** berechnet sich aus

$$sensitivity = \frac{\text{Anzahl true positives}}{\text{Anzahl false negatives + Anzahl true positives}}$$

3.5.2.2 Reciever Operating Characteristic Kurve

Im Allgemeinen entspricht der Spezifitäts- und Sensitivitätswert eines Klassifizierers zu einem Test einem Punkt im zweidimensionalen Spezifitäts-/Sensitivitätsraums. Für einen weichen Klassifizierer mit unterschiedlichen Diskriminanzschwellen[16] entsteht so eine sog. ROC-Kurve.

Abbildung 3.5: Beispiel zweier Roc-Kurven

Hierbei lassen sich bei jeweiligem Verlauf einer ROC Kurve verschiedene Interpretationen zum entsprechenden Klassifikator anstellen:

- entspricht z.B. die ROC Kurve einer Diagonalen, so ist der Test nicht besser als eine Zufallsentscheidung, also der Klassifikator unbrauchbar.

- Beim Vergleich von zwei verschiedenen Klassifikatoren wie die in Abbildung 3.5 gezeigte blaue bzw. rote Kurve, wäre der Klassifikator der blauen Kurve aufgrund von stets größeren Funktionswerten ein besserer Diskriminator als der zur roten Kurve zugehörige Klassifikator.

Auf diese Art können verschiedene Klassifizierer (oder ein Klassifizierer mit verschiedenen Diskriminanzschwellen) untereinander verglichen werden.

[16] gemeint ist ein Schwellwert, der die Entscheidungsgrenze eines binären Klassifizierers darstellt

Kapitel 4

Experimente und Ergebnisse

Nur ein Narr macht keine Experimente
- Charles Darwin

Das folgende letzte Kapitel dieser Arbeit stellt Experimente und deren Ergebnisse zu den bisher vorgestellten Methodiken aus dem un- bzw. überwachten Lernen vor. Diese stellen aufgrund des verwendeten Datensatzes eine praxisnahe Anwendung der bereits vorgestellten Methodiken dar. Im folgenden soll zunächst der verwendete Datensatz vorgestellt und auf zusätzliche Details in Bezug auf die Umsetzung und Implementierung der jeweiligen Methodiken eingegangen werden. Einer Auflistung von Ergebnissen folgen jeweils unmittelbar Beobachtungen zu diesen.

4.1 Datensatz und Hilfsmittel

4.1.1 Verwendeter Datensatz

Der in den Experimenten verwendete Datensatz stammt aus medizinischen Messungen von Nebennieren-Tumor Patienten. Die Nebenniere ist ein hormonproduzierendes Organ, welches sog. Corticosteroide (Stereoid-Hormone) herstellt. Hintergrund dieser Messungen ist folgender: Die Diagnose einer Nebennieren-Karzinom Erkrankung bedarf grundsätzlich Blutproben eines Patienten. Die hier vorliegenden Messdaten dagegen wurden mit Hilfe einer Spektrogramm-Analyse des Urins über bestimmte Stoffkonzentrationen (Metaboliten[1] der besagten Corticosteroide) gewonnen. Folgend die Anzahl an Patienten für die jeweilig vorliegenden Klassen {ACA,ACC,CTRL}:

- 113 **ACA** Patienten (gutartiges Karzinom)
- 45 **ACC** Patienten (bösartiges Karzinom)
- 88 **CTRL** Patienten (gesunde Patienten)

[1] Dies sind Zwischenprodukte in biochemischen Prozessen

Es lagen Daten zu folgenden 32 Messmerkmalen vor:

Nr.	Merkmalsname	**Nr.**	Merkmalsname
1.	ANDROS	17.	PT
2.	ETIO	18.	PT-ONE
3.	DHEA	19.	THS
4.	16OH-DHEA	20.	Cortisol
5.	5-PT	21.	6b-OH-F
6.	5pd-plus-pregna	22.	THF
7.	THA	23.	5a-THF
8.	5a-THA	24.	a-cortol
9.	THB	25.	b-cortol
10.	5a-THB	26.	11b-OH-ANDRO
11.	3a5b-THALDO	27.	11b-OH-ETIO
12.	TH-DOC	28.	Cortisone
13.	5a-TH-DOC	29.	THE
14.	PD	30.	a-cortolone
15.	3a5a 17HP	31.	b-cortolone
16.	17-HP	32.	11-OXO-Et

Tabelle 4.1: Messmerkmale des verwendeten Datensatzes

Die Nummerierung der Merkmale aus Tabelle 4.1 dient als Referenz, d.h. Merkmale sind in folgenden Tests häufig kurz durch ihre entsprechende Nummer vertreten (statt durch ihren vollen Namen). In folgenden Ausführungen wird für die Untersuchung von zwei Klassen, z.B. ACA und ACC, kurz *ACA vs. ACC* oder *ACA/ACC* notiert.

4.1.1.1 Vorverarbeitung der Messdaten

Vor dem Einsatz in den Experimenten wurde der Datensatz als solcher aus folgenden Gründen vorverarbeitet:

1. Die Maßeinheiten der Merkmalsmessungen sind teilweise sehr unterschiedlich gewesen, zudem existierten Messwerte, die charakteristisch für *Mess-Ausreißer* (verursacht durch evtl. Messfehler) waren und die die Ergebnisse ungünstig beeinflussen könnten[2].

2. Messungen zu bestimmten Merkmalen bei einigen Patienten waren unvollständig (im vorliegenden Datensatz betraff dies insgesamt 56

[2]diese Beobachtung entstand durch vorhergende Analyse der Wertebereiche aller Messmerkmale

fehlende Messwerte der Klasse **ACC** mit 45 Patienten und 471 Messwerte der Klasse **CTRL** mit 88 Patienten[3]). Das Vorhandensein war aber bei den vorgestellten Methodiken Vorraussetzung für deren Anwendbarkeit. Einem Entfernen der entsprechenden Patienten aus dem Datensatz widersprach die ohnehin geringe Größe des Datensatzes und die Gefahr des Verlorengehens evtl. nützlichen Wissens.

Bezüglich obiger Punkte wurde der Roh-Datensatz X zur Glättung zunächst logarithmiert:

$$X_{Neu} = log(X) = \{x_1, x_2, x_3, ..., x_n\}$$

Hiernach wurde aus X_{Neu} ein Mittelwertsvektor μ errechnet, indem Mittelwerte für die d-Merkmale der Patienten $x_i \in X_{Neu}$ errechnet wurden:

$$x_i = \begin{pmatrix} x_{i1} \\ x_{i2} \\ . \\ . \\ x_{id} \end{pmatrix}, \mu = \begin{pmatrix} \mu_1 \\ \mu_2 \\ . \\ . \\ \mu_d \end{pmatrix} \text{ und } \mu_k = \frac{1}{n} \sum_{i=1}^{n} x_{ik},$$

wobei μ_k als k-te Komponente von μ den Mittelwert des k-ten Merkmals aus X_{Neu} darstellt. Im Anschluss wurde dieser Mittelwert μ_k als Ersatzwert für jedes nicht vorhandene k-te Merkmal eines i-ten Patienten eingesetzt: $x_{ik} = \mu_k$.
Zur Erhaltung der Klassencharakteristika und der Gefahr diese durch künstliche Schaffung von Messwerten evtl. zu verfälschen, wurde diese Ersetzung von fehlenden Messwerten jeweils getrennt auf den einzelnen Klassendaten durchgeführt.

Zu den Experimenten in folgenden Abschnitten wurden stets die, wie hier geschilderten, vorverarbeiteten Daten verwendet.

4.1.2 Verwendete Hilfsmittel

Aufgrund des Umfangs des Themengebietes, wurden bestimmte bereits existierende Implementierungen als Startpunkt für eigene Implementierungen vom Aufgabensteller empfohlen und zugelassen. Beim *unüberwachten Lernen* wurden Implementiereungen hauptsächlich aus [NabNey] verwendet und erweitert. Eigene Implementierungen zur binären Klassifikation mit Gaußschen Prozessen (und den *Automatic Relevance Detection* Varianten dieser) bauten auf dem Softwarepaket aus [Rasmus] auf. Für beide Software Pakete als auch die eigenen Erweiterungen diente MATLAB als Entwicklungsumgebung. Ferner wurde folgende Soft- und Hardeware Konfigurationen verwendet[4]:

[3] Das sind im Falle der Klasse ACC ca. 3% und für die Klasse CTRL 16 % fehlende Daten

[4] diese Angaben dienen als Richtwerte für die evtl. Reproduktion einiger Experimente, die in bestimmten Fällen mehrere Stunden bis zu einem Tag andauerten

- PC mit WindowsXP
- MATLAB 7.1 (R14)
- AMD 3200GHz Prozessor
- 1.5Gb RAM Arbeitsspeicher

4.2 Unüberwachtes Lernen

Folgender Abschnitt listet die Versuchsergebnisse zur Hauptkomponentenanalyse der Messdaten auf. Da die Ergebnisse der PPCA Untersuchungen qualitativ identisch zu denen der PCA Untersuchungen waren, sind hier lediglich die Ergebnisse zu den PCA Untersuchungen aufgelistet. Ergebnisse zu Untersuchungen mit GMM sind, aufgrund des Einsatzes zur Klassifikation, im Kapitel 4.3.3 zu finden.

4.2.1 Untersuchungen der Messdaten durch Hauptkomponentenanalyse

Bezüglich der folgenden Ergebnisse zu Untersuchungen der Patientendaten durch die PCA Analyse soll das Hauptaugenmerk darauf liegen, wie gut Klassenunterschiede durch die Hauptkomponenten erfasst werden können. Eine Darstellung der Daten im 2-D Hauptkomponentenraum[5] soll zudem visuell eine Einschätzung zur Separierbarkeit der Klassen ermöglichen.

4.2.1.1 Versuchsanordnung

Auf den Messdaten (alle Merkmale), die wie in Kapitel 4.1.1.1 vorverarbeitet wurden, wurde eine PCA Analyse durchgeführt. Anschliessend wurden die Messdaten in den durch die beiden ersten Hauptkomponenten *COMP1* und *COMP2* aufgespannten Raum projiziert.

4.2.1.2 Ergebnisse

Die Ergebnisse sind jeweils in den Abbildungen 4.1 bis 4.3 zu sehen, wobei linke Plots die Daten im Hauptkomponentenraum zeigen. Die rechten Plots in den Abbildungen 4.1 bis 4.3 zeigen wieviel Varianz der Messdaten durch die einzelnen Hauptkomponenten anteilig erklärt wird (blaue Balken)[6]. Zudem werden in den rechten Plots ebenfalls durch die blaue Kurve die kumulierten Varianzanteile dargestellt.

[5] aufgespannt durch die beiden ersten Hauptkomponenten

[6] Es werden nur die ersten 10 Hauptkomponenten, die kumuliert insgesamt ca. 90% der Datenvarianz erklären, gezeigt

Abbildung 4.1: Daten ACA vs. ACC projiziert auf die ersten beiden PCA Hauptkomponenten

Abbildung 4.2: Daten ACA vs. CTRL projiziert auf die ersten beiden PCA Hauptkomponenten

Abbildung 4.3: Daten ACC vs. CTRL projiziert auf die ersten beiden PCA Hauptkomponenten

4.2.1.3 Beobachtungen

Folgende Beobachtungen zu den linken Plots der Abbildungen 4.1 bis 4.3 sind möglich: Die Daten liegen für *ACA vs. ACC* bzw. *ACA vs. CTRL* nur moderat getrennt, für *ACC vs. CTRL* jedoch etwas besser, jedoch auch nicht mit einer eindeutigen sauberen Trennung der Klassenbereiche[7]. D.h. die ersten beiden Hauptkomponenten erfassen hier Eigenschaften der Daten, die zur Unterscheidung zwischen Patienten zu *ACA vs. ACC* bzw. *ACA vs. CTRL* nur mäßig, bei *ACC vs. CTRL* (wenn auch nicht völlig zufriedenstellend) schon besser dienen. Auffällig sind ebenfalls die Varianzanteile der Hauptkomponenten in den rechten Plots aus Abbildungen 4.1 bis 4.3: Während für *ACA vs. CTRL* der Anteil an erklärender Varianz für die ersten beiden Hauptkomponenten nur ca. 52% beträgt (siehe blaue Kurve im rechten Plot aus Abb. 4.2), liegt diese für *ACA vs. ACC* bzw. *ACC vs. CTRL* bei ca. 65% bzw. 62% (blaue Kurven in rechten Plots aus Abb. 4.1 und 4.3).

Eine Annahme, dass Klassenunterschiede (mit Hilfe der ersten zwei Hauptkomponenten) bei *ACA vs. CTRL* schwieriger als bei *ACA vs. ACC* bzw. *ACC vs. CTRL* zu erfassen sind, ist rein spekulativ aber möglich.

[7] D.h. es liegen zumindest keine linear trennbaren Bereiche vor

4.3 Überwachtes Lernen

Im folgenden sind Ergebnisse zu Untersuchungen mit Methodiken des überwachten Lernens aufgeführt. Obwohl die GMM thematisch dem unüberwachten Lernen untergeordnet worden sind (siehe Kapitel 2.3), finden sich Ergebnisse zu GMM, aufgrund der Benutzung zur Klassifikation, ebenfalls hier. Die ausschliessliche Auflistung von GP Ergebnissen mit den beiden Kovarianzfunktionen **covSEard** und **covLINard** (siehe Kapitel 3.3.1.2) hat folgende Gründe:

- Zum einen waren Ergebnisse zu diesen beiden Kovarianzfunktionen in Bezug auf Performanz des Klassifikators am vielversprechensten[8]. Eine vollständige Auflistung von Ergebnissen zu allen vorgestellten Kovarianzfunktionen hätte zudem den Rahmen dieses Kapitels gesprengt.
- Ferner wurden Untersuchungen im Hinblick auf Merkmalsstärken des Datensatzes durchgeführt, die nur durch ARD Kovarianzfunktionen durchgeführt werden konnten.

Klassifikatoren wurden ausschliesslich zur binären Klassifikation verwendet, d.h. verwendete Daten $X = X_{C_i} \cup X_{C_j}$ zu einem einzelnen Test wurden durch Zusammenschluss der Daten X_{C_i} und X_{C_j} zweier Klassen C_i,$C_j \in \{ACA, ACC, CTRL\}$ mit $i \neq j$ erzeugt.

In folgenden Untersuchungen werden häufig Indizes fehlklassifizierter Patienten angegeben. Diese Indizes beziehen sich auf die vom Aufgabensteller zur Verfügung gestellten Datentabellen zu den einzelnen Patientenklassen und sind als Zeilennummern der entsprechenden Datentabellen zu verstehen (d.h. ein fehlklassifizierter ACC Patient mit dem Index 3 bezieht sich auf den in Zeile 3 aufgeführten Patienten aus der Datentabelle zur Klasse ACC).

[8] In vorhergehenden Untersuchungen schnitten diese deutlich besser ab als z.B. *covSEiso* (siehe S. 47)

4.3.1 Untersuchungen zu Merkmalsstärken mit ARD Kovarianzfunktionen

Im folgenden Unterkapitel werden Ergebnisse zu Untersuchungen von Merkmalsstärken mit Hilfe der zwei ARD Kovarianzfunktionen **covSEard** und **covLINard** vorgestellt. Hauptaugenmerk lag hierbei darauf, ob es bestimmte Merkmale bzw. Merkmalsreihenfolgen gibt, die beim Vergleich zweier Klassen evtl. einen Hinweis auf bestimmte klassentypische Eigenschaften geben könnten[9].

4.3.1.1 Versuchsanordnung

Zunächst wurde durch Zusammenschluss der Daten X_{C_i} und X_{C_j} zweier Klassen C_i bzw. $C_j \in \{ACA, ACC, CTRL\}$ mit $i \neq j$ eine Trainingsmenge $X_{Train} = X_{C_i} \cup X_{C_j}$ gebildet. Diese enthielt alle 32 Messmerkmale. Hiernach wurde ein GP mit einer der beiden ARD-Kovarianzfunktionen **covSEard** bzw. **covLINard** auf der Trainingsmenge X_{Train} trainiert.

4.3.1.2 Ergebnisse

Zunächst sind in Tabelle 4.2 die Merkmalsrangfolgen in Bezug zu den entsprechenden Hyperparameterstärken sortiert von stark nach schwach (nach dem Trainieren) aufgeführt. Hierbei ist die Rangfolge stark nach schwach von oben nach unten zu lesen. Die aufgeführten Merkmalsindizes beziehen sich wieder auf die bereits in Tabelle 4.1 aufgeführte Nummerierung. Hiernach zeigt also die erste Spalte die Rangfolge der Hyperparameter nach dem Trainieren mit der Kovarianzfunktion **covSEard** auf den durch die Klassen ACA und ACC gebildeten Daten, wobei das 18. Merkmal das stärkste, das dritte Merkmal das zweitstärkste ist usw. Korrespondierend zu den sechs Spalten der Tabelle 4.2 sind in den Abbildungen 4.4 bis 4.9 die am Ende des Trainings vorliegenden Ausprägungen der *length-scale* Hyperparameter $l_1, l_2, .., l_{32}$ als Plots abgebildet. Alle Abbildungen bestehen aus zwei Plots, wobei der erste (obere) Plot die Hyperparameterstärken darstellt[10]. Im zweiten (unteren) Plot sind die Hyperparameterwerte jeweils nochmals mit der Standardabweichung der Merkmale multipliziert zu sehen. Auf den X-Achsen sind entsprechend die Merkmals-Indizes aus Tabelle 4.1 aufgetragen.

[9] Dies könnte z.B. aus medizinischer Sicht bzgl. Diagnostik interessant sein

[10] length-scale Hyperparameterwerte l_i wurden durch die verwendeten Implementierungen zu GP aus [Rasmus] als $log(l_i)$ zurückgeliefert, daher wurde vor dem Plotten auf diese die e-Funktion angewendet

covSEard			**covLINard**		
1	2	3	4	5	6
ACA/ACC	ACA/CTRL	ACC/CTRL	ACA/ACC	ACA/CTRL	ACC/CTRL
18	3	3	3	3	16
3	18	18	28	4	14
4	27	27	31	27	21
11	14	14	4	15	18
27	4	25	15	12	15
16	16	16	1	2	4
13	32	32	16	9	23
15	13	4	23	16	27
32	25	23	21	17	32
9	11	9	18	11	17
19	12	26	7	24	25
5	10	10	14	30	2
14	9	13	29	10	10
21	26	31	22	31	26
2	15	2	24	7	13
23	29	6	32	25	9
6	2	1	2	13	31
28	17	7	9	22	24
25	22	29	13	14	20
10	31	21	6	32	22
26	30	24	30	18	28
20	5	22	10	21	29
7	6	17	27	26	3
1	7	28	20	5	6
12	24	20	12	1	7
31	1	12	11	29	1
17	19	15	17	8	12
24	8	30	26	6	11
30	23	8	5	20	5
29	28	11	8	23	30
8	20	5	19	28	8
22	21	19	25	19	19

Tabelle 4.2: Merkmalsreihenfolgen nach Hyperparameterstärken ermittelt durch **covSEard** und **covLINard**

Abbildung 4.4: Hyperparameterverlauf ACA vs. ACC mit Kovarianzfunktion covSEard

Abbildung 4.5: Hyperparameterverlauf ACA vs. CTRL mit Kovarianzfunktion covSEard

Abbildung 4.6: Hyperparameterverlauf ACC vs. CTRL mit Kovarianzfunktion covSEard

Abbildung 4.7: Hyperparameterverlauf ACA vs. ACC mit Kovarianzfunktion covLINard

Abbildung 4.8: Hyperparameterverlauf ACA vs. CTRL mit Kovarianzfunktion covLINard

Abbildung 4.9: Hyperparameterverlauf ACC vs. CTRL mit Kovarianzfunktion covLINard

4.3.1.3 Beobachtungen

In Bezug auf die Kovarianzfunktion **covSEard** fallen die Merkmale 18 und drei auf, welche in allen Klassenvergleichen (Spalten eins bis drei der Tabelle 4.2) an der Spitze zu finden sind. Hierbei scheint Merkmal drei besonders ACA bzw. ACC typische Eigenschaften zu erfassen: z.B. ist der Ausschlag der Hyperparameterkurve bei Merkmal drei beim Vergleich *ACA vs. CTRL* (Abb. 4.5) bzw. *ACC vs. CTRL* (Abb. 4.6) im Vergleich zu anderen Hyperparametern besonders hoch. Bei einem Vergleich *ACA vs. ACC* jedoch ist der Ausschlag der Hyperparameterkurve bei Merkmal drei im Vergleich zu allen anderen Merkmalen nicht mehr so dominant (Abb. 4.4).

In Bezug auf die Merkmalsrangfolgen bei Benutzung der Kovarianzfunktion **covLINard** (Spalten drei bis sechs der Tabelle 4.2) ist kein eindeutiger Trend (wie bei den beiden Merkmalen 18 und drei für **covSEard**) bei den Klassenvergleichen zu erkennen. Hierzu ist zudem in den Abbildungen 4.7 bis 4.9 auffällig, dass es keine eindeutig dominanten Ausschläge der Hyperparameterkurven bei bestimmten Merkmalen gibt (wie z.B. in Abb. 4.6).

4.3.2 Klassifikation mit GP

Folgendes Kapitel beinhaltet hauptsächlich Ergebnisse zur Anwendung der in Kapitel 3.5 bereits vorgestellten Krossvalidierungsverfahren *LOO-CV* und *90/10 CV* bzw. der ROC-Analyse. Vorhergende Untersuchungen zur Klassifikation mit GP zeigten die beiden Kovarianzfunktionen **covSEard** und **covLINard** bei den vorliegenden Daten hinsichtlich der Klassifikationsperformanz als die am vielversprechensten. Daher sind aufgrund des Umfangs der Testergebnisse im Folgenden lediglich die Testergebnisse zu diesen beiden Kovarianzfunktionen aufgeführt.

4.3.2.1 Trainingsfehler auf komplettem Merkmalsraum

Der in Kapitel 3.5.1.1 bereits vorgestellte Trainingsfehler war für alle Klassenkonstellationen *(ACA vs. ACC)*, *(ACA vs. CTRL)* bzw. *(ACC vs. CTRL)* mit Betrachtung aller Merkmale für beide Kovarianzfunktionen **covSEard** und **covLINard** fehlerlos, d.h. es wurde jeweils nach dem Trainieren die Trainingsmenge selber als Testmenge verwendet und alle Klassenindividuen korrekt klassifiziert.

4.3.2.2 Versuchsanordnung zu *LOO-CV*

Die LOO-CV wurde entsprechend dem Algorithmus 1 aus Kapitel 3.5.1.2 durchgeführt. Für jeweils einen LOO-CV Durchlauf wurde ein GP mit einer der beiden Kovarianzfunktionen **covSEard** bzw. **covLINard** auf dem kompletten Merkmalsraum trainiert und klassifiziert.

4.3.2.3 Ergebnisse LOO-CV

In Tabelle 4.3 sind spaltenweise die Indizes aller Patienten aufgelistet, die nach einem LOO-CV Durchlauf fehlklassifiziert wurden. D.h. in Spalte eins sind Indizes der Patienten aufgeführt, die nach der LOO-CV auf einem GP mit der Kovarianzfunktion **covSEard** zur binären Klassifikation *ACA vs. ACC* fehlklassifiziert wurden usw. Die letzte Zeile enthält die prozentualen Fehlklassifikationsraten nach einem kompletten LOO-CV Durchlauf. Beispielsweise waren bei LOO-CV zu *ACA vs. ACC* (Spalte eins) ein ACA und sieben ACC Patienten fehlklassifiziert, was bei ingesamt 113 ACA und 45 ACC Patienten einem Fehler von $\frac{1}{113} \approx 0.8\%$ für ACA und $\frac{7}{45} \approx 15.5\%$ für ACC entspricht.

covSEard						**covLINard**					
1		2		3		4		5		6	
ACA	ACC	ACA	CTRL	ACC	CTRL	ACA	ACC	ACA	CTRL	ACC	CTRL
78	13	14	1	26		8	2	14	4	2	4
	24	40	3			19	5	61	8	26	
	25	53	4			55	10	70	13	28	
	26	61	8			57	13	76	14	39	
	28	76	13			73	24	99	15		
	32	78	14			90	26		43		
	39	80	15				39		44		
			18								
0.8%	15.5%	6.1%	9%	2.2%	0%	5.3%	15.5%	4.4%	7.9%	3.5%	1.1%

Tabelle 4.3: Indizes fehlklassifizierter Patienten nach LOO-CV auf 32 Merkmalen

4.3.2.4 Beobachtungen LOO-CV

Folgende Punkte sind im Vergleich zwischen den beiden Kovarianzfunktionen **covSEard** und **covLINard** in Tabelle 4.3 auffällig:

1. **covSEard** scheint für eine Unterscheidung *ACC vs. CTRL* besser geeignet zu sein als **covLINard** (Spalten drei und sechs)

2. Bzgl. *ACA vs. ACC* fällt die geringe Anzahl an fehlklassfizierten für ACA in Spalte eins gegenüber den fehlklassfizierten ACA in Spalte vier auf, womit **covSEard** beim Vergleich von *ACA vs. ACC* zur Ermittlung von ACA typischen Eigenschaften besser abschneidet.

3. Zu Gemeinsamkeiten der beiden Kovarianzfunktionen **covSEard** und **covLINard** fallen die ACC Patienten mit Index 26 und 39 auf, die in fast jeder Spalte zu finden sind. Eine Vermutung, dass diese beiden Patienten untypische ACC Eigenschaften besitzen oder aber ihre Messwerte evtl. Messfehler aufweisen, ist bedenkenswert[11].

4.3.2.5 Versuchsanordnung zu *90/10 CV*

In folgendem Abschnitt werden Ergebnisse zur GP Klassifikation bei Betrachtung von verschiedenen Merkmalsuntermengen vorgestellt. Diese Untersuchungen wurden ausschließlich mit den beiden ARD Kovarianzfunktionen **covSEard** und **covLINard** durchgeführt. Hierbei sollte untersucht werden, wie sich die Klassifikationsperformanz eines GP Klassifikators hinsichtlich Anzahl betrachteter Messmerkmale verhält. Die Untersuchungen

[11] Eine Inspizierung entsprechender Messwerte der genannten beiden ACC Patienten durch einen medizinischen Experten könnte evtl. Aufschluss geben

wurden in Kombination mit der bereits vorgestellten 90/10 Krossvalidierungsmethode durchgeführt. Hauptsächlich wurde diesbezüglich ein mittlerer Klassifikationsfehler **MCE** (Mean Classification Error) für jeweils beide betrachteten Klassen errechnet. Hierzu wurde der in Kapitel 3.5.1.3 vorgestellte 90/10 CV Algorithmus 2 modifiziert. Die besagten Veränderungen, welche in Algorithmus 3 (siehe Seite 71) farbig aufgeführt sind, beziehen sich generell auf den Schritt, in welchem ein GP Klassifikator die ihm präsentierten Testdaten (nach dem Trainieren) klassifizieren soll. Bevor auf die einzelnen Besonderheiten des Algorithmus 3 eingegangen wird, sollen kurz folgende Pseudo-Code Bezeichner erklärt werden:

- `H = DoTraining(TrainSet, covFunc)`

  → Funktion, die auf einem gegebenen Datensatz $TrainSet$ mit D Merkmalen einen GP mit der Kovarianzfunktion *covFunc* trainiert und als Resultat Hyperparameterwerte nach dem Training liefert. Hierbei kann

  $$H = [h_1, h_2, .., h_D]$$

  als Liste von D Hyperparamterwerten verstanden werden, wobei z.B. Hyperparameterwert h_i der Stärke des i-ten Merkmals entspricht. Im Falle der Kovarianzfunktion **covSEard** enthält H ein zusätzliches Element für die Signalvarianz. Dieses spielt jedoch bei folgenden Untersuchungen zu Merkmalsuntermengen keine Rolle.

- `ShortTestSet = FilterAttributes(TestSet, HShort)`

  → Funktion, die aus D Merkmalen bestehenden Testdaten $TestSet$ lediglich die Merkmale extrahiert, deren korrespondierende Hyperparameterwerte in der gegeben Hyperparameterliste $HShort$ enthalten sind. Als Ergebnis wird $ShortTestSet$ geliefert, welche eine Merkmalsreduzierte Form der Testdaten $TestSet$ repräsentiert.

- `p = Classify(TestSet, covFunc, H)`

  → Funktion, welche die aus K Merkmalen bestehende Testmenge $TestSet$ mit einem GP klassifiziert und einen Vektor p mit Wahrscheinlichkeiten für die Klassenzugehörigkeit der einzelnen Elemente aus $TestSet$ liefert. Hierbei ist $covFunc$ die Kovarianzfunktion mit der der GP trainiert wurde und H eine Liste von K Hyperparametern, die zu den K Merkmalen der Testmenge $TestSet$ korrespondiert[12].

- `[fA, fB] = EvaluateClassification(p, YTest)`

[12] Im Falle der Benutzung der Kovarianzfunktion *covSEard* enthält H für den Klassifikationsschritt immer zusätzlich ein $K + 1$-tes Element für die Signalvarianz

→ Funktion, welche die durch einen GP gelieferten Wahrscheinlichkeiten der Klassenzugehörigkeiten p anhand der durch $YTest$ gegebenen tatsächlichen Klassenzugehörigkeiten bewertet. Zurückgeliefert werden in fA bzw. fB die Anzahl fehlklassifizierter Individuen aus Klasse A bzw. B.

Folgend werden nun die farbigen und nummerierten Abschnitte aus Algorithmus 3 (siehe Seite 71) erläutert:

1. In Zeile 1 wird ein GP auf der aktuellen Trainingsmenge $X_{TrainSet}$ mit der Kovarianzfunktion *covFunc* trainiert und als Ergebnis die Liste von Hyperparameterwerten $H_{trained}$ nach dem Training gewonnen.

2. In Zeile 2 wird für weitere Ausführungen eine leere Liste H_{mce} von Hyperparametern initialisiert.

3. Nun wird über alle D Merkmale der Trainingsmenge $X_{TrainSet}$ iteriert (Zeilen 3 bis 12):

   - In Zeile 4 wird der stärkste Hyperparameter der Liste $H_{trained}$ ermittelt und aus dieser entfernt (Zeile 5).
   - In Zeile 6 wird der entnommene Hyperparameter in die Liste H_{mce} eingefügt.
   - In Zeile 7 werden die Testdaten X_{Test} auf die Merkmale reduziert, deren Hyperparameter in H_{mce} vertreten sind.
   - In Zeilen 8 und 9 werden die merkmalsreduzierten Testdaten X_{Short} klassifiziert und die Anzahl fA bzw. fB an fehlklassifizierten Patienten für beide Klassen A und B ermittelt.
   - In Zeilen 10 bis 12 werden die Fehlklassifizierungsraten der beiden Klassen bei Betrachtung von k Merkmalen aktualisiert. Hierbei sind MCE_A,MCE_B bzw. MCE_{A+B} als Listen mit D Elementen zu verstehen, die vor Anwendung des Algorithmus auf null initialisiert wurden. Diese enthalten als k-tes Element, die Fehlklassifikationsraten bei Betrachtung von k Merkmalen zu Klassifikation. Der Faktor $\frac{1}{MaxIter}$ dient dazu, die errechneten Fehlklassifizierungsraten auf Durchschnittswerte über *MaxIter* Iterationen zu skalieren.

An dieser Stelle sei in Bezug auf Algorithmus 3 und den Zeilen 7 und 8 folgende Anmerkung gemacht: Es gab eine vom Aufgabensteller vorgeschlagene andere alternative Methodik zur Klassifikation, in welcher die Testmenge nicht merkmalsreduziert werden sollte, sondern stattdessen die Hyperparameter von nicht zu betrachtenden Merkmalen auf den Wert null gesetzt werden sollten. Diese Vorgehensweise führte jedoch leider zu Schwierigkeiten zur Umsetzung mit Hilfe der durch [Rasmus] bereits existierenden

GP Funktionen. Einer entsprechenden Umstellung der aus [Rasmus] gegebenen Software-Strukturen widersprach deren Komplexität und der nicht abschätzbare Aufwand hierfür. Diese Schwierigkeiten wurden dem Aufgabensteller mitgeteilt und die hier stattdessen vorgestellte Vorgehensweise abgesprochen[13].

```
    [X_A, X_B] ← DevideClassSets (X)
    for i ← 1 to MaxIter do
        [X_TrainSetA, X_TestSetA] ← Shf9010 (X_A)
        [X_TrainSetB, X_TestSetB] ← Shf9010 (X_B)
        X_TrainSet ← X_TrainSetA ∪ X_TrainSetB
        X_TestSet ← X_TestSetA ∪ X_TestSetB
 1      H_trained ← DoTraining (X_TrainSet, covFunc)
 2      H_mce ← ∅
 3      for k ← 1 to D do
 4          h_max ← max(H_trained)
 5          H_trained ← H_trained\{h_max}
 6          H_mce ← H_mce ∪ {h_max}
 7          X_Short ← FilterAttributes (X_Test, H_mce)
 8          p ← Classify (X_Short, covFunc, H_mce)
 9          [f_A, f_B] ←EvaluateClassification (p, Y_TestSet)
10          MCE_A(k) ← 1/MaxIter [MCE_A(k) + f_A/|X_TestSetA|]
11          MCE_B(k) ← 1/MaxIter [MCE_B(k) + f_B/|X_TestSetB|]
12          MCE_A+B(k) ← 1/MaxIter [MCE_A+B(k) + (f_A+f_B)/(|X_TestSetA|+|X_TestSetB|)]
        end
    end
```

Algorithmus 3 : 90/10 CV mit MCE Ermittlung

4.3.2.6 Ergebnisse 90/10 CV

In den Abbildungen 4.10 bis 4.15 sind die Ergebnisse nach Anwendung des Algorithmus 3 auf entsprechende Klassenpaare nach 200 Iterationen abgebildet[14]. Hierbei zeigen die oberen Plots die Kurven zu den jeweiligen Fehlklassifikationsraten der einzelnen Klassen bei steigender Anzahl von Merkmalen bei der Klassifikation. Die Fehlklassifikationsraten sind in Prozent angegeben[15]. Beispielsweise hat in oberem Plot aus Abbildung 4.10 die Klasse *ACC* nach 200 Iterationen bei Betrachtung von einem Merkmal einen MCE Wert von ca. 75 Prozent, der bei Betrachtung von zwei Merkmalen hingegen auf

[13] Es bleibt daher an dieser Stelle offen, ob diese besagte alternative Vorgehensweise evtl. gravierend andere Ergebnisse, als die hier aufgeführten, gebracht hätte

[14] D.h. die äussere Schleife in Algorithmus 3 wurde $MaxIter = 200$ mal durchgeführt

[15] skaliert auf den Bereich [0,1]

ca. 40 Prozent sinkt usw. In der letzten Zeile der Tabelle 4.4 (siehe Seite 76) sind hierzu die MCE-Minima der Kurven aus oberen Plots der Abbildungen 4.10 bis 4.15 aufgeführt mit zusätzlicher Angabe der Anzahl an Merkmalen (vorletzte Zeile), bei dem das jeweilige Minimum der MCE Kurve zu finden ist. Demnach also ist z.B. in Spalte eins der Tabelle 4.4 bezüglich der MCE Kurven MCE_{ACA} und MCE_{ACC} aus oberen Plots der Abbildung 4.10 zu lesen, dass das Minimum für MCE_{ACA} bzw. MCE_{ACC} bei 1.5% bzw. 13% liegt und bei einer Anzahl von 29 bzw. 32 Merkmalen zu finden ist.

Die unteren Plots aus Abbildungen 4.10 bis 4.15 zeigen die Varianzen in Bezug auf die in den oberen Plots ermittelten MCE Werte, wobei zum Verständnis folgendes Beispiel helfen soll:

Angenommen Algorithmus 3 würde statt der 200 Iterationen nur 3 Iterationen iteriert und es würden in jeder Iteration 10 ACC Patienten in einer Testmenge vertreten sein. Ferner sei angenommen, dass bei Betrachtung von z.B. nur einem Merkmal in den besagten drei Durchläufen zwei, fünf und acht ACC Patienten fehlklassifiziert wurden. Dann errechnet sich der MCE-Wert (für ein Merkmal) aus

$$MCE_{ACC}(1) = \tfrac{1}{3}[\tfrac{2}{10} + \tfrac{5}{10} + \tfrac{8}{10}] = 0.5$$

Der korrespondierende Varianzwert $VA_{ACC}(1)$ hierzu würde dann aus der Zahlenfolge $[\frac{2}{10}, \frac{5}{10}, \frac{8}{10}]$ berechnet werden:

$$VA_{ACC}(1) = var([\tfrac{2}{10}, \tfrac{5}{10}, \tfrac{8}{10}]) = 0.09$$

Der Wert $MCE_{ACC}(1)$ ist dann beispielsweise in oberem Plot der Abbildung 4.10 auf der X-Achse bei eins aufgetragen (entsprechend $VA_{ACC}(1)$ in unterem Plot auf der X-Achse bei eins).

Abbildung 4.10: MCE nach 200 Iterationen *90/10 CV* für *ACA vs. ACC* bei steigender Anzahl von Merkmalen nach *covSEard* Reihenfolgen

Abbildung 4.11: MCE nach 200 Iterationen *90/10 CV* für *ACA vs. CTRL* bei steigender Anzahl von Merkmalen nach *covSEard* Reihenfolgen

Abbildung 4.12: MCE nach 200 Iterationen *90/10 CV* für *ACC vs. CTRL* bei steigender Anzahl von Merkmalen nach *covSEard* Reihenfolgen

Abbildung 4.13: MCE nach 200 Iterationen *90/10 CV* für *ACA vs. ACC* bei steigender Anzahl von Merkmalen nach *covLINard* Reihenfolgen

Abbildung 4.14: MCE nach 200 Iterationen *90/10 CV* für *ACA vs. CTRL* bei steigender Anzahl von Merkmalen nach *covLINard* Reihenfolgen

Abbildung 4.15: MCE nach 200 Iterationen *90/10 CV* für *ACC vs. CTRL* bei steigender Anzahl von Merkmalen nach *covLINard* Reihenfolgen

covSEard						**covLINard**					
1		2		3		4		5		6	
ACA	ACC	ACA	CTRL	ACC	CTRL	ACA	ACC	ACA	CTRL	ACC	CTRL
29	32	31	29	23	24	1	29	1	1	31	1
1.59%	13%	5%	11%	3%	0%	0%	12%	0%	0%	4.2%	0%

Tabelle 4.4: MCE Minima aus den oberen Plots der Abb. 4.10 bis 4.15

4.3.2.7 Beobachtungen

Allgemein fällt in den oberen Plots der Abbildungen 4.10 bis 4.15 auf, dass nicht immer bei steigender Anzahl von betrachteten Merkmalen ein kontinuierliches fallen der MCE Werte zu erwarten ist. Ein möglicher Grund hierfür könnte folgender sein: Es scheint, dass bei Betrachtung von n Merkmalen die Hinzunahme eines (n+1)-ten Merkmals weniger Wissen bzgl. gewisser Klasseneigenschaften einbringt, als dass es das Wissen in den bereits vorhandenen n Merkmalen verzerrt und somit den Klassifikationsfehler erhöht.

Auffälligkeiten zu **covSEard** (Abb. 4.10 bis 4.12)

- Es sind ähnliche MCE Kurvenverläufe bis zur Betrachtung von 15 Merkmalen zu sehen. In Abbildung 4.12 allerdings fällt der plötzliche Anstieg der MCE_{CTRL} Kurve ab Betrachtung von 27 Merkmalen auf.
- Ab einer Betrachtung von 15 Merkmalen haben MCE Kurven eine relativ konstante Höhe. Minima werden erst ab ca. 23 Merkmalen erreicht (siehe hierzu auch Spalten eins bis drei der Tabelle 4.4).
- Auffällig beim Vergleich von ACA vs. ACC (Abb. 4.10) sind die recht hohen MCE Werte für ACC im Gegensatz zu niedrigen MCE Werten für ACA bis zur Betrachtung von 3 Merkmalen.

Auffälligkeiten zu **covLINard** (Abb. 4.13 bis 4.15)

- Der plötzliche Anstieg beim Vergleich *ACC vs. CTRL* der MCE_{CTRL} Kurve ab Betrachtung von 27 Merkmalen fällt ebenfalls (wie zuvor auch bei covSEard) auf.
- Auffällig ist die recht niedrige Anzahl von betrachteten Merkmalen, bei denen die MCE Kurven ihre Minima haben und insbesondere die fehlerlosen Klassifikationen für die Klassen *ACA* und *CTRL* (siehe hierzu auch Spalten vier bis sechs der Tabelle 4.4).
- Für den Vergleich *ACA vs. ACC* fällt für die MCE-Kurve der Klasse ACC (Abb. 4.13) das Steigen des MCE Wertes bei der Betrachtung von zwei Merkmalen gegenüber einem Merkmal auf.

Insgesamt schneidet *covLINard* in Bezug auf die kleinsten MCE Werte (siehe Tabelle 4.4) besser ab als *covSEard*.

4.3.2.8 Versuchsanordnung zu ROC Kurven mit *90/10 CV*

Für die Erzeugung von ROC Kurven wurden wieder die Klassenpaare *ACA vs. ACC*, *ACA vs. CTRL* und *ACC vs. CTRL* für die binäre Klassifikation durch GP betrachtet. Zum Trainieren und Klassifizieren wurden hierzu jeweils die beiden Kovarianzfunktionen **covSEard** und **covLINard** auf dem kompletten, unreduzierten Merkmalsraum (alle 32 Merkmale) verwendet. In der Hoffnung auf noch aussagefähigere Ergebnisse, wurden die ROC Kurven in Kombination mit der Anwendung der 90/10 CV erzeugt und hierbei wie folgt vorgegangen (Zeilenangaben beziehen sich auf Algorithmus 2 (Seite 51):

- 90/10 CV wurde 200 mal iteriert (Schleife in Zeile 2 wurde 200 mal durchgeführt)
- Im i-ten Schleifendurchlauf wurde im Klassifikationsschritt (Zeilen 8 und 9) für eine zu klassifizierende Testmenge X_{Test} die Diskriminanzschwelle (Entscheidungsgrenze) im Intervall $[0, 1]$ mit einer Schrittweite von 0.05 verschoben und nach jeder Verschiebung X_{Test} neu klassifiziert und entsprechende Sensitivitäts- bzw. Spezifitätswerte vermerkt.
- Nach Beendigung der 200 Iterationen 90/10 CV wurden aus den vermerkten Spezifitäts- bzw. Sensitivitätswerten zu den betrachteten Diskriminanzschwellen Durchschnittswerte über die 200 Iterationen errechnet.

4.3.2.9 Ergebnisse zu ROC Kurven mit *90/10 CV*

In den Abbildungen 4.16 bis 4.18 sind die durchschnittlichen ROC Kurven nach 200 Iterationen 90/10 CV mit jeweils **covSEard** bzw. **covLINard** abgebildet.

Abbildung 4.16: Durchschnittlicher ROC Kurvenverlauf nach 200 Iterationen *90/10 CV* für *ACA vs. ACC*

Abbildung 4.17: Durchschnittlicher ROC Kurvenverlauf nach 200 Iterationen *90/10 CV* für *ACA vs. CTRL*

Abbildung 4.18: Durchschnittlicher ROC Kurvenverlauf nach 200 Iterationen *90/10 CV* für *ACC vs. CTRL*

4.3.2.10 Beobachtungen zu ROC Kurven mit *90/10 CV*

Alle ROC Kurvenverläufe aus Abb. 4.16 bis 4.18 zeigen, dass der Klassenentscheid der Klassifikatoren nicht zufällig erfolgt (Kurven sind nicht nahe der Diagonalen). Beim Vergleich *ACC vs. CTRL* (Abb. 4.18) fällt die deutlich schlechtere *covSEard* ROC-Kurve im Gegensatz zur *covLINard* ROC-Kurve auf.

4.3.3 Klassifikation mit GMM

Im folgenden Abschnitt werden GMM zur binären Klassifikation verwendet (siehe hierzu auch Kapitel 3.4). Die Tests sind grob in drei Teile unterteilt und richten sich nach der Anzahl betrachteter Messmerkmale. Hierzu wurden die in Tabelle 4.2 bereits aufgeführten Rangfolgen zur Merkmalsstärke genommen und die Klassifikation durchgeführt im Merkmalsraum mit

- Betrachtung aller 32 Merkmale (gesamter Merkmalsraum)
- Betrachtung der 15 stärksten Merkmale
- Betrachtung der zwei stärksten Merkmale

4.3.3.1 Versuchsanordnung

Es wurden zunächst zwei GMM gebildet, wobei ein GMM auf den Daten X_{C_i} einer Klasse C_i, und das andere GMM auf den Daten X_{C_j} der Klasse C_j trainiert wurde mit $i \neq j$ und C_i bzw. $C_j \in \{ACA, ACC, CTRL\}$. Hiernach wurden beiden GMM die Testdaten $X_{Test} = X_{C_i} \cup X_{C_j}$ präsentiert und die posterior Wahrscheinlichkeit $p(C_k|X_{Test})$ wie in Kapitel 3.4 beschrieben ausgerechnet und klassifiziert.
Folgende Initialisierungen der GMM wurden vor dem Trainieren durchgeführt:

- Es wurden ausschliesslich volle Kovarianzmatrizen für die GMM Komponenten verwendet
- Die Anzahl an Komponenten (Kernels) für ein GMM beschränkte sich auf zwei bzw. drei Komponenten je GMM[16]. Für folgende Zusammenstellungen zur binären Klassifikation waren diese:
  - [2 Komponenten ACA vs. 3 Komponenten ACC]
  - [3 Komponenten ACA vs. 2 Komponenten CTRL]
  - [3 Komponenten ACC vs. 2 Komponenten CTRL]

[16] Diese gewählten Anzahl an Komponenten hatte sich in vorhergehenden Untersuchungen als geeignet erwiesen

- Es wurde für 20 Iterationen der K-Means Algorithmus als Initialisierung eines GMM durchgeführt

Das Trainieren eines GMM erfolgte durch Anwendung des EM-Algorithmus für 160 Iterationen (diese Anzahl wurde durch Vorprüfungen für eine Konvergenz des EM-Algorithmus bei obiger Konfiguration und gegebenen Daten als ausreichend befunden).

4.3.3.2 Ergebnisse zur Klassifikation bei Betrachtung aller 32, 15 bzw. zwei stärksten Merkmale

In den Abbildungen 4.19 bis 4.24 werden zur Klassifikation nur die zwei stärksten Messmerkmale der Spalten eins bis drei (covSEard) aus Tabelle 4.2 betrachtet. Die Abbildungen 4.25 bis 4.30 dagegen betrachten die zwei stärksten Merkmale der Spalten vier bis sechs (covLINard) aus Tabelle 4.2. Jede der Abbildungen beinhaltet im linken Plot die Trainingsdaten und Lage des jeweiligen GMM im Datenraum. Die rechten Plots zeigen wieder im Datenraum die Lage der fehlklassifizierten Datenpunkte, die zur Klasse des im linken Plot gezeigten GMM gehören. Zudem ist in den rechten Plots ebenfalls die posterior Wahrscheinlichkeit des entsprechenden GMM über den abgebildeten Datenraum als Konturplot zu sehen. Der hierzu abgebildete Farbbalken in den rechten Plots mit entsprechenden Farbwerten dient als Lesehilfe des Konturplots, wonach die Farbwerte als (posterior) Wahrscheinlichkeitswerte[17] zu verstehen sind.

In der Tabelle 4.5 (S.87) sind Indizes fehlklassifizierter Patienten aus rechten Plots der Abbildungen 4.19 bis 4.30 aufgeführt: Spalte eins der Tabelle 4.5 enthält Indizes fehlklassifizierter Patienten zur Klassifikation *ACA vs. ACC* (Abbildungen 4.19 und 4.20), Spalte zwei Indizes von fehlklassifizierten zur Klassifikation *ACA vs. CTRL* (Abbildungen 4.21 und 4.22) usw. Die letzte Zeile der Tabelle 4.5 enthält die Fehklassifikationsraten der einzelnen Klassen in Prozent[18].

Tabelle 4.6 (S.88) enthält Klassifikationresultate bei Betrachtung der 15 stärksten Merkmale aus Tabelle 4.2 und ist wie Tabelle 4.5 zu lesen.

Bei der Klassifikation auf dem gesamten Merkmalsraum (Betrachtung aller 32 Merkmale) wurden stets alle Patienten richtig klassifiziert mit folgenden Ausnahmen:

- ACC-Patient 26 bei *ACA vs. ACC*
- ACC-Patienten 26 und 28 bei *ACC vs. CTRL*

[17] skaliert auf den Bereich $[0, 1]$

[18] Bei der Klassifikation ACA vs. ACC in Spalte eins der Tabelle 4.5 gab es z.B. vier fehlklassifizierte ACA Patienten (siehe auch rechter Plot in Abbildung 4.19), was bei einer Gesamtanzahl von 113 ACA Patienten einer Fehlklassifikationsrate für ACA von $\frac{4}{113} \approx 3.5\%$ entspricht

Abbildung 4.19: ***ACA vs. ACC***, Lage und Predictions des **ACA GMM** im Datenraum (2 stärksten Merkmale covSEard)

Abbildung 4.20: ***ACA vs. ACC***, Lage und Predictions des **ACC GMM** im Datenraum (2 stärksten Merkmale covSEard)

Abbildung 4.21: ***ACA vs. CTRL***, Lage und Predictions des **ACA GMM** im Datenraum (2 stärksten Merkmale covSEard)

Abbildung 4.22: ***ACA vs. CTRL***, Lage und Predictions des **CTRL GMM** im Datenraum (2 stärksten Merkmale covSEard)

Abbildung 4.23: ***ACC vs. CTRL***, Lage und Predictions des **ACC GMM** im Datenraum (2 stärksten Merkmale covSEard)

Abbildung 4.24: ***ACC vs. CTRL***, Lage und Predictions des **CTRL GMM** im Datenraum (2 stärksten Merkmale covSEard)

Abbildung 4.25: ***ACA vs. ACC***, Lage und Predictions des **ACA GMM** im Datenraum (2 stärksten Merkmale covLINard)

Abbildung 4.26: ***ACA vs. ACC***, Lage und Predictions des **ACC GMM** im Datenraum (2 stärksten Merkmale covLINard)

Abbildung 4.27: ***ACA vs. CTRL***, Lage und Predictions des **ACA GMM** im Datenraum (2 stärksten Merkmale covLINard)

Abbildung 4.28: ***ACA vs. CTRL***, Lage und Predictions des **CTRL GMM** im Datenraum (2 stärksten Merkmale covLINard)

Abbildung 4.29: ***ACC vs. CTRL***, Lage und Predictions des **ACC GMM** im Datenraum (2 stärksten Merkmale covLINard)

Abbildung 4.30: ***ACC vs. CTRL***, Lage und Predictions des **CTRL GMM** im Datenraum (2 stärksten Merkmale covLINard)

covSEard						**covLINard**					
1		2		3		4		5		6	
ACA	ACC	ACA	CTRL	ACC	CTRL	ACA	ACC	ACA	CTRL	ACC	CTRL
1	1	15	5	1	50	1	2	3	1	2	35
82	2	16	7	2	84	24	6	14	2	6	39
98	3	17	8	3		54	7	17	3	8	40
105	5	19	9	5		57	13	29	4	11	85
	7	20	16	6		98	14	67	6	13	86
	13	22	18	7			20	76	8	15	
	14	24	19	14			24	80	9	20	
	17	39	22	17			25	107	10	24	
	20	40	24	20			26		11	26	
	24	47	28	24			30		12	28	
	25	48	31	25			32		13	32	
	26	52	34	26			35		14	34	
	30	53	37	32			37		16	35	
	32	54	38	35			39		17	43	
	35	55	39	38			42		18	44	
	38	57	40	43			43		19		
	42	61	41	44			44		54		
	43	62	42	45			45		55		
	44	70	44						56		
	45	72	50						57		
		73	51						58		
		76	52						59		
		77	53						60		
		78	60						61		
		84	64						62		
		87	65								
		88	66								
		93	74								
		94	82								
		99									
		100									
		104									
		106									
		108									
3.5%	44.4%	30.0%	32.9%	40.0%	2.2%	4.4%	40.0%	7.0%	28.4%	33.3%	5.6%

Tabelle 4.5: Indizes fehlklassifizierter Patienten bei GMM Klassifizierung mit zwei Merkmalen

covSEard						**covLINard**					
1		2		3		4		5		6	
ACA	ACC	ACA	CTRL	ACC	CTRL	ACA	ACC	ACA	CTRL	ACC	CTRL
	26			26		73	26		6	14	2
							43		10	26	
							44		12		
									15		
0%	2.2%	0%	0%	2.2%	0%	0.8%	6.6%	0%	4.5%	4.4%	1.1%

Tabelle 4.6: Indizes fehlklassifizierter Patienten bei GMM Klassifizierung mit 15 Merkmalen

4.3.3.3 Beobachtungen

Betrachtung der zwei stärksten Merkmale

Für die in Tabelle 4.5 angegebenen Indizes von fehlklassifizierten Patienten ist in den Abbildungen 4.19 bis 4.30 zu sehen, dass diese sich häufig in Bereichen mit GMM Prediction-Werten nahe bei 0.5 befinden. Vergleicht man die Anzahl fehlklassifizierter Patienten aus Spalten 1-3 mit Spalten 4-6 in Tabelle 4.5, so fallen die deutlich weniger fehlklassifizierten Patienten zugunsten von *covLINard* auf.

Betrachtung der 15 bzw. 32 stärksten Merkmale

Auffällig in Tabelle 4.6 sind die insgesamt geringen fehlklassifizierten Patienten, wobei bei Betrachtung der 15 stärksten Merkmale nach *covSEard* Reihenfolgen der Klassifikationsfehler deutlich niedriger ausfällt als bei *covLINard* Reihenfolgen. Ferner ist ACC Patient 26 zu bemerken, der in allen Spalten der Tabelle 4.6 zu finden ist und auch bei der Klassifikation bei Betrachtung des kompletten Merkmalsraums erneut auftritt.

4.4 Diskussion und Ausblick

Insgesamt ist im Hinblick auf den verwendeten Datensatz für die Experimente folgendes anzumerken:

- Die Klassen waren durch sehr unterschiedliche Anzahl von Patienten vertreten, wobei es im Fall ACC mit 45 Patienten offen bleibt, ob diese geringe Menge ausreichend repräsentativ für ACC Eigenschaften war.
- Es bleibt zudem offen, ob die Verwendung von kompletten und lückenlosen Datensätzen (statt den wie hier künstlich gefüllten) evtl. gravierend andere Ergebnisse geliefert hätte.

Beim unüberwachten Lernen und der PCA Analyse sind Unterschiede zwischen den Daten der jeweiligen Klassen im zweidimensionalen PCA Raum zumindest visuell erkennbar gewesen, jedoch mit teilweise recht grossen Überschneidungen der Klassenbereiche, die keine eindeutige (z.B. lineare) Trennbarkeit vermuten liessen. Die Untersuchungen zur Merkmalstärke in den gegebenen Datensätzen zeigten deutlich verschiedene Merkmalsrangfolgen im Vergleich *covSEard* zu *covLINard*. Ob die hier aufgelisteten Merkmalsrangfolgen eine medizinische Relevanz besitzen ist unklar. Sofern es möglich ist durch entsprechendes Expertenwissen eine medizinisch relevante Merkmalsrangfolge zusammenzustellen, so wären weitere Untersuchungen mit GP, unter Benutzung der besagten Rangfolgen, für eine weitere Bewertung der Klassifikationsverfahren interessant. Beim überwachten Lernen fiel für GP der fehlerlose Trainingsfehler im kompletten Merkmalsraum in Gegensatz zur LOO-CV für GP mit diversen fehlklassifizierten Patienten (ebenfalls kompletter Merkmalsraum) auf. Darüber hinaus zeigen gewonnene Ergebnisse zu Untersuchungen von Merkmalsuntermengen mit 90/10 CV und den darin ermittelten MCE Klassifikationsfehlern eine Überlegenheit der *covLINard* Kovarianzfunktion gegebenüber *covSEard*. Bei der Ermittlung von ROC Kurven war insgesamt zu erkennen, dass Klassenentscheide von verwendeten GP Klassifikatoren deutlich besser als eine Zufallsentscheidung waren. Aufgeführte Untersuchungen zur Klassifikation mit GMM (mit Verwendung der GP Ergebnisse zu Merkmalsrangfolgen) zeigten schlechte Ergebnisse bei Betrachtung von nur zwei stärksten Merkmalen. Bei der Betrachtung von 15 stärksten bzw. 32 Merkmalen zeigte sich jedoch eine deutliche Verbesserung. Im Hinblick auf einen Klassifikationsfehler sind die Ergebnisse zu GMM im Vergleich[19] zu beispielsweise LOO-CV bei GP als besser einzustufen.

[19] Sofern die beiden Klassifikationsmethoden GMM und GP überhaupt qualitativ vergleichbar sind

Zusammenfassung

Ziel dieser Arbeit war es, eine Datenanalyse von gegebenen Steroiddaten mit Hilfe von Methoden des unüberwachten und überwachten Lernens durchzuführen. Hierzu sollten relevante Merkmale bzw. Merkmalsrangfolgen aus den Daten automatisch extrahiert werden. Anschliessend sollte mit Hilfe dieser eine Klassifizierung zu bestimmten Klassen erfolgen. In der vorliegenden Arbeit wurden hierzu Methoden des unüberwachten und überwachten Lernens vorgestellt. Zum einen beinhalteten diese für den unüberwachten Fall Methoden zum Auffinden von Strukturen in höherdimensionierten Merkmalsräumen. Zum anderen wurden probabilistische Methodiken aus dem überwachten Lernen zur automatischen Ermittlung von Merkmalsrelevanzen und anschliessender Klassifikation aufgeführt. Ferner wurden diverse Krossvalidierungstechniken als Bewertungsverfahren für Klassifikatoren erklärt und teilweise neu entwickelt.

Abbildungsverzeichnis

Tabellenverzeichnis

Liste der Algorithmen

Literaturverzeichnis

[ChrBish] Christopher Bishop: *Pattern Recognition and Machine Learning*, Springer Verlag, 2006, ISBN-10: 0-387-31073-8

[NabNey] Ian T. Nabney: *Netlab, Algorithms for Pattern Recognition*, Springer Verlag, 2002, ISBN 1-85233-440-1

[SchoeSmo] Bernhard Schölkopf, Alex Smola:*Learning with Kernels Support Vector Machines, Regularization, Optimization and Beyond*, MIT Press, Cambridge, MA, 2002

[HarHot] Harold Hoteling:*Analysis of complex statistical variables into principal components*,Journal of educational psychology, 24:417, 1933, Seiten 41

[TipBish] Michael E. Tipping, Christopher M. Bishop: *Probabilistic Principal Component Analysis*, Journal of the Royal Statistical Society, Series 61, Part 3,pp. 611-622, September 1999

[Rasmus] C.E.Rasmussen, C.K.I. Williams: *Gaussian Processes for Machine Learning*, The MIT Press, 2006, ISBN 026218253X

[Dempster] Dempster A.P., Laird N.M.: *Maximum Likelihood from Incomplete Data via the EM Algorithm*, Journal of the royal Statistical Society, 1977, Vol. 39 (S.1 bis S.38)

[Fawcett] Tom Fawcett: *An introduction to ROC analysis*, Pattern Recognition Letters 27, 2006, (861-874)

[OpperWinther] Opper M., Winther O.: *Gaussian Process for Classification, Mean Field Algorithms*, Neural Computation, 2000, 12(11):(2655-2684)

[Minka] Minka T.P.: *A Family of Algorithms for Approximate Bayesian Inference*, PhD thesis MIT, 2001, (Seiten 41-52)

[Neal] Neal R. M: *Bayesian Learning for Neural Networks*, Springer, New York, 1996, (Lecture Notes in Statistics 118)