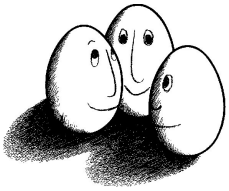
Diplomarbeit

Zeitreihenprognose mittels lokaler Modelle und ihrer Globalisierung

Anne Antonia Scheidler



Diplomarbeit
am Fachbereich Informatik
der Technischen Universität Dortmund

Dortmund, 28. April 2008

Betreuer:

Prof. Dr. Katharina Morik
Dipl.-Inform. Ingo Mierswa

Danksagungen

Diese Diplomarbeit entstand am Lehrstuhl für Künstliche Intelligenz der Technischen Universität Dortmund unter der Leitung von Frau Prof. Katharina Morik. Besonders möchte ich mich bei meinem Betreuer Ingo Mierswa für die freundliche und engagierte Betreuung bedanken, zudem bei Karsten Kubicke für das Lektorat. Ebenso möchte ich mich an dieser Stelle herzlich bei Gregor Mordal sowie meinem Freundeskreis und meiner Familie für deren Unterstützung in jeglicher Hinsicht während der Dauer der Diplomarbeit und des Studiums bedanken.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung

"Prognosen sind schwierig
- besonders, wenn sie die
Zukunft betreffen."

Mark Twain

Die Beschäftigung mit Zeitreihen ist eine hochaktuelle und interessante Thematik. Bei näherer Betrachtung dieser stellen wir fest, daß sie uns vielfach im Alltag begegnet. Dies fordert geradezu heraus, geeignete Prognosen für Zeitreihen zu entwickeln. Daran anknüpfend haben inzwischen Forschungen bezüglich Lokalität und Globalität Einzug gehalten. Zeitreihenprognose, Lokalität und Globalität sind die zentralen Begriffe dieser Arbeit.

Zahlreiche Arbeiten dienten als Inspiration zur Entwicklung eigener Methoden, die an entsprechenden Stellen referenziert wurden. Die vorliegende Arbeit dokumentiert nicht nur die Methodik selbst, sondern zeigt oftmals über Abgrenzungen, welche Gedankengänge zur Entwicklung eigener Methodik geführt haben. Die Arbeit ist als vollständiges Prognoseverfahren anzusehen - von den Rohdaten bis zur Vorhersage. Die Ideenfindung kann kapitelweise nachvollzogen werden, wobei die Kapitel sich gemäß der Funktionalität im Gesamtverfahren ergeben. Die Vielzahl an Aufgabenstellungen, die sich durch die Grundlage eines abgeschlossenen Verfahrens ergeben haben, sowie die Suche nach Lösungen, wirkten dabei motivierend auf den Arbeitsprozeß ein.

Da die Arbeit die Methodik zur Zeitreihenprognose mittels Fallunterscheidung differenziert, soll im Folgenden ein kleiner Überblick über die Struktur gegeben werden, um dem Leser eine Vorabsortierung der Themen zu gestatten.

Zunächst erfolgt in Kapitel 2 eine Einführung von Begrifflichkeiten, die als grundlegendes Werkzeug für das Methodikverständnis in der nachfolgenden Arbeit dienen. Hier werden Begriffe wie Zeitreihe, Instanzen und Muster vorgestellt und miteinander in Verbindung gebracht. Die Abbildung 1.1 vermittelt einen ersten Eindruck dieser Begriffe. Sie zeigt symbolisch eine Zeitreihe, in der man zwei Muster unterscheiden kann. Das Auftreten der einzelnen Strukturen, die als Klassenbeispiele dienen, bezeichnet man als Instanzen. Während Muster A drei Instanzen beinhaltet (1,3 und 5), besteht Muster B aus zwei Instanzen (2 und 4). Grundlegend betrachtet ist ein Klassenbeispiel ein strukturbestimmender Abschnitt der Zeitreihe, bestehend aus Datenpunkten. In welcher Form die Reihe

nun fortgesetzt wird inklusive der Methoden, ist Hauptthema der Arbeit.

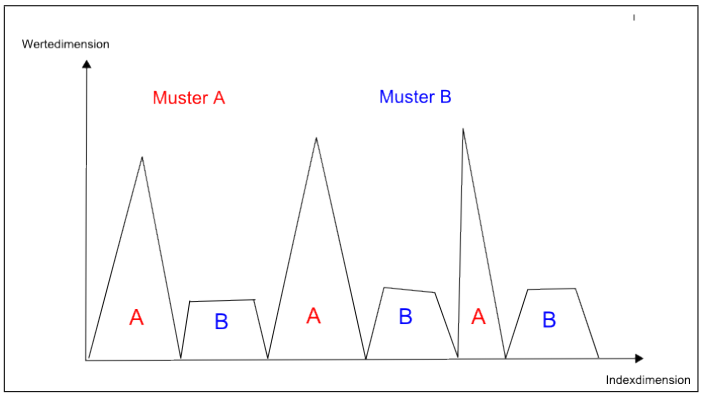


Abbildung 1.1.: Musterbeispiel

Die Zeitreihenprognose und die dafür verwendete Methodik wird in verschiedenen Abschnitten erläutert. Der Strukturbaum ist über das Kapitel 3 zu verfolgen. Dieses Kapitel führt uns von den globalen Modellen schrittweise zu den lokalen Modellen anhand bekannter Prognoseverfahren. Die Prognose unterteilen wir in zwei große Hauptgebiete. Der erste Bereich beschäftigt sich mit globalen Modellen (3.1) und stellt deren aktuelle Techniken vor. Im Anschluß an die Prognose mittels globaler Modelle werden Randbereiche aufgezeigt und Methoden vorgestellt, die erste Schritte in den Bereich der Lokalisierung aufzeigen. Schließlich widmen wir uns den lokalen Modellen, die den zweiten Hauptbereich bestimmen, und zu denen wir das hier entwickelte Verfahren in seinem Basiskonzept zählen dürfen. Die Modellierungen (lokaler und globaler Bereich) werden in Verbindung mit der Support Vector Machine (SVM) gebracht. Dabei wird aufgezeigt, welche Ergebnisse die SVM bei verschiedenen Ansätzen liefert. Über das Aufgreifen der SVM bzw. Prognoseverfahren, die diese Technik nutzen, wird letztendlich ersichtlich, warum Stufen des hier behandelten Konzeptes sich der SVM bedienen und deren Nachteile durch eigene Methodik überwinden. Die genaue Betrachtung der eigentlichen Techniken erfolgt in Kapitel 5. In Kapitel 4 wird der Bereich der Vorverarbeitung vorgestellt, der das Anwenden der Prognosetechniken auf den Datensätzen ermöglicht. Abschnitt 5.1, der die Behandlung von langfristigen Prognosen methodisch erläutert, zeigt ungenutztes, in den Datensätzen verborgenes Wissen auf. Im Anschluß wird in Kapitel 6 eine Methode zur kurzfristigen Prognose vorgestellt, die genau dieses Wissen nutzt. Diese Methode greift nicht mehr ausschließlich auf die Muster zu, sondern zusätzlich auf die Instanzen. Bei der Grundidee handelt es sich um einen Nearest Neighbour-Ansatz, der über eine Balancefunktion zwei unterschiedliche Gewichtungen des Suchraumes gestattet. Kapitel 9 zeigt einen kleinen Ausblick bezüglich möglicher Erweiterungen und Automatisierungen. Darin betrachten wir Ansätze zur Automatisierung in den Verfahren und auch multivariate Datensätze. In Kapitel 7 sind ausgewählte Experimente graphisch und parametergenau dokumentiert. Für statistische Kennzahlen wird auf den Abschnitt 8.1 sowie den Anhang A und B verwiesen. Hier finden sich auch in der Arbeit nicht explizit erläuterte Definitionen, die im Rahmen der Versuche von Bedeutung sind.

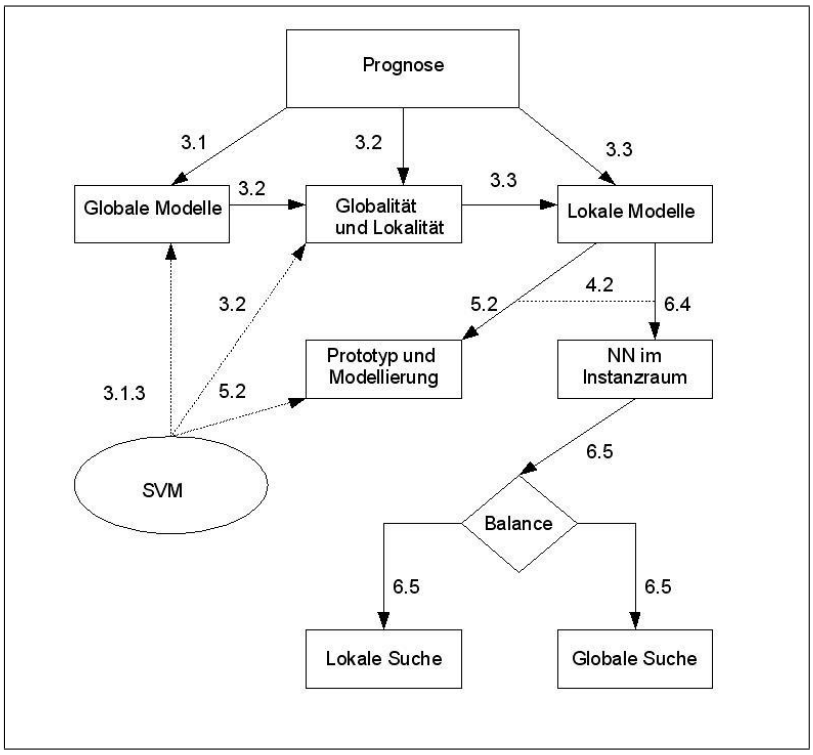


Abbildung 1.2.: Organigramm

Schon in der Einführung tauchen die Begriffe Lokalität und Globalität auf, die auf unterschiedlichen Granularitätsebenen und aus unterschiedlichen Blickwinkeln immer wieder eine erweiterte Definition finden werden. Dies jedoch, ohne eine Inkonsistenz zur ursprünglichen Annahme zu erzeugen. Einige Anwendungen der Begriffe gestatten eine weitere, tiefgehende Beleuchtung. Im Rahmen dieser Arbeit wird immer dann auf ihre Ausführung verzichtet, wenn die Erkenntnis für die entwickelten Verfahren nicht gewinnbringend ist. Nehmen wir zum Beispiel die Saisonalkomponente, die aus dem Blickwinkel der Zeitreihe ein lokales Muster bildet. Doch ein solches lokales Muster besteht gegebenenfalls aus Submustern. Ändern wir den Blickwinkel, so ist das Muster, das das Submuster beinhaltet, ein globales Muster bezüglich des lokalen Musters (Und ein Submuster hat Attribute, eine Menge von Punkten ...).

2. Grundlegende Begrifflichkeiten

In den folgenden Abschnitten werden einige grundlegende Definitionen zu dem Begriff der Zeitreihe und dem Thema Muster eingeführt. Anschließend werden die Definitionen der Themenbereiche aufeinander abgebildet mit dem Ziel, Schnittpunkte herauszuarbeiten.

2.1. Zeitreihen

Die in dieser Arbeit vorgestellten Verfahren arbeiten auf Zeitreihen. Bei Zeitreihen erfolgt die Festlegung auf die Zeitskala für eine Dimension - sie stellen somit nur eine Beschränkung der allgemeinen Wertereihe dar. In späteren Abschnitten finden die auf ihnen angewandten Methoden nähere Betrachtung.

2.1.1. Mathematische Annäherung

Wertereihen sind im Alltag allgegenwärtig - wir finden sie in Bereichen der Betriebswirtschaft, Mathematik, Ökonomie - kaum ein Bereich, aus dem sie wegzudenken wären. Ihre Eigenschaft, graphisch einen schnellen Einblick in den Sachverhalt zu vermitteln, wird im Bereich der Finanzmathematik beispielsweise für den Verlauf von Börsenkursen genutzt.

Wertereihen können als Folge von Datenpunkten aufgefaßt werden. Dies setzt voraus, daß die zugrundeliegenden Daten diskret vorliegen. Eine besonders häufig anzutreffende Unterklasse der Wertereihe ist die Zeitreihe. Der Begriff Zeitreihe rührt daher, daß eine Dimension die Zeit beschreibt, also eine zeitabhängige Folge von Punkten vorliegt. Die Zeitpunkte, die den Daten zugeordnet werden, können äquidistant oder unregelmäßig sein. Das ist zumeist direkt durch die zu erfassenden Werte gegeben. Die Zeitinformation kann explizit, als Zeitstempel (time stamps), gehalten werden oder implizit als Werteentnahme aus zeitvariierenden Prozessen [Rüping 2001]. Je nach Datennatur bilden Abtastung oder auch Erhebungszeitpunkte der Daten eine Strukturvorgabe für die Zeitskala. Abtastung (Englisch: Sampling) bedeutet das Registrieren von Messwerten zu bestimmten Zeitpunkten. Ein Element der Zeitreihe ist ein geordnetes Paar $x_i = (d_i, w_i)$ mit $d_i \in \mathbb{N}$, $w_i \in \mathbb{C}^m$ und $i = 1...n$. Ferner bezeichnet d_i die Indexdimension und w_i die Wertedimension. Im Allgemeinen entspricht die Ordnung der x_i der Ordnung der d_i, der Index d_1, d_2 , d_3 verweist somit auf das erste, zweite, dritte Glied der Folge. Liegt zu jedem d_i ein Wert w_i vor, ist die Reihe univariat - liegt eine Mehrzahl von Zahlenwerten in Form eines Vektors der Länge m vor, bezeichnet man die Reihe als multivariat.

Definition 2.1.1 (Zeitreihe)**.** *Eine Zeitreihe ist eine zeitlich geordnete Folge von Werten. Unter einer Folge versteht man die Abbildung:*

$$x : \mathbb{N} \to \mathbb{R} \times \mathbb{C}^m; m \in \mathbb{N}$$

2.1.2. Statistische Annäherung

Eine andere Herangehensweise an die Zeitreihe ist, sie als stochastischen Prozess aufzufassen.

Definition 2.1.2 (Stochastischer Prozess)**.** *Ein stochastischer Prozess ist eine Familie von Zufallsvariablen mit* $X_t : \Omega \to Z$ *mit T = Indexdimension.*

Wird T aus dem Bereich $\mathbb{Z}$ gewählt, so spricht man von einer stochastischen Folge. Wenn T aus $\mathbb{Z}$ stammt oder abzählbar ist, so ist der Prozess diskret.

Definition 2.1.3 (Zufallsvariable)**.** *Sei* $(\Omega\Sigma P)$ *ein Wahrscheinlichkeitsraum mit* Ω *= Ergebnismenge,* Σ *= Ereignisalgebra und* P*= Wahrscheinlichkeitsmaß auf* Σ *sowie* $(\Omega'\Sigma')$ *ein Messraum. Die Abbildung* $X : \Omega \to \Omega'$ *bezeichnet dann die* Ω' *Zufallsvariable auf* Ω*.*

Wenn die Indexdimension als Zeit aufgefaßt wird, wird deutlich, daß die o.a. Definitionen den gleichen Sachverhalt beschreiben. Man unterscheidet beide Definitionen bezüglich ihrer praktischen Verwendung. Definition 2.1.2 ist als theoretisches Konzept mit Augenmerk auf Stochastik und Zufallsfunktion verbreitet. Definition 2.1.1 hingegen findet häufig zur Beschreibung konkreter Realisierungen von Zeitreihen Verwendung. In der vorliegenden Arbeit wird mit der mathematischen Definition 2.1.1 gearbeitet und diese von nun an als Standard betrachtet. Sollten statistische Sichtweisen verwendet werden, wird darauf explizit hingewiesen.

Ausgehend von Definition 2.1.2 gibt es eine Vielzahl von statistischen Kennzahlen, die die Zeitreihe näher beschreiben können. Man nennt diese Kennzahlen *Momente*. Die wesentlichen *Momente* sollen hier kurz vorgestellt werden, für weitere Kenngrößen sei auf [Andel 1984] verwiesen.

Mittelwertfunktion $\mu(t) := E[X_t]$, d.h. der Erwartungswert für den Zeitpunkt t.

Varianzfunktion $\sigma^2(t) := Var[X_t]$.

Autokovarianzfunktion Seien $r, s \in T, Var(X_t) < \infty$, dann ist

$$\gamma(r,s) = Cov(X_r), X_s = E((X_r - \mu(r))(X_s - \mu(s))) \quad \forall r, s \in T.$$

Strikt Stationär Der Prozess X_t heißt strikt stationär, wenn gilt:

$$E|X_t|^2 < \infty \quad \forall t$$
$$E(X_t) = \mu\infty \quad \forall t \in T$$
$$\gamma x(r,s) = \gamma x(r+h, s+h) \forall r, s, h : r, s, r+h, s+h \in T.$$

Schwach Stationär Der Prozess X_t heißt schwach stationär, wenn gilt:

$$\gamma x(r,s) = \gamma x(r-s, 0) \forall r, s.$$

Autokorrelationsfunktion Der Prozess X_t sei schwach stationär. Dann ist die Autokorrelationsfunktion gegeben durch:

$$p_x(h) = \frac{Cov(X_{t+h}, X_t)}{\sqrt{Var(X_{t+h}, X_t)}}, h \in \mathbb{Z}.$$

2.1.3. Darstellung von Zeitreihen

Im Hinblick auf die visuelle Mustererkennung spielt die graphische Darstellung von Zeitreihen eine zentrale Ordnung. Die grundlegende Darstellungsform, die hier verwendet wird, ist das Eintragen der x_i in ein karthesisches Koordinatensystem. Auf der Abzisse wird die Indexdimension abgetragen und auf der Ordinate die Wertedimension. Im Regelfall werden die (d_i, w_i) einzeln dargestellt. Wann immer diese Darstellung benutzt wird, wird der Begriff *Zeitreihe* für die Darstellung verwendet. Durch Interpolation der Werte entsteht ein Polygonenzug. Derartige Darstellungen sind unter dem Begriff *Zeitreihenpolygon* deklariert. Je nach Visualisierungsziel und den sich unter Umständen daran anschließenden Lernaufgaben gibt es eine Vielzahl von Visualisierungsmöglichkeiten und Darstellungsschemata (z.B. [Keogh und Pazzani 1998]).

Beispiel 2.1 (Zeitreihe und Darstellung)**.** *Am Ende ihrer Lebensdauer durchlaufen die meisten Sterne ein Stadium der Instabilität, das dazu führt, daß sie ihr Licht verändern. Die Lichtveränderungen wurden zu äquidistanten Zeitpunkten (Täglich um Mitternacht) summiert über den Tag entnommen. Die Abzisse gibt die Zeitachse in Tagen gemessen an, die Ordinate die aufgezeichneten Helligkeitswerte. Da nur die Helligkeit - also ein Wert - erhoben wurde, beträgt die Wertedimension m=1. Die Darstellungen Zeitreihe und Zeitreihenpolygon (siehe Abbildung 2.1) wurden mit RapidMiner erzeugt und zeigen nur einen Ausschnitt der gesamten Wertereihe [University 2006].*

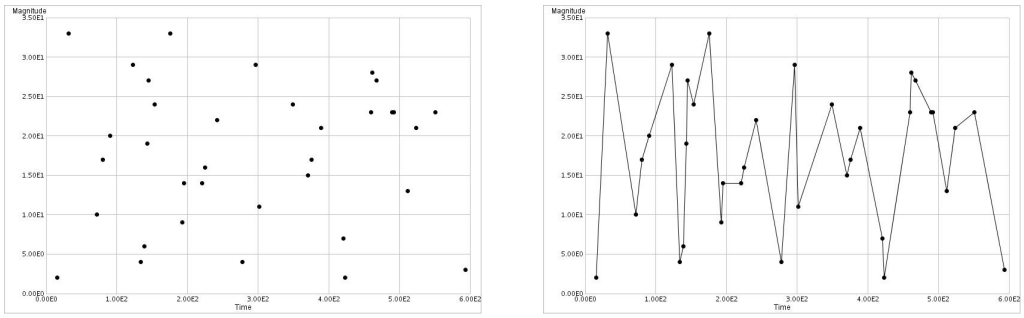


Abbildung 2.1.: Darstellung Zeitreihe und Zeitreihenpolygon

2.1.4. Zeitreihenkomponenten

Die Statistik beschreibt die Komponenten der Zeitreihe über das klassisch additive Zerlegungsmodell. Die Zeitreihe S entspricht der Addition $T + S + R$. Das erste Element T bezeichnet die Trendkomponente, eine im Zeitverlauf nur langsam variierende Funktion, die die Grundtendenz angibt. S bezeichnet die saisonalen Komponenten - sie entsprechen periodischen Ereignissen in Zeitreihen. R wiederum ist eine stationäre Zeitreihe und wird als irreguläre Rauschkomponente oder als statistischer Rest bezeichnet. Unter *Irregulär* sind unvorhersehbare Schwankungen in der Reihe zu verstehen, die nicht durch Trend oder Saisonalität erklärbar sind. Die drei Komponenten können auch über andere mathematische Operationen miteinander verknüpft werden. Anwendung findet verbreitet das *Multiplikative Zerlegungsmodell*, in dem die Zeitreihe als Produkt der drei Komponen-

ten abgeleitet wird. Die Einzelkomponenten sind graphisch in Abbildung 2.2 verdeutlicht [TU Clausthal 2007].

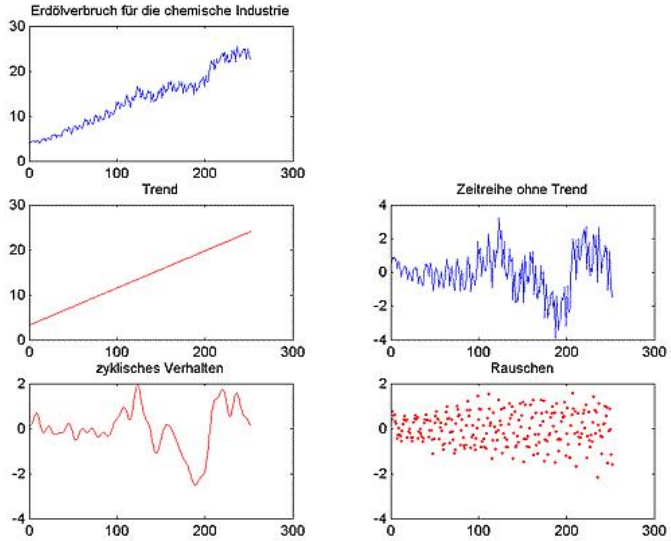


Abbildung 2.2.: Komponenten

2.1.5. Ziele der Zeitreihenanalyse

Der Begriff Zeitreihenanalyse umfaßt verschiedenartige Ziele, die bei der Arbeit mit Reihendaten im Vordergrund stehen können. Zum einen gibt es den großen Bereich der eigentlichen Analyse. Hier liegt der Schwerpunkt im Auffinden interessanter Strukturen wie potentielle Maxima im Wertebereich. Dann schließt sich in der Regel die Überlegung an, von welchen äußeren Einflüssen *(Parametern)* diese Werte oder auch komplexe Strukturen abhängig sind. Man stelle sich einen Produktionsprozess in der Industrie vor: Die Darstellung der Zeitreihe der Abverkäufe einer Produktgruppe zeigt auf die Zeitachse in Monatsabschnitten bezogen alle 6 Monate ein lokales Extremum an. Welche Faktoren intern oder extern können dieses hervorgerufen haben ? Im Anschluß daran könnte die Zeitreihe mit weiteren Zeitreihen verglichen werden. Fragen, ob die Zeitreihe Ähnlichkeiten in der Struktur zu Zeitreihen anderer Produktgruppen aufwirft oder sich gar die Zeitreihe mit einer weiteren Reihe erklären läßt, sind von zentraler Bedeutung. Ähnlichkeit läßt sich jedoch nicht immer aus der bloßen graphischen Struktur erkennen. Hier schließen sich die umfassenden Gebiete der Ähnlichkeitsberechnung und Merkmalsextraktion an, die auch von Interesse sind, wenn Zeitreihen klassifiziert werden sollen.

Neben der Analyse, deren Ziele in dieser Arbeit nur eine untergeordnete Rolle spielen werden, gibt es den Bereich der Prognose.

Die Vorhersage von Zeitreihen ist ein mannigfaltiges Gebiet. Die Techniken richten sich danach, mit welcher Genauigkeit etwas vorhergesagt werden soll. Umso relevanter ist der

praktische Nutzen - ein *exakt* prognostizierter Aktienverlauf ist unbezahlbares Wissen. Doch *exakt* ist bei Realdaten kaum möglich, daher suchen wir eine möglichst gute Annäherung an die wahren Zukunftswerte der Zeitreihe. Ein Überblick über den aktuellen Forschungsstand und die Methoden wird in Kapitel 3 gegeben.

Definition 2.1.4 (Prognose und Prognosehorizont)**.** *Unter einer Prognose für eine Zeitreihe* X_i *mit Elementen* $x_i \in (d_i, w_i)$ *und* $i = 1...n$ *verstehen wir das Berechnen der* w_j *für* d_j *(das Bilden der* x_j*) für* $j = n + 1...m$*. Der Prognosehorizont* ph *ist definiert als:* $ph = m - n + 1$*.*

Bevor das in 2.1.4 verwendete Wort *berechnen* und der in den nächsten Abschnitten eingeführte Ansatz vorgestellt werden, ist es unerlässlich, ein weiteres Gebiet zu betrachten. Der nächste Abschnitt beschäftigt sich mit dem Begriff des Musters.

2.2. Muster

Betrachtet man die Verwendung des Wortes *Muster* im Alltag, fällt zuerst ein basisbildender Aspekt auf, die Wiederholung. Wiederholung kann in divergentem Kontext, je nach Natur der Sache, definiert werden. Ein Denkmuster beschreibt einen sich wiederholenden Vorgang, ein Strickmuster stellt eine Anleitung dar, die zur Reproduktion dient. So kann *Muster* durch sich wiederholende Strukturen oder allgemein gleichbleibende Merkmale einer sich wiederholenden Sache aufgefaßt werden. Haben wir Merkmale, sind wir in der Lage, zu unterscheiden und somit auch zuzuordnen, also zu klassifizieren. Mustererkennung ist also nichts anderes als die Zuordnung eines unbekannten Musters zu einer Klasse.

2.2.1. Der Musterbegriff

Eine Formalisierung des Begriffes ist nicht einfach und eine allgemein gültige mathematische Definition ist bis dato noch nicht gefunden worden[Niemann 2006]. Daher nähern wir uns dem Begriff an, um im nächsten Abschnitt das Problem zielgenau zu formulieren und auf die zugrundeliegende Struktur - die der Zeitreihe - anzupassen.

2.2.2. Lokale und Globale Muster

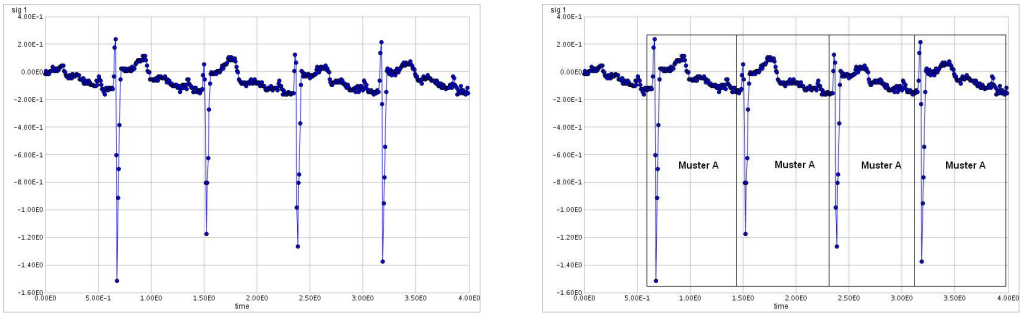
Basis für die Anschauung sei eine Menge von Punkten. Greifen wir einen dieser Punkte a heraus und definieren eine Nachbarschaft, so bezeichnen wir diesen Bereich als lokal. Die Nachbarschaft kann als ϵ - Umgebung mit $U_{\epsilon(a)} := \{x \in \mathbb{R} : |x - a| < \epsilon\}$ aufgefaßt werden [Bronstein und Semendjajew 1996]. Da die zugrundeliegenden Daten als diskret angenommen werden, ist es jedoch sinnvoller, sich die Umgebung als Menge von Punkten vorzustellen, die zu a und einer Metrik (Vergleiche zum Begriff Metrik die Definition 6.1) die geringste Distanz aufweisen. Analog bezeichnen wir dann als globales Muster alle Punkte, die Teil der Grundmenge sind.

2.3. Matching von Zeitreihen und Mustern

In den nachfolgenden Abschnitten sollen nun die eingeführten Begriffe miteinander verknüpft werden. Die grundlegende Frage lautet hierbei: Was definieren wir als Muster bei einer Zeitreihe? Betrachten wir eine beliebige Zeitreihe und versuchen dort, Muster zu entdecken. Um etwas als Muster zu erkennnen, muß es eine signifikante Struktur geben, die diese von der Umwelt unterscheidet [Morik und Köpcke 2005]. Nur durch Wiederholung ist die Struktur als solche zu identifizieren.

2.3.1. Intuitive Mustererkennung in Zeitreihen

Folgendes Zeitpolygon (es handelt sich um den Datensatz normal8 aus [E. Keogh 2006]) stellt eine Messreihe von menschlichen Pulswerten dar.



(a) (b)

Abbildung 2.3.: Die Abbildung zeigt die Zeitreihe Puls (a) und mögliche Musteridentifikationen auf der Zeitreihe (b).

Augenscheinlich gibt es sich wiederholende Strukturen (hier: Peaks), gefolgt von leichten Erhebungen. Die Spannweite von einem Peak bis zum nächsten Peak bildet in dem betrachteten Polygon ein Muster (Muster A in 2.3.1). Die einzelnen Strukturelemente, die sich graphisch unterscheiden lassen, werden als Submuster bezeichnet [Han et al. 1999]. In Abbildung 2.4 sind drei zu erkennende Submuster in Musterinstanz A dargestellt. Musterinstanzen, die einem Muster zugeordnet werden, bestehen immer aus derselben Abfolge von Submustern. Sie spielen in dieser Betrachtung keine weitere Rolle, da sich die Reihe aus sich wiederholenden Mustern zusammensetzt und eine Aufspaltung in Submuster keinen Vorteil bringt. Sobald eine Struktur einzeln auftaucht und nicht als Submuster einem übergeordneten Muster zugeordnet werden kann, bildet sie ein separates Muster. Die Existenz von Submustern ergibt sich aus den Zeitrelationen, die das Verhältnis der Teilstrukturen untereinander beschreiben. Unter den möglichen 13 primitiven Zeitrelationen befindet sich ‘B during A’, ‘B starts A’ und ‘B finishes A’ [Allen 1983]. In dieser Sichtweise repräsentiert B ein Submuster und A ein Muster.

Das Muster A der Zeitreihe ist mithilfe der Peaks ausgewählt worden. Peaks sind graphische Merkmale, die bei gut gewähltem Skalierungsfaktor mit bloßem Auge erkennbar

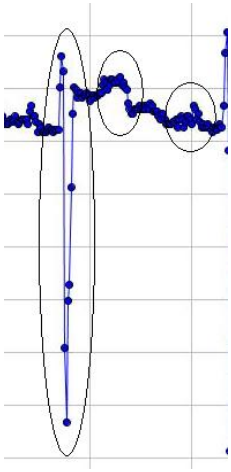


Abbildung 2.4.: Die Abbildung zeigt auf graphischer Ebene ein Muster mit gekennzeichneten Subsmustern. Die Relation zwischen Strukturen der Zeitreihe kann formell durch das Allen Kalkül, daß von J.F. Allen entwickelt wurde, beschrieben werden.

sind. Der Ansatz der *Peak-Similarity* findet als Ähnlichkeitsmaß auch beim Vergleich von Zeitreihen untereinander Verwendung [Pratt 2001]. Um Muster von Hand auswählen zu können und um Ansatzpunkte für die manuelle Mustersuche in den Datensätzen zu haben, ist die richtige Skalierung sowie die Menge der visualisierten Datenpunkte im ersten Schritt elementar [Das et al. 1997]. Verständlich, denn anhand von 5 Datenpunkten dürfte kaum eine Wiederholung komplexer Strukturen sichtbar sein, da die betrachtete Grundmenge zu klein ist. Ausreichend Datenpunkte in der Visualisierung liefern eine gute Chance, Muster aufzufinden (sofern vorhanden), da nur in einer ausreichenden Grundmenge eine Wiederholung ersichtlich wird. Die Skalierung ist der zweite wichtige Faktor bei der Erstbeobachtung. Eng damit verknüpft ist der Begriff der Normalisierung. Wenn einzelne Werte der Zeitreihe stark abweichen, sind aufgrund des großen Wertebereiches feinere Strukturen, die einen nur kleinen Bereich im Werteraum abdecken, schwerer zu identifizieren. Dies läßt sich durch die Achsenskalierung begründen. Zeitreihen, die diese Struktur aufweisen, werden normalisiert, d.h. es findet eine mathematische Transformation des Wertebereiches statt.

Definition 2.3.1 (Normalisierung)**.** *Jedes Element* x_i *der Zeitreihe* X_i *wird mit* σ *in in* x_i' *überführt, so daß gilt:*

$$x_i' = \sigma(x_i)$$

Es gibt verschiedenartige Transformationsfunktionen σ. Einige wichtige sind:

Dezimalskalierung (Teilen der Werte durch die kleinste Zehnerpotenz, so daß der größte Wert betragsmäßig nicht größer als eins wird.)

Summe Z (Transformation, die sicherstellt, daß die Summe der Elemente des transformierten Vektors Z beträgt. Am häufigsten wird für Z der Wert eins eingesetzt.)

Intervalltransformation (Hierbei werden alle Elemente der Reihe proportional in ein Intervall transformiert, so daß der Wertebereich des transformierten Vektors durch die Intervallgrenzen abgeschlossen wird. Ein gängiger Intervallbereich ist $[0, 1]$.)

Ausgehend von der in 2.1.1 eingeführten Zeitreihenbeschreibung und den vorgestellten Begriffen soll nun díe Bezeichnung Muster formalisiert werden.

2.3.2. Formalisierung des Musterbegriffes

Die Auswahl der Muster erfolgt in dieser Arbeit per Hand. Diesem Vorgang liegt die Tatsache zugrunde, daß das menschliche Gehirn in der Lage ist, aus einer ungeheuren Datenflut die wichtigen Informationen zu extrahieren und abzuspeichern , so daß ein Wiedererkennungsprozeß möglich ist.

Die vorgestellte Definition beschreibt also formell, was im ersten Arbeitsschritt manuell ausgewählt wurde. In Anlehnung an die Musterdefinition von Singh [Singh 1999] definieren wir ein Muster wie folgt:

Definition 2.3.2 (Muster)**.** *Das Muster* M_j *ist ein Objekt der Zeitreihe* X_i*. Die Instanz* m_i *mit* $i = 1...n$ *ist ein Objekt des Musters* M_j *mit* $j = 1...m$*. Alle* m_i *mit demselben Klassifikationsergebnis werden einem Muster zugeordnet. Das Muster selbst repräsentiert somit eine Klasse bezüglich aller ihm gemäß einer Klassifikationsaufgabe zugeordneten Instanzen. Wenn* $m_i \neq m_j$ *bezüglich der Klassenzugehörigkeit, dann gilt* $M_i \neq M_j$*.*

Definition 2.3.3 (Musterinstanz)**.** *Gegeben sei eine Zeitreihe* X_i *mit* $x_i \in X_i, i = 1...n$ *Dann ist eine Musterinstanz, kurz Instanz,* m_i *eine Menge von* x_i *laufend von* i *bis* $i + l$ *und* $\forall i = 1...n, l = 0...n, i + l = 1...n$*. Die* x_i *werden als Attribute bezeichnet. Den Parameter* l *nennen wir die Länge der Musterinstanz.*

Definition 2.3.4 (Klassifikation)**.** *Einzelne Klassen (Muster) werden mittels Klassifizierung, das heißt durch die Einteilungen von Objekten (Instanzen) anhand bestimmter Merkmale, gebildet.*

Das Pulsbeispiel, das zuvor behandelt wurde, hat nur ein Muster - bezeichnet mit A. Die Attribute, die das Muster bilden, sind im ersten Auftreten des Musters (d.h. die erste Instanz) gekennzeichnet durch x_{70} bis x_{190}, die Länge des Musters lautet somit l=120. Das Muster wird vier mal in der Zeitreihe wiederholt, wobei die Instanzgrenzen ineinander übergehen. Letzterer Fakt ist darauf begründet, daß jeder Punkt der Reihe mindestens einem Muster zugeordnet werden kann. Lücken in der Musterabfolge sind nicht notwendig.

In dem betrachteten Beispiel liegt nur ein Muster und somit eine Klasse vor. Sie beinhaltet 4 Objekte, welche genau die 4 Wiederholungen des Musters A darstellen, wobei jede Wiederholung für ein Objekt, bzw. Klassenbeispiel steht. Eine Zeitreihe kann verschiedene Muster beinhalten, wobei jede Instanz zu dem ihr entsprechenden Muster gehört. Somit können verschiedene Muster einer Reihe zugeordnet werden. Die Menge der Instanzen, die zu einer Klasse gehören, ist unbeschränkt. In Realdaten tauchen Muster

natürlich nicht unendlich oft auf, und durch technische Einschränkungen ist der betrachtete Zeitbereich endlich.

Alle Beispiele (Instanzen) vom selben Typus werden einem Muster zugeordnet. Der *selbe Typus*, d.h. gleiche Merkmalsausprägung, bedeutet anschaulich, daß sich die Abschnitte der Zeitreihe, die das Muster bilden, über einen großen Anteil der Spanne bezüglich ihres Verlaufes gleich verhalten. Wie *gleiches* Verhalten sich in Merkmalsausprägungen ausdrückt (oder gar in welchen Merkmalen - Merkmalskombinationen), ist jedoch nicht Schwerpunkt dieser Arbeit. Daher wurde in den Definitionen die allgemeine Aufgabe der Klassifikation beschrieben. Da in dieser Arbeit die Zuordnung von Hand erfolgt, wird der Punkt der Klassifikation und der Merkmalsauswahl nicht näher betrachtet. Im Ausblick (Kapitel 9) wird die Idee der automatischen Extraktion der Instanzen aufgegriffen. In diesem Rahmen wird ebenfalls ein arbeitsbezogener Überblick zu bereits bestehen Verfahren gegeben, um eine Einarbeitung in das Gebiet im Hinblick auf weiterführende Arbeiten zu ermöglichen.

Wenn eine Zeitreihe nun aus n Punkten (Attributen) besteht, dann ist es leicht ersichtlich, daß Lokalität sich im Allgemeinen auf einen Bereich, der kleiner als n ist, erstrecken wird. Globalität hingegen betrachtet die Gesamtmenge der n Punkte. Ausgehend von dieser Betrachtung kann nun festgehalten werden, daß ein lokales Muster eine Teilmenge der n Punkte ist und ein Globales Muster die Gesamtheit n beinhaltet.

2.3.3. Muster und das Zerlegungsmodell

Betrachten wir das in 2.1.4 vorgestellte Modell. Basierend auf der Aussage von Hand [Hand 2002]

$$\text{data} = \text{background model} + \text{local patterns} + \text{random}$$

kann nun das statistische Komponentenmodell auf den Musterbegriff übertragen werden. Das *background model* entspricht der Trendkomponente. Die Trendkomponente wiederum ist ein Teilbereich der Globalität, wobei der Trend für die Lokalität ohne Bedeutung ist. Dies ist erklärbar, da der Trend über die Menge aller Punkte ersichtlich wird und globale Muster ebenfalls über die Grundgesamtheit definiert wurden. Betrachtet man die Tatsache, daß der Trend als eine über den gesamten Zeitraum laufende Funktion angesehen wird, nämlich den Darstellbereich der Abzisse abdeckend, so ist klar, daß jeder Wert der Daten Teil dieser Funktion ist. Die *local patterns (Lokale Muster)* sind mit der Saisonalität zu identifizieren. Saisonalität ist in Zeitreihen ein periodisches Auftreten von Auffälligkeiten. Auffälligkeiten sind nichts anderes als die musterbestimmenden Strukturen, und jedes Auftreten in Perioden ist eine Wiederholung des Musters und daher eine Instanz des zugeordneten Musters. Jede Saisonkomponente ist damit ein Muster. Der irreguläre Rest mit ungenauer Zuordnung ist der Faktor *random* in der Gleichung von *Hand*. Das *background model* sowie die *local patterns* sollen identifiziert und erlernt werden, um anhand ihrer eine Prognose zu erstellen. Rauschkomponenten sollen möglichst eliminiert werden.

Abschließend sollen die vorgestellten Begriffe in einem Beispiel verdeutlicht werden.

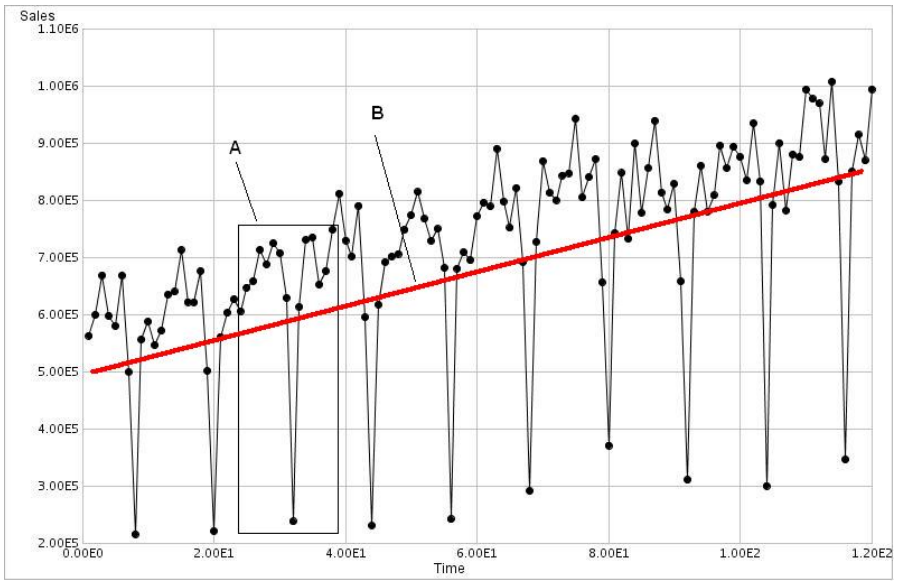


Abbildung 2.5.: Abverkäufe Papierindustrie

Beispiel 2.2 (Begriffserläuterung am Beispiel Papierverkäufe)**.** *Unten anstehend sind die Papierverkäufe über einen Zeitraum von Januar 1963 bis Dezember 1972 eines französischen Industrieunternehmens aufgezeichnet [*University *2006]. Als Darstellungsform wurde ein Zeitreihenpolygon gewählt, wobei auf der Abzisse die Zeit und der Ordinate die Verkäufe abgetragen wurden. Die Zeitreihenelemente* x_i *laufen von* i *=1-120. Das globale Muster umfaßt die gesamte Zeitreihe und hat somit eine Länge L=120. Jedes* x_i *stellt einen Messzeitpunkt dar, hier pro Monat. Die Erhebungspunkte sind äquidistant. Dem Globalen Muster ist der Trend unterlegt, dargestellt als Linear Funktion B in Abbildung 2.5. Die Funktion ist steigend und gut an den wachsenden Y-Werten der Tiefpunkte zu erkennen. Die sich zyklisch wiederholende Struktur, gekennzeichnet in 2.5 durch die Bezeichnung A, ist die Saisonalkomponente. Die Saisonalkomponente A ist die Instanz* m_3 *mit* x_i*,* i *=24-38. Jede der neun Wiederholungen stellt eine Instanz* m_i *dar, mit* $i = 1-9$ *(Die zehnte Instanz ist nicht vollständig erkennbar). Das Muster* M_1 *hat somit neun Objekte und ist die einzige Musterklasse der Reihe. Der Wertebereich aller* x_i *läuft von 215187 bis 1006852.*

Ziel ist, die Reihendaten der Zukunft vorherzusagen. Es werden im vorangestellten Beispiel die Werte x_{121} bis x_{121+l} gesucht. Welche Möglichkeiten hierfür existieren, betrachten wir im folgenden Kapitel.

3. Prognose mittels globaler und lokaler Modelle

Im folgenden Kapitel wird ein Überblick über praxisbewährte Methoden aus dem Gebiet der Zeitreihenprognostik gegeben. Da dieses Gebiet sehr umfangreich ist, wird an entsprechenden Stellen auf weiterführende Literatur verwiesen. Auf erster Ebene erfolgt ein Einblick in die verschiedenen Prognoseansätze, bevor dann die Vorhersage mittels lokaler Muster näher betrachtet wird und die verwendete Methodik Begründung findet. Im Hinblick auf die Unterscheidung zwischen globalen und lokalen Modellen kann die Verfahrensabgrenzung als Weg vom globalen Modell (GM) zum lokalen Modell (LM) nachvollzogen werden. Das Verfahren selbst sowie der mathematische Kern werden in den nachfolgenden Kapiteln formell dargestellt.

3.1. Globale Modelle

Als erste Annäherung an die Prognose wird versucht, die gesamte Zeitreihe (die Menge aller Punkte) durch ein Modell abzubilden, und über das Modell die Zukunftswerte vorherzusagen. Da man die Zeitreihe als Ganzes betrachtet, sind die gefundenen Modelle globaler Art. Ein Überblick über die Grundverfahren, die als Resultat ein GM liefern, soll nun gegeben werden.

3.1.1. Klassische Zeitreihenprognose

Der erste Ansatz zur Prognostik entspricht der mathematischen Vorgehensweise, ein Polynom zu finden, das als möglichst gute Schätzfunktion für alle Werte der Reihe dient. Diese Grundidee, die als Resultat ein globales Modell liefert, führt direkt ins Gebiet der klassischen Zeitreihenprognose und somit zur Statistik. Das Gebiet der Zeitreihenprognose wurde lange von eben diesen statistischen Methoden beherrscht. Dies galt sowohl für den Bereich der Forschung als auch für die Techniken, die in der Praxis zur Anwendung kamen. Die Methoden der Statistik setzen in vielen Fällen eine Vorverarbeitung der Daten voraus. Die drei Grundpfeiler sind das Garantieren der Stationärität, die sich daraus ergebende Trendelimination und die Elimination von Saisonalität. Hierfür steht ein großer Methodenraum zur Verfügung wie z.B. das Differenzieren (Vergleiche [Andel 1984]). Der Nachteil ist in der Definition begründet. Ein Teil der Zeitreihen muß vorverarbeitet werden bzw. scheidet aufgrund der inhärenten Datenstruktur für bestimmte Prognoseverfahren aus. Je nach Beschaffenheit der Daten nach der Preprocessing Phase kommen also vollkommen unterschiedliche Verfahren zur Anwendung. Beispielsweise kann eine Zeitreihe, die nicht das Kriterium der Saisonalität erfüllt, mittels der Regression auf die Zeit vorhergesagt werden, jedoch nur unter der Voraussetzung, daß der Trend stabil über die

Zeit ist. Im einfachsten Fall erfolgt die statistische Prognose mittels des Modells der einfachen linearen Regression, bei der angenommen wird, daß der Zusammenhang zwischen der Indexdimension d_i und der Wertedimension w_i linear ist. In Räumen, die eine höhere Dimensionalität aufweisen, ist das Verfahren als multiple Regressionsanalyse bekannt. Die Statistik unterscheidet durch den Zusatz *deskriptiv* bzw. *wahrscheinlichkeitstheoretisch*, von welcher Art der Zusammenhang der Variablen ist. Deskriptiv bedeutet, daß der Zusammenhang nicht vom Zufall abhängt, dieser ist also deterministisch. Wahrscheinlichkeitstheoretisch drückt aus, daß der Zusammenhang der Variablen nur geschätzt werden kann. Die Vorhersage besteht nun darin, mittels der unabhängigen, bekannten Variable d_i und der Regressionsgleichung die abhängige Variable w_i vorherzusagen. Eine weit verbreitete Modellklasse im Bereich der Regression, die auf den oben genannten Zusammenhängen basiert, sind die autoregressiven Modelle AR, wobei ein Wert durch seinen AR(1)- oder seine AR(p)-Vorgänger erklärt wird. Mathematischer Kern dieser Modelle sind lineare Gleichungssysteme (LGS), die oftmals mittels einfacher linearer Regression geschätzt werden können. Hingegen werden die Modelle selbst - von der Kernschätzung zu unterscheiden - mit nichtlinearer Regression abgeschätzt (z.B. die Modelle der ARMA Familie).

Betrachten wir einen weiteren biometrischen Datensatz (vernarh4.dat, Signal 1) und versuchen, die komplexen Formen durch lineare Regression vorherzusagen, so stellen wir fest, daß das Ergebnis unzureichend ist. In Abbildung 3.1 ist zum Vergleich die Originalzeitreihe in ihrer komplexen Form sowie die Regressionsgerade dargestellt.

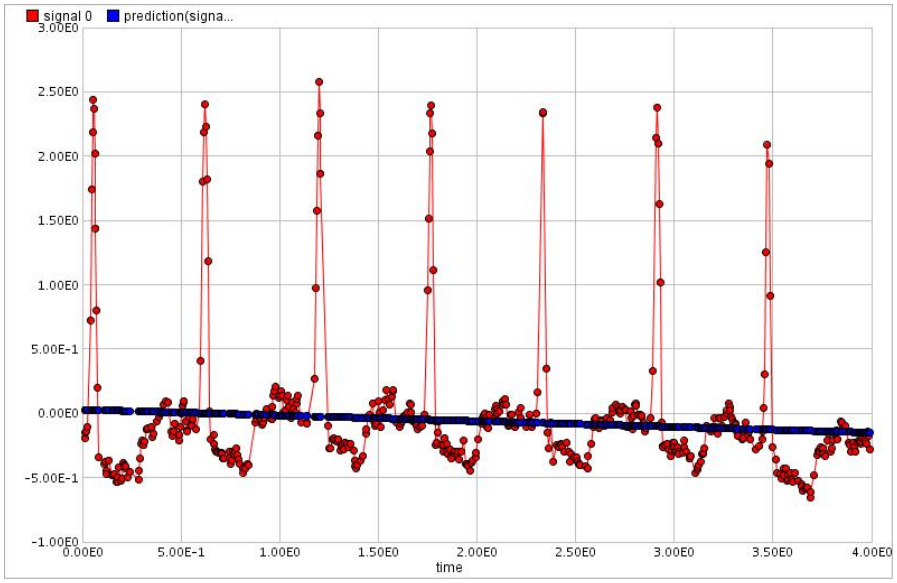


Abbildung 3.1.: Lineare Regression auf biometrischen Daten

Oft ist nicht direkt ersichtlich, welches der statistischen Modelle das beste Resultat erzielt, so daß in der Praxis zunehmend individuelle Kombinationen Verwendung finden. Da viele Vorhersagemodelle unterschiedliche Annahmen voraussetzen, ist eine Methodenkombination oftmals schwierig [Yang und Zou 2002]. Abgesehen von den herkömmlichen

Methoden und den damit verbundenen Beschränkungen im Bereich der Nichtlinearität lohnt sich ein Seitenblick in das Gebiet des Soft-Computings. Der Begriff geht auf seinen Begründer Lotfi A. Zadeh zurück, und soll die Unsicherheit bzw. Wagheit, die den Verfahren innewohnt, betonen.

3.1.2. Soft-Computing-Verfahren

Im Folgenden sollen exemplarisch praxisbewährte Methoden des Soft-Computings zwecks Abgrenzung vorgestellt werden.

Neuronale Netze

Die Prognose mittels Neuronaler Netze (ANN) basiert auf einem Zeitfenster der Größe n, das über die Zeitreihe geschoben wird. Das Training der NN erfolgt über die bereits vorhandenen Werte der Reihe. Die n Fensterwerte bilden die Inputschicht des Neuronalen Netzes. Die nächsten X Werte, die dem Prognosehorizont der Größe X entsprechen, können über die Outputschicht abgelesen werden. Das Training ist beendet, wenn die Outputwerte den eigentlichen Werten *zu Genüge* entsprechen. Nach dem Beenden der Trainingsphase kann dann an der Ausgabeschicht pro Neuron ein Zukunftswert der Reihe abgelesen werden. Aufwendig ist hierbei, daß äußere Faktoren, die eventuell bekannt sind und den Verlauf der Reihe mitbestimmen, als separate Neuronen kodiert werden müssen. Diese werden dann zusätzlich zu den Eingabeneuronen als Teil der Inputschicht verarbeitet. Vorteile bieten Neuronale Netze in chaotischen und nichtlinearen Systemen, da sie gut mit dem Rauschanteil der Daten umgehen können und selbständig funktionale Zusammenhänge erlernen [Azoff und Azoff 1994]. Diesem Vorteil stehen jedoch auch Nachteile gegenüber. Einige Probleme sind allgemeiner Natur - die Gefahr der Überanpassung (Overfitting) an die Trainingsdaten durch zuviele Neuronen und damit verbundene schlechte Ergebnisse im Bereich der Prognose sind bei ANN ebenso gegenwärtig wie bei vielen anderen Prognosemethoden. Der größte Nachteil ist für die hiesige Problemstellung jedoch, daß bekannte Zusammenhänge nicht im Vorhinein in das Netz kodiert werden können, da der Wissenserwerb der Netze in der Trainingsphase *verborgen* geschieht. Da aber gerade der hier präferierte Denkansatz auf bekannten Zusammenhängen - beispielsweise auf der Abfolge der einzelnen Musterklassen nach Wahrscheinlichkeiten - basiert, ist der Ansatz der NN nicht in der Lage, das Wissen geeignet in die Strukturen zu übernehmen. Des Weiteren wurde Wert auf das Unterscheiden der variablen Horizontgrößen gelegt, was mit einem NN kaum zu realisieren ist. Zwar kann eine unterschiedliche Anzahl von Ausgabeneuronen rein technisch aktiviert werden. Um dies zu simulieren, wäre jedoch das zugrundeliegende Netz logischerweise strukturell in beiden Fällen identisch. Somit ist auch im Bereich Horizontvariation kein Vorteil zu erhoffen. Letztendlich stellen die zumeist langen Trainingszeiten der Netze selbst einen Nachteil für die Praxis dar. Versuche im Bereich NN und Vergleichsdaten zu den bereits erwähnten ARMA Modellen finden sich in der Arbeit von [Faraway und Chatfield 1995].

Evolutionäre Algorithmen

Ein anderes Strategiefeld zur Vorhersage greift *Evolutionäre Algorithmen* auf. Die Stärke dieser Verfahren liegt darin, daß mit ihnen andere Prognoseverfahren modelliert und optimiert werden können. Erste Ansätze basierten auf Evolutionärer Programmierung und modellierten endliche Automaten so, daß sie Vorhersagen für Zeitreihen machen konnten. Ein praktisches Beispiel aus der jüngeren Zeit ist Timeweaver. Das Verfahren nutzt einen genetischen Algorithmus, um Musterkomplexe zu identifizieren, mit deren Hilfe nach dem if-then Prinzip besondere Ereignisse der Zukunft vorhergesagt werden sollen (Prediction-Rules). Dieses Verfahren bezieht sich jedoch nur auf die Vorhersage von Einzelereignissen, die im Original als 'rare' bezeichnet werden. Da es jedoch bei der hier gewünschten Vorhersage um eine kontinuierliche Fortsetzung der Zeitreihe geht, scheiden Verfahren, die nur hervorgehobene Einzelereignisse berechnen, prinzipiell aus [Weiss 1999].

Randgebiete der Prognosemethodik

Es gibt eine Menge von weiteren Verfahren zur Prognose von Zeitreihen, die jedoch in ihren Grundannahmen auf den vorgestellten Konzepten beruhen. Beispielsweise gibt es den Bereich der Neuro-Fuzzy-Systeme, die zur Prognose von Zeitreihen verwendet werden. Hierbei wird entweder kooperativ oder hybrid das NN mit einem Fuzzy-System kombiniert. Die Fuzzy-Logik findet auch bei der Regelbildung im Bereich Prognostik Verwendung. In diesem Rahmen gibt es Querverbindungen zu dem Bereich der Evolutionären Algorithmen, da das Problem, die optimale *Fuzzy-Membership* zu definieren, mit genetischen Algorithmen bestimmt werden kann. Somit wird deutlich, daß es eine große Anzahl von Arbeiten beruhend auf Verfahrenskombinationen gibt [Rüping und Morik 2003]. Da die Modelle größtenteils für spezielle Sektoren wie z.B. den Finanzmarkt entworfen wurden, ist eine allgemeine Abgrenzung am besten mit den zugrundeliegenden Basismethoden zu erreichen. Im Anhang A erfolgt eine Übersicht, mit welchen allgemeinen Kriterien (Statistische Kennzahlen) unterschiedliche Methoden verglichen werden können.

3.1.3. SVM Robustheit, Nichtlinearität und Mehrdimensionalität

Betrachten wir nun eine Einführung in die Thematik der Support Vector Machine.

Stärken und Schwächen der SVM Theorie

Ein Lernverfahren, das sich als vielversprechend im Gebiet der Prognostik herausgestellt hat, ist unter dem Namen Support Vector Machine (SVM) bekannt und basiert auf dem Ansatz des überwachten Lernens. Genauere mathematische Beschreibungen der Konzepte, die der SVM zugrundeliegen, werden im Abschnitt 5.1 vorgestellt, da sie dort zum Verständnis der benutzten Parameter vonnöten sind. SVMs können in der Praxis hervorragend für Funktionsschätzungen eingesetzt werden. Der mathematische Kern kann je nach Art der Daten und Aufgabenstellung, die an die Prognose gestellt wird, andersartig sein. Zusätzlich ermöglicht ein Satz von Funktionsparametern Kontrolle über die Qualität der Schätzung. Betrachtet man zuerst die Klasse der linear vorhersagbaren Funktionen, so

stellt man fest, daß eine SVM mit linearem Kern sich nicht anders verhält als die bereits erwähnten AR Modelle. Dies ist nicht verwunderlich, denn die AR Modelle lassen sich von einer SVM mit linearem Kern lernen. Zusätzlich konnte jedoch von Morik und Rüping gezeigt werden, daß die SVM auf Testdatensätzen eine größere Stabilität bezüglich Datenausreißern als die AR Modelle hat (Vergleiche hierzu [Rüping und Morik 2003]). Die Kernfunktionen stellen auch die eigentliche Mächtigkeit der SVM dar. Hinterlegen wir andere Kerne, so erhalten wir Zugriff auf die Klasse der nichtlinearen Modellfunktionen [Vapnik 1998]. Schwierigkeiten treten im Zusammenhang mit den Kernfunktionen derart auf, daß für eine Anzahl von Problemen gute Kernfunktionen erst ermittelt werden müssen. Viele Vorarbeiten und bekannte Kernfunktionen geben jedoch häufig Ansatzpunkte für das Ermitteln der geeigneten Kernfunktion. Ein Vorteil, der im Bezug auf Erweiterung des Konzeptes eine Rolle spielt, ist die Eigenschaft, daß SVMs sehr gute Lernergebnisse im Bereich der mehrdimensionalen Daten erzielen. Die Begründung liegt darin, daß der Generalization Error nur auf der Breite der Hyperplanmarge und nicht der Dimensionalität der Daten beruht (Abschnitt 5). Nachteilig wirkt sich aus, daß das Bestimmen der optimalen Hyperebene im Rahmen des Lernens zeitaufwendig ist, jedoch im Durchschnitt nicht so sehr wie bei NN. Dem Lernaufwand gegenüber steht der klare Vorteil der hohen Präzision und der geringen Fehlerwahrscheinlichkeit, die der SVM zueigen ist. Zusätzlich ist die SVM konvex. Somit ist es theoretisch immer möglich ,bei geeigneter Problemklasse das globale Optimum zu finden. Ferner sind Ergebnisse der SVM wegen des zugrunde liegenden mathematischen Kalküls reproduzierbar. Diese Vorteile erwirkten das nähere Betrachten der SVM.

SVM als Globales Modell

In den ersten Ansätzen, die SVM zu Prognosezwecken zu verwenden, wird mit ihrer Hilfe ein Globales Modell erstellt. Hierfür verwenden wir die SVM als Regressionstechnik. In den folgenden Abschnitten und den entwickelten Verfahren ist mit dem Begriff SVM immer die Erweiterung Support Vector Regression gemeint [Burges 1998]. Also betrachten wir die biometrischen Daten aus 3.1 und versuchen zunächst, die bekannten Werte mit Hilfe einer SVM in einem GM zu verwerten. Das Finden der Regressionsfunktion ist erfolgreich (siehe 3.2), jedoch scheitert die eigentliche Prognose. Da die SVM die Regression unter Zuhilfenahme von Trainingsdaten und den so bestimmten Stützvektoren durchführt, ist dies verständlich. Für den Prognosezeitraum existieren diese Trainingsdaten nicht und die SVM hat keine Stützvektoren für das Ermitteln der Regression zur Verfügung. Die Ergebnisse der Prognose wären unzureichend, da die SVM in ihren Grundmechanismen die Reihe *fortsetzt*. Abbildung 3.3 zeigt das Resultat der Prognose für einen Wert bezüglich der Indexdimension t+x der betrachteten Zeitreihe.

3.2. Von der Globalität zur Lokalität

Alle bereits angesprochenen Prognosetechniken liefern als Basiskonzept ein globales Modell. Erwähnt sei dazu, daß natürlich einzelne im Bereich GM angesprochene Techniken auch bei lokalen Modellen Anwendung finden können. Betrachtet man die Schwierigkeiten, ein Modell über alle Werte der Reihe im Bereich von nicht-stationären und chaoti-

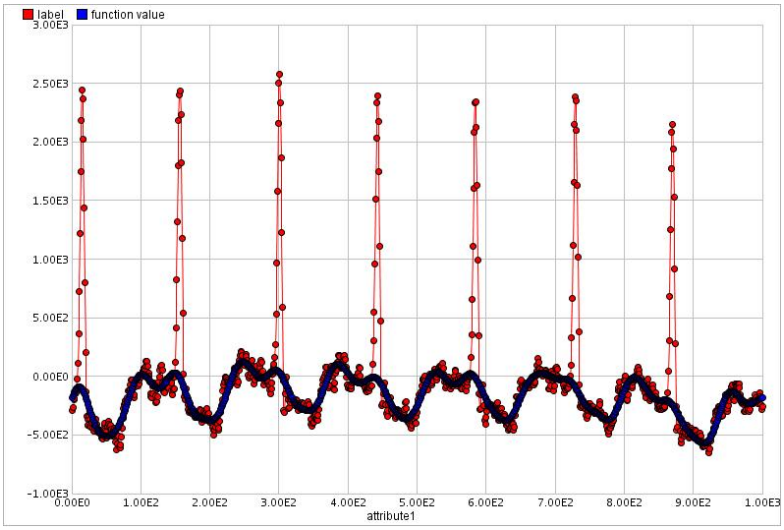


Abbildung 3.2.: Die Abbildung zeigt ein globales SVM Modell, daß über eine Zeitreihe erlernt wurde.

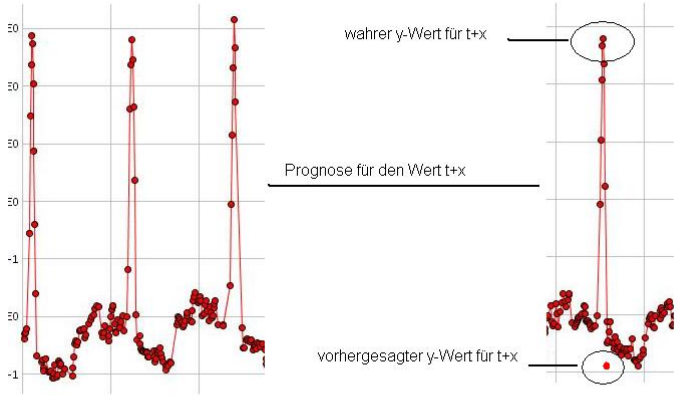


Abbildung 3.3.: Die Abbildung ist im Rahmen eines Experimentes entstanden. Sie zeigt den Prognosewert, den die SVM in ihrer Funktion als globales Modell prognostiziert.

schen Zeitreihen zu bilden, so ist es naheliegend, sich der Lokalität zuzuwenden. Intuitiv können wir dann erhoffen, bessere Abschätzungen für kleinere Teilbereiche zu finden, und somit die Probleme der globalen Modelle zu überwinden.

Erste Schritte hin zur Lokalisierung findet man bei dem Verfahren von Rüping. Die Regressionsfunktion der SVM, mit deren Hilfe die Prognose erfolgt, wird hier nicht mehr über die gesamte Zeitreihe gebildet. Die Lokalität besteht insofern, daß ein gleitendes Fenster der Größe n genutzt wird, und damit eine Unterteilung der Zeitreihe in separate Abschnitte (Vektoren) erfolgt. Auf den Datenpunkten der Vektoren (in der Abbildung 3.4 die Attribute) wird die SVM angewendet und die Regressionsfunktion ermittelt. Die Vorhersage erfolgt dann für einen Wert k+h bei einem Prognosehorizont von h und k als letztem Datenpunkt in dem gleitenden Fenster. In der Summe der Vektoren untersuchen wir aber letztendlich immer noch die Zeitreihe als Gesamtheit [Rüping 1999].

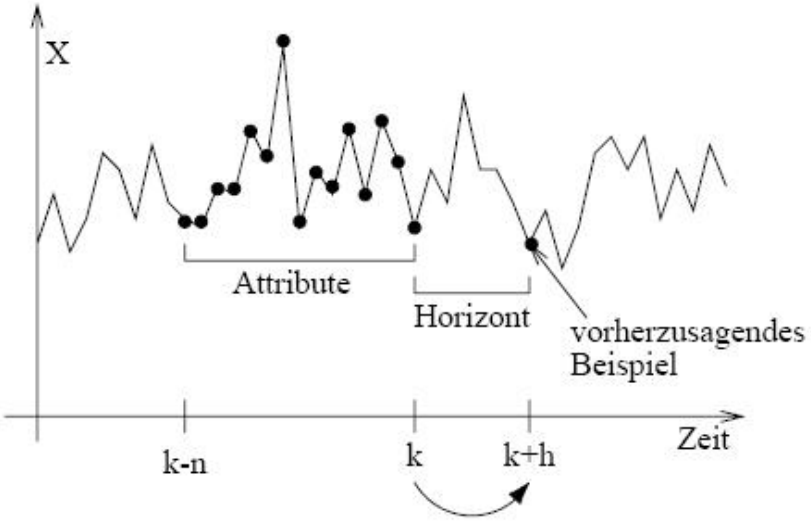


Abbildung 3.4.: Prognoseverfahren mit SVM auf Teilbereichen

3.3. Lokale Modelle

Basierend auf dem Kapitel 2 ist ein neuer Ansatz die Vorhersage mittels lokaler Modelle. In dieser Sichtweise können wir als Grundidee das Vorhersagen unbekannter Datenpunkte als Fortsetzung von lokalen Modellen, die aus den bekannten Werten gebildet wurden, sehen. Ein LM finden wir in unserem Musterbegriff wie folgt - die Instanzen eines Musters sind im Allgemeinen nur auf Teilbereichen der Zeitreihe aufzufinden, und das Muster als solches repräsentiert auch nur einen lokalen Bereich der Reihe, wobei der Begriff Muster mit seiner Repräsentanz durch den Prototyp als Modell verstanden werden darf. Extrahierten Mustern können in nachfolgenden Schritten globale Modelle unterlegt werden, um so eine realistische Vorhersage zu treffen. Hierbei treten zwei zentrale Fragestellungen auf: Wie werden die lokalen Muster identifiziert, und wie werden sie in

der Vorhersage kombiniert? Eine Methode, Muster zu bilden, besteht darin, die Zeitreihe in Subsequenzen zu unterteilen und die Subsequenzen zu Clustern. Das Ergebnis sind Elementarformen (Original: Primitive shapes), die über Regeln miteinander verbunden werden können. Mit Hilfe dieser Regeln können Aussagen 'wenn A auftritt, erfolgt B innerhalb einer Periode T' getroffen werden. Das Basiskonstrukt, das Methoden, die auf lokaler Mustersuche basieren, gemeinsam haben, wird in der oben beschriebenen Arbeit von [Gautam Das und Smyth 1998] verdeutlicht. Das Basiskonstrukt lautet: Muster zu identifizieren und diese nach Regeln miteinander zu verknüpfen, um so die Zeitreihe auch für zukünftige Zeitpunkte zu modellieren. Die Verknüpfung einzelner Segmente zu einem Gesamtkonstrukt ist eine implizite Problematik lokaler Modelle, der wir uns bei den vorherigen Betrachtungen globaler Modelle nicht zu stellen hatten.

3.3.1. Modellierung der Zeitreihe

Die Regelbildung, die zur Musterabfolgesequenz führen soll, ist oftmals problematisch [Gautam Das und Smyth 1998]. Zum einen muß eine symbolische Repräsentation der Reihendaten gefunden werden, auf der die Regeln angewendet werden können, was allgemein mit einem Trade-Off zwischen der Abstraktionsqualität und den damit erlernten Regeln verbunden ist. Zum anderen nehmen Verfahrensschritte wie beispielsweise die Fenstergröße, die für das Erstellen der Subsequenzen verwendet wird, Einfluß auf die spätere Regelbildung. Ein anderer Ansatz versucht, die Musterabfolge über Entscheidungsbäume zu realisieren und mit ihnen sequenzweise die Prognose durchzuführen [Geurts 2001]. Diese Verfahren haben oftmals den Schritt der Musteridentifikation als Teilphase implementiert, so daß Ergebnisse bezüglich der Modellierung teilweise eng mit der integrierten Implementation und den verwendeten Parametern zusammenhängen. Dies wird sehr anschaulich in einem weiteren Ansatz, in dem die Zeitreihe als Intervall-Sequenz repräsentiert wird. Die Idee ist, Muster, die mit einer Mindesthäufigkeit auftreten, zu extrahieren. Diese werden dann weiter gefiltert und letztendlich nach Regeln, die nach Informationsgewinn ausgewählt werden, miteinander verknüpft [Höppner 2002]. Die gewählte Darstellung und die Definition des Musters als Intervallsequenz variierender Länge ist also grundlegend für die anschließende Konkatenation der Muster in dieser Methode.

Da der Bereich der Musteridentifikation manuell erfolgt, existieren die Muster als Teilstücke der zu modellierenden Kette, und die eigentliche Fragestellung ist nun: In welcher Weise werden diese Teilstücke aneinandergehängt? Betrachtet man die vorhandene Zeitreihe, so kann beobachtet werden, mit welchen Wahrscheinlichkeiten Muster ineinander übergehen. Falls auf eine Instanz des Musters A in der Zeitreihe immer ein Auftreten einer Instanz des Muster B erfolgt, so können wir sagen: Die Wahrscheinlichkeit, daß Muster B nach Muster A auftritt, beträgt 100%. Dieser Ansatz bietet den Vorteil, daß auch komplexe Zusammenhänge wahrscheinlichkeitstheoretisch erfaßt werden können. Denkbar wären Ansätze wie 'wenn A und B unmittelbar hintereinander auftreten, erfolgt C mit größerer Wahrscheinlichkeit als D'. Diese Aussage hätte in der Phase der Prognose zur Folge, daß eine Zeitreihe zum Zeitpunkt $\mathtt{t}$, deren $\mathtt{t} - \mathtt{a} - \mathtt{b}$ Werte das Muster A der Länge a und das Muster B der Länge b zeigten, mit Muster C fortgesetzt würde. Diese Grundidee könnte z.B. über Markov Ketten aufgegriffen werden. Die Muster entsprechen Zuständen der Kette, und die Abfolge der Muster ist an Wahrscheinlichkeiten

für die Übergänge gekoppelt. In Kapitel 5.3 wird dieses Thema behandelt und eine Methode basierend auf Nearest Neighbour Klassifikation vorgestellt, die zur Lösung dieser Aufgabenstellung verwendet wurde.

3.3.2. Musteridentifikation und Prototyp

Musteridentifikation mittels Clusterung ist bei dem hier verwendeten Ansatz nicht nötig - vielmehr liegt eine Problemverschiebung vor. Diese Muster werden manuell identifiziert. Gleichartige Musterinstanzen, die im Allgemeinen länger als primitive Segmentstrukturen sind, werden wie in Kapitel 2 beschrieben zu Mustern zusammengefasst. Das Muster selbst wird durch einen Prototyp repräsentiert. Der Gedankengang, der dieser Idee zugrundeliegt, ist der, daß der Prototyp eines Musters der *beste Vertreter* für das zukünftige Auftreten dieser Struktur ist. Die Formulierung *beste Vertreter* bedeutet, daß die Übereinstimmung der Vorhersage mit den tatsächlichen Zukunftswerten durch den Prototyp prozentual die größte Überdeckung aufweist. Nun gibt es verschiedene Möglichkeiten, aus der Menge von Punkten einen Musterprototyp zu erlernen. Natürlich sind rudimentäre Ansätze wie Mittelwertbildung und die bereits in diesem Kapitel beschriebenen Regressionstechniken zur Erfüllung der Lernaufgabe möglich, scheiden aber aus bereits erwähnten Gründen aus.

In Anlehnung an bisherige Prognosemethoden wird die Lernaufgabe *Bilden des Prototypes* nun mit Hilfe der SVM bearbeitet. Ein offensichtlicher Anlaß ist, daß die gesuchten Schätzer für die Prototypen der Nichtlinearität zuzuordnen sind, da die Muster aufgrund ihrer Komplexität im Regelfall nicht durch Geradengleichungen abgeschätzt werden können. Betrachtet man rückblickend die Schwäche der SVM als globales Modell zur Prognose, so fällt in der differenzierten Aufgabenstellung auf, daß die SVM hier auch nicht die Aufgabe erfüllen muß, unbekannte Werte vorherzusagen. Die vorgestellten Stärken der SVM werden nun benutzt, um aus der Instanzmenge eines Musters den Prototyp zu erlernen. Die eigentliche Vorhersage (der Punkt, an dem die SVM ohne Fensterung sich als untauglich erwiesen hat) ist dann Aufgabe der Prototypen und ihrer Abfolge. Im nachfolgenden Kapitel werden die Voraussetzungen an die Daten sowie deren Vorverarbeitung näher erläutert, denn darauf arbeiten die Methoden, und einige Punkte sind für die Anwendung zu beachten. Anschließend wird genau betrachtet, wie das Erlernen des Prototypes mit Hilfe der SVM erfolgt.

4. Vorverarbeitung und Vorüberlegungen

Das folgende Kapitel beschäftigt sich mit der Vorverarbeitung und Auswahl der Datensätze, die im Rahmen dieser Arbeit durchgeführt wurden. Es dient der Vollständigkeit der Arbeitsschritte und soll keinesfalls eine allgemeine Abhandlung über Daten Preprocessing darstellen. Der letzte Abschnitt beschäftigt sich mit Vorüberlegungen zum Prognosehorizont, die im Anschluß zu zwei verschiedenen Vorhersagemethoden führt.

4.1. Datenbeschaffung und Auswahlkriterien

Die erste Aufgabe bestand in der Beschaffung und Sichtung der Daten. Im Rahmen der Datenbeschaffung wurde auf Time Series-Datenbanken zugegriffen, insbesondere die von Eamon Keogh [E. Keogh 2006] und die Time Series Data Library [University 2006]. Um die Performanz der entwickelten Methoden gegen Vergleichsmethoden testen zu können, wurden auch Datensätze des UCI Datenarchivs verwendet [Hettich und Bay 2006]. In erster Instanz mußten die großen Datenmengen auf Eignung geprüft werden. Die Eignung richtete sich nach Kriterien, die eng mit dem Musterbegriff und seiner Definition von Wiederholung und Struktur verbunden sind (Vergleiche 2.3.2).

Muster Die Zeitreihe muß mindestens eine zu identifizierende Struktur aufweisen.

Zeitreihenlänge Die Zeitreihe muß über eine gewisse Mindestlänge verfügen, da nur so Wiederholungen in Strukturen sichtbar sind.

Kontinuierliche Abtastung Der Zeitreihe muß eine möglichst kontinuierliche Werteentnahme zugrundeliegen, da sonst die identifizierten Instanzen und die erlernten Prototypen für die Prognose untauglich sind.

Das erste Argument ist selbsterklärend, da wir ohne das sichtbare Vorhandensein von Mustern auch keine erlernen können (Beispiel im Anhang B). Das zweite Argument begründet sich neben der Musteridentifikation auch noch darauf, daß die Regressionsgüte der SVM, die zur Prototypmodellierung verwendet wird, von der Anzahl der Lernbeispiele abhängt. Ist die Zeitreihe nicht lang genug (also zuwenig Instanzen), wird letztendlich der Prototyp im Schnitt für die Prognose weniger geeignet sein. Der letzte Punkt erklärt sich dadurch, daß die lückenhafte Abtastung über große Zeiteinheiten zu einer verfälschten Strukturbildung führt. Es werden dadurch Muster erlernt, die durch die fehlenden Zeiteinheiten in der Realität nicht auftreten können - es wird ja nur Unvollständiges abgebildet. Somit können die auf den Mustern gebildeten Prototypen auch zukünftige Werte nicht korrekt approximieren. In Zeitreihen waren fehlende Daten gekennzeichnet und wurden vor Anwendung der Regression mittels eines Wertefilters aus der Beispielmenge entfernt. Das Vorhandensein unvollständiger Zeitreihen sollte jedoch nicht vergessen werden. Im Ausblick wird gezeigt, wie mit den entwickelten Verfahren auch die Aufgabenstellung der

Zeitreihenrekonstruktion gelöst werden kann (9.2.3).

Zeitreihenlänge und Abtastung konnten anhand der Wertetabellen überprüft werden und bildeten das erste Ausschlußkriterium. Um das Kriterium des Musters zu überprüfen, war es jedoch nötig, die Zeitreihen zu visualisieren und ihre Struktur zu prüfen. Nur Zeitreihen, die alle Kriterien erfüllen, wurden zur Entwicklung der Verfahren herangezogen. Aufgrund der Quellenangaben der Daten konnte eine erste Selektion getroffen werden. Von Daten, denen ein sich wiederholender Prozess zugrunde liegt, durfte erhofft werden, daß sich die Zeitsymmetrie auf Datenebene (und damit unabweichlich in der Zeitreihe selbst) widerspiegelte. Aus diesem Grund wurden vor allem Messdaten aus dem Bereich der Vitalfunktionen (beispielsweise Muskelkontraktionen), Zeitreihen mit zyklischen Ereignissen (beispielsweise Verkaufsreihen von Produkten, die Saisonalschwankungen unterliegen) und Bewegungsabläufe im Raum einer näheren Betrachtung unterzogen.

Die so ausgewählten Zeitreihen erfüllen im Rahmen dieser Arbeit verschiedene Aufgaben. Bei der Problemeinführung wurden oftmals Zeitreihen konstruiert oder Realdaten modifiziert, um so dem Leser das Verständnis für besondere Problemstellungen zu ermöglichen. Im Rahmen der Experimente sind reale Zeitreihen und die Bearbeitung jener zur Anwendung gekommen, um die Arbeit methodisch zu dokumentieren. In der Verfahrensbewertung wurden Zeitreihen ausgewählt, für die Vergleichsdaten existieren, um die entwickelten Methoden gegen Vergleichsmethoden zu testen. Alle Zeitreihen sind bei ihrer Einführung referenziert oder als konstruiert gekennzeichnet.

4.1.1. Allgemeines Pre- und Postprocessing

Die entwickelten Methoden wurden auf univariaten Datensätzen geprüft und evaluiert. Obwohl eine Erweiterung auf multivariate Datensätze im Ausblick gegeben wird (9.2.1), bestand der erste Schritt nach der Datenauswahl in der Dimensionsreduktion des Wertebereiches. Viele Datensätze (gerade aus dem Bereich EEG) wiesen mehrere signifikante Werte per Index auf. Daher wurden die Datensätze aufgeteilt, so daß jedes Merkmal in einer eigenen Zeitreihe dargestellt werden konnte. Da die Datensätze aus verschiedenen Bezugsquellen stammen, ist kein einheitliches Datenformat gegeben. Somit mußten die Files manuell vorverarbeitet werden, um in die Lernumgebung importiert werden zu können. Dies betrifft insbesondere das Benennen von Attributen, da RM für gewisse Operationen (wie beispielsweise Merge) gleiche Attributsbezeichnungen fordert. Des Weiteren mußte die Anordnung der Werte einheitlich sein. Die Zeitreihe in unbearbeiteter Form mußte in Gestalt einer Wertetabelle für weitere Arbeitsschritte vorliegen. Dies konnte durch Tabellenkalkulationsprogramme auf einfache Weise sichergestellt werden. Benötigt wurde zusätzlich oftmals eine Reihe von Datentransformationen wie Vertauschung von Zeilen und Spalten oder einheitliche Interpunktionszeichen. Viele Vorverarbeitungsschritte ließen sich durch verschiedene Werkzeuge unterstützen. Häufig standen mehrere Möglichkeiten zur Auswahl - es wurde jedoch soviel wie möglich in der Lernumgebung RM umgesetzt, da diese eine große Bandbreite an Preprocessing-Methoden mitführt.

Bei praxisbezogener Anwendung kann es in einem Postprocessing-Schritt sinnvoll sein, die Funktionswerte der Prognose mathematisch zu runden. Wenn z.B. die Anzahl der

Menschen an einem bestimmten Ort vorhergesagt werden soll, sind ganzzahlige Prognosewerte angezeigt. Bei der Vorhersage von Temperaturentwicklungen mag eine größere Genauigkeit gefordert werden. Da das nachträgliche Runden der Zahlen keine methodische Besonderheit darstellt, wird in den hier ermittelten Regressionsmodellen sowie tabellarischen Dokumentationen darauf verzichtet.

4.1.2. Transformationen

Zu Beginn der Arbeit sind allgemeine Begrifflichkeiten eingeführt worden, unter ihnen die Normalisierung, die als Vorverarbeitungsschritt für die Daten benötigt wird. Die Transformation der Attributsdomänen ist unerläßlich im Bereich der SVM. Arbeiten aus dem Bereich SVM haben gezeigt, daß die Vorverarbeitung mit der Normalisierung als elementarstem Bestandteil das Lernergebnis in einem gleichstarken Maß wie die Parameterwahl im Kernel zu beinflussen vermag [Sven F. Crone und Weber 2006]. Durch die Transformation der Attributsdomäne (z.B. die Intervalltransformation) soll verhindert werden, daß Attribute mit großer numerischer Spannweite diejenigen dominieren, die nur über eine kleinere Spannweite verfügen. Dies ist für die hier behandelte Anwendung bedeutsam, da jeder Abtastpunkt ein Attribut repräsentiert. Blicken wir vorausschauend in die anstehenden Arbeitsschritte und betrachten die übereinandergeschobenen Instanzen, so fällt auf, daß genau in diesen Bereichen häufig große Attributsspannweiten auftreten können (Vergleiche Abbildung 4.1). Ein weiterer Grund für die Anwendung der Normalisierung ist, daß sehr große Werte bei Attributen zu Berechnungsproblemen bezüglich ausgewählter Kernelfunktion führen können. Dieses Faktum bezieht sich auf Polynomial und Linear Kernel (Definition 5.1.1), die bei der entwickelten Methodik nicht zur Anwendungen kamen, jedoch vollständigkeitshalber erwähnt werden sollen.

Die Daten werden zu Beginn der Lernaufgabe normalisiert - dieser Vorgang erstreckt sich auf die Daten, die zum Erstellen des Vorhersagemodells mittels SVM benötigt werden (d.h. die Trainings- und Validationsdaten). Das erlernte Modell und somit die Attribute, die dieses bilden, werden nach Abschluß der Lernaufgabe der inversen Transformation unterzogen. Die eigentliche Prognose mittels des Modells postuliert wieder nichtnormalisierte Daten. Diese Daten können somit in der Bewertungsphase des Modells direkt mit den Testdaten verglichen werden.

Literaturrecherche ([Schölkopf und Smola 2002] und weiterführende Arbeiten zur Normalisierung des Feature Space und der Kernfunktionen [Graf und Borer 2001]) sowie die Funktionsweise der SVM haben zum Rückschluß geführt, daß auch für die SVM als Regressionsmodell vorangestellte Transformationen des Wertebereichs von größerer Bedeutung sein können. Der Beweis dieser Aussage könnte durch einen informellen Gegenbeweis (wie folgt) vorgenommen werden.

These 4.1.1. *Würde die Art der Transformationen die Gestalt der Regressionsmodelle nicht beinflussen, so müßten die gewonnenen Regressionsmodelle nach inverser Transformation deckungsgleich zu dem Regressionsmodell mit Standard Transformation (Intervalltransformation [0,1]) sein.*

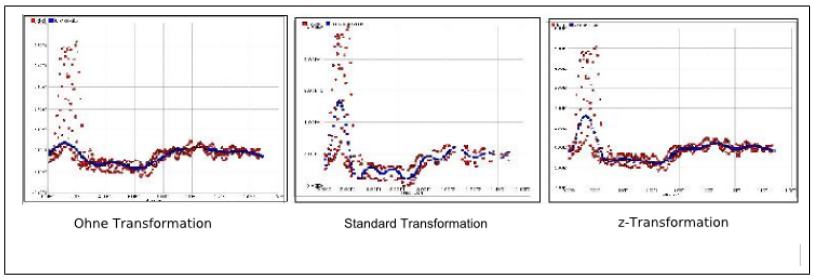


Abbildung 4.1.: Transformationsfunktionen

Da sich die Ergebnisse der SVM reproduzieren lassen, sind die diversen Perfomancevektoren auf gleicher Eingabe bei unterschiedlichen Transformationsfunktionen als Beweis dieser Aussage zu deuten. Experimente im Rahmen dieser Arbeiten führten jedoch zu identischen Performancevektoren (RMSE, AE, NAE, RRSE (siehe Anhang A)), da nur der Normalization Operator vorgeschaltet werden konnte. Die hier verwendete SVM Implementierung nutzt eine interne Normalisierung, so daß keine separate Datenbehandlung möglich war. Die interne Normalisierung ließ sich bei der LibSVM nicht auf Anwenderebene durch andere Transformationsfunktionen ersetzen (dies ist bei der überwiegenden Anzahl der Implementierungen der Fall), so daß vorgeschaltete Operatoren durch die wiederholte operatorinterne Normalisierung negiert wurden. Ein Experiment, daß obige These bestärkt, konnte in RM mittels der W-SVMreg Implementierung durchgeführt werden. Hierbei wurde die z-Transformation, die Standard Normalisierung und das Unterlassen jeglicher Normalisierung auf verschiedenen Datensätzen getestet - variierende Intervalltransformationen waren nicht möglich. Die Werte geben Grund zur Annahme, daß trotz des expliziten Verzichts eine interne Normalisierung verwendet wurde. Diese Aussage konnte durch Tests mit anderen SVM Implementierungen, die durch ihre interne Normalisierung vergleichbare Performancevektoren produzieren, verifiziert werden. Aus genannten Gründen darf das Experiment nur als Ansatzpunkt für die Auswirkung der Transformationsfunktionen und keinesfalls als Güteeinschätzung verstanden werden. Abbildung 4.1 zeigt eine Visualisierung einer Versuchsreihe, Tabelle 4.1 die Performancevektoren (Datenquelle: [E. Keogh 2006]).

4.1.3. Trendelimination

Zu Beginn sind die Komponenten der klassischen Zeitreihenanalyse vorgestellt worden, unter ihnen der Trend (Beispiel in Abbildung 2.5). Die Elimination der Trendkomponente ist ausschlaggebend für das korrekte Erlernen des Prototyps. In der referenzierten Abbildung unterliegt die Zeitreihe einem Trendverlauf, und das Muster A weist Wertedifferenzen von 15 Einheiten auf (man beachte die Peaks). Trendelimination bedeutet das Übereinanderschieben von Instanzen bezüglich der Wertedimension. Das gleiche Prinzip kommt später auch für die Indexdimension zur Anwendung, beschrieben durch den Vorgang der Indexsubtraktion. Nur über Instanzen, die sich möglichst überdecken, kann ein Regressionsmodell erlernt werden (Vergleiche hierzu 3.3.2). Auch für die kurzfristigen Prognosen muß eine trendbereinigte Reihe in die Methoden eingehen. Die Nearest Neighbour-Suche, die auf einer Distanzmetrik beruht, verwertet den Trend andernfalls

Versuchsreihe A

Transformationsfunktion	RMSE	AE	NAE	RRSE
Ohne Transformation	8.537 +/- 0.598	6.356 +/- 0.443	0.553 +/- 0.026	0.630 +/- 0.033
z-Transformation	12.178 +/- 0.576	9.738 +/- 0.534	0.848 +/- 0.028	0.898 +/- 0.028
Standard Transformation	11.516 +/- 0.556	9.071 +/- 0.510	0.790 +/- 0.033	0.850 +/- 0.034

Versuchsreihe B

Transformationsfunktion	RMSE	AE	NAE	RRSE
Ohne Transformation	492.871 +/- 77.057	222.794 +/- 20.305	0.809 +/- 0.044	0.975 +/- 0.036
z-Transformation	540.929 +/- 95.094	256.228 +/- 35.518	0.924 +/- 0.009	1.068 +/- 0.059
Standard Transformation	536.695 +/- 97.293	254.157 +/- 36.893	0.916 +/- 0.012	1.059 +/- 0.064

Versuchsreihe C

Transformationsfunktion	RMSE	AE	NAE	RRSE
Ohne Transformation	509.962 +/- 67.232	261.337 +/- 17.951	0.951 +/- 0.062	1.011 +/- 0.014
z-Transformation	540.815 +/- 95.358	256.441 +/- 35.441	0.925 +/- 0.008	1.067 +/- 0.059
Standard Transformation	536.566 +/- 97.168	254.096 +/- 36.783	0.916 +/- 0.012	1.058 +/- 0.064

Tabelle 4.1.: Die Tabelle zeigt die Performancevektoren, die auf verschiedenen Mustern bei unterschiedlicher Normalisierung ermittelt wurden. Verwendet wurde der W-SVMreg Operator mit RBF Kernel in RM, sowie 10-fache Kreuzvalidierung und optimiertes C.

additiv im Ergebnis der Distanzberechnung. In der Statistik sind für die Komponentenextraktion verschiedene Filterverfahren zur Trenderkennung bekannt und werden in parametrische und nichtparametrische Verfahren unterteilt. Der Begriff Filterverfahren deutet an, daß die Verfahren darauf beruhen, den Trend bzw. die übrigen Komponenten der klassischen Zeitreihenanalysen zu separieren. Durch das Separieren läßt sich der Trend entfernen, um die lokalen Muster korrekt weiterzuverarbeiten. Für die eigentliche prognostische Modellierung wird dann der Trend als additive Komponente unterlegt, um eine korrekte Prognose zu erzeugen. Die wichtigsten Filterverfahren der Statistik aus dem parametrischen Bereich sind der Maximum-Likelihood-Schätzer bzw. die Methode des Minimum-Quadrat-Schätzers. Unter den Methoden der nichtparametrischen Verfahren findet sich der Moving-Average, und als allumfassende Methode die exponentielle Glättung. Allumfassend beschreibt die Tatsache, daß die exponentielle Glättung nicht nur lineare Trendkomponenten, sondern auch exponentielle und gedämpfte extrahieren kann ([Donner 2007]). Je nach Art der Zeitreihe ist eine andere Trendapproximation angebracht. Da ein großer Datensatz aus dem EEG Bereich stammt, liegt naturgemäß kein Trendverlauf vor. Zeitreihen, bei denen eine Trendelimination benötigt wurde, wurden mit Hilfe des Programmes B4V.1 bearbeitet [Wiesbaden 2004]. Hierfür wurde der Trend mittels der Formel separiert :

Trend = Originalzeitreihe - Saisonalkomponente - Rauschkomponente.

Die Trendfunktion (Abbildung 4.2) wurde durch grundlegende Funktionen geschätzt (z.B. Geradengleichungen). Danach wurde die Zeitreihe mittels Subtraktion der Trendkomponente bereinigt, um die Bedingung für die eigentliche Methodik sicherzustellen. Zur Visualisierung wurde die Originalzeitreihe modelliert, und die Attribute des Prognosebereichs wurden mit der Trendfunktion unterlegt. Dies bedeutet, daß zu den Attributwerten die Funktionswerte der Trendfunktion addiert werden.

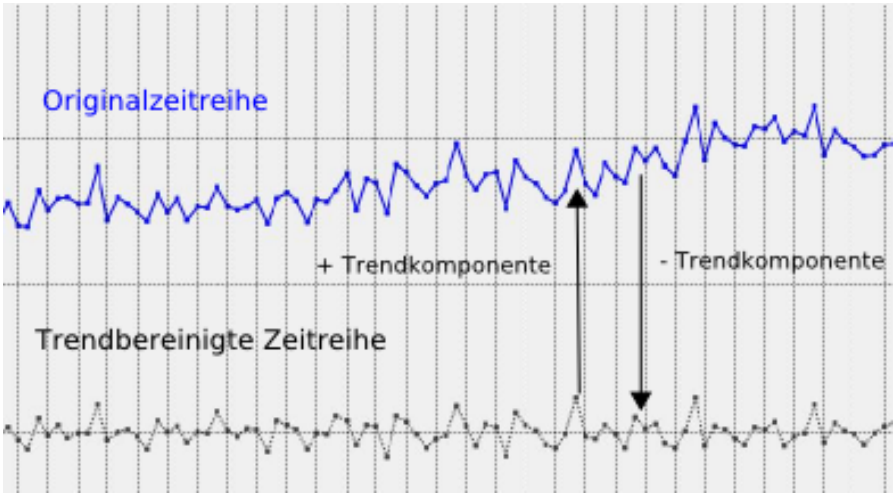


Abbildung 4.2.: Das Beispiel zeigt das Resultat einer Trendbereinigung anschaulich und wurde mit dem Programm B4V.1 ermittelt.

4.2. Prognosehorizonte - Fallunterscheidung durch individuelle Anforderung

In der Einführung wurde der Begriff Prognosehorizont eingeführt (Vergleiche 2.1.4). Der Prognosehorizont, d.h. die zeitliche Reichweite der Vorhersage, bildet die Basis für die Methodenwahl. Begriffe wie *langfristig* und *kurzfristig* dürfen nicht ohne Kontext der zugrundeliegenden Zeitreihe verwendet werden. Als langfristige Prognose bezeichnen wir eine Vorhersage über eine Zeitspanne, die mehr als die maximale Musterdurchschnittslänge umfaßt, in der Regel jedoch um ein Vielfaches länger ist. Eine kurzfristige Prognose erstreckt sich über nur wenige Attribute und ist im Allgemeinen nicht länger als die Durchschnittslänge der Muster. So wird deutlich, daß die Auswahl des Prognosehorizontes nur in Beziehung zur gesamten Zeitreihe zur Methodenwahl genutzt werden kann. Genauer gesagt: In der hier entwickelten Prognosemethodik ist nur die Durchschnittslänge der Muster ein Anhaltspunkt für die Entscheidung, ob der Wert t+x über die Methodik der kurzfristigen Prognose vorhergesagt werden kann. Dieser Fakt findet darin Begründung, daß Instanzen nur vervollständigt werden und somit nicht größer als das ihnen zugeordnete Muster werden können.

Beispiel 4.1 (Prognosehorizont)**.** *Gegeben sei eine Zeitreihe mit 800 Attributen und zwei Mustern. Das längere Muster umfaßt im Durchschnitt 80 Attribute. Dann ist die Vorhersage für den Zeitraum von 20 Attributen kurzfristig, für den Zeitraum von 400 Attributen jedoch langfristig. Würde das längere Muster nur 10 Einheiten umfassen, so wäre der Prognosezeitraum von 20 Attributen schon in den Bereich der langfristigen Prognose einzuordnen.*

Die langfristige Prognose darf als Strukturprognose der Zeitreihe betrachtet werden. In den Begriff der Strukturprognose geht der Sachverhalt ein, daß mittels der lokalen Muster (Strukturen) die Vorhersage ermöglicht wird. Da diese Verfahrensweise eine allgemeine

Lösungsmöglichkeit der Prognoseaufgabe darstellt (eine globale Methode), soll sie zuerst vorgestellt werden. Sie kann in jedem Fall zur Anwendung kommen und ermöglicht auch Vorhersagen für den Bereich, der mittels kurzfristiger Prognosen modelliert werden kann. Die kurzfristige Vorhersage verwendet zusätzlich ungenutztes Detailwissen, um möglichst exakte Werte für die nahe Zukunft vorherzusagen. Sie bildet in diesem Sinne eine lokale Technik , da sie nur auf lokal begrenzte Zeiträume anwendbar ist und neben globalem Wissen (Musterabfolge) lokales Wissen (unvollständige Instanz) nutzt. Beide Vorgehensweisen können bei Bedarf miteinander kombiniert werden. Zum Beispiel kann eine kurzfristige Prognose erfolgen und die zusätzlich zu modellierenden Werte über die langfristige Prognosemethodik gebildet werden. Aufgrund der Optionalität der kurzfristigen Prognose, deren Ziel eine höhere Genauigkeit für bestimmte Prognosebereiche ist, wird die kurzfristige Prognose nachfolgend manchmal als Verfeinerung der Strukturprognose bezeichnet. In den nächsten Kapiteln sollen die Prognosemethoden ausführlich betrachtet werden.

5. Langfristige Prognosen mittels SVM

Der Abschnitt 3.3.2 führte zur Grundidee, Prognosen unter Zuhilfenahme der SVM durchzuführen. Der Grundablauf, der zur Gewinnung des essentiellen Prototypes führt, ist in Abbildung 5.1 dargestellt. Die Segmentierung wurde bereits in den vorherigen Kapiteln erläutert. Daher soll nun hier die Dynamik der SVM-Regression näher betrachtet werden. Die Modellierung der Prototypen und ihre Abfolge bildet den Abschluß des Kapitels, so daß wir am Ende die Module des Verfahrens vollständig aufgeschlüsselt haben. Diese Verfahren werden verständlich erklärt - für den genauen Versuchsaufbau sei auf das Beispielexperiment A.1. in Abschnitt 7.1.1 verwiesen.

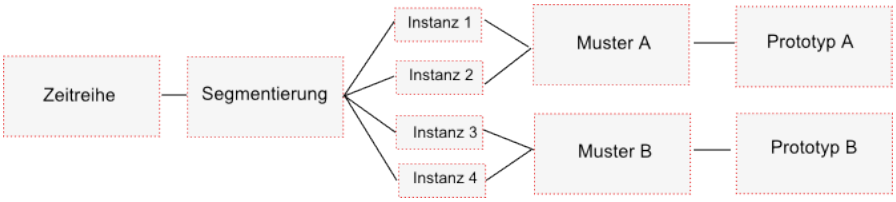


Abbildung 5.1.: Moduldarstellung Prognose mittels SVM

5.1. SVM Einführung

In diesem Abschnitt wird eine kurze Einführung in die Thematik der Support Vector Machines (SVM) mit dem arbeitsbezogenen Schwerpunkt auf die Regressionsanalyse gegeben. Für Vertiefungen der Thematik und allgemeine Klassifizierung mittels SVM sei auf [Burges 1998] und [Schölkopf und Smola 2002] verwiesen.

5.1.1. Kernidee SVM

SVM ist die Abkürzung für Support Vector Machine und kann als Klassifikator betrachtet werden. In deutschen Texten findet man die gebräuchliche Übersetzung Stützvektormethode. Sie zählt zur Kategorie der Large Margin Classificators. Dieser Begriff basiert darauf, daß sie in ihrer Grundfunktion eine Menge von Objekten in zwei Klassen teilt. Zur Unterteilung dieser Klassen wird eine Klassengrenze gezogen, die so genannte Hyperebene. Bei linear separierbaren Problemen gibt es jedoch unendlich viele Möglichkeiten, lineare Aufteilungen zu finden. Folgendes Kriterium gilt es darum zu erfüllen: Der Bereich um die Hyperebene, in dem sich keine Objekte befinden, soll maximal sein - also soll die Hyperebene einen breiten Rand haben (Englisch: large margin). Eine Hyperebene, die dieses Kriterium erfüllt, wird optimale Hyperebene genannt. Die Funktion des breiten Randes ist es, später zu klassifizierende Objekte möglichst gut einzusortieren

(Generalisierungsfähigkeit). Zur mathematisch exakten Lagebestimmung sind nur die der Hyperebene am nächsten liegenden Objekte von Relevanz. Sie bestimmen die exakte Positionierung der Klassengrenze und tragen den verfahrensgebenden Namen Stützvektoren (support vectors).

Hyperebenen können nicht verformt werden, also scheint zunächst die Beschränkung auf linear trennbare Objektmengen notwendig zu sein. Die Mächtigkeit der SVM ergibt sich aus dem Kern-Trick. Die Objekte bzw. die Merkmalsvektoren werden durch eine nichtlineare Funktion vom Eingaberaum (Input Space) in einen hochdimensionalen Raum (Feature Space) abgebildet. Im hochdimensionalen Raum lernt die SVM nun den linearen Klassifikator, obwohl der eigentliche Eingaberaum nicht linear separabel ist. Für den Eingaberaum bedeutet dies, daß beliebig komplexe nichtlineare Klassifikatoren ihre Anwendung finden können. Oben beschriebene Abbildungen in höherdimensionale Räume sind häufig schwer zu berechnen. Unter Verwendung von Kernfunktionen können Hin- und Rückrechnungen in die verschiedenen Räume durchgeführt werden, ohne sie tatsächlich berechnen zu müssen. Bei dem Schritt der Rückrechnungen muß jedoch die Einschränkung gemacht werden, daß dieser zwar theoretisch durchführbar ist, jedoch praktisch oftmals nicht umsetzbar, z.B. bei bestimmten Kernfunktionen und unendlichdimensionalem Feature Space. Kernfunktionen müssen bestimmte Bedingungen erfüllen (Mercer's Theorem), um zu gewährleisten, daß bei der Abbildung bestimmte Eigenschaften erfüllt werden. Nicht jede Funktion kann folglich als Kernfunktion verwendet werden. Im Folgenden sind die gebräuchlichsten Kernfunktionen nach [Schölkopf und Smola 2002] definiert.

Definition 5.1.1 (Kernel)**.**

$$\textit{Linear}\ k\langle x, x'\rangle = x^T x' \tag{5.1}$$

$$\textit{Polynomial}\ k\langle x, x'\rangle = (\langle x, x'\rangle + v)^d \tag{5.2}$$

$$\textit{Radial Basis Function}\ k\langle x, x'\rangle = exp\left(\frac{\|x - x'\|^2}{2\sigma^2}\right) \tag{5.3}$$

$$\textit{Sigmoid}\ k\langle x, x'\rangle = tanh\ (\kappa\langle x, x'\rangle + v) \tag{5.4}$$

5.1.2. Regressionsanalyse mittels SVM - SVR

Der Einsatz der SVM als Mittel zur Regressionsanalyse entspricht einer methodischen Erweiterung der zuvor beschriebenen SVM Struktur. Der Betrachtungsschwerpunkt wurde auf das Grundverständnis und explizite Gleichungen, deren Parameter später das Aussehen des Prototyps bestimmen, gelegt. Das Prinzip der Regression wurde bereits in den vorherigen Abschnitten eingeführt. Anstelle der statistischen Methoden zur Funktionsschätzung (3.1.1) wird diese Aufgabe mittels SVM gelöst. Gegeben sei eine Menge von Trainingsdaten (Musterinstanzen),

$$(x_1, y_1; ...; x_l, y_l) \subset \mathbb{R}^d \times \mathbb{R}$$

auf denen mittels SVM eine Funktionsschätzung von f(x) durchgeführt werden soll. Für den linearen Fall bedeutet das:

$$f(x) = \langle w, x \rangle + b, w \in \mathbb{R}^d, x \in \mathbb{R}.$$

.
Die Funktion f(x) sollte immer so *flach* wie möglich sein, was gleichbedeutend mit der Minimierung der Norm w ist. Der anschauliche Begriff *flach* bedeutet, daß die Steigung der Kurventangente in jedem Punkt so minimal wie möglich ist. Da möglicherweise nicht alle Punkte exakt auf der Funktion liegen, wird eine Fehlerfunktion eingeführt, die Strafkosten für die abweichenden Punkte angibt - die sogenannte Loss Funktion. Es gibt verschiedene Arten von Loss Funktionen: Die Quadratische, die Laplace, die Huber und ϵ-insensitive Loss Funktion. Die ϵ-insensitive Loss Funktion ermöglicht Einsparungen im Bereich der Support Vectors (SV) und wurde als Erweiterung der Huber Funktion entwickelt. Sie kann als Trade-Off zwischen der Robustheit der Huber Funktion und den Einsparmöglichkeiten bezüglich der Anzahl der SV angesehen werden. Die ϵ-insensitive Loss Funktion hat die Eigenschaft, nur Datenpunkte mit einem Fehler größer als ϵ zu bestrafen (Vergleiche Abbildung 5.2, entnommen aus [Smola und Schölkopf 1998]). Sie erfüllt das Prinzip der strukturellen Risikominimierung und wird somit als Fehlerfunktion verwendet. Die ϵ-insensitive Loss Funktion lautet:

$$|\xi|_\epsilon = \begin{cases} 0 \text{ wenn } |\xi| \leq \epsilon \\ \text{sonst } |\xi| - \epsilon \end{cases}$$

Der Funktion liegt folgender Gedankengang zugrunde: Um Fehler außerhalb des ϵ Randes zuzulassen (was aufgrund des konvexen Optimierungsproblems und seinen Bedingungen nötig sein kann - siehe [Smola und Schölkopf 1998]), können Schlupfvariablen (Slack Variable) ξ_i, ξ_i^* eingeführt werden. Sie sind eine Abschwächung der Nebenbedingungen, so daß für Punkte außerhalb des ϵ Randes die Strafe proportional zum Ausmaß der Abweichung ausfällt. Damit wird nun die Minimierung folgender Gleichung gesucht:

$$\frac{1}{2} \|w\|^2 + C \sum_{i=1}^{l} (\xi_i + \xi_i^*).$$

Unter den Bedingungen:

$$\begin{aligned} y_i - \langle w, x_i \rangle - b &\leq \epsilon + \xi_i \\ \langle w, x_i \rangle + b - y_i &\leq \epsilon + \xi_i^* \\ \xi_i, \xi_i^* &\geq 0 \end{aligned}$$

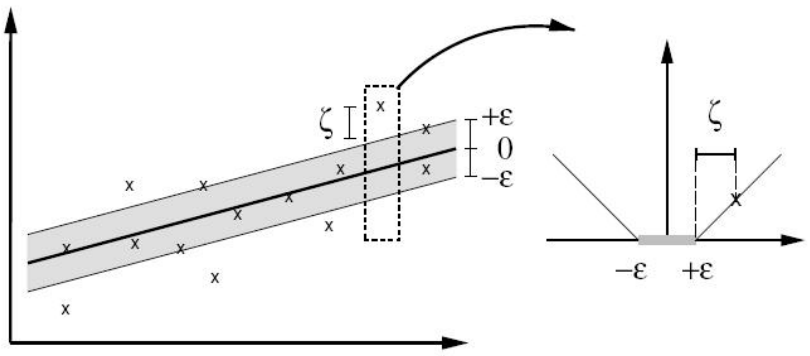


Abbildung 5.2.: ϵ-Margin und Slack Variablen

Die Konstante C steuert den Trade-Off zwischen der oben angesprochenen Flachheit der Funktion und den Punkten, die außerhalb der ϵ - Margin gestattet werden. Sie kann als eine Art Fehlergewicht verstanden werden. Aufgrund dieser Gleichungen wird im Folgenden die SVM zur Regression genutzt und die Parameter eingestellt.

5.2. Lokale Muster erlernen

Im nächsten Schritt soll aufgezeigt werden, welche Vorbedingungen zugrunde liegen müssen, um die SVM anzuwenden. Zuerst muß die Menge der Instanzen bezüglich ihrer Musterzugehörigkeit verwaltet werden. Sämtliche Instanzen, die einem Muster zugeordnet werden (Vergleiche 2.3.2), sind tabellarisch erfaßt. Das Erstellen der Tabelle geschieht über die Mergeoperation in RM. Hierfür ist es notwendig, in einem Preprocessing-Schritt die Indexdimension von Null/Eins beginnen zu lassen und dann inkrementell entsprechend der Abtastabstände der Originalwerte zu erhöhen. Dies entspricht lediglich der Subtraktion einer Konstanten bezüglich der Indexdimension der zu betrachtenden Instanz. Aus diesem Grund wird im folgenden Text dieser Vorgang mit Indexsubtraktion gekennzeichnet. Dies muß für jede Instanz, die Teil der Mergeoperation ist, geschehen, da sonst mathematisch keine Verschmelzung der Instanzen, sondern eine Verkettung erfolgt. Bei der Instanzverschmelzung bedeutet Regression das Ermitteln eines Prototyps, der stellvertretend für die Instanzen steht. Die Verkettung führt nur zur Regression der einzelnen Instanzen. Bei nur einem auftretenden Muster würde ohne Vorverarbeitung die gesamte Zeitreihe der Regression durch die SVM unterworfen werden (Ein Beispiel für das Resultat wurde bereits in Abbildung 3.2 gegeben). Das Resultat wäre ein globales Modell, gesucht wird jedoch ein lokales Modell.

5.2.1. Ergebnis der Regressionsanalyse - Der Prototyp

Betrachten wir nun genauer, was die SVM in der hier verwendeten Form zu leisten vermag. Zur Veranschaulichung dient der biometrische Datensatz, auf den bereits mehrere globale und lokale Modelle angewendet wurden (beispielsweise 3.1). Die Segmentierung liefert Instanzen, die in RM nach Ausführen der Indexsubtraktion der Mergeoperation

unterzogen werden. Anschaulich darf das Ergebnis dieser Operation als *Musterwolke* betrachtet werden, in der der Prototyp verborgen ist. Die *Musterwolke*, die aus den ersten 4 Instanzen der Zeitreihe gewonnen werden kann, ist in Abbildung 5.3 dargestellt. Der Begriff Musterwolke bezeichnet die gemergten Instanzen bezüglich eines korrespondierenden Musters, da der Begriff Muster über die Menge aller Instanzen ohne den Mergevorgang definiert ist.

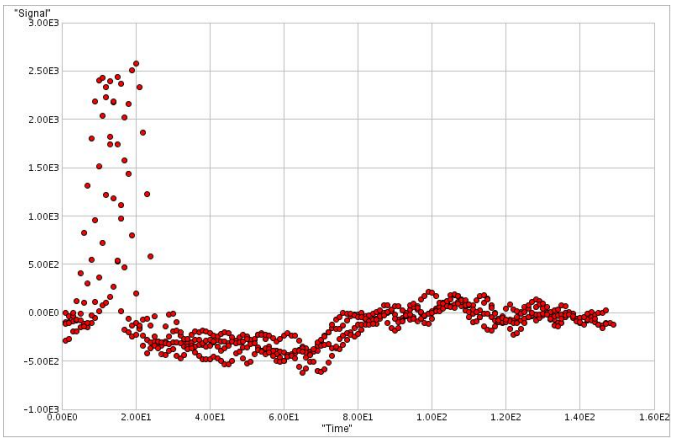


Abbildung 5.3.: Musterwolke der biometrischen Daten

Nun stellt sich die Frage, wie aus diesen Musterwolken der Prototyp für das Muster (also der *beste Vertreter* der Instanzen) gewonnen werden kann? Eben dies ist Aufgabe der SVM, die wir (wie bereits vorgestellt) zur Regressionsanalyse verwenden [Smola 1996]. Die Attribute der Instanzen bilden den Eingaberaum der SVM, auf der diese arbeitet. Aufgrund der Regressionsaufgabe wurde hier die epsilon-SVR Implementierung des LibSVMLearners ausgewählt. Die erste Entscheidung betrifft den Kernel, der der SVM zugrunde liegt. In der hier verwendeten SVM wird der Radial Basis Function Kernel (RBF) - Definition in 5.1.1 - verwendet, der in der Literatur oftmals auch unter der Bezeichnung Gauß Kernel zu finden ist [Schölkopf und Smola 2002]. Der RBF Kernel wurde bevorzugt, da er weniger Parameter als der Polynomial und Sigmoid Kernel aufweist, was in der eigentlichen Parameterbestimmung bedeutet, daß weniger Werte zu spezifizieren sind. Bei hoher Gradwahl kann es außerdem zu Berechnungsproblemen bezüglich des Polynomial Kernels kommen. Zusätzlich ist der Sigmoid Kernel unter einigen Parametern nicht definiert und wird somit als Risikokandidat ausgeschlossen (nur bestimmte κ und δ erfüllen das Mercer Theorem, Vergleiche 5.1.1). Der Linear Kernel muß nicht gesondert betrachtet werden, da er nur einen Spezialfall des RBF Kernels unter Restriktionen (Penalty Parameter) darstellt.

Die zweite Aufgabe betrifft den Bereich der Parametereinstellung. Wie in den vorangegangenen Abschnitten erläutert wurde, hängt das Ergebnis der Schätzung von den einzustellenden Parametern in den Gleichungen der SVM ab. Nur eine SVM mit *guten* Parametern ermöglicht einen Prototyp, der Instanzen ersetzt - eine SVM ohne Parameteranpassung ist ungeeignet. Zusätzlich sollte jedoch immer das eigentliche Ziel (die

Prognose) im Auge behalten werden. Eine hohe Genauigkeit bezüglich der Trainingsdaten muß jedoch nicht zu guten Prognoseergebnissen führen (Overfitting). Eine gute Möglichkeit, Overfitting zu verhindern, ist die Kreuzvalidierung. Grundprinzip ist, die Menge der Instanzen in zwei Teile zu teilen, von denen der eine zum Lernen (Trainingsdaten), der andere zum Testen (Testdaten) der Generalisierungsfähigkeit bestimmt ist. Mittels Kreuzvalidierung werden nun verschiedene Parametersätze getestet.

Definition 5.2.1 (Kreuzvalidierung)**.** *Aufteilung der Beispielmenge in n Mengen. Für i=1 bis i=n wähle die i-te Menge als Testmenge, die übrigen Mengen bilden die Lernmenge. Ermittle Korrektheit und Vollständigkeit bezüglich der i-ten Menge. Berechne das Mittel der Werte über alle n Durchläufe, um so ein Maß für das Lernergebnis zu bilden.*

Derjenige Parametersatz, der nach vorgegebenen Kriterien (Kriterien im Anhang A) am besten abschneidet, wird als Parametersatz für die SVM übernommen. Das Auswählen der Parameter wurde hier mittels Grid Search durchgeführt. Nehmen wir an, es gibt drei zu optimierende Parameter a,b und c, dann berechnet Grid Search alle kombinatorischen Möglichkeiten aus den zuvor festgelegten Werten. Im Bezug auf die Arbeit [Chih-Wei Hsu und Lin 2007] im Bereich der SVM-Klassifikation wurde zuerst eine exponentiell wachsende Zahlenreihe für die Parameter gewählt (Coarse Grid Search). Das Ergebnis wurde als Ansatzpunkt für weitere zu testende Parameter gewählt (Fine Grid Search). Hierfür wurde das Intervall mit den Grenzen *Vorgänger* und *Nachfolger* des Suchpunktes gebildet und dann äquidistant unterteilt. Dieses Verfahren wurde rekursiv durchgeführt (7.1.2).

Bei der epsilon-SVR mit RBF Kernel sind die zu bestimmenden Parameter epsilon, p, C und gamma. Die Herkunft der Parameter läßt sich mit dem Vorwissen aus Abschnitt 5.1.2 nun wie folgt erklären: Der Parameter epsilon gibt die Toleranz der Abbruchbedingung an. Die Veränderliche p ist das ϵ aus der ϵ-insensitive Loss Funktion. Der Parameter C ist bereits als Fehlergewicht aus den SVR Gleichungen bekannt. Die SVM wird der Parameteroptimierung unterzogen, bevor der Prototyp mit optimierten Parametern mittels Funktionsregression über die Musterwolke erlernt wird. Der Parameter gamma entspricht der Veränderlichen aus der Kernfunktion in 5.3.

Manuelles Postprocessing

Eine besondere Bedeutung im Rahmen dieser Arbeit kommt dem Parameter C zu. In praktischer Hinsicht sind seine Auswirkungen derart, daß bei großem C Punkte, die eine große Distanz zur ϵ-Margin aufweisen, stärker in die Funktionsschätzung einbezogen werden [Smola und Schölkopf 1998]. Bei kleinem C wird die Funktion flacher, da diese Punkte an Einfluß verlieren. Aufgrund der variierenden Musterstruktur und den daraus resultierenden Musterwolken erfolgte teilweise eine manuelle Anpassung des Parameters C. Dies wird gesondert erwähnt, da hier von der maschinell optimierten Lösung abgewichen wurde. Beispiele hierzu bilden die Fälle, in denen aufgrund der hinterlegten Struktur der zugrunde liegende Verlauf als zu flach erachtet wurde. Das bekannte Problem des Over- und Underfittings spielt auch bei benutzergesteuerten Veränderungen des Parameters C eine Rolle. Wird C zu hoch gewählt, so können feine Ausreißer in einzelnen Instanzen zu Strukturveränderungen führen - wird C zu niedrig gewählt, werden

signifikante Strukturteile, die nur durch einzelne Attribute gegeben sind, unzureichend approximiert. Dieses Problem trat auf, wenn ein Muster nur durch wenige Instanzen in der Zeitreihe vertreten war. Wie bereits bei der SVM erläutert, benötigt die SVM jedoch eine gewisse Anzahl an Instanzen, um gute Ergebnisse aufzuweisen. Um jedoch nicht auf einzelne Muster gänzlich zu verzichten und somit die Prognose wissentlich zu verfälschen, wurde in einem solchen Fall der Parameter C angepaßt. In Abbildung 5.4 sind die Auswirkungen von C bei konstantem Parametervektor anhand einer Musterwolke des biometrischen Datensatzes aufgezeigt.

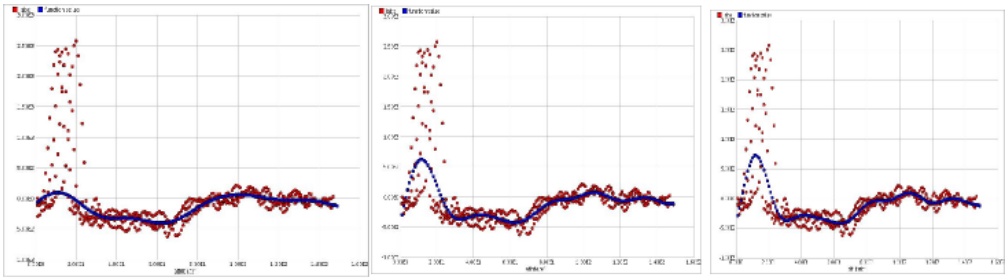


Abbildung 5.4.: Auswirkung des Parameters C auf die Prototypgestalt

5.3. Musterabfolgen und deren Übergänge

Bis zum jetzigen Stand haben wir Instanzen Mustern zugeordnet und dann über die Musterwolke eine Regression durchgeführt, um den Prototyp zu erhalten. Doch wie verhält es sich mit der Beziehung der Prototypen untereinander ? Stellen wir die Frage nach den lokalen Zusammenhängen: Besteht eine Zeitreihe aus mehr als einem Muster, so benötigen wir Auswahlregeln, die bestimmen, welcher Prototyp zwecks Prognose eingesetzt werden soll. Bei einem langfristigen Prognosehorizont kann die Vorhersage sogar mehrere Prototypen umfassen, da der Prognosezeitraum dieses erfordert. Somit erweitert sich die ursprüngliche Aufgabenstellung hin zur Modellierung einer Abfolge von Prototypen, hier: die Sequenzmodellierung.

5.3.1. Die Zeitreihe als Objektsequenz

Eine Zeitreihe wurde als Abfolge von Mustern betrachtet, wobei ein Muster als Objekt der Zeitreihe definiert wurde (Vergleiche 2.3.2). Die Abfolge von einzelnen Mustern ist ein lokale Sequenz, die Musterabfolge der gesamten Zeitreihe bezeichnen wir als globale Sequenz. Die Zeitreihe kann in der Phase der manuellen Instanzauswahl zeitgleich in eine Objektsequenz überführt werden, ohne zusätzlichen Aufwand zu produzieren.

Beispiel 5.1 (Objektsequenz)**.** *Die dargestellte Zeitreihe 5.5 repräsentiert ein menschliches Biosignal mit äquidistanten Abtastwerten aus einem multivariaten EEG Datensatz . Es lassen sich vier unterschiedliche Muster, bezeichnet mit A B C und D, identifizieren, die als Objektsequenz verwaltet werden können.*

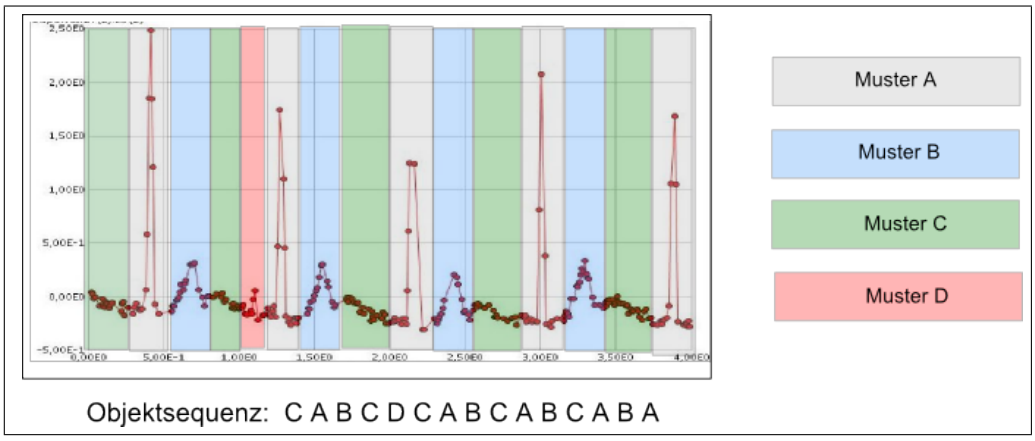


Abbildung 5.5.: Objektsequenz

5.3.2. Überlegungen und Abgrenzung

Das Ziel lautet nun, aus dem Beispiel der Objektsequenz lokale Zusammenhänge zu erlernen. Mittels dieser lokalen Zusammenhänge kann dann für den Prognosezeitraum eine Objektsequenz gefunden werden, die sich an die bereits bekannte Objektsequenz der Zeitreihe anschließt. Wir sind an Aussagen interessiert, die uns ermöglichen, aus der letzten aufgezeichneten Objektmenge, engl.Itemset (z.B. *BAB*) einen Rückschluß auf das Folgeobjekt (z.B. *C*) bzw. die Folgesequenz (z.B. *CBC*) zu erhalten. Aus einer gegebenen Objektsequenz können prinzipiell alle Regeln abgeleitet werden, die sich aus einer vollständigen Kombination einzelner Objekte oder Objektmengen ergeben. Der Musterbegriff, der schon oftmals Verwendung fand, begegnet uns hier auf einem anderen Abstraktionslevel. Die lokalen Zusammenhänge, die der Wiederholung unterliegen, können nämlich ebenfalls als Muster betrachtet werden. Und wie später noch anhand der kurzfristigen Prognose verdeutlicht wird, dürfen wir über das Wissen *Muster* erhoffen, unvollständige Bereiche über sie zu komplettieren. Unvollständigkeit kennzeichnet sich hier durch den Prognosezeitraum (i.A. die Folgesequenz), die wir prognostizieren möchten.

Formeller kann nun die Aufgabenstellung als das Finden von Assoziationsregeln definiert werden [Morik 2000]:

Definition 5.3.1 (Finden von Assoziationsregeln)**.** *Gegeben sei*

I eine Menge von Objekten

t eine Transaktion, $t \subseteq I$

T eine Menge von Transaktionen

$s_{min} \in [0, 1]$ *die benutzergegebene Minimalhäufigkeit*

$c_{min} \in [0, 1]$ *die benutzergegebene Minimalkonfidenz*

Gesucht werden alle Regeln der Form:

$X \rightarrow Y$ *wobei* $X \subseteq I$, $Y \subseteq I$, $X \cup Y = \{\}$ *und es gilt:*

$s(r) := \frac{|\{t \in T | X \cup Y \in T\}|}{|T|} \geq s_{min}$

$c(r) := \frac{|\{t \in T | X \cup Y \in T\}|}{|\{t \in T, X \in T\}|} \geq c_{min}$

Im Zusammenhang mit den Assoziationsregeln ist der Begriff der häufigen Mengen aufzufinden. Unter der Häufigkeit verstehen wir das prozentuale Vorkommen dieser Menge bezüglich der Gesamtmenge an Elementen (Support bzw. Threshold). Häufige Mengen (frequent itemsets (FI)) sind diejenigen, welche über dem benutzerdefinierten Support liegen und somit als häufig vorkommend gelten. FIs werden erzeugt, indem man häufige Teilmengen zu einer Menge hinzufügt und diese Mengen dann auf Häufigkeit testet. Mengen mit einer seltenen Teilmenge werden nicht weiter betrachtet. Dieser Gedanke geht auf die Monotonie der Seltenheit zurück, die besagt, daß bei der Seltenheit einer Teilmenge auch die sie enthaltende Menge selten sein muß. Die Konfidenz c einer Assoziationsregel ist definiert als der Prozentsatz aller Transaktionen, die Y enthalten, in der Teilmenge aller Transaktionen, die X enthalten. Die Faktoren s_{min} und c_{min} sind somit eine Möglichkeit zur Beschneidung der Regelmenge.

Es gibt eine Vielzahl von Verfahren aus dem Bereich Assoziationsanalyse. Ein verbreitetes Verfahren ist Apriori, das mit dem Auffinden von einfachen Mengenzusammenhängen arbeitet, sowie diverse Abwandlungen von Apriori. Im ersten Schritt werden hier häufige Itemsets ermittelt und danach auf ihnen Regeln generiert. Abwandlungen und Weiterentwicklungen von Apriori beziehen sich i.A. darauf, wie im ersten Schritt die häufigen Itemsets gefunden werden [Petersohn 2005]. Apriori ist schlimmstenfalls exponentiell in der Regelmenge, welche zudem höchst redundant ist. Mehrere Ansatzpunkte wie kondensierte Repräsentationen oder andere Beschränkungen zur Beschneidung der Regelmenge gehen in neu entwickelte Verfahren ein. Ein Verfahren, das die sogenannte Kandidatengenerierung von Apriori nutzt, ist *Winepi* [Mannila und Toivonen 1996]. Eine Methode, häufige Mengen zu finden, die nicht auf der Kandidatengenerierung beruht, ist FP Growth. FP Growth ist die Abkürzung für Frequent Pattern Growth und basiert auf den FP Trees. Der FP Tree gibt Präfixbäume für ein Suffix an und basiert auf der Ordnungsrelation Objekthäufigkeit, die absteigend verwaltet wird. FP Growth ist schneller als Apriori. Die Gründe hierfür sind zunächst der Verzicht auf die bereits angesprochene Kandidatenerzeugung und das Testen derer, sowie das Basieren auf einer kompakten Datenstruktur, die die Vermeidung häufiger Datenbankdurchläufe bedingt. Auch die Basisoperation Zählen und der FP Tree-Aufbau sind komfortabel zu handhaben [Morik 2007]. Auf den so gefundenen Mengen werden dann Regeln generiert. Eine implizite Problematik der hier behandelten Aufgabenstellung soll anhand FP Growth exemplarisch verdeutlicht werden, bevor aufgezeigt wird, daß auch andere Ansätze dieses nicht zu lösen vermögen.

FP Growth basiert auf sogenannten Transaktionsdatenbanken. Vorliegend ist die Objektsequenz (eine globale Transaktion), die die Zeitreihe als Summe beschreibt. Um eine Menge von Transaktionen zu erhalten, muß die Zeitreihe bezüglich der Dimension Zeit gespalten werden. Dieser Vorgang erzeugt künstliche lokale Transaktionen, die uns ermöglichen, nach Wiederholungen und Strukturen in der Objektsequenz als Summe zu suchen. Das Aufspalten können wir durch das sogenannte Windowing (Fensterung) er-

zeugen. Es wird ein Fenster mit Schrittgröße eins (bezogen auf die Muster) über die Zeitreihe geschoben. Das Fenster kann variierende Größen annehmen. Die Größe des Kontextfensters ist ausschlaggebend für die Regelfindung, d.h. über welche Indexdimension Zusammenhänge zwischen den Mustern gesucht werden. Im einfachen Fall schieben wir ein Fenster der Größe zwei über die Objektsequenz und erhalten so eine Menge von zweielementigen Transaktionen. Für die Prognose wäre die Auswirkung derart, daß aus dem letzten vollständigen Muster die Assoziationsregeln eine präferierte Konklusion liefern würden - genau über das Folgemuster. Doch ein zu klein gewähltes Fenster kann Wissen verschenken. Diesen Vorgang verdeutlichen wir an einem Beispiel:

Beispiel 5.2 (Fenstergröße)**.** *Betrachten wir die Objektsequenz ABCDABCDEBDFEBDFABCDA und stellen uns vor, die Zeitreihe repräsentiert die Temperaturerhebungen eines Jahres. Das Muster E stellt einen außergewöhnlich kalten Frühling dar, auf den ein außergewöhnlich warmer Winter F folgt. Sie sind so außergewöhnlich, daß ihre Struktur zu einem eigenen Muster geführt hat. Der bedingte jahreszeitliche Zusammenhang ist in der Zeitreihe versteckt. Selbst wenn wir B und C nicht vorhersagen könnten ,so könnten wir aus dem Zeitreihenende E prognostisches Wissen erlangen, in dem wir folgern, daß der Winter (also das dritte Muster) der Prototyp F (weil er bei geeignetem Support eine Konfidenz von 1 aufweist) sein muß. Auch multivariate Lernaufgaben (z.B. aus A,B folgt E - hypothetisch, nicht bezogen auf die Sequenz) sind nun zu lösen. Betrachten wir dann die letzten beiden Muster der Zeitreihe, so würden wir E prognostizieren und nicht, gegebenenfalls nach purer Wahrscheinlichkeit von zweielementigen Transaktionen, zwischen D und E würfeln.*

Daß die Fenstergröße Auswirkungen auf die Regelmenge hat, ist leicht erklärbar, denn aus ihr resultiert die Menge der Transaktionen. Diese Problematik wurde bereits im Rahmen der Methodikfindung angesprochen und in den Arbeiten von [Das et al. 1997] aufgezeigt. Auch auf diesen Vorgang trifft die Problematik zu. In Abschnitt 5.3.3 wird gezeigt, daß die Grundproblematik des Fenstergrößen Trade-Offs in die meisten Verfahren eingeht, da die Fensterung Teil der Vorverarbeitung ist. Neben der Fenstergröße gilt es noch, die anderen beiden Benutzerparameter $\mathsf{s}_{\mathsf{min}}$ und $\mathsf{c}_{\mathsf{min}}$ zu wählen. Alle drei Parameter korrespondieren miteinander. Die Fenstergröße beeinflußt die zu findenden Regeln zusammen mit dem Kriterium $\mathsf{s}_{\mathsf{min}}$. Setzen wir einen niedrigen Wert für $\mathsf{s}_{\mathsf{min}}$ an, ist $\mathsf{c}_{\mathsf{min}}$ nicht aussagekräftig. Was sagt eine Konfidenz von 1 aus, wenn diese Objektsequenz nur einmal vorliegt? Für eine weitere Auswahl an Beispielen, die den Zusammenhang von Support und Konfidenz anschaulich erläutern, sei auf [Morik 2007] verwiesen.

Für die Zeitreihenprognose in der hier verwendeten Art ist das Verfahren ungeeignet. Wir sind an allen Regeln (d.h. Transaktionen) interessiert. Eine Menge der lokalen Zusammenhänge führt jedoch nicht unbedingt zu globalen Aussagen. Die oben beschriebenen Vorgänge können auch als Funktionslernen aufgefaßt werden [Morik 2000]. Unsere Aufgabe bestünde nun darin, alle partiellen Funktionen für den gesamten Instanzraum zu erlernen. Bei Experimenten mit niedrigen Schwellenwerten ($\mathsf{s}_{\mathsf{min}} = 0{,}1$ und $\mathsf{c}_{\mathsf{min}} = 0{,}1$) war die Regelmenge sehr groß und enthielt keine geeignete Regel. Abbildung 5.6 zeigt die Regelmenge für das zuvor eingeführte Jahreszeitenbeispiel. Aufgrund dieser Schwierigkeit mußten Assoziationsregeln als global gültiger Lösungsweg verworfen werden.

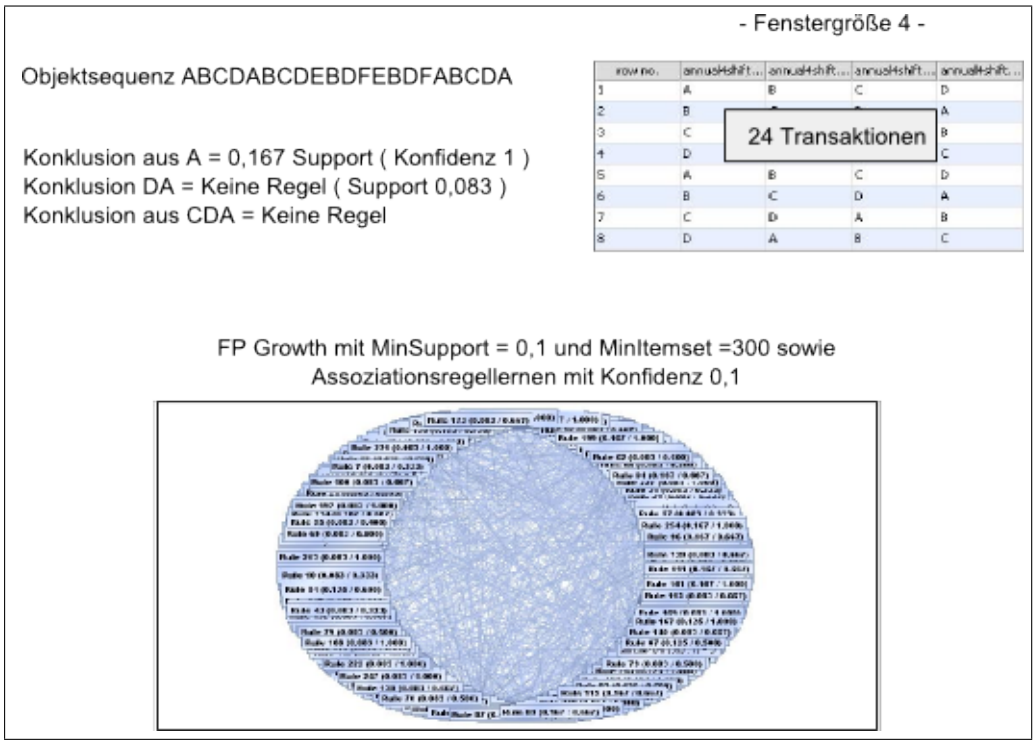


Abbildung 5.6.: Assoziationsregellernen auf Objektsequenzen

Kehren wir zum Begriff Funktionslernen zurück und prüfen, welche sonstigen Möglichkeiten zur Realisierung gegeben sind. Ein Lösungsansatz für diese Aufgabe wurde in Abschnitt 3.1.2 eingeführt. Eine typische Aufgabe für ANNs ist die Approximation einer unbekannten und analytisch schwer beschreibbaren Funktion. Das ANN hat jetzt nicht mehr (wie in der ursprünglichen Einführung) die komplexe Aufgabe der gesamten Prognose zu bewerkstelligen, sondern soll nur noch die Teilaufgabe der Sequenzvorhersage erfüllen. Die zuvor aufgestellte Grundüberlegung bleibt weiterhin bestehen. Die Objektsequenz muß in eine Menge von Beispielen überführt werden. Dieser Schritt wird mittels der Fensterung durchgeführt. Das gesuchte Folgeobjekt konnten wir aufgrund der Parameterrestriktionen nicht als Konklusion finden. Nun soll das Folgeobjekt mittels ANNs erlernt werden. Die Eingabe für das ANN ist nach Fensterung eine Menge von Beispielen mit einem ausgewiesenen Ziel (Folgeobjekt). Auf dieser Eingabe erfolgt das Training. Die letzten n Attribute - wobei n der Fenstergröße entspricht - bilden nun das zu klassifizierende Beispiel für das trainierte Netz. An der Ausgabeschicht kann daraufhin das Ergebnis der Klassifikation abgelesen werden. Es ist nicht notwendig, daß die zu klassifizierenden Daten identisch mit den Trainingsdaten des ANN sind. Vielmehr genügt eine Ähnlichkeit der aktuellen Daten mit den bereits vorhanden Trainingsdaten. Dies ist ein notwendiges Kriterium, denn es gilt, ggf. auch Sequenzen zu klassifizieren, die es in der Objektreihenfolge der Trainingsdaten nicht gibt.

Zur Realisierung wurde ein Multilayer Perceptron Netz (MLP) unter RM verwendet. Im ersten Schritt wurde ein gleitendes Fenster mit n = 3 über die Daten geschoben. Das

Folgeobjekt der so gewonnenen Sequenzen wurde als Ziel definiert, anschließend wurde das Netz trainiert. Die letzten drei Objekte der Zeitreihe stellen die zu klassifizierende Sequenz dar, und das Netz liefert das Folgeobjekt (Abbildung 5.7).

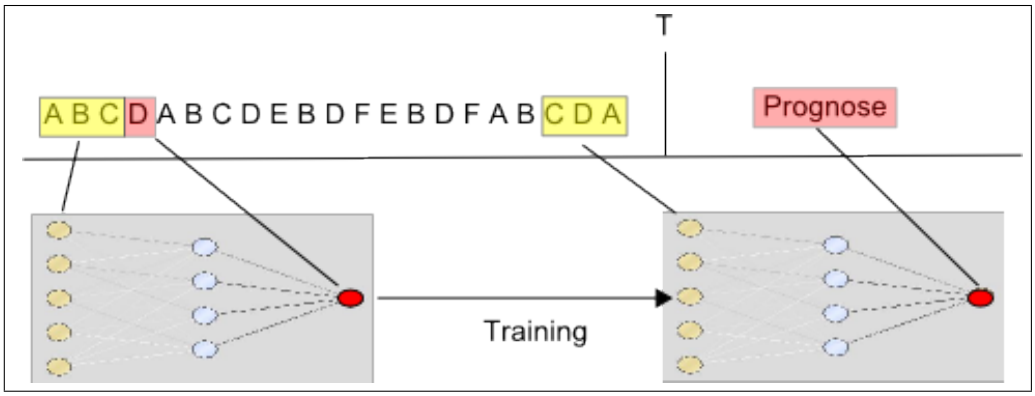


Abbildung 5.7.: MLP auf Objektsequenz

Der eigentliche Aufwand besteht aber in der geeigneten Codierung der Eingabeschicht, da ein MLP per Definition keine nominalen Attribute verwerten kann. Die Muster sind jedoch nominal, d.h. wir haben unterschiedliche Typen, können jedoch keine Rangfolge in einer Skala definieren. Auch wenn die einzelnen Muster durch Zahlen kodiert werden können, sind mathematische Operationen mit diesen Zahlen nicht sinnvoll, da sie eben kein numerischer Wert sind. Vielmehr lassen sich solche Attribute nur auf Gleichheit oder Ungleichheit testen. Denkbar wäre der Ansatz, jeder Stelle im Kontextfenster eine Menge aus Eingabeneuronen zuzuweisen, die der Anzahl der möglichen Klassen entspricht. Eine binäre Codierung könnte somit die Eingabe verwalten und das prognostizierte Muster entsprechend in der Ausgabeschicht codieren. Doch dieser Weg ist zeitintensiv, und die Codierung ist abhängig von der Anzahl der gegebenen Musterklassen. Somit ist es theoretisch möglich, das bestehende Problem mit Neuronalen Netzen zu lösen, jedoch soll im Folgenden eine einfache und effiziente Methode vorgestellt werden.

5.3.3. Modellieren der Objektsequenz - Eine Klassifikationsaufgabe

Basierend auf den vorangegangenen Abgrenzungen und Überlegungen greifen wir die Fensterung der Objektsequenz auf und betrachten einen weiteren Ansatz zur Problemlösung. Die Regelfindung garantiert keine Vollständigkeit, und das Funktionslernen mit anderen Methoden erfordert oftmals eine komplexe und problemspezifische Codierung. Eine andere Option ist, die Folgepositionen des Kontextfensters als Klassensuche zu betrachten (2.3.4). Die Anzahl der möglichen Klassen ist durch die bekannten Muster vorgegeben. Es kann nur eine bereits vorhandene Klasse prognostiziert werden. Die Muster im Kontextfenster sind nominale Ausprägungen der Attribute, die auf das Folgemuster (die gesuchte Klasse) schließen lassen. Falls die Folgeposition in der Objektsequenz unbekannt ist, müssen wir die letzte bekannte Objektsequenz klassifizieren, um so die Klasse zu prognostizieren. Der hier präferierte Ansatz zur Klassifikation ist der Nearest Neighbour-Ansatz, der im folgenden Kapitel aufgegriffen wird (6.1.2). An dieser Stelle wird zum

Beispiele	Position 1	Position 2	Position 3	Label
Beispiel 1	A	B	C	A
Beispiel 2	A	D	C	D
Beispiel 3	B	B	A	B
Beispiel 4	A	B	D	B
Beispiel 5	C	B	C	C

Tabelle 5.1.: Kodierung der Problematik für Nearest Neighbour-Klassifizierung

einfachen Verständnis auf explizite Erklärungen verzichtet, da die detaillierten mathematischen Beschreibungen an zuvor referenzierter Stelle im geeigneten Kontext näher untersucht werden.

Um die Klassifikationsaufgabe zu lösen, codieren wir das Problem folgendermaßen: Die Einträge der Spalten repräsentieren die Positionen der Elemente im Kontextfenster. Jede Zeile entspricht einem Beispiel, d.h. die Menge der Beispiele ist Resultat der Fensterung. Die Klasse entspricht dem Folgeelement des Fensters. Eine neue Eingabe soll dann mittels der Beispiele korrekt klassifiziert werden, also die Klasse des Folgeelements liefern. Es wird die Klasse des Beispiels, das der Klassifikationsanfrage am ähnlichsten ist, ausgewählt. Da wir auf nominalen Werten arbeiten, sollen die einzelnen Attributwerte nur auf Gleichheit und Ungleichheit getestet werden (Vergleiche 5.3.2). Das ähnlichste Beispiel ist jenes, das die größte Anzahl an gleichen Attributausprägungen aufweist. Tabelle 5.1 verdeutlicht den Aufbau, wobei die Menge $\{\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}, \mathsf{E}\}$ wie bisher Muster repräsentiert. Bei einem Prognosehorizont, dessen Indexanzahl größer als ein Muster ist, gibt es zwei mögliche Vorgehensweisen. Nachstehende Überlegungen treffen auch für die zuvor verworfenen ANNs sowie sämtliche Klassifizierungsverfahren mit univariatem Label zu. Zum einen können unterschiedliche Klassen (Label) mittels variierender Schrittweite in der Fensterung erstellt und auf den so gewonnenen Eingaben eigenständige Modelle trainiert werden. Zum anderen kann das trainierte Modell die Prognose schrittweise aufbauen. Dies wird rekursiv ausgeführt, bis der Prognosezeitraum abgedeckt wurde. Hierfür wird die Eingabe nach jeder Klassifikation um eine Stelle nach rechts verschoben und um das Ergebnis der Klassifikationsaufgabe ergänzt. Das erscheint zunächst einfacher, da nur ein Modell erlernt werden muß. In Experimenten hat sich jedoch gezeigt, daß bei ungeeigneter Fensterwahl ein Zustand erreicht wird, der nicht mehr zu verlassen ist. Dieser Vorgang läßt sich mittels der Automatentheorie ([Wegener 1999]) verdeutlichen: Die Menge der möglichen Label entspricht der Menge der Zustände und die Eingaben stellen die Zustandsübergänge dar. Durch die Aufnahme des Labels in die Eingabe entsteht ein Zyklus, und mögliche andere Label werden unerreichbar.

Beispiel 5.3 (Zyklusproblem)**.** *Die Abbildung 5.8 zeigt einen Ausschnitt des Zustandsautomaten, der auf dem Dodgers-Datensatz (Vergleiche Abschnitt B) erstellt wurde. Die beiden Zustände repräsentieren die beiden Klassen Dodgers-Spiel (D) und kein Spiel (N). Der Wert n ist die Fenstergröße - hier n = 6. Sobald die Klasse N vorhergesagt wird, wird N im nächsten Schritt an die Eingabe gehängt (z.B. DNNNNN → NNNNNN). Dieses Eingabemuster kann nicht mehr verlassen werden, da es von N keinen Pfad zu D gibt, d.h. NN kann D nicht als Label vorhersagen. Der Verein Dodgers würde wegen der Zyklusproblematik in der modellierten Zeitreihe nie mehr als spielend vorhergesagt werden - eine sicherlich fehlerhafte Aussage.*

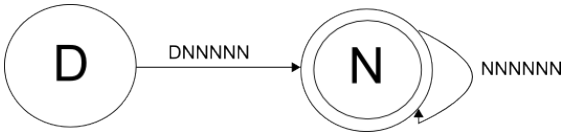


Abbildung 5.8.: Zyklus im Zustandsautomat bei rekursiver Eingabe

Ein Anhaltspunkt dafür ist, daß trotz variierender Eingaben und Fenstergrößen der Accuracy-Wert (prozentuale Übereinstimmung der Vorhersage mit der eigentlichen Sequenz) bis auf marginale Abweichungen konstant blieb, da durch das Erreichen eines solchen Zustandes immer die selbe Sequenz produziert wird. Aus diesem Grund wurde jede Musterposition im Prognosezeitraum separat gelabelt und die Positionsvorhersagen mittels eigener NN Modelle in jedem Schritt durchgeführt. Im Anhang findet sich unter B.6 eine Versuchsdokumentation zur Zyklusproblematik.

Wir haben nun eine praxistaugliche Methode, um Objektsequenzen zu prognostizieren. Sie ist in ihrer Struktur unabhängig von der Anzahl der Muster. Des Weiteren ist sie über eine geeignete Metrik in der Lage, auch ohne Vorverarbeitung nominale Attribute zu vergleichen und zu verwerten. Dem im Abschnitt 5.3.2 verdeutlichten Trade-Off der Fenstergröße kommt hier eine gesonderte Bedeutung zu. Wurde beim Assoziationsregellernen im ungünstigsten Fall keine passende Regel gefunden, da die Fenstergröße nicht geeignet war, so kann hier bei zu großer bzw. zu kleiner Fensterwahl ein verfälschtes Ergebnis entstehen. Die Tatsache findet darin Begründung, daß evtl. irrelevante Attribute Gleichheit aufweisen und somit eine falsche Klasse bestimmt wird. Daher wird hier auf verbindliche Aussagen zur Fenstergröße verzichtet, da sie wie bereits angesprochen zwar die Ergebnisse beinflussen, jedoch zu stark von der zugrunde liegenden Zeitreihe abhängen. Als Anhaltspunkt wurden die Daten im Verhältnis 9:1 in Trainingsdaten und Validierungsdaten unterteilt. Danach wurden mögliche Fenstergrößen getestet und die Fenstergröße mit dem besten Accuracy-Wert auf den Validierungsdaten für die eigentliche Prognose genutzt (Beispiel in Tabelle B.6). Unter dem Abschnitt Experimente 7.1.3 kann der gesamte Versuchsaufbau schrittweise nachvollzogen werden.

6. Kurzfristige Prognosen

Im vorherigen Kapitel wurde eine Technik vorgestellt, mit der man langfristige Prognosen erstellen kann. Im Folgenden erfolgt nun die Betrachtung von Methoden, die zum Einsatz bei kurzfristigen Prognosezeiträumen in Erwägung gezogen wurden.

6.1. Einführung und Vorüberlegungen

Um die scheinbar divergente Sichtweise in den folgenden Methoden nachzuvollziehen, soll zuerst der Globalitätsbegriff nähere Betrachtung finden. Wir betrachten eine Menge von zusammenhängenden Punkten - eine Lokalität - und wollen nun den Begriff Globalität in Relation zu ihnen setzen. Hierfür gibt es zwei verschiedene Sichtweisen. Zum einen jene, die bei der Einführung von Modellen auf Zeitreihen zum Tragen kam (Vergleiche 2.2.2). In diesem Modell betrachten wir Globalität als die Menge aller Punkte und somit die Menge aller Instanzen. Verständlich, da die Mustersuche gerade erst begonnen hatte und das einzige Maß für die Globalität die gesamte Zeitreihe war. Zum anderen wurde aber bereits eine andere Art der Globalität bei der SVM-Methodik deutlich. Es wurde dargestellt, daß jede Instanz Objekt einer Klasse - dem Muster - war. Hier repräsentiert die Instanz wie gehabt die Lokalität, jedoch der Globalitätsbegriff bezieht sich auf die Menge der gleichartigen Instanzen, die durch den Prototyp repräsentiert werden. Der Prototyp als lokales Modell (Vergleiche 3.3) ist aus Sicht der korrespondierenden Instanzen eine Globalisierung eben dieser und bildet die Globalität. Andererseits wird die Menge aller Muster aus der Menge aller Instanzen gebildet, so daß sich hier eine konstante Definition, jedoch auf verschiedenen Granularitätsebenen, erkennen läßt. Im Folgenden sollen Methoden betrachtet werden, die auf diesen Begrifflichkeiten aufbauen und zur kurzfristigen Prognose verwendet werden. Dieses Kapitel zeigt verschiedene Lösungsansätze auf und erklärt, warum manche im Rahmen der Aufgabenstellung verworfen werden mußten. Es zeigt ebenso den zeitlichen Weg der Ideenfindung, der zur Lösung geführt hat. Dieser eignet sich gut, dem Leser die impliziten Problematiken zu verdeutlichen.

6.1.1. Unvollständige Instanzen

Um die Ideen der Instanzvervollständigung nachzuvollziehen, betrachten wir zuerst das Ende einer Zeitreihe (siehe Abbildung 6.1) mit n Datenpunkten, gegeben durch den letzten Beobachtungszeitpunkt x_n. Sei m_i das letzte vollständig zu beobachtende Muster, dessen letzter Datenpunkt x_i nicht mit x_n übereinstimmt, d.h. $i < n$. Am Ende der Zeitreihe befindet sich somit eine Menge von Datenpunkten $\{x_{i+1}...x_n\}$. Diese Daten wurden bis dato noch nicht genutzt. Grundidee ist nun, die kurzfristige Prognose nicht über den wahrscheinlichsten Prototyp folgend auf den Wert x_i zu modellieren, sondern aufgrund der vorhandenen Attribute die *ähnlichste* Instanz auszuwählen. Die Motivation

für die Suche im Instanzraum ist die, daß wir erhoffen dürfen, über das Zusatzwissen in Form der $\{x_{i+1}...x_n\}$ eine Instanz zu finden, die die Werte x_{n+1} besser prognostiziert als der Prototyp. Sie ist also ein *besserer Vertreter*. Dies ist naheliegend, denn die Datenpunkte nach x_i bilden veranschaulicht die Anfangspunkte einer neuen Instanz, die zur Prognose vervollständigt werden soll. Um geometrische Strukturen zu vervollständigen, erfolgt die Suche nach einer kompletten, möglichst ähnlichen Struktur, um über ihre Gesamtheit die fehlenden Datenpunkte der Teilstruktur zu ergänzen. Bevor der Suchraum strukturiert werden kann und die Punkteanzahl der Teilinstanz Einfluß auf die Suche nimmt, soll der Begriff Ähnlichkeit formalisiert werden. Dies ist zentral, da unabhängig von der verwendeten Sichtweise im Bezug auf die Globalität der erste Schritt immer derjenige ist, die *ähnlichste* Instanz zu suchen. In diesem Fall müssen wir Ähnlichkeit mathematisch definieren, da wir nun nicht mehr manuell eingreifen, so wie es noch bei der Instanzauswahl für Muster der Fall war (Vergleiche Kapitel 2.2). Wir wollen dies am Beispiel des Nearest Neighbour-Ansatzes zeigen, da er als Basisgedanke in alle Verfahren eingeht.

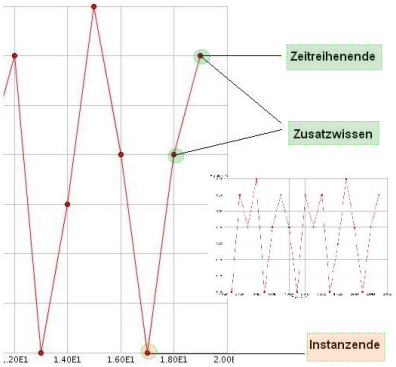


Abbildung 6.1.: Zeitreihenende

6.1.2. Ähnlichkeit von Instanzen

Die Bestimmung des Ähnlichkeitsmaßes ist eine zentrale Frage bei den Nearest Neighbour-Methoden. Das Ähnlichkeitsmaß dient zur Erfüllung der Aufgabe, aus der Menge der Beispiele (die einzeln abgespeicherten Musterinstanzen) die ähnlichsten herauszusuchen. Hierbei kann es sich um das ähnlichste (nearest neighbour) oder die k ähnlichsten (k nearest neighbour) Beispiele handeln. Je nach Art der zugrunde liegenden Daten muß ein geeignetes Ähnlichkeitsmaß gefunden werden (das Ähnlichkeitsmaß ist domänenabhängig). Angenommen, Strings sollten auf Ähnlichkeit geprüft werden - mögliche Maße wären beispielsweise die Anzahl der verschiedenen Buchstaben oder die Länge der Strings. Doch wie verhält es sich mit den Instanzen, die aus Zeitreihen gewonnen werden ? Da wir auf Zeitreihen arbeiten, bestehen unsere Elemente (die Punkte) nur aus numerischen Attributen. Es ist somit naheliegend, die Ähnlichkeit über den Abstand zu definieren. Weisen zwei Elemente eine große Ähnlichkeit auf, kommt dies in einem kleinen Wert des Distanzmaßes, also einem geringen Abstand, zum Ausdruck. Um Abstände zu messen, muß eine Abstandsfunktion $d(x, y)$ für alle Elementepaare x, y erklärt sein, eine soge-

nannte Metrik. Sie ist elementar für den Vergleich von Elementen, denn sie gestattet uns, den Objekten nicht negative reelle Werte (Abstand) zuzuordnen. Eine Metrik muß folgenden Axiomen genügen:

$$\text{Definitheit } d(x,y) = 0 \Rightarrow x = y \tag{6.1}$$

$$\text{Symmetrie } d(x,y) = d(y,x) \tag{6.2}$$

$$\text{Dreiecksungleichung } d(x,y) \leq d(x,z) + d(z,y) \tag{6.3}$$

Ein Raum, für dessen Elementarpaare eine solche Abstandsfunktion erklärt ist, wird als Metrischer Raum bezeichnet. Doch wie genau sieht die analytische Gestalt einer solchen Abstandsfunktion aus ? Da Ähnlichkeit zwischen Instanzen gemessen werden soll, ist es notwendig, zuvor eine Begriffserweiterung einzuführen. Um die Instanzzugehörigkeit formell zu unterscheiden, erweitern wir zunächst den Musterbegriff aus 2.3.3 wie folgt:

Definition 6.1.1 (Erweiterte Indizierung)**.** *Gegeben sei eine Menge von Instanzen* m_i *mit* $x_i \in X_i, i = 1...n$*. Um die Punktzugehörigkeit bei mehreren Instanzen zu kennzeichnen, bezeichnen wir mit* x_i^j *das i-te Element der j-ten Instanz. Analog gibt* w_i^j *den Wert des i-ten Elements der j-ten Instanz an. Bei dem konkreten Bezug auf eine Instanz wird weiterhin die verkürzte Schreibweise genutzt.*

Nun kann die Distanzfunktion für Attribute und Instanzen definiert werden. Durch die arbeitsbezogene Einschränkung auf Zeitreihen sind nachfolgende Definitionen im Raum $\mathbb{R}^2$ angelegt. Im Bezug auf die Arbeit von [Morik 2000] definieren wir die Distanzfunktion über den Euklidischen Abstand (ED). Es gibt eine Vielzahl weiterer Distanzfunktionen, die jedoch aufgrund der Beschränkung auf numerische Attribute und Zeitreihen keiner näheren Betrachtung bedürfen.

Definition 6.1.2 (Distanzfunktion für Attribute)**.** *Gegeben sei eine Instanz* m_i*. Wir definieren die Ähnlichkeitsfunktion für zwei Attribute* $x_1^i \in m_i$ *und* $x_2^i \in m_i$ *wie folgt:*

$$d\left(x_1^i, x_2^i\right) = \left|x_1^i - x_2^i\right|.$$

Definition 6.1.3 (Distanzfunktion für Instanzen)**.** *Die Distanzfunktion für zwei Instanzen* m_1 *und* m_2 *aus* M_i*, mit* x_i*, i=1...m, ist definiert als:*

$$d(m_1, m_2) = |m_1, m_2| = \sqrt{\sum_{i=1}^{n} \left(x_i^1 - x_i^2\right)^2}.$$

Es ist leicht ersichtlich, daß die Distanzfunktion für Attribute identisch zur Distanzfunktion für Instanzen mit Beschränkung auf ein Attribut pro Instanz ist.

6.2. Globalität als Menge aller Instanzen

Die nun folgenden Überlegungen sollen die vorgestellten Begriffe vertiefen und erste Lösungsversuche zeigen. Die Herangehensweise beruht auf dem Globalitätsbegriff, der über die Menge aller Instanzen gebildet wurde.

6.2.1. Ähnlichkeit im Instanzraum

Betrachten wir einen Beispieldatensatz, um so die Suche nach ähnlichen Instanzen zu verdeutlichen. Bei den Daten handelt es sich um eine konstruierte Zeitreihe (Alpha), um so dem Leser das Nachvollziehen der einzelnen Schritte auch auf numerischer Ebene zu erleichtern. Die Zeitreihe Alpha ist in Abbildung 6.2 dargestellt. Beginnend von x_1 bilden

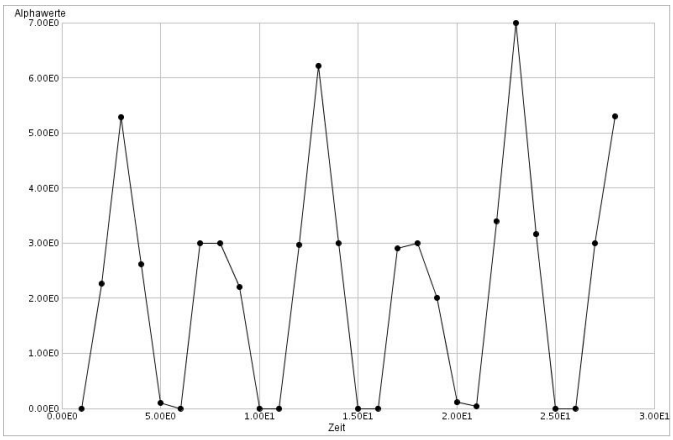


Abbildung 6.2.: Zeitreihe Alpha

jeweils 5 Punkte eine Instanz - die Musterzugehörigkeit ist für das Problemverständnis vorerst irrelevant. Wir erhalten somit fünf vollständige Instanzen, die in Abbildung 6.3 dargestellt sind. Durch die vorherigen Operationen, in denen die Musterinstanzen bereits von Hand ausgewählt und für den Merge Operator der Indexsubtraktion unterzogen wurden, fällt bei diesem Schritt keine weitere Datenbearbeitung an. Die x_i repräsentieren die Attribute, und die Instanzen sind zur Identifizierung numeriert. Ziel ist es nun, für die unvollständige Instanz m_6, gegeben durch die Punkte $(26, 0)$, $(27, 3)$, $(28, 5.3)$, die ähnlichste Instanz zu berechnen. Das Ergebnis ist die Instanz m_1. Bei der Zeitreihe

row no.	Beispiel-N...	Beispiel-N...	Beispiel-N...	Beispiel-N...	Beispiel-N...	Beispiel-N...
1	1	0	2.260	5.290	2.620	0.100
2	2	0	3	3	2.200	0
3	3	0	2.970	6.220	3	0
4	4	0	2.900	3	2.010	0.110
5	5	0.040	3.400	7	3.160	0

Abbildung 6.3.: Datensatz Modell-Lernen Alpha

Alpha bildeten sämtliche Musterinstanzen die Basis für den Suchraum, und der gesamte Raum wurde in der Ähnlichkeitssuche berücksichtigt. Ohne Hinzunahme von weiterem Wissen wurde somit die ähnlichste Instanz über die Menge alle Instanzen gesucht. Doch ist dies auch sinnvoll ? Kehren wir zur Zeitreihe Alpha zurück und stellen uns vor, das Attribut x_{28} sei unbekannt... Die unvollständige Instanz m_6 wird somit nur noch durch zwei Datenpunkte repräsentiert. Nearest Neighbour liefert nun ein anderes Ergebnis -

die Instanz $\mathfrak{m}_2$. Betrachten wir das bereits vorhandene Wissen über die Zeitreihe, so fällt auf, daß das Wissen über die Musterzugehörigkeit der Instanzen und die damit verbundene Musterabfolge noch nicht genutzt wurde. Da dieses Wissen bereits in den Datensätzen gehalten wird, nutzen wir es für weitere Strukturierungen innerhalb der Daten. Jede Instanz ist eindeutig einem Muster zugeordnet. Bis jetzt haben wir auf dem gesamten Instanzraum gearbeitet und dabei die Musterzugehörigkeit vernachlässigt. Im Instanzraum wird die Lokalität durch den Vermerk der Musterzugehörigkeit in einem zusätzlichen Attribut repräsentiert. Lokale Suche ist in dieser Sichtweise somit die Suche in einer Musterklasse, globale Suche die Suche im gesamten Instanzraum.

6.2.2. Verschiedene Umsetzungen der Grundidee

Im Rahmen dieser Arbeit sind verschiedene Ansätze, die auf den vorgestellten Grundprinzipien beruhen, entwickelt worden. Die drei Grundideen sollen hier kurz vorgestellt werden, um danach aufzeigen zu können, warum sie als Lösungsweg auszuschließen sind.

Klassenbasierte Suche

Dieser Ansatz verwendet als Ausgangsbasis eine Tabelle mit allen Instanzen pro auftretendem Muster. Jede Tabelle stellt eine eigene Klasse bezüglich der Suche dar. Zusätzlich werden die Wahrscheinlichkeiten der Markov-Kette verwertet. Als Ausgangsbasis dient die Vorgängerinstanz der unvollständigen Instanz. Mittels der Markov-Kette kann eine Zuordnung von Auftrittswahrscheinlichkeiten zu den Klassen verwaltet werden. Die Klassen werden Top-Down bezüglich der Wahrscheinlichkeiten verwaltet. Die Suche nach der ähnlichsten Instanz startet in der Klasse, die der letzten vollständigen Instanz mit größter Wahrscheinlichkeit folgt. Mit Hilfe eines Benutzerparameters ξ als Toleranzkriterium kann so gesteuert werden, ob die gefundene Instanz hinreichend ist. Hierfür wird getestet, ob der euklidische Abstand (ED) $< \xi$ ist. Bei negativer Antwort erfolgt ein Abstieg in die Folgeklasse, und die NN Suche wird erneut durchgeführt sowie das Toleranzkriterium geprüft. Das Verfahren endet, wenn alle Instanzen durchsucht worden sind (Worst-Case) oder (ED) $< \xi$ eingetroffen ist.

Gewichte und Balancefunktion

Der zweite Ansatz versucht, eine Gewichtung zwischen globaler und lokaler Suche mittels einer Balancefunktion zu gewährleisten. Hierfür wird die Tabelle in 6.2 erweitert, siehe 6.4. Die Attribute sind über a-e verwaltet. l bezeichnet die Instanz (ist also auf algorithmischer Ebene das Label und Ziel der Vorhersage), und w als neues Attribut ist die Gewichtung der Instanz bezüglich NN mit ED Metrik. Der Grundgedanke ist, über die Gewichte die lokale und globale Suche zu steuern. Ein größer Wert bei w soll eine stärkere Instanzgewichtung bezüglich NN gewährleisten. Hierfür wird zuerst die globale Suche initialisiert (w=1), was eine gleichstarke Gewichtung aller Instanzen erzeugt. Danach wird für die wahrscheinlichste Musterklasse (Vergleiche 6.2.2) der Wert für w nach einer Balancefunktion berechnet. Je geringer die Anzahl der Attribute, desto größer wird der Ergebniswert der Balancefunktion λ. Über die Neuberechnung der Gewichte für die wahrscheinlichste Musterklasse nach der Formel $w + \lambda$ soll so sichergestellt werden, daß

bei nur wenigen Attributen die lokale Suche präferiert wird. Im Fall von z.B. nur einem Attribut soll die lokale NN-Suche in der wahrscheinlichsten Musterklasse durch die hohe Gewichtung garantiert werden. Bei einer höheren Attributanzahl kann $\lambda = 0$ werden, und eine globale Suche über alle Attribute kann durch identische Gewichtung erfolgen.

row no.	w	l	a	b	c	d	e
1	2	1	0	2.260	5.290	2.620	0.100
2	1	2	0	3	3	2.200	0
3	1	3	0	2.970	6.220	3	0
4	1	4	0	2.900	3	2.010	0.110
5	0	5	0.040	3.400	7	3.160	0

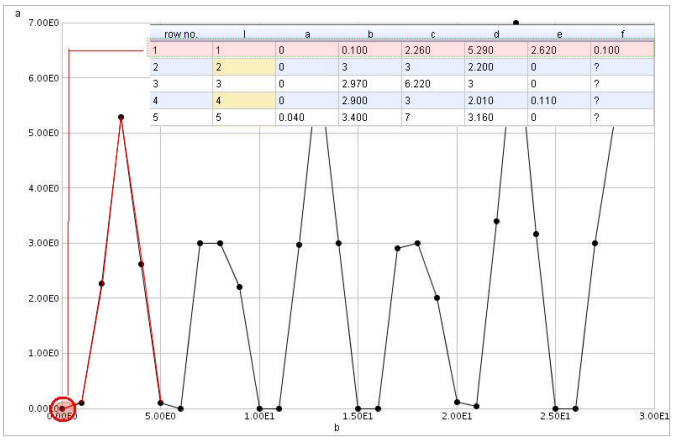
Abbildung 6.4.: Modell-Lernen Alpha mit Gewichten

Die Problematik der Methoden

Betrachten wir nun, warum beide Ansätze in der Praxis nicht direkt zur Instanzprognose genutzt werden können. Die Klassenbasierte Suche sowie die Methode der Balancefunktion basieren auf Tabellen, die den Suchraum des NN-Verfahrens bilden. Da die einzelnen Attribute miteinander verglichen werden, d.h. die Indexwerte in Relation zueinander gesetzt werden, ist leicht ersichtlich, daß dies nur bei den wenigsten Aufgaben möglich ist. Ein einfaches Beispiel sind Zeitreihen, die nicht äquidistante Datenpunkte aufweisen. Hier wird die Vergleichbarkeit der Werte unmöglich, da nicht jedem Punkt der unvollständigen Instanz in jeder zu vergleichenden Instanz ein Punkt (also ein vorhandener Abtastwert) zugeordnet werden kann. Somit kann eine geringfügige Verschiebung (z.B. durch einen zusätzlichen Abtastwert) verfälschte Ergebnisse liefern. Ein Beispiel ist in 6.5 durch den Datensatz Beta gegeben. Hier führt die Tatsache, daß Instanz 1 ein Attribut mehr besitzt, als im Ursprungsdatensatz Alpha vorhanden ist, zu anderen Vergleichswerten und somit zu unzuverlässigen Ergebnissen. Daher können auch abgewandelte Ansätze, z.B. Tabellen fixer Größe zu erzeugen und die Instanzen bezüglich ihrer Schwerpunkte zu mergen, sowie den Randbereich mit Default-Werten zu füllen, kein brauchbares Ergebnis aus den oben genannten Gründen liefern. Zusätzlich ist bei der Klassenbasierten Suche der Klassenabstieg über einen Benutzerparameter geregelt. Dies ist zu vermeiden, da die Wahl des Parameters zu großen Einfluß auf das Suchergebnis hat und somit ebenfalls zu unsicher ist.

6.3. Lösungsversuche

Da die unvollständige Instanz nicht in einen direkten Vergleich mit den bereits vorhandenen Instanzen zu setzen ist, müssen neue Ansätze gefunden werden, um Vergleichsmöglichkeiten zu liefern.



row no.	l	a	b	c	d	e	f
1	1	0	0.100	2.260	5.290	2.620	0.100
2	2	0	3	3	2.200	0	?
3	3	0	2.970	6.220	3	0	?
4	4	0	2.900	3	2.010	0.110	?
5	5	0.040	3.400	7	3.160	0	?

Abbildung 6.5.: Datensatz Beta

6.3.1. Feature Extraction

Die bereits präsentierten Ansätze basierten auf der direkten Verwertung der Rohdaten. Statt die Tabelle in 6.2 jedoch mit den Rohdaten zu füllen, existiert die Möglichkeit, jede Instanz durch einen Vektor von Merkmalen zu repräsentieren. Die Momente der Statistik, die in der Einführung vorgestellt wurden, sind ein Beispiel für repräsentative Merkmale. Die NN-Suche würde somit Merkmale auf ihre Ähnlichkeit untersuchen und nicht mehr eigentliche Koordinaten.

Um das Problem der unterschiedlichen Instanzlängen und ihrer Verschiebungen zu umgehen, wurde die Methode Feature Extraction betrachtet. Da durch eine fixe Anzahl an Merkmalen repräsentative Vektoren gleicher Länge pro Instanz erzeugt werden, ist das oben vorgestellte Problem umgangen worden. Jedoch ist es notwendig, die unvollständige Instanz ebenfalls durch einen Vektor von festgelegten Merkmalen darzustellen. Da jedoch im Mittel nur eine kleine Anzahl von Punkten (im Bezug auf die durchschnittliche Instanzlänge der Reihe) zur Verfügung steht, ist die Auswahl der Merkmale oftmals nicht möglich. Somit kann die eigentliche Ähnlichkeit nicht bestimmt werden, da die unvollständige Instanz nicht als Merkmalsvektor dargestellt werden kann. Mit dieser Überlegung können auch weitere Verfahrensgruppen, die in der Geometrie Verwendung finden, ausgeschlossen werden. Dies sind all jene Verfahren, die auf Merkmalen beruhen - selbst auf einem einzigen wie z.B. Referenzpunktmethoden. Die unvollständige Instanz, die im Worst Case nur aus einem Attribut besteht, reicht nicht aus, um Vergleichskriterien zu bilden, die im Allgemeinen über vollständige Strukturen definiert sind.

6.3.2. Regressionsmodelle

In einer weiteren Methode, das Problem zu behandeln, werden Regressionsmodelle erzeugt. Da jedoch in dieser Sichtweise alle Instanzen als möglicher nächster Nachbar in Frage kommen, müßte für jede Instanz ein eigenes Regressionsmodell erzeugt und abgespeichert werden. Selbst wenn wir uns auf die wahrscheinlichste Musterklasse als Suchraum beschränken, müßten spätestens im folgenden Schritt, der eine Erweiterung des Such-

raums in Relation auf die vorhandene Punktemenge vorsieht, alle Instanzen als Regressionsmodell verwaltet werden. Hierbei kommen verschiedene Möglichkeiten zur Regression in Frage - beispielsweise das hier bereits vorgestellte Verfahren der SVM-Regression, das bezüglich der Prototypgewinnung zur Anwendung kam. Da jedoch unabhängig von der verwendeten Regressionsmethode alle Instanzen separat der Regression unterworfen und abgespeichert werden müssen, ist diese Methode für große Zeitreihen impraktikabel.

6.4. Lösungsmethode

Basierend auf der Arbeit von Andrej Pawlenko [Pawlenko 2005] zeigen wir nun eine Methode, die das Problem der Nearest Neighbour-Findung bei nicht direkt vergleichbaren Werten löst. Sei der Datenpunkt x_i gegeben, dann wird im Suchraum für jede Instanz der Vorgänger und Nachfolger der Indexdimension bezüglich x_i gesucht. Diese Punkte werden interpoliert, um somit den Vergleichswert für das Nearest Neighbour-Verfahren zu schätzen (Abbildung 6.6). Dieses Verfahren wird für jedes Attribut der unvollständigen Instanz wiederholt. Über das Nearest Neighbour-Verfahren kann nun punktweise die Distanz berechnet und aufsummiert werden. Die Instanz, die ihre Attribute als Vervollständigung beisteuert, weist das geringste Distanzmaß auf und ist somit der nächste Nachbar.

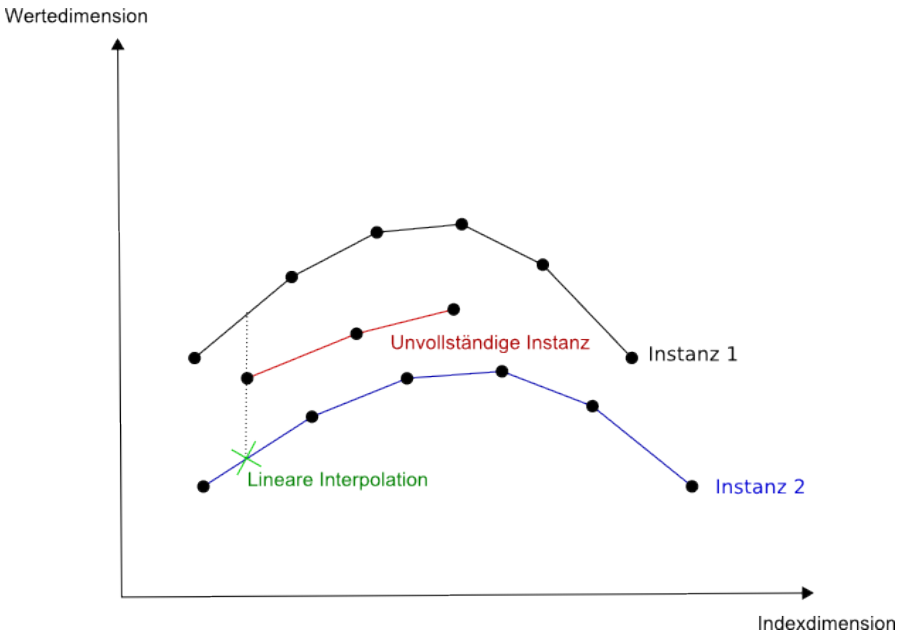


Abbildung 6.6.: Interpolation

Mit diesem zusätzlichen *Werkzeug* können erneut die Methoden 6.2.2 und 6.2.2 einer Prüfung unterzogen werden. Die Methode der Klassenbasierten Suche ist durch Interpolation zwar durchführbar geworden, weist jedoch immer noch den als kritisch zu erachtenden Benutzerparameter auf. Die Methode, die sich der Gewichte und Balancefunktion be-

dient, ist in der ersten Phase ebenfalls durch Interpolation durchführbar geworden. Die Schwierigkeiten treten im Bezug auf die Balancefunktion und den Globalitätsbegriff als Menge aller Instanzen auf. Genauer gesagt: Die Methode der NN-Interpolation basiert bezüglich des Entscheidungskriteriums auf einer Distanzmetrik. Aus diesem Grund läßt sich die Methode in der hier benutzen Form nur auf ähnliche Strukturen anwenden. Das Kriterium wurde bei der Suche innerhalb eines Musters erfüllt, da die Instanzen, die diesem zugeordnet wurden, quasi strukturidentisch waren (quasi wird hier im Bezug auf die manuelle Instanzauswahl gesehen und beschreibt den Vorgang des menschlichen Musterklassifizierens 2.2). Strukturdivergente Instanzen können jedoch nicht über die ausgewählte Metrik verglichen werden. Dies ist leicht ersichtlich, da Instanzen als Polygonzüge betrachtet werden können. Die Menge aller Polygonzüge in der Ebene wird (z.B durch den Hausdorff Abstand) zu einem metrischen Raum, und nicht durch die Euklidische Distanz. Um die Balancefunktion als Idee weiterhin zu nutzen, gibt es somit zwei mögliche Lösungsansätze. Zum einen die Suche nach einer neuen Metrik, zum anderen das Prüfen der hier verwendeten Globalität. Die zweite Sichtweise der Globalität betrachtete sie als Prototyp eines Musters (Prototyp Globalität - PG). Der nächste Abschnitt wird zeigen, daß diese Sichtweise dem bereits vorhandenen Wissen und der Balancefunktion gerecht wird.

6.5. Prototyp als Globalität

Strukturunterschiede, die zwischen Mustern auftreten (sonst wären sie schließlich nicht eigenständige Muster) hatten zu Schwierigkeiten bezüglich der Metrikanforderung geführt. Betrachten wir das Verfahren 6.2.2 unter Verwendung der PG-Sichtweise: Beginnend bei der lokalen Suche ist der Eingaberaum immer noch der aller Referenzinstanzen. Die globale Suche beinhaltet jetzt jedoch nicht mehr einen Raum mit vielfältigen Möglichkeiten, die es zu prüfen gilt, sondern nur noch eine Lösung. Ergebnis der globalen Suche bezüglich einer unvollständigen Instanz ist immer der Prototyp des Musters, zu dem sie aufgrund der Wahrscheinlichkeit gehört.

6.5.1. Verfahrensspezifikation - MNN

Im Folgenden soll das Verfahren spezifiziert werden, wobei das Verständnis für die Abläufe im Vordergrund steht. MNN ist die Abkürzung für Modifiziertes Nearest Neighbour und dient in Experimenten zur Kennzeichnung der nachfolgend beschriebenen Methodik. Eine konkrete Anwendung anhand von Datensätzen ist im Experiment 7.2.1 gegeben.

Nearest Neighbour - lokale Suche

Im ersten Schritt werden die Attribute der unvollständigen Instanz separiert. Aufgrund der Wahrscheinlichkeiten bezüglich der Musterabfolge wird das Muster mit der höchsten Auftrittswahrscheinlichkeit für den Indexbereich der unvollständigen Instanz bestimmt - dieses Muster bezeichnen wir als Referenzmuster bezüglich der unvollständigen Instanz. Die Instanzen des Referenzmusters, die durch Erstellung des Prototyps bereits musterabhängig verwaltet werden, bilden nun den Suchraum bezüglich der Nearest Neighbour-Anfrage. Für jedes Attribut wird danach der Indexwert bestimmt. Nun erfolgt das Inter-

polationsverfahren, das für alle Instanzen einen Vergleichswert bezüglich des aktuellen Attributes approximiert. Dies geschieht, indem das Intervall, welches die relevante Indexdimension beinhaltet, für jede Instanz des Suchraums bestimmt wird. Der so gefundene Intervallbereich wird der Interpolation unterworfen. Somit können die Werte für den relevanten Indexwert approximiert und nach der Formel ED 6.1.3 der Abstand berechnet werden. Die Abstände werden pro Instanz aufsummiert und bilden den Wert für das Abstandsranking. Die Instanz mit dem geringsten Abstand ist die gesuchte Lösung der Nearest Neighbour-Anfrage.

Ohne die Idee der Gewichtung zwischen Globalität und Lokalität wäre die Methode abgeschlossen. Im einfachen Fall hat die Funktion nur zwei zulässige Szenarien: Die vollständig lokale Suche oder die vollständig globale Suche. Als Resultat wären entweder der nächste Nachbar oder der Prototyp zu modellieren. Nun stellt sich die Frage: Welches Kriterium (bis auf die unerwünschten Benutzerparameter) kann an die Funktion zwecks einer solchen Entscheidung übergeben werden? Es ist die Anzahl der Punkte, die die unvollständige Instanz bilden. Sie sind, wie es die Instanz zuvor war, bis dato ungenutztes Wissen und werden zur Modellierung der Balancefunktion verwendet.

Balancefunktion - Möglichkeit der globalen Suche

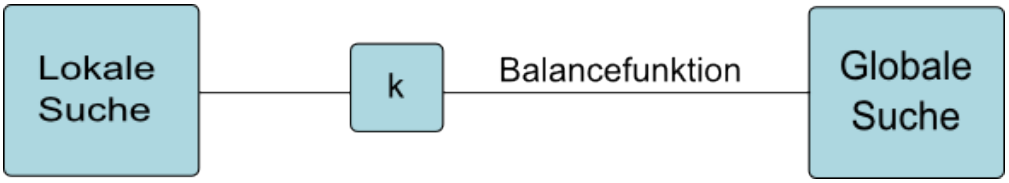


Abbildung 6.7.: Balancefunktion

Die unvollständige Instanz kann eine variierende Anzahl von Attributen besitzen. Diese Instanz wird nachfolgend äquivalent zu den m_i mit u_1 mit Attributen x_i $i = 1...k$ bezeichnet, wobei k die Anzahl der Attribute angibt (Einfache Indizierung, da nur eine unvollständige Instanz). Eine Aussage im quantitativen Sinne kann jedoch nicht ohne Referenz aufgestellt werden. Da Muster und somit die sie bildenden Instanzen unterschiedliche Länge aufweisen, ist eine unspezifizierte Sicht (z.B. im Verhältnis zur Gesamtanzahl der Attribute einer Zeitreihe) nicht sinnvoll. Für die unvollständige Instanz wird mittels der Wahrscheinlichkeitsabfolge das Referenzmuster bestimmt. Die Instanzen des Referenzmusters gelten als Maßstab, wodurch die Durchschnittslänge der Instanz im Verhältnis zu k gesetzt wird - und somit eine quantitative Aussage möglich ist. Im Folgenden beziehen sich die Aussagen über große und kleine k also immer auf dieses Verhältnis, bis letztendlich die daraus entwickelten Formeln den Vorgang formalisieren.

Eine geringe Anzahl an Attributen bedeutet, daß nur wenige Punkte für einen Ähnlichkeitsvergleich zur Verfügung stehen. In der graphischen Interpretation der Zeitreihe ist also nur eine kleine Teilstruktur sichtbar. Aufgrund dieses geringen Wissens erfolgt die Prognose tendenziell mit dem Prototyp, da dieser die gesuchte Struktur im Mittel

am besten prognostiziert. Im Fall k=0 liegt überhaupt kein Vergleichswert vor, und der Prototyp muß modelliert werden. Anders verhält es sich bei großen k. Hier darf gehofft werden, über die große Anzahl an Attributen eine möglichst ähnliche (NN-Suche) vollständige Instanz zu finden. In Abhängigkeit der k verändern sich die Prognosewerte - sie bewegen sich jedoch nur innerhalb des Intervalls, das durch NN-Instanz und Prototyp vorgegeben ist. Die Grundvoraussetzung an die Balancefunktion ist somit, bei kleinem k die Prognosewerte gegen den Prototyp und bei großem k gegen die Nearest Neighbour-Attribute konvergieren zu lassen. Abbildung 6.8 veranschaulicht graphisch den Intervallbereich zwischen einer Instanz und dem dazugehörigen Prototyp. Der Intervallgedanke führt somit für die Prognoseberechnung zu folgendem Modell:

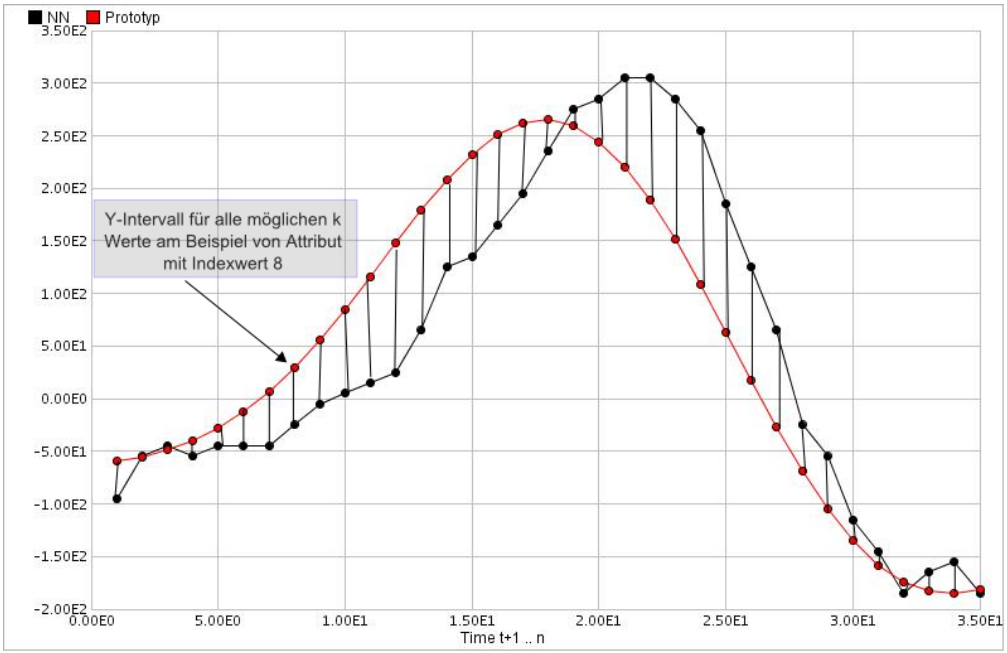


Abbildung 6.8.: Intervallbereich zwischen Globalität und Lokalität

Definition 6.5.1 (Balancefunktion)**.**

$$x_j^{u_1} = x_j^{m_i} - M \cdot A, j = k+1...n, \ i = \textit{NN Instanz ID} \tag{6.4}$$

$$A = x_j^{m_i} - x_j^{m_p}, j = k+1...n, \ p = \textit{Prototyp} \tag{6.5}$$

$$M = \frac{tanh(6\eta - 3) + 1}{2} \tag{6.6}$$

$$\eta = 1 - \frac{k}{h}, h = \frac{\textit{Anzahl der Attribute } \forall m \in \textit{Referenzmuster}}{\textit{Anzahl der Instanzen } m \in \textit{Referenzmuster}} \tag{6.7}$$

Formel 6.4 ist die eigentliche Balancefunktion. Mit ihr werden für die Attribute $k+1...n$, die u_1 vervollständigen, die neuen Werte bezüglich der Wertedimension berechnet. Die eigentliche Berechnung erfolgt, indem der nächste Nachbar innerhalb eines Intervalls gegen den Prototyp verschoben wird. Ob die Balancefunktion intern von der Globalität zur

Lokalität oder umgekehrt berechnet, ist für die Prognose irrelevant. Jedoch ist die Sichtweise, die NN-Instanz als Startpunkt zu wählen, methodisch günstiger. Da wir Attribute gegeneinander aufrechnen und indexbezogen vergleichen (siehe Problematik 6.2.2), ersparen wir uns eine mögliche neue Interpolation der Attribute. Der Prototyp liegt bereits als Regressionsmodell vor, und Werte können aus beliebigen Zeitachsenwerten entnommen werden. Das Intervall ist durch A 6.5 gegeben und stellt die Differenz des Prototypen attributsbezogen zur NN-Instanz dar. Da wir gegen den Prototypen verschieben, darf hier kein Absolutwert eingesetzt werden, denn möglicherweise schneiden sich die beiden Strukturen im Raum. In Formel 6.6 ist der Multiplikationsterm gegeben, der das Maß der Verschiebung steuert. Seine Entwicklung leitet sich aus folgenden Grundsätzen ab: Gesucht wurde eine Wachstumsfunktion im Bereich (0,1)- das offene Intervall ergibt sich aus der Konvergenz gegen Prototyp und Balancefunktion. Des Weiteren wird eine monoton steigende, stetige Funktion gefordert. Da sich gerade an den Endpunkten des Intervalls die marginalen Unterschiede bezüglich k nicht gravierend auswirken sollen, wird der Tangens Hyperbolicus angewendet. Dieser wurde über Stauchung und Verschiebung in die gewünschte Form gebracht (Abbildung 6.9). In die Wachstumsfunktion geht k über den Faktor η (Formel 6.7) ein, welcher den Durchschnittswert der Anzahl der Attribute im Referenzmuster angibt. Der Multiplikationsterm kann bei Bedarf durch andere Funktionen (z.B. die Gerade $f(\eta) = \eta$) ersetzt werden.

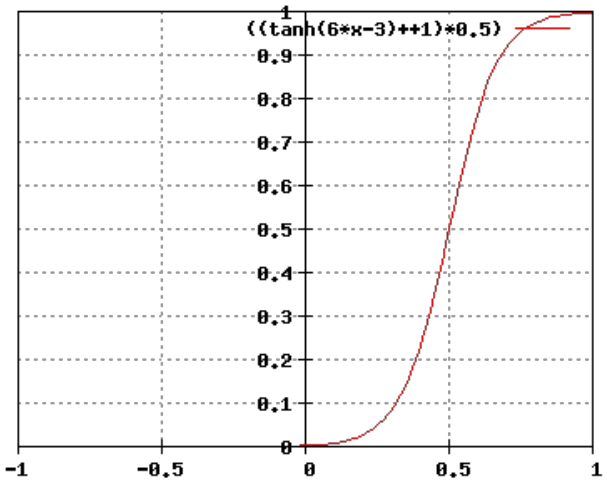


Abbildung 6.9.: Verlauf des Tangens Hyperbolicus

Von der Theorie zur Praxis

Nachdem nun alle theoretischen Grundlagen und Fälle betrachtet wurden, kann das Verfahren angewendet werden. Der nächste Nachbar wird bestimmt und die $n-k+1$ Attribute, die zur Prognose dienen, werden errechnet. Dazu reicht aus, für den letzten Indexwert $d_k^{u_1}$ den nächstgrößeren Indexwert in der NN-Instanz zu bestimmen. Die Attributwerte, die u_1 vervollständigen, werden der Balancefunktion unterzogen. Das Ergebnis liegt in Form der neuen w_i vor. Die Indexwerte werden durch Addition von Konstanten an die

Zeitreihe angepaßt. Dies ist erforderlich, da die Zeitreihenskala nun ab t+1 beginnt, und die ursprünglichen Indexwerte nur noch bezüglich ihres Abstandes untereinander zu verwerten sind. Die so gewonnene Liste von Attributen kann nun an die Zeitreihe angehängt werden und liefert die Prognose. Experiment B.1 (7.2.1) und B.2 (7.2.2) dienen zur Veranschaulichung der Vorgänge.

Betrachten wir abschließend folgenden Gedankengang: Bei der Suche nach dem nächsten Nachbar haben wir den Prototyp ausgeschlossen, da er ja laut der hier angestellten Überlegungen die Globalität repräsentiert (PG). Es wäre jedoch denkbar, daß der Prototyp selbst bereits der nächste Nachbar ist. Durch eine große Anzahl an bekannten Attributen würde die Balancefunktion in die Lokalität tendieren und somit über die Instanz prognostizieren. Um diesen Fall aufzufangen, kann der Prototyp in die Menge der Instanzen in der Funktion einer eigenständigen Instanz aufgenommen werden. Falls er das Resultat der NN-Suche darstellt, stoppt das Verfahren, da methodisch nichts mehr zu gewinnen ist und das bestmögliche Ergebnis vorliegt.

7. Praktische Anwendungen

In diesem Kapitel sind die Verfahrensschritte anhand von Beispieldaten dokumentiert. Die Grafik 7.1 gestattet das Einsortieren der Experimente in den Gesamtkontext der Arbeit. Die nachstehenden Experimente sowie Datenaufbereitung und graphische Darstellungen sind unter Rapid Miner entstanden. Rapid Miner - ehemals Yale - ist eine Open Source Data Mining Software. Die Software bietet eine breitgefächerte Anzahl von Methoden aus den Bereichen Maschinelle Lernverfahren, Daten Preprocessing, Daten Postprocessing, komfortable I/O Schnittstellen, sowie verschiedenartige Visualisierungsmöglichkeiten. Somit konnten die Ansprüche bezüglich der Datenverarbeitung und Lernumgebung durch eine Software abgedeckt werden [Mierswa et al. 2006].

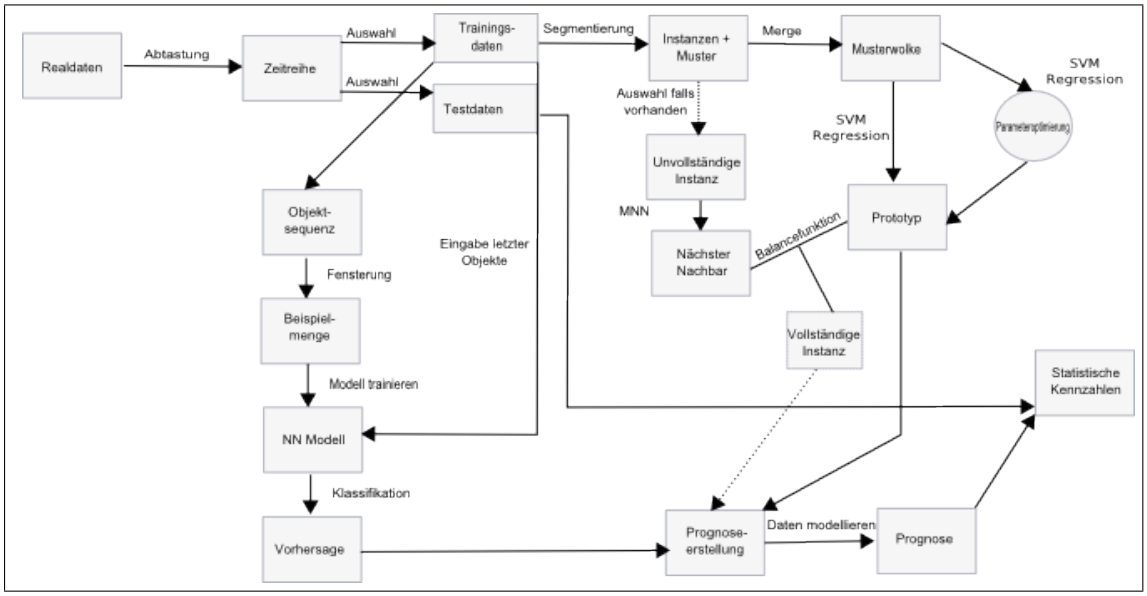


Abbildung 7.1.: Die Abbildung zeigt den Gesamtablauf der Prognose anhand der einzelnen Arbeitsschritte. Neben der Aufgabe des methodischen Überblicks soll damit die Einsortierung der einzelnen Experimente innerhalb der Verfahrenskette erleichtert werden.

7.1. Versuchsreihe SVM

Nachstehend sind ausgewählte Experimente aus dem Bereich der SVM (Methode Proto - Vergleiche Kapitel5) dokumentiert. Der Verlauf der Experimente umfaßt den vollständigen Arbeitsweg - von den Rohdaten zum Prototyp als Vertreter des Musters. Aufgrund

der Komplexität sind Experimente teilweise gesplittet worden, um so ein besseres Verständnis für die einzelnen Arbeitsschritte zu erzielen.

7.1.1. Instanzidentifikation und Mergeoperator (A.1)

Das Experiment beschreibt das Einlesen der Daten in die Lernumgebung Rapid Miner (RM), die Visualisierung dieser, sowie das manuelle Segmentieren zwecks Musteridentifikation. Jedes Segment ist eine Instanz, d.h. ein Beispiel für das Lernverfahren, das in Experiment A.2 dokumentiert wird. Zwecks Überschaubarkeit der Daten ist nur ein kleinerer Datenbereich gewählt worden, dafür wird aber der gesamte Instanzraum verwendet. Anschließend erfolgt das Überlagern der Instanzen, um so die erforderliche Form für A.2 zu erhalten.

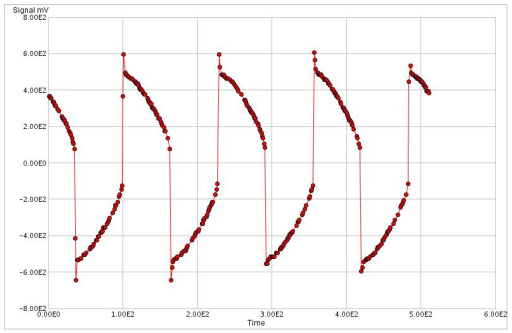


Abbildung 7.2.: Datenvisualisierung

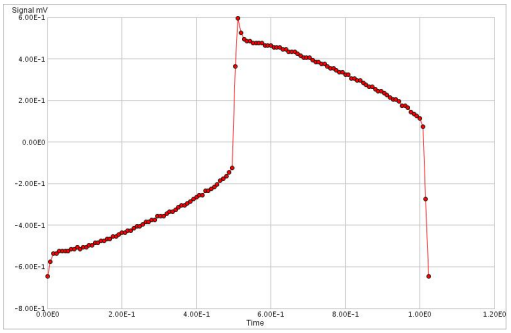


Abbildung 7.3.: Musterinstanz

Der Versuch A.1 erfolgt auf dem Datensatz normal14 von E.Keogh [E. Keogh 2006]. Ausschlaggebend für die Auswahl ist die klare Definition einer Musterstruktur, die sich punktgenau auf Zahlenebene im Datensatz widerspiegelt. Startpunkt des Musters (bezeichnet als S) sowie Endpunkt des Musters (bezeichnet als E) sind auf mathematischer Ebene Extrema. Da die Struktur ansonsten keine lokalen Tiefpunkte aufweist, können E und S im Datensatz exakt bestimmt werden und somit die Instanzen separiert werden.

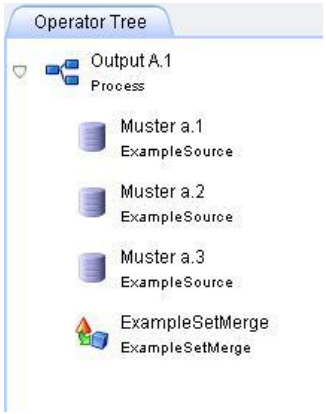


Abbildung 7.4.: Merge RM

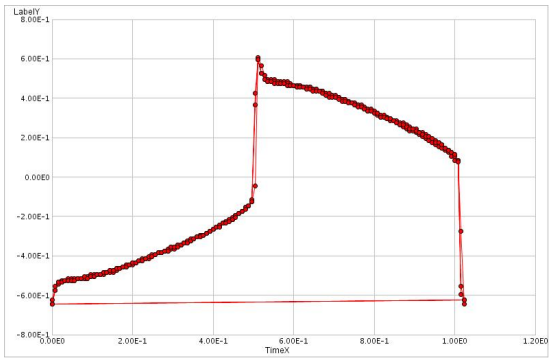


Abbildung 7.5.: Resultat der Mergeoperation

Die erste Instanz läuft über die Attribute x_{39} bis x_{167}. Die Daten sind manuell aus der Originaldatei extrahiert und als Musterinstanz a.1. abgespeichert worden. Derselbe Vorgang wurde für die anderen beiden Instanzen des Musters A wiederholt. Hierbei wurde für jede Instanz die Zeitachse auf $x_S = 0$ gesetzt und pro nachfolgendem Wert inkrementell erhöht, um so die Mergeoperation zu ermöglichen (Indexsubtraktion). Das Ergebnis wurde als example set data file und als attribute description file abgespeichert und dient dem Experiment A.2 als direkte Eingabe im I/O Example Source.

7.1.2. Parametersuche und Prototyp (A.2)

Im Experiment A.2. werden die Parameter für die SVR bestimmt und anschließend mit diesen Parametern die SVR trainiert. Das Ergebnis ist der Prototyp A, der in der langfristigen Prognose als Prototyp des Musters A Gebrauch findet.

Zur Verwendung ist der LibSVMLearner mit RBF Kernel und epsilon-SVR Typus gekommen. Die Parametersuche erstreckt sich über C, epsilon,p und gamma (Auswahl der ersten Phase), deren Werte in Tabelle 7.1 abgebildet sind.

C	50	100	150	200	250	500
epsilon	0.01	0.03	0.08	0.001	0.1	0.5
p	0.1	0.3	0.6	0.01	0.05	0.8
gamma	0.0	0.05	0.1	1.0	1.0	2.0

Tabelle 7.1.: Kandidaten der Parametersuche

Hierbei wird der Metalearner GridSearch verwendet und mit einer 10fachen Kreuzvalidierung in RM kombiniert. Die Parameter, die in der ersten Phase bestimmt werden, werden in einer zweiten Phase verfeinert. Das bedeutet, daß die direkte Umgebung, d.h. die Nachbarn des Parameterwertes, der als optimal für C, epsilon, gamma und p bestimmt wurde, als Eingabe für die Operation *ParameterOptimization* gewählt wurde. Ein potentiell besseres Ergebnis kann erneut als Ausgangspunkt für eine Verfeinerung gewählt werden. Somit kann dieses Verfahren beliebig oft verwendet werden, bis das Ergebnis zufriedenstellend oder die Abweichung so gering ist, daß sie für die hier durchgeführte Parametersuche nicht mehr ins Gewicht fällt. Die Operatorline von RM zeigt den Versuchsablauf in Abbildung 7.6.

Die erste Phase ergab als Parametersatz C=250, epsilon=0.05, gamma=0.1 und p=0.3. Nach mehrmaligem Verfeinern beliefen sich die optimalen Werte auf C=300, epsilon=0.05, gamma=0.1 und p=0.3. Mit diesem Parametersatz wurde die SVR auf dem Datensatz trainiert - das Ergebnis ist in 7.7 dargestellt. Der dazugehörige Performancevektor beläuft sich auf:

- root mean squared error: 93.364 +/- 53.794 (mikro: 107.672 +/- 0.000)
- absolute error: 36.294 +/- 20.853 (mikro: 36.318 +/- 101.362)
- relative error: -0.067 +/- 0.061 (mikro: -0.067 +/- 0.442)

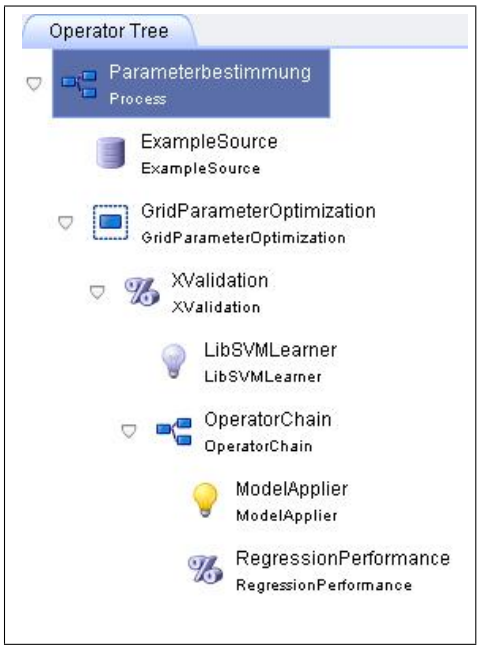


Abbildung 7.6.: Parameterbestimmung in RM

- normalized absolute error: 0.100 +/- 0.053 (mikro: 0.101)
- root relative squared error: 0.240 +/- 0.133 (mikro: 0.281)

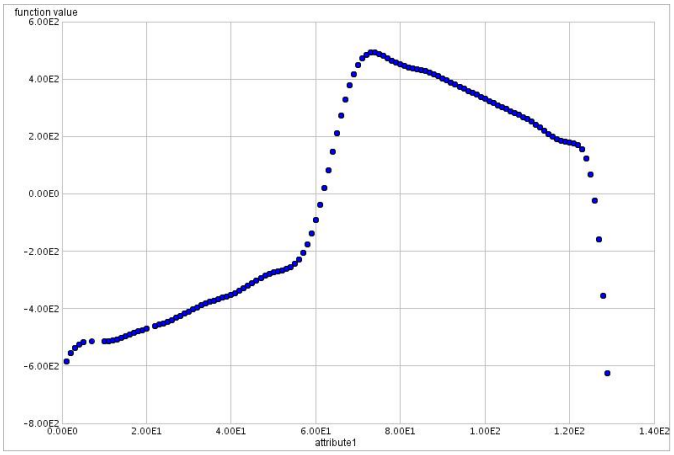


Abbildung 7.7.: Prototyp A

7.1.3. Sequenzprognose und Modellierung (A.3)

Das folgende Experiment beschreibt das Erlernen verschiedener Prototypen auf einer Zeitreihe mit mehreren Mustern. Der Schwerpunkt der Dokumentation liegt auf der Mo-

dellierung der Mustersequenz mittels Klassifikation.

Die zugrunde liegende Zeitreihe stammt von E.Keogh und ist unter dem Stichwort Kalpakis Datensatz (eeg.13) zu finden. Die Zeitreihe wurde einer Permutation bezüglich der Instanzabfolge unterzogen, um so verschiedenartige Objektsequenzen zu erzeugen und deren Auswirkung auf die Prognose zu testen.

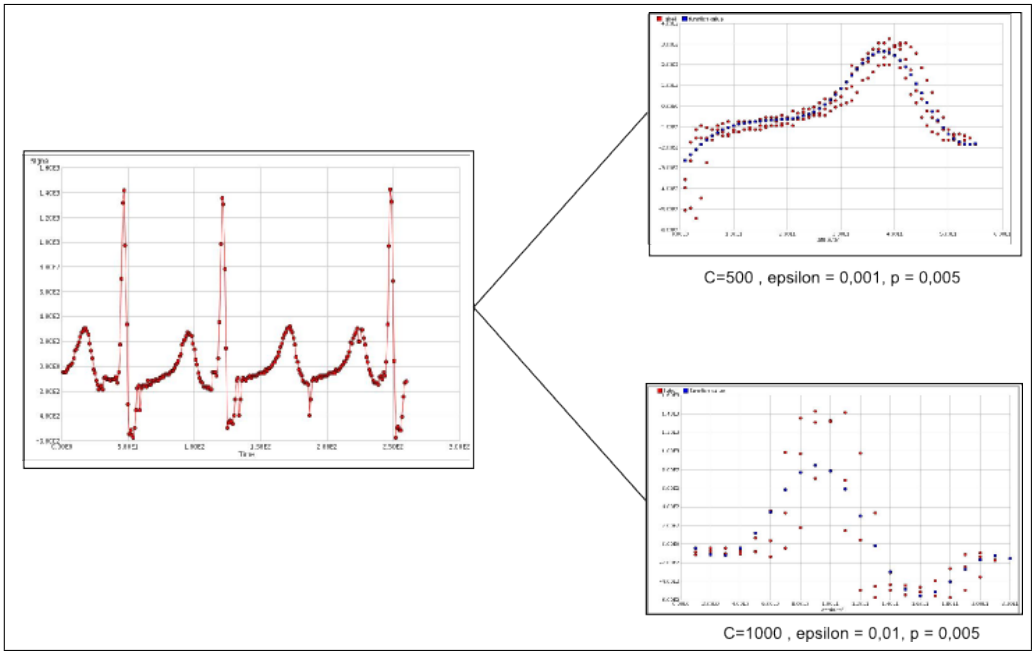


Abbildung 7.8.: Verschiedene Prototypen pro Zeitreihe

Das Erlernen von Prototypen wurde bereits in den vorangestellten Experimenten hinreichend erläutert, und das Vorhandensein mehrerer Muster ändert die Technik als solche nicht. Lediglich zu Beginn müssen die Instanzen bezüglich ihres korrespondierenden Musters geordnet werden. Danach erfolgt das Erzeugen der Musterwolken (eine für jedes Muster), die Parametersuche für die SVM, sowie das Bilden des Prototyps (Vergleiche Abbildung 7.8). Im ersten Schritt (der Instanzbestimmung) wurde die Zeitreihe manuell in die Objektsequenz {ABABAABABABAABAABABABABAAB} überführt (7.2). Anschließend wurde mittels RM die Objektsequenz mit n=3 (Fenstergröße) und s=1 (Schrittweite) gefenstert (Series2WindowExamples). Intern mußten dafür die nominalen Attribute in numerische Werte überführt werden, um die Anwendung der Fensterfunktion zu ermöglichen. Der resultierende *.dat File wurde danach mittels Ersetzung in die ursprüngliche numerische Codierung umgewandelt. Dieser File dient RM als Eingabemenge für das NN-Modell. Daraufhin wurde ein NN-Modell mit k=1 und numerischer Distanzfunktion erstellt. Das Zeitreihenende {AAB} wurde als Klassifikationsanfrage an das Modell übergeben und die Klasse bestimmt. Das Ergebnis lautet A und bedeutet für die Prognose, daß das nachfolgende Muster durch den Prototyp A repräsentiert wird.
Um den Prognosehorizont zu vergrößern, wurde die Zeitreihe mittels erneuter Fensterung

Position 1	Position 2	Position 3	Label
A	B	A	B
B	A	B	A
A	B	A	A
B	A	A	B
A	A	B	A
A	B	A	B
B	A	B	A
A	B	A	B
B	A	B	A
A	B	A	A
B	A	A	B
A	A	B	A
A	B	A	A
B	A	A	B
A	A	B	A
A	B	A	B
B	A	B	A
A	B	A	B
B	A	B	A
A	B	A	A
B	A	A	B

Tabelle 7.2.: Die Tabelle zeigt das Ergebnis der Fensterung aus dem Experiment. Die Fenstergröße beträgt n=3, die Schrittweite s=1 und der Horizont h=1.

und divergentem Horizont im Bezug auf das Label in neue Beispielmengen überführt. Auf jeder Beispielmenge wurde ein eigenständiges Modell erlernt, d.h. für jedes zu postulierende Muster eine Beispielmenge und ein dazugehöriges Modell. Tabelle 7.3 zeigt die Vorhersage für einen Prognosehorizont von drei Mustern. Gemessen an der Musterdurchschnittslänge der Zeitreihe entspricht das ≈ 75 Attributen.

Klassifizierung	Position 1	Position 2	Position 3	Prediction
Aufruf 1	A	A	B	A
Aufruf 2	A	B	A	B
Aufruf 3	B	A	B	A

Tabelle 7.3.: Nearest Neighbour-Aufrufe in Abhängigkeit des Prognosehorizontes

Auf numerischer Ebene werden die Datenpunkte der Regressionsmodelle mit fortlaufenden Indizes gemäß der ermittelten Reihenfolge an die Zeitreihe angehängt. Abbildung 7.9 stellt die Visualisierung der so gewonnenen Zeitreihenprognose mittels RM dar.

In dieser Variante wurde das Zeitreihenende verworfen (Abbildung 7.10) und der Bereich durch einen Prototypen modelliert. Da für kurzfristige Prognosen die erweiterte Methodik (Methode MNN - Vergleiche Kapitel6) zur Verfügung steht, wird auf eine gesonderte Fallbehandlung für den Prototypen verzichtet und auf das Beispielexperiment in Abschnitt 7.2.1 verwiesen.

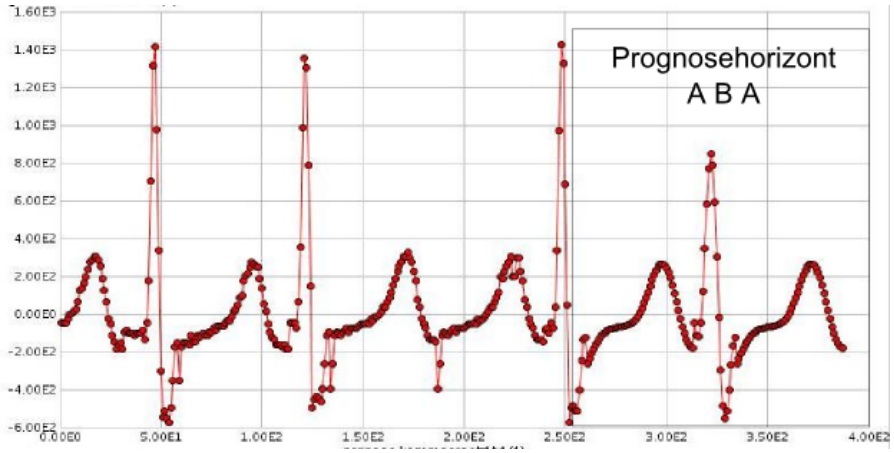


Abbildung 7.9.: Visualisierung des Prognoseresultates mittels RM

7.2. Versuchsreihe Nearest Neighbour und Balancefunktion

Nachstehend sind Experimente zum Bereich 6 dokumentiert. Der Verlauf der Experimente umfaßt den vollständigen Arbeitsweg - von den Rohdaten zur Prognose. Aufgrund der Komplexität sind Experimente teilweise gesplittet worden, um so ein besseres Verständnis für die einzelnen Arbeitsschritte zu erzielen.

7.2.1. Nearest Neighbour im Instanzraum (B.1)

Das Experiment beschreibt die Nearest Neighbour-Suche mittels Interpolation. Verarbeitungsschritte, die nicht explizit erklärt sind, sind vorangestellten Experimenten zu entnehmen. Grundlage bildet die Zeitreihe aus Experiment A.3 (Graphische Darstellung z.B. in 7.8), deren Indexdimension nun bis zum Wert x_{281} läuft 7.10 und mit einer unvollständigen Instanz abschließt. Zuerst werden die Attribute der unvollständigen Instanz u_1 der Indexsubtraktion unterzogen. Die Auswertung der Wahrscheinlichkeiten liefert Muster 1 als Referenzmuster (Abbildung 7.11). Nun wird für jede Instanz des Referenzmusters der Abstand bezüglich u_1 ermittelt. Dies geschieht mittels einer Excel-Tabelle. Die Entscheidung fiel auf Excel, da die Instanzen bereits als Excel-Tabellen vorliegen und RM Excel-Tabellen auslesen kann. Somit können Ergebnisse dieses Schrittes ohne großen Aufwand weiterverarbeitet werden. Der Aufbau der Excel-Tabelle sowie die funktionalen Ausdrücke sind in der Tabelle 7.4 gegeben. Abbildung 7.12 verdeutlicht den Tabellenaufbau anhand der Abstandsberechnung von u_1 zu $m_1 \in M_1$.
Im Folgenden werden nun die restlichen Instanzen zunächst der Interpolation und anschließend der Abstandsberechnung unterzogen. Die Ergebnisse sind in Tabelle 7.5 dokumentiert. Somit ist m_1 der nächste Nachbar bezüglich u_1. Die NN-Suche wurde hier über die Abstandsberechnung in Excel ausgeführt. Bei großen Datenmengen müssen jedoch die Abstände manuell verwaltet werden. Aus diesem Grund sei hier auf die zweite Möglichkeit mit RM hingewiesen. Der grundsätzliche Aufbau der RM-Datenstrukturen wurde bereits in vorherigen Kapiteln (z.B 6.2) vorgestellt. Jede Zeile repräsentiert wie in den vorangestellten Beispielen eine Instanz der Referenzklasse. Jedoch werden die Attri-

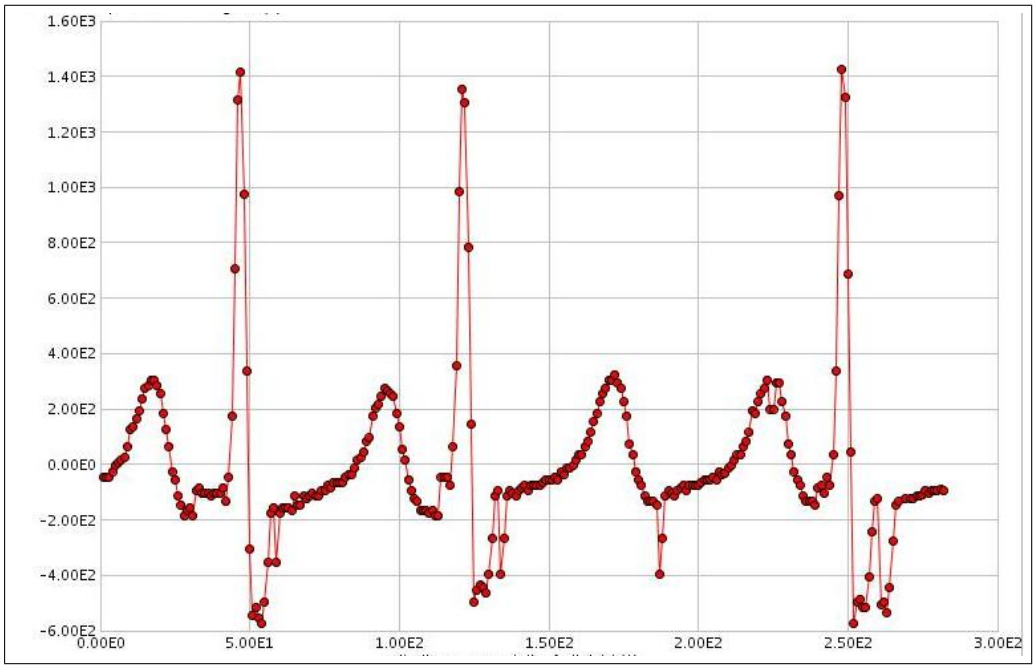


Abbildung 7.10.: Zeitreihe mit unvollständiger Instanz

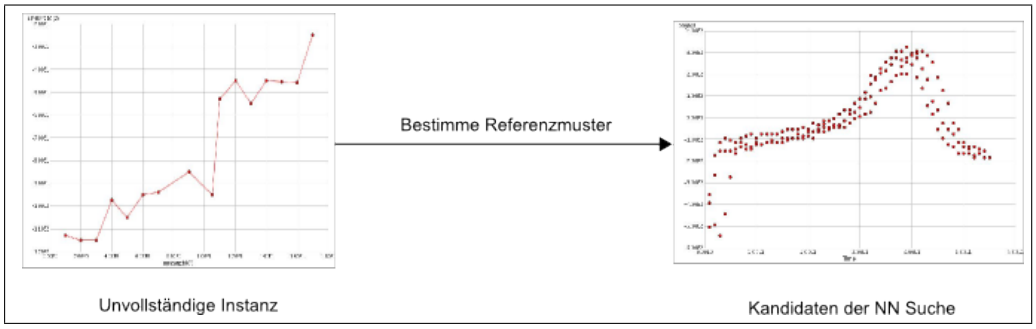


Abbildung 7.11.: Kandidaten der NN-Suche

Spalte	Funktion	Excelaufruf
A	m_i Index	Wert
B	m_i Attributwert	Wert
C	Interpolierter Wert	=Trend(Verschiebung(B1,HX;0,2,1);Verschiebung(A1,HX,0,2,1)DX)
D	Index u_1	Wert
E	Attributwert u_1	Wert
F	Abstand	Wurzel(Potenz(EX-CX;2))
G	Summe Abstand	SummeF1:FX
H	Hilfstabelle	Vergleich(DX;A2,A200:Wenn(A2>A3;-1;1))

Tabelle 7.4.: Die Tabelle zeigt die Funktionsanweisung nach Spalten in Excel an. Der Wert X symbolisiert die Zeilenreferenz, z.B. H1 für Zeile 1. Diese Tabelle muß für jede zu prüfende Instanz m_i aufgerufen werden bzw. optional die A Werte angepasst werden.

Time	Signal	Interpolierte Werte	Attribut - Index	Attribut – Wert	Abstand	Summe Abstand	Hilfstabelle
1	-505	-505	1	-505	0	99,9	1
2	-495	-495	2	-495	0		2
3	-545	-545	3	-533	12		3
4	-445	-445	4	-445	0		4
5	-275	-275	5	-275,8	0,8		5
6	-145	-145	6	-145	0		6
7	-135	-135	7	-135	0		7
8	-125	-135	9	-125	10		9
9	-135	-125	10,5	-135,4	10,4		10
10	-125	-125	11	-125	0		11
11	-125	-115	12	-122	7		12
12	-115	-115	13	-115	0		13
13	-115	-113	14,1	-115	2		14
14	-115	-95	15	-111	16		15
15	-95	-105	16	-92	13		16
16	-105	-95	17	-105	10		17
17	-95	-95	18	-95	0		18
18	-95	-95	19	-93	2		19
19	-95	-85	20	-95	10		20
20	-85	-95	21	-88,3	6,7		21

Abbildung 7.12.: Wertetabelle Interpolation (Auszug)

Instanz	m_1	m_2	m_3	m_4
Distanz	99,9	1449,5	1449,5	1950,2

Tabelle 7.5.: Abstandswerte in ausgewählter Musterklasse

butsspalten über die interpolierten Werte gebildet und nicht mehr über Rohdaten. Dies bewirkt eine identische und im metrischen Sinne vergleichbare Anzahl an Attributen. Das Modell wird dann auf die Instanz u_1 angewendet und die ED-Werte der Instanzen werden berechnet (7.6). In beiden Fällen wird die Zeitreihe über die zusätzlichen Attribute der NN-Instanz modelliert. Das erste Attribut von m_i, welches das Kriterium $d_i \in m_i > d_k \in u_1$ erfüllt, bildet den Prognosewert für den Zeitpunkt $t + 1$. Die Folgeattribute $d_{i+1}...d_n$ werden den Zeitabständen entsprechend an die Reihe angehängt. Wir erhalten $22 \in m_1 > 21 \in u_1$ bezüglich der Suche und setzen die Zeitreihe mit den Werten bis zum letzten Attribut x_{315} von m_1 fort. Das Ergebnis ist in Abbildung 7.13 dargestellt.

Instanzen	I_1	I_2	...	I_n
m_1	w	w	...	w
m_2	w	w	...	w
...	...	...	...	...
m_m	w	w	...	w
u_1	w	w	...	w

Tabelle 7.6.: Die Tabelle zeigt den Aufbau in RM. Die Werte I_i repräsentieren die Indexwerte, die mittels Interpolation ermittelt wurden. Die m_i sind die Instanzen einer Musterklasse und repräsentieren auf Modellebene die vorherzusagenden Label. Die w repräsentieren die Y-Werte der interpolierten Stelle. Alle m_i dienen als Beispielmenge, über die das NN-Modell erlernt wird. Die Instanz u_1 wird in einem zweiten Arbeitsschritt dem Modell übergeben und das Ergebnis der Klassifizierungsaufgabe ermittelt (das gesuchte Label).

7.2.2. Modellierung der Balancefunktion (B.2)

Das nachfolgende Experiment stellt die Umsetzung der Balancefunktion vor. Hierfür wurde das Experiment aus B.1 (7.2.1) erweitert, um die Auswirkungen der verschiedenen k auf die Prognose zu demonstrieren. Zuerst werden bezüglich der gesuchten Indexdimension die Werte des Prototypes mittels RM ermittelt. Der Wert für h kann über ein einfaches Count der Werte im Mergefile ermittelt werden. Wir erhalten für $\eta = 1 - (20/(215 : 4)) \approx 0,626$. Die Berechnung der Balancefunktion sowie der benötigten Zwischenwerte wurde mittels Excel realisiert. Tabelle 7.14 gibt die Wertetabelle an, die wie im vorherigen Experiment zur direkten Prognose genutzt wird. In Tabelle 7.15 sind verschiedene Prognosewerte in Abhängigkeit von k dargestellt, die im Rahmen der Versuche erstellt wurden. Die Visualisierung der entsprechenden Teilstruktur in Abbildung 7.16 verdeutlicht die Auswirkungen auf die Prognose.

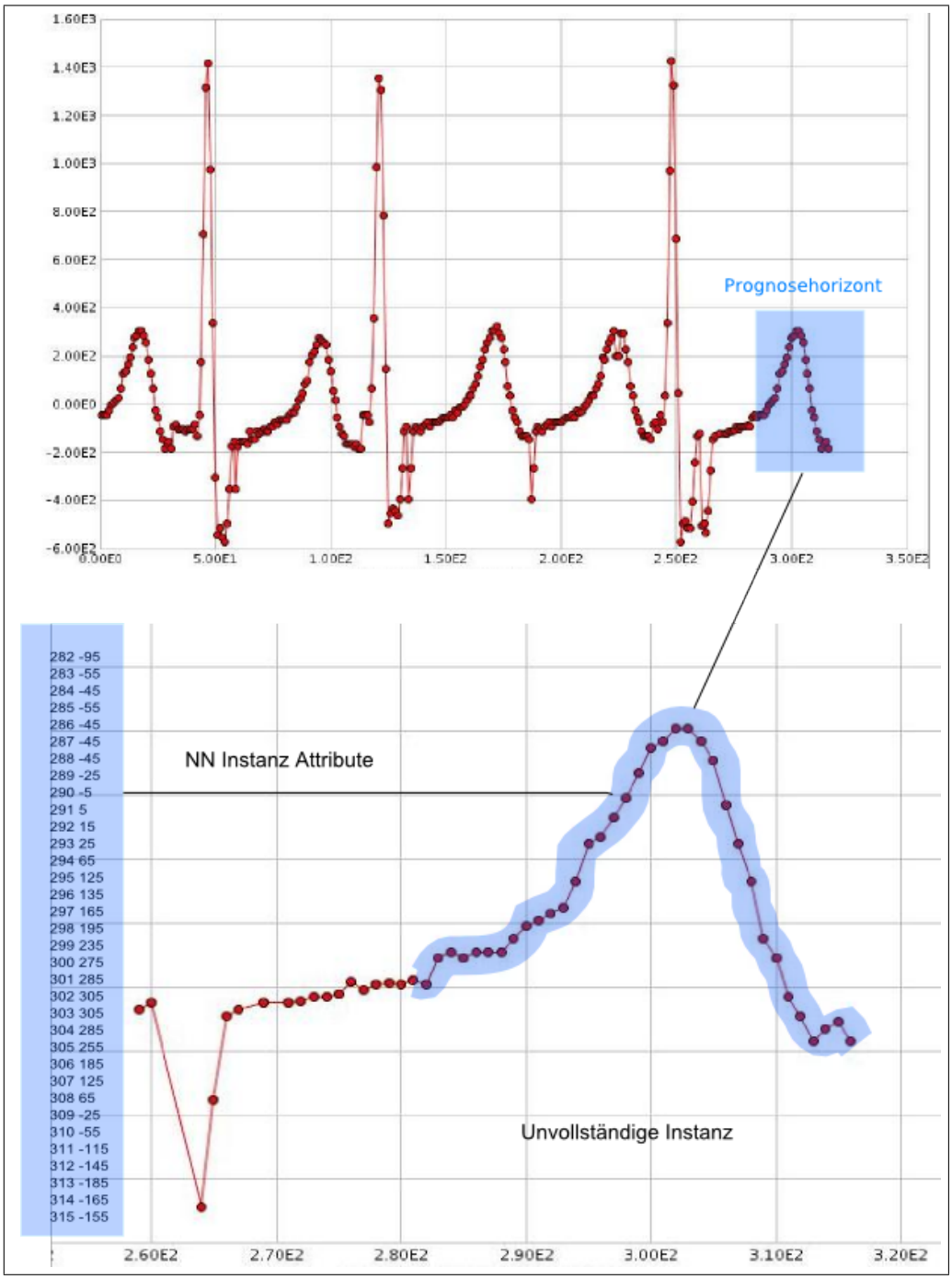


Abbildung 7.13.: Modellieren des kurzfristigen Prognosehorizontes

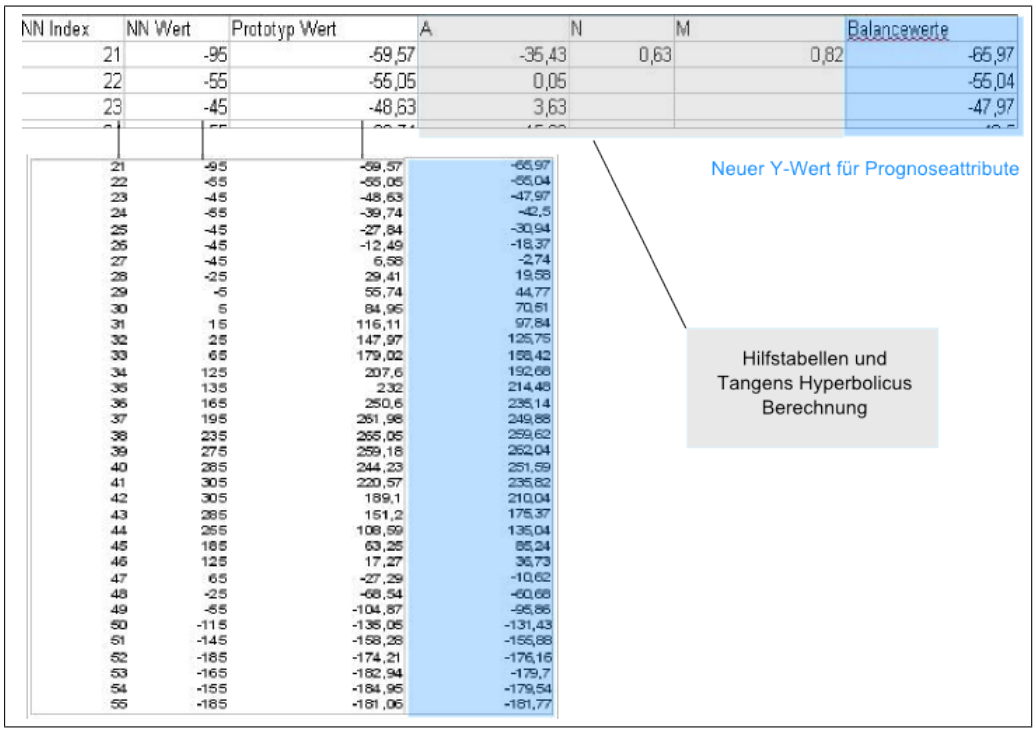


Abbildung 7.14.: Berechnung der Balancefunktion

row no.	NN	Prototyp	K=10	K=1	K=40	K=50	K=25	Time t+1 .. n
1	-95	-59.572	-60.370	-59.680	-93.200	-94.800	-82.450	1
2	-55	-55.046	-55.050	-55.050	-55	-55	-55.020	2
3	-45	-48.627	-48.550	-48.620	-45.180	-45.020	-46.290	3
4	-55	-39.741	-40.090	-39.790	-54.230	-54.910	-49.590	4
5	-45	-27.845	-28.230	-27.900	-44.130	-44.900	-38.920	5
6	-45	-12.495	-13.230	-12.600	-43.350	-44.810	-33.480	6
7	-45	6.581	5.420	6.420	-42.380	-44.710	-26.720	7
8	-25	29.414	28.190	29.250	-22.240	-24.690	-5.720	8
9	-5	55.737	54.370	55.550	-1.920	-4.650	16.520	9
10	5	84.948	83.140	84.700	9.060	5.460	33.330	10
11	15	116.108	113.820	115.800	20.130	15.580	50.830	11
12	25	147.965	145.190	147.590	31.240	25.700	68.570	12
13	65	179.017	176.440	178.670	70.790	65.650	105.400	13
14	125	207.600	205.740	207.340	129.190	125.470	154.270	14
15	135	232.003	229.810	231.700	139.930	135.550	169.370	15
16	165	250.600	248.670	250.340	169.350	165.490	195.330	16
17	195	261.976	260.460	261.770	198.400	195.380	218.730	17
18	235	265.052	264.370	264.960	236.530	235.170	245.650	18
19	275	259.184	259.540	259.230	274.200	274.910	269.400	19
20	285	244.226	245.150	244.350	282.930	284.770	270.550	20

Abbildung 7.15.: Prognosewerte bei variierendem k

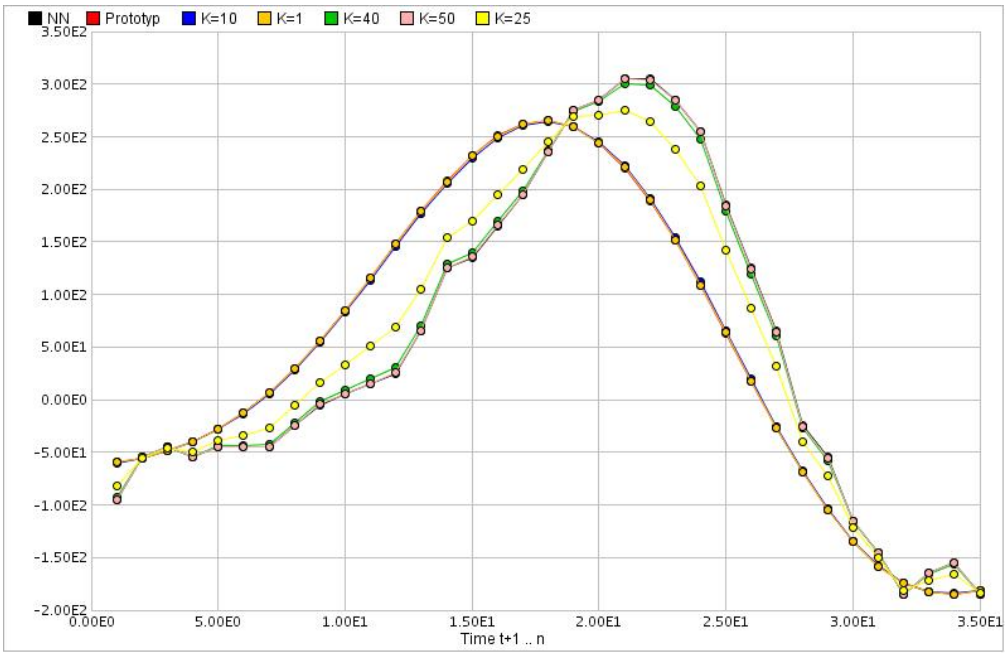


Abbildung 7.16.: Die Abbildung zeigt die Veränderungen des Strukturverlaufs bei variierendem k. Die Intervallbegrenzungen sind durch die NN-Instanz und den Prototyp gegeben. Für Visualisierungen der Balancefunktion auf Testdaten sei auf die M-Serie in Abschnitt 8.1 verwiesen.

7.3. Versuchsreihe Rauschen

Im Ausblick wird unter 9.2.3 ein Konzept zur Rauschelimination vorgestellt. Nachfolgendes Experiment soll die Vorgehensweise zur Rauschelimination bei Zeitreihen unter Verwendung der Erkenntnisse dieser Arbeit aufzeigen.

7.3.1. Rauschelimination (C.1)

Ziel der Aufgabe ist, mögliches Rauschen über die Remodellierung der Zeitreihe mittels Prototypen zu eliminieren. Hierzu wurde der Datensatz Dodgers mittels des RM Preprocessing Operators NoiseGenerator mit einer Verzerrung unterlegt. Die Muster wurden ausgewählt und mittels des Merge Operators überlagert, so daß die bereits bekannten Musterwolken entstehen. Daraufhin wurde für jede Musterwolke mittels der Operatorlinie aus Experiment 7.1.2 die Parameteroptimierung der SVM durchgeführt. Der über die Regression gewonnene Prototyp (Vergleiche Abbildung 7.17) ersetzt die korrespondierenden Instanzen der Zeitreihe entsprechend ihrer Musterzugehörigkeit. Die Güte der Ergebnisse bei variierender Rauschüberlagerung wurde mittels Validierung in RM bestimmt. Hierfür wurde der Prototyp gegen eine Auswahl von 30 Instanzen aus der Originalzeitreihe getestet, um somit ein Maß für die Rauschelimination vorzuweisen. Abbildung 7.18 verdeutlicht das Ergebnis für einen Ausschnitt der Zeitreihe, die 2 Muster umfaßt, auf graphischer Ebene. Für statistische Kennzahlen zu den Ergebnissen der Rauschelimination sei auf Anhang B verwiesen.

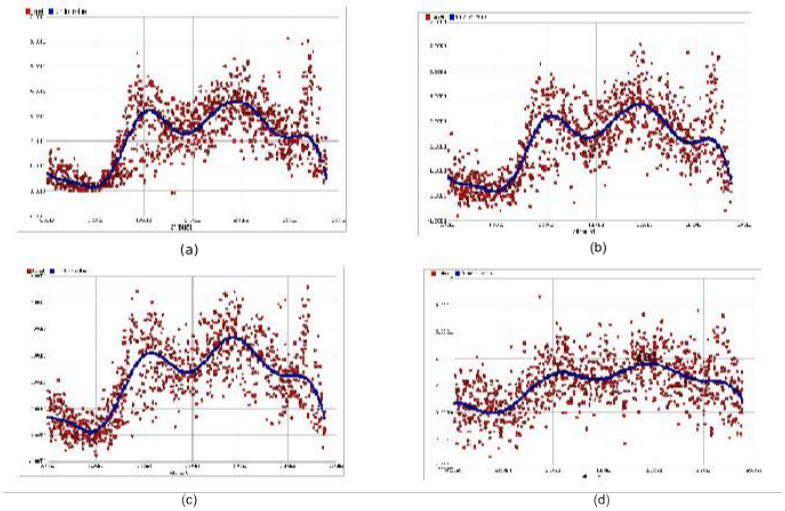


Abbildung 7.17.: Die Abbildung zeigt die Prototypvariationen, die durch Rauschüberlagerung der Zeitreihe entstehen. (a) ist auf der Originalzeitreihe gewonnen, (b) hat eine Rauschüberlagerung des Labels von 0.05, (c) hat eine Rauschüberlagerung des Labels von 0.1 und (d) schließlich von 0.2.

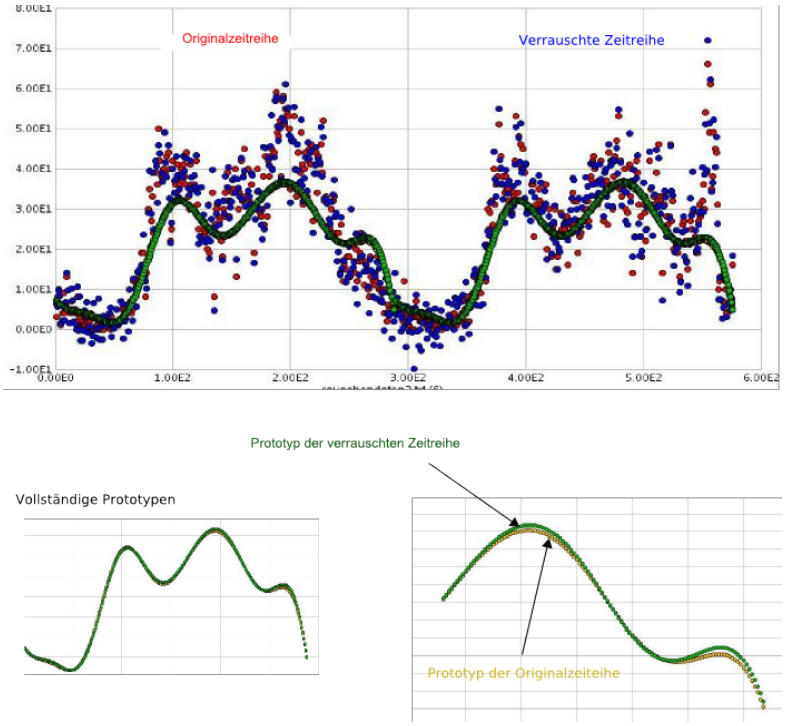


Abbildung 7.18.: Die Darstellung dokumentiert die Ergebnisse der Rauschentfernung mittels Prototypmodellierung auf graphischer Ebene. Die Verzerrung des Labels beträgt 0.05, und der darauf erstellte Prototyp weist nur eine Abweichung im Bereich von 100steln der Ordinateneinheit zum Prototypen der Originalzeitreihe auf.

8. Verfahrensbewertung mittels statistischer Kennzahlen

In diesem Kapitel soll eine abschließende Verfahrensbewertung getroffen werden, die Stärken und Schwächen der entwickelten Methodik darstellt. Um die Aussagen der Verfahrensbewertung zu verifizieren, wird die Methodik dieser Arbeit gegen ausgewählte Methoden getestet und die Ergebnisse nach statistischen Kriterien (Vergleiche hierzu Anhang A) ausgewertet.

8.1. Methodentest

Im Rahmen abschließender Experimente wird die hier entwickelte Methode (gekennzeichnet mit Proto) gegen ausgewählte Methoden getestet. Das Verfahren Proto wird in einigen Experimenten unter verschiedenen Restriktionen durchgeführt (unterschiedliche Musterwahl und optionales MNN). Dies ist in den Tabellen numerisch gekennzeichnet und unter den Methodenangaben dokumentiert. Bei den Methoden, die als Vergleichsverfahren herangezogen werden, handelt es sich um die Methode von Stefan Rüping (gekennzeichnet mit Rüping), sowie die Statistikmethode Arma. Die Methode von Rüping kann unter [Rüping 1999] nachvollzogen werden. Für die Experimente mittels der Rüping-Methode werden die Bedingungen aus der referenzierten Literatur übernommen, um dem Autor und seiner Methode gerecht zu werden. Das Arma Modell ist in den statistischen Versuchen in das Standardmodell Arma ohne zusätzliche Modifikationen (Arma1) sowie Arima mit automatischer Modellbildung und Ausreißerelimination (Arma2) unterteilt. Alle Methoden werden unter gleichen statistischen Bedingungen getestet. Die Experimente (bis auf die Prognose mittels Arma) können in RM durchgeführt werden. Das Arma Modell wird mittels der Statistiksoftware TSW [Pollock 2002] erstellt und die resultierenden Prognosewerte in Excel exportiert. Die Excel-Tabellen können dann der Hauptsoftware RM für die Berechnung statistischer Kennzahlen zur Verfügung gestellt werden.

8.1.1. Versuchsreihe M1

Datenquelle Metallproduktion in Italien (Datensatz Prodme), monatsweise Erhebung von 1990 - 1998. Entnommen aus [Pollock 2002].

Versuchsaufbau Insgesamt 74 Attribute, davon 54 Attribute Trainingsdaten (inklusive Validierungsdaten) $\approx 70\%$ und 20 Attribute Testdaten $\approx 30\%$.

Spezifische Methodenangaben Bei **Rüping**: JMySvmLearner mit Anova Kernel und fensterweiser Parameteroptimierung (sigma, L, epsilon, C, convergence-epsilon, loss)

sowie konstantem degree = 2, Fensterung n=14 (getestet 3,4,8,14,16). **Proto**: Trendschätzung mit linearer Trendfunktion $f(x) = 0,01x$, drei Muster mit jeweils optimierter (gamma, epsilon, p, C) LibSVM-epsilonSVR mit rbf Kernel (**Proto**) bzw. zwei Muster mit jeweils optimierter (gamma, epsilon, p, C) LibSVM-epsilonSVR mit rbf Kernel (**Proto2**), Fenstergröße n=3 (getestet 2,3,4). Die Musterauswahl von Proto und Proto2 ist Abbildung 8.2 zu entnehmen. **Arma1** und **Arma2** mittels automatischer Parameterwahl in TSW ermittelt.

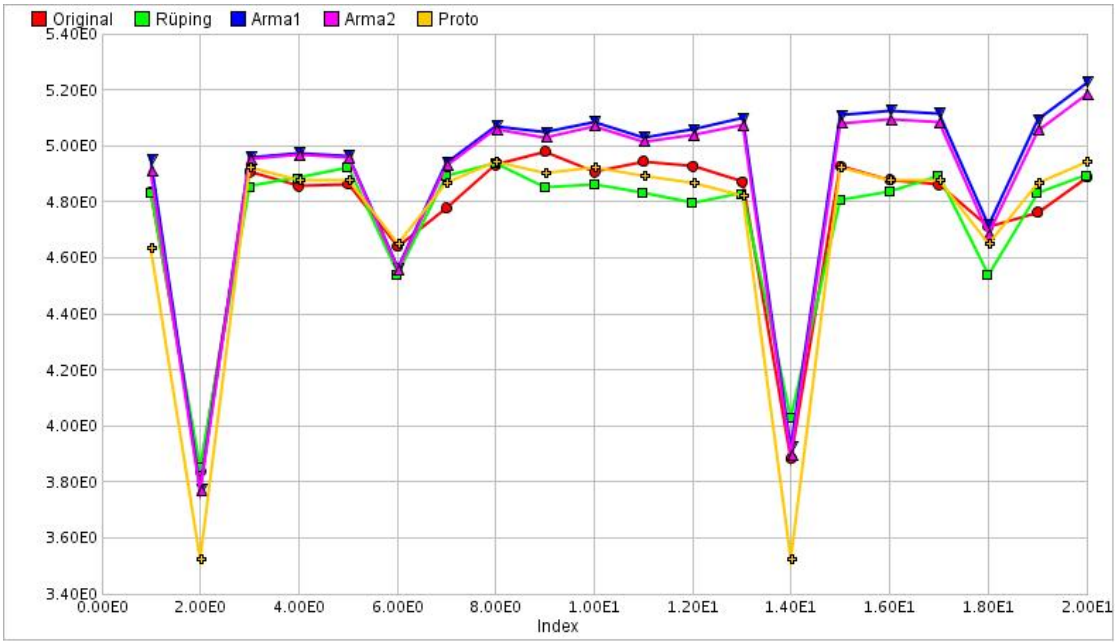


Abbildung 8.1.: Die Darstellung zeigt den modellierten Prognosebereich der getesteten Methoden im Vergleich zu den Testdaten.

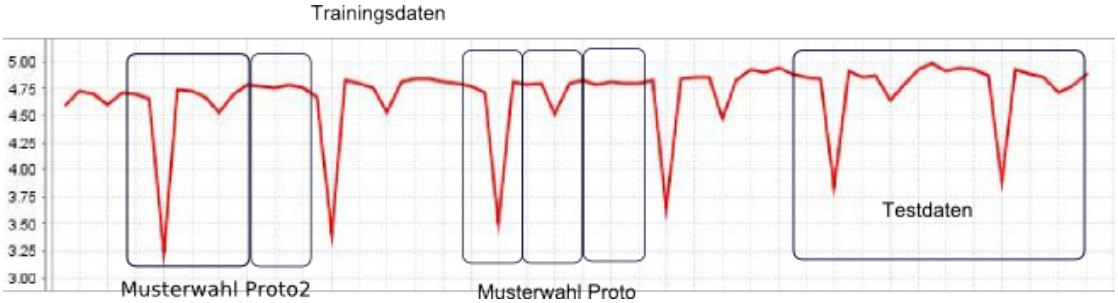


Abbildung 8.2.: Die Darstellung zeigt die Zeitreihe M1. Auf den Trainingsdaten ist exemplarisch die jeweilige Musterwahl für die Methode Proto visualisiert.

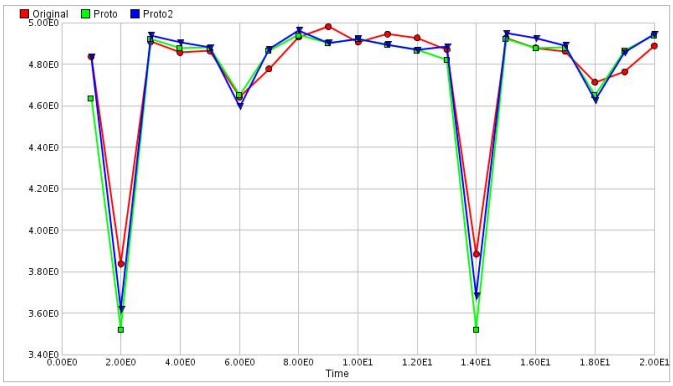


Abbildung 8.3.: Die Darstellung zeigt die Testdaten, Proto und Proto2 auf dem Prognoseintervall. Sie verdeutlicht die Veränderung in der Prognose, die durch die divergente Musterwahl in Experiment M1 erzielt werden konnte. Die Begründung für die alternative Musterwahl, bezeichnet als Proto2, ist die unzureichende Menge von Attributen pro Muster in Proto.

Rang	Methode	RMSE	AE	RE	NAE	RRSE	SE
1	Proto2	0.084	0.062 +/- 0.057	0.014 +/- 0.015	0.314	0.271	0.007 +/- 0.013
2	Rüping	0.087	0.071 +/- 0.050	0.015 +/- 0.011	0.361	0.280	0.008 +/- 0.008
3	Proto	0.125	0.077 +/- 0.099	0.018 +/- 0.025	0.390	0.402	0.016 +/- 0.035
4	Arma2	0.152	0.129 +/- 0.081	0.027 +/- 0.016	0.655	0.489	0.023 +/- 0.026
5	Arma1	0.172	0.146 +/- 0.092	0.030 +/- 0.019	0.742	0.555	0.030 +/- 0.033

Tabelle 8.1.: Die Tabelle zeigt die statistischen Kennzahlen der Versuchsreihe M1, die auf den Testdaten ermittelt wurden. Der Rang wurde absteigend nach dem gewählten Hauptkriterium (HK) ermittelt.

8.1.2. Versuchsreihe M2

Datenquelle Verkehrsaufkommen bei Dodgers-Spielen (Datensatz dodgers), 50400 Attribute, Sensorausfall 2800 Attribute (Attributwert =-1, Anzahl ermittelt mit Attribute value filter in RM). Entnommen aus [Hettich und Bay 2006].

Versuchsaufbau Trainingsdatensatz 80% (davon 10% Validierungsdatensatz für SVM) sowie Testdatensatz 20%. Die statistischen Werte wurden aufgrund des großen Horizontes von 10000 Attributen mittels Stichprobe ermittelt (Indexwerte der Stichprobe: 1,2,5,6,7,8,9,10,15,20,50,100,200,300,500,1000,2000,3000,5000).

Spezifische Methodenangaben Bei **Rüping**: JMySvmLearner mit Anova Kernel und einmaliger Parameteroptimierung (sigma, L, epsilon, C, convergence-epsilon, loss) sowie konstantem degree = 2, Fensterung n=250 (getestet 10,100,250,500). **Proto**: Zwei Muster mit jeweils optimierter (gamma, epsilon, p, C) LibSVM-epsilonSVR mit rbf Kernel (Musterwahl in Abbildung 8.4), Fenstergröße n=4 (getestet 2,3,4,6). **Arma1** und **Arma2** mittels automatischer Parameterwahl in TSW ermittelt.

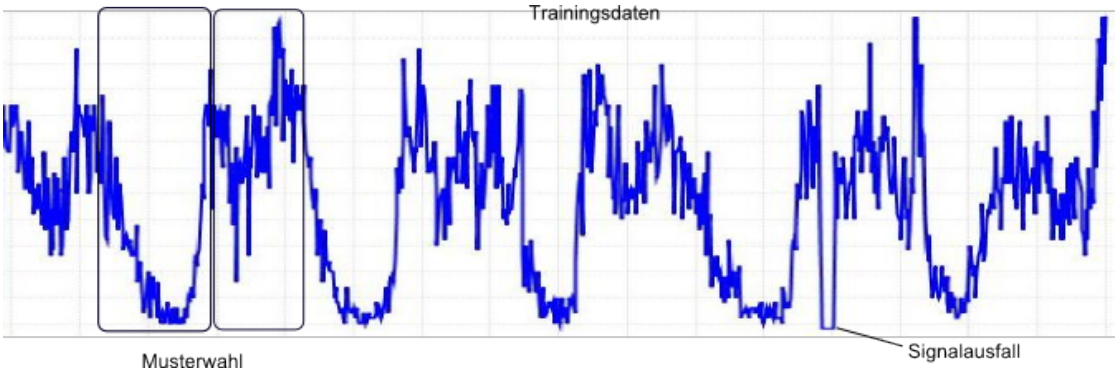


Abbildung 8.4.: Aufgrund der großen Datenmenge zeigt die Abbildung nur eine Auswahl der Zeitreihe von 1500 Attributen. In diesem Ausschnitt können die Musterwahl und die zugrunde liegende Struktur der Datenreihe M2 nachvollzogen werden.

Rang	Methode	RMSE	AE	RE	NAE	RRSE	SE
1	Proto	7.111	4.335 +/- 5.637	0.563 +/- 0.576	0.364	0.507	50.571 +/- 135.616
2	Rüping	15.321	14.397 +/- 5.239	3.569 +/- 3.052	1.210	1.093	234.722 +/- 138.692
3	Arma2	17.223	15.528 +/- 7.451	4.264 +/- 3.827	1.305	1.229	296.645 +/- 213.252
4	Arma1	20.123	18.373 +/- 8.208	5.060 +/- 4.455	1.544	1.435	404.936 +/- 274.014

Tabelle 8.2.: Die Tabelle zeigt die statistischen Kennzahlen der Versuchsreihe M2, die auf einer Stichprobe der Testdaten ermittelt wurden. Der Rang wurde absteigend nach dem gewählten Hauptkriterium (HK) ermittelt.

Abbildung 8.5.: Die Zeichnung verdeutlicht das Problem der Rüping-Methode auf dem Datensatz M2. Die Attribute stellen die Prognosewerte der Stichprobe dar und wurden gemäß ihrer Indexwerte aufsteigend visualisiert (Die Wertereihe Original ist die Prognosekurve der Stichprobe und kann nicht als Strukturelement in den Testdaten gefunden werden). Trotz verschiedener Horizonte, Schrittweiten und Fenstergrößen waren die Strukturen in einem umfangreichen Experiment nicht zu approximieren. Die Vorhersagewerte schwanken um den Mittelwert der Attribute, und Strukturelemente können nur unzureichend vorhergesagt werden. Ursächlich für das Resultat scheinen die große Datenmenge und die daraus bedingten Horizontgrößen der Vorhersage zu sein. Des Weiteren erlaubt die Variation der Musterlängen keinen geeigneten Ansatz für eine günstigere Fenstergröße.

8.1.3. Versuchsreihe M3

Datenquelle Weinverkäufe in Australien von 1980 - 1995, monatsweise Erhebung (Datensatz Sparklng). Entnommen aus [University 2006].

Versuchsaufbau Insgesamt 187 Attribute, davon 150 Attribute Trainingsdaten (inklusive Validierungsdaten) ≈ 80% und 37 Attribute Testdaten ≈ 20%.

Spezifische Methodenangaben Bei **Rüping**: JMySvmLearner mit Anova Kernel und einmaliger Parameteroptimierung (sigma, L, epsilon, C, convergence-epsilon, loss) sowie konstantem degree = 2, Fensterung n=14 (getestet 3,4,8,14,16). **Proto**: Ein Muster (Vergleiche Abbildung 8.6) mit optimierter (gamma, epsilon, p, C) LibSVM-epsilonSVR mit rbf Kernel, Objektsequenz entfällt. Trendfunktion mittels B4V.1 geschätzt und auf lineare Trendkomponente gerundet (+ 0.1x). **Arma1** und **Arma2** mittels automatischer Parameterwahl in TSW ermittelt.

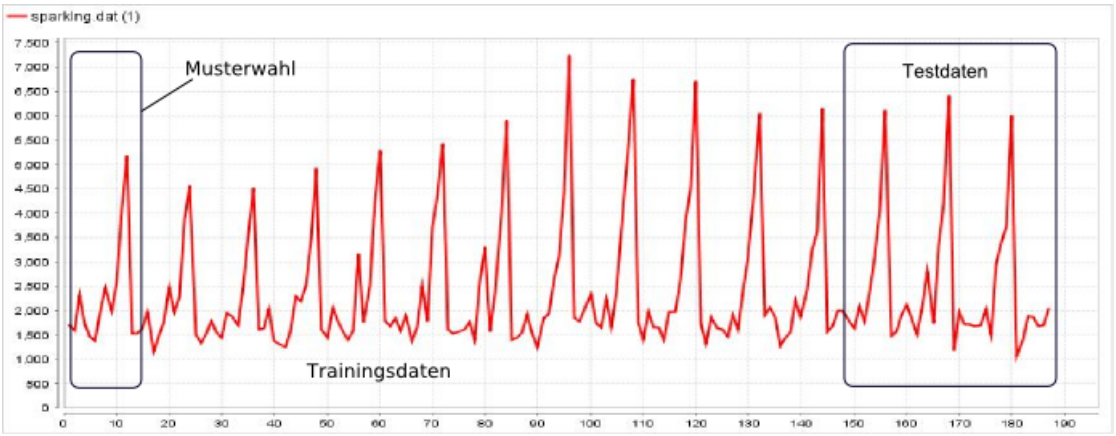


Abbildung 8.6.: Die Abbildung zeigt die Originalzeitreihe mit der Aufteilung in die angegebenen Trainings - und Testdaten. Die Zeitreihe beinhaltet ein Muster (mögliche Submuster weisen nicht genügend Attribute auf) mit jeweils 12 Attributen.

Rang	Methode	RMSE	AE	RE	NAE	RRSE	SE
1	Arma2	325.306	238.243 +/- 221.504	0.123 +/- 0.142	0.233	0.243	105,823.865
2	Proto	388.436	307.219 +/- 237.695	0.149 +/- 0.157	0.301	0.290	150,882.607
3	Rüping	643.686	473.599 +/- 435.931	0.224 +/- 0.222	0.464	0.480	414,331.184
4	Arma1	934.458	479.432 +/- 802.094	0.186 +/- 0.187	0.470	0.697	873,211.000

Tabelle 8.3.: Die Tabelle zeigt die statistischen Kennzahlen der Versuchsreihe M3, die auf den Testdaten ermittelt wurden. Der Rang wurde absteigend nach dem gewählten HK ermittelt.

8.1.4. Versuchsreihe M4

Datenquelle Weißweinverkäufe in Australien von Jan 1980 - Jul 1995, monatsweise Erhebung (Datensatz Drywhite). Entnommen aus [University 2006].

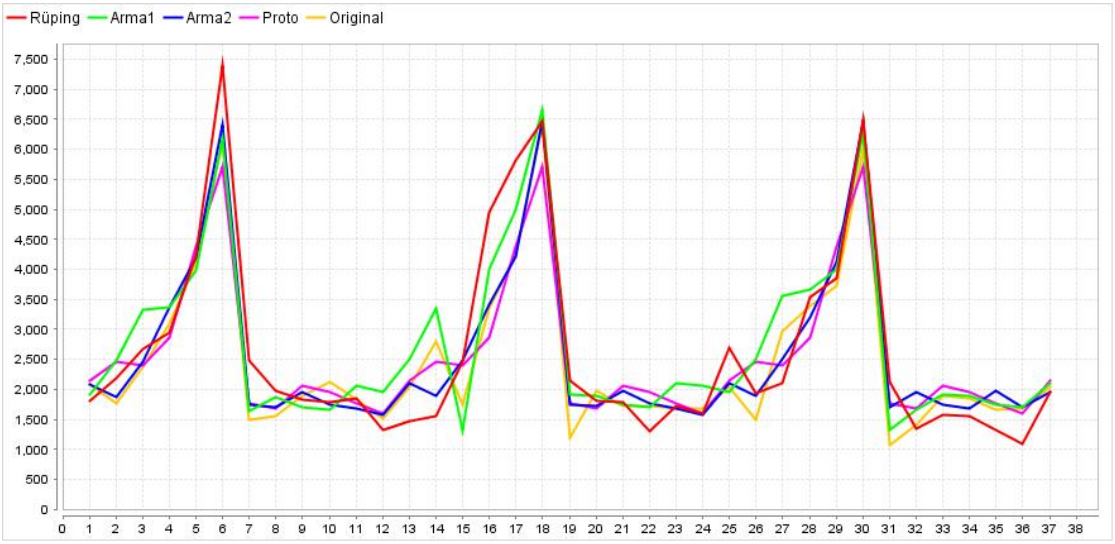


Abbildung 8.7.: Die Abbildung zeigt den modellierten Prognosebereich der einzelnen Methoden im Vergleich zu den Testdaten M3.

Versuchsaufbau Datenauswahl x_1-x_{170}, davon 150 Attribute Trainingsdaten (inklusive Validierungsdaten) $\approx$ 85% und 20 Attribute Testdaten $\approx$ 15%.

Spezifische Methodenangaben Bei **Rüping**: JMySvmLearner mit Anova Kernel und einmaliger Parameteroptimierung (sigma, L, epsilon, C, convergence-epsilon, loss) sowie konstantem degree = 2, Fensterung n=12 (getestet 3,4,8,12,14). **Proto**: Ein Muster (Vergleiche 8.8) mit optimierter (gamma, epsilon, p, C) LibSVM-epsilonSVR mit rbf Kernel, Objektsequenz entfällt. Zyklische Trendkomponente wurde mittels B4V.1 ermittelt. **Proto2** ist mittels des entwickelten MNN-Verfahrens erstellt worden. Für die ersten sieben Prognoseattribute konnte somit die Vorhersagegüte (Darstellung in 8.10) verändert werden. **Arma1** und **Arma2** mittels automatischer Parameterwahl in TSW ermittelt.

Rang	Methode	RMSE	AE	RE	NAE	RRSE	SE
1	Proto2	401.272	295.544 +/- 271.428	0.087 +/- 0.077	0.473	0.477	161,019.178
2	Arma2	413.441	278.750 +/- 305.339	0.079 +/- 0.078	0.446	0.491	170,933.450
3	Proto	444.869	348.213 +/- 276.869	0.101 +/- 0.080	0.558	0.529	197,908.503 5
4	Arma1	449.330	303.152 +/- 331.658	0.082 +/- 0.079	0.485	0.534	201,897.660
5	Rüping	688.069	558.748 +/- 401.546	0.153 +/- 0.103	0.895	0.817	473,438.542

Tabelle 8.4.: Die Tabelle zeigt die statistischen Kennzahlen der Versuchsreihe M4, die auf den Testdaten ermittelt wurden. Der Rang wurde absteigend nach dem gewählten HK ermittelt.

8.1.5. Versuchsreihe M5

Datenquelle Verkaufszahlen eines Produktes aus der Plastikindustrie, monatsweise Erhebung (Datensatz Splastics). Entnommen aus [University 2006].

Abbildung 8.8.: Die Abbildung zeigt die Zeitreihe des Experimentes M4. Es wurde ein Muster gewählt, das auf semantischer Ebene einem Jahreszyklus entspricht. Der Jahreszyklus besteht aus 12 kumulierten Erhebungen am jeweiligen Monatsende - das Muster wird durch 12 Attribute repräsentiert.

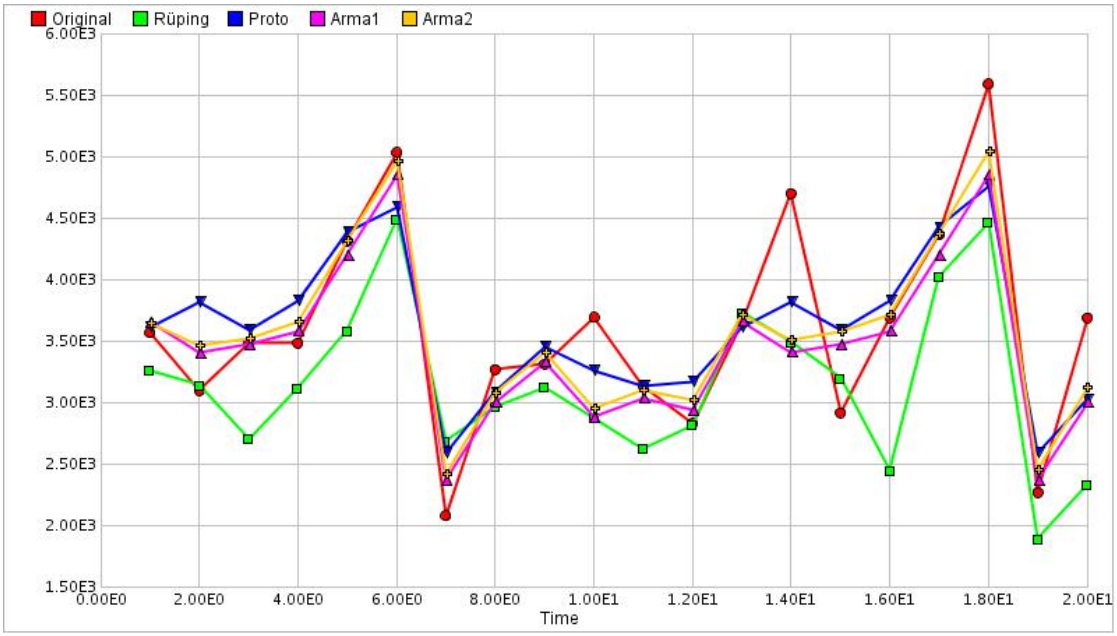


Abbildung 8.9.: Die Abbildung zeigt die Ergebnisse der getesteten Verfahren für den Prognosebereich von M4 an. Die Testdaten sind mit der Bezeichnung Original in der Abbildung verzeichnet.

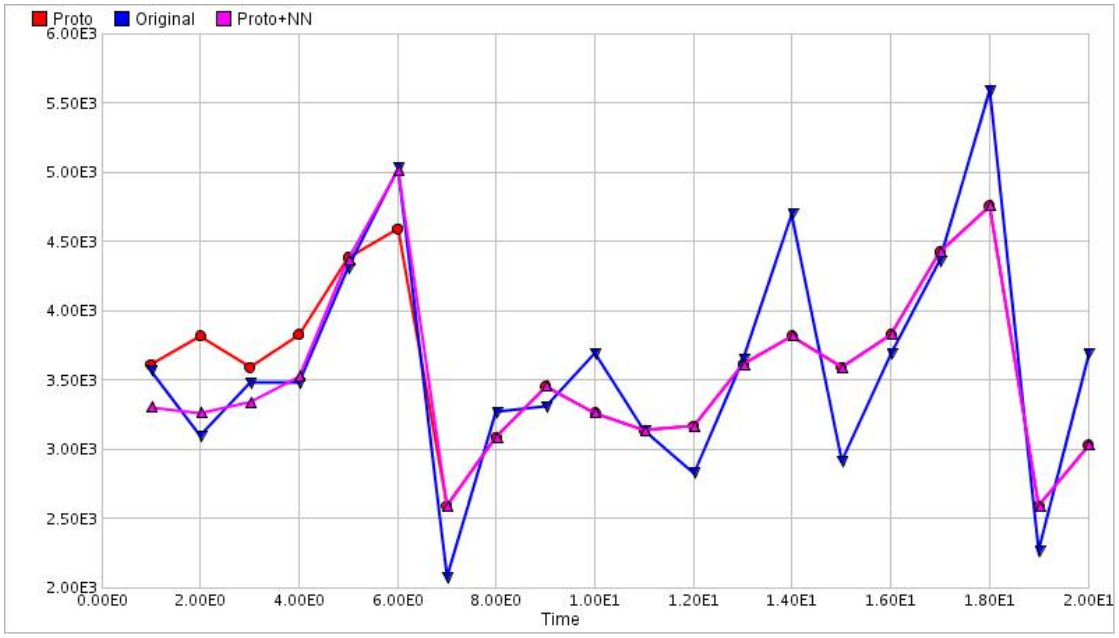


Abbildung 8.10.: Die Abbildung zeigt die Ergebnisse der Proto-Verfahren für das Prognoseintervall von M4. Proto kennzeichnet die Strukturprognose, Proto + NN stellt die Verfeinerung dar. In Proto + NN wurde die erste Instanz des Prognosehorizontes mittels der MNN und Balancefunktion vervollständigt. Die Kennzahlen sind unter der Bezeichnung Proto2 dokumentiert (Tabelle 8.4).

Versuchsaufbau Insgesamt 60 Attribute, davon Trainingsdatensatz 51 Attribute = 85% sowie Testdatensatz 9 Attribute = 15%.

Spezifische Methodenangaben Bei **Rüping**: JMySvmLearner mit Anova Kernel und einmaliger Parameteroptimierung (sigma, L, epsilon, C, convergence-epsilon, loss) sowie konstantem degree = 2, Fensterung n=13 (getestet 10,12,13,14). **Proto**: Ein Muster mit jeweils optimierter (gamma, epsilon, p, C) LibSVM-epsilonSVR mit rbf Kernel (Musterwahl in Abbildung 8.11), Objektsequenz entfällt. Trendkomponenten mittels B4V.1 ermittelt und auf lineare Trendfunktion approximiert (+ 0.13x). **Arma1** und **Arma2** mittels automatischer Parameterwahl in TSW ermittelt.

Abbildung 8.11.: Die Abbildung zeigt die Trainings- und Testdatenwahl des Experimentes M5. Zusätzlich wurde die Instanzwahl für das Muster des Verfahrens Proto exemplarisch markiert.

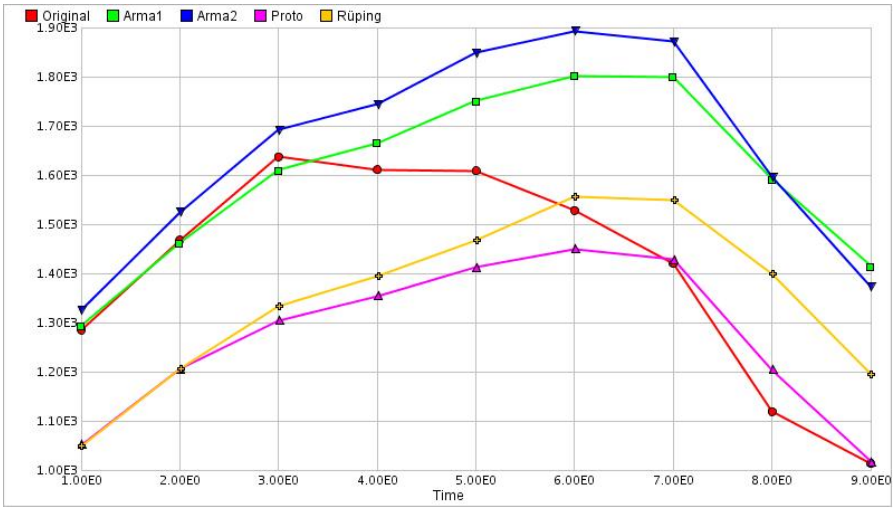


Abbildung 8.12.: Die Abbildung veranschaulicht die Ergebnisse der getesteten Verfahren für den Prognosebereich von M5. Die Testdaten sind mit der Bezeichnung Original in der Abbildung gekennzeichnet. In der graphischen Darstellung ist zu erkennen, daß das unstetige Trendverhalten für alle Methoden eine Schwierigkeit darstellt.

Rang	Methode	RMSE	AE	RE	NAE	RRSE	SE
1	Proto	197.957	161.988 +/- 113.784	0.109 +/- 0.073	0.897	0.932	39,186.899
2	Rüping	213.954	197.491 +/- 82.302	0.145 +/- 0.066	1.094	1.007	45,776.421
3	Arma1	264.041	196.222 +/- 176.676	0.157 +/- 0.158	1.087	1.242	69,717.556
4	Arma2	293.675	241.860 +/- 166.580	0.186 +/- 0.144	1.339	1.382	86,245.054

Tabelle 8.5.: Die Tabelle zeigt die statistischen Kennzahlen der Versuchsreihe M5, die auf den Testdaten ermittelt wurden. Der Rang wurde absteigend nach dem gewählten HK ermittelt.

8.2. Verfahrensbewertung

Methode	M1	M2	M3	M4	M5	$\sum$M
Proto	1	1	2	1	1	6
Rüping	2	2	3	4	2	13
Arma2	3	3	1	2	4	13
Arma1	4	4	4	3	3	18

Tabelle 8.6.: Die Kennzahlen der Tabelle entsprechen dem ermittelten Ranking der Versuchsreihen M. Die Spalte $\sum$M ist die Aufsummierung der Rankingwerte.

Abschließend soll eine Verfahrensbewertung für die Methode Proto erarbeitet werden, die Stärken und Schwächen aufzeigt. Neben den bereits gewonnenen Erkenntnissen fließen Rückschlüsse aus der Versuchsreihe M in die Bewertung ein.

Im Rahmen dieser Arbeit wurden Voraussetzungen an die zugrunde liegende Zeitreihe gestellt, um so die Qualität der entwickelten Verfahren zu garantieren. Mit der Definition Muster als lokales Modell wurde die Anwendungsmöglichkeit zugunsten der Stärken der SVM eingeschränkt. Dies ist wünschenswert, da somit das Feld der Anwendungsmöglichkeiten klar definiert wurde. Abgrenzungen haben gezeigt, daß das Verletzen der Grundvoraussetzungen zu unbefriedigenden Ergebnissen führt (Vergleiche hierzu Anhang B.1). Es bleibt jedoch festzuhalten, daß die Methoden aus diesen Gründen keine allgemeingültigen Prognosemöglichkeiten von Zeitreihen sind. Sind die Bedingungen jedoch erfüllt, erzielen wir mit dem Prototyp gute Ergebnisse (Tabelle 8.6). Eine wesentliche Beobachtung aus der Serie M ist, daß die Methode Proto auf keinem Datensatz eine der unteren Rankingpositionen belegt hat. Dies findet darin seine Begründung, daß unter den oben angegebenen Voraussetzungen die Zeitreihe immer strukturell zu erfassen war. Das Risiko eines methodischen Versagens der Strukturprognose kann auf einem geeignetem Datensatz (Anforderungskriterien in 4) somit ausgeschlossen werden.

In Experiment M2 konnte gezeigt werden, daß das Zusammenfassen von Strukturen zu Mustern innerhalb der Methode Proto vorteilhaft ist, denn lokale Modelle unterschiedlicher Indexlänge können in einem Verfahrensschritt bearbeitet werden. Das schlechtere Abschneiden der Methode Rüping kann auf dem Experiment M2 damit erklärt werden, daß unterschiedliche Musterlängen mit konstanter Fenstergröße schwer zu erfassen sind und daher immer eine mögliche Fehlerquelle darstellen. Da wir auf Mustern arbeiten,

können wir auf konstante Fenstergrößen mit Benutzerparametern zur Strukturerfassung verzichten. Im Rahmen der Versuchsreihe M wurde auch der Horizont der Prognose variiert, um zu testen, ob eine der Methoden vorteilhaft für bestimmte Prognosebereiche ist. Hier hat sich jedoch herausgestellt, daß die Methode mit den günstigsten Werten unabhängig von der Horizontgröße das Ranking anführte. Geringfügige Verbesserungen konnten für statistische Kennzahlen der Methode Proto ermittelt werden, indem die Trainingsdaten so gewählt wurden, daß die kurzfristige Prognose über das MNN modelliert wurde. Da entweder der Testbereich sehr groß (M2) oder die Muster sehr klein waren, fielen die Veränderungen oftmals nur marginal aus. Weil in den Versuchsreihen M1 und M4 die Ergebnisse ähnliche Kennzahlen aufwiesen, konnte in diesem Fall jedoch mit der Verfeinerung über MNN eine verbesserte Position in der Rangordnung erzielt werden. Möglichst exakte Werte für kurze Prognosen zu ermitteln, erscheint jedoch nicht nur aufgrund statistischer Vergleiche sinnvoll. Vielmehr sind in der Realität oftmals Aufgabenstellungen zu finden, bei denen gerade die Prognose der unmittelbaren Zukunft gefragt ist. Mit der Balancefunktion wurde eine Möglichkeit vorgestellt, das Konvergenzverhalten der prognostizierten Instanz anzupassen. Hierfür kann benutzerorientiert die gewählte Funktion mit einmaligem Aufwand durch eine datenangepaßte Funktion ersetzt werden (6.5.1).

Die Modellierung der Zeitreihe ist ohne Aufwand auszuführen, da lediglich die Prototypen gemäß der Objektsequenz konkateniert werden müssen. Im ungünstigen Fall muß der gesamte Wertebereich der Prognose um den Trend ergänzt werden. Eine explizite Berechnung einzelner Horizonte oder erneute Modifikationen entfallen nach Berechnung der Prototypen und der Objektsequenz vollständig. Somit können ohne erneuten Aufwand beliebig lange Sequenzen modelliert werden. Im Hinblick auf optionalen praktischen Einsatz der Methode ist es jedoch ratsam, die Trainingsdaten nach einer Zeit durch neugewonnene Daten zu ergänzen und anschließend die Prototypen sowie die Objektsequenz erneut zu berechnen. Die einfache manuelle Modellierungsmöglichkeit ist ein Vorteil gegenüber der Rüping-Methode. Jeder zu prognostizierende Wert mußte in dieser Methode über ein eigenes Modell ermittelt werden. Die rekursive Eingabe, d.h. die Verwendung nur eines Modells, führte auf den Testdatensätzen der M-Serie zu einer niedrigeren Positionierung innerhalb des Rankings. In der Versuchsreihe M1 wurde zusätzlich eine Parameteroptimierung für jedes erstellte SVM-Modell der Rüping-Methode ermittelt. Dieser Vorgang hat die Durchführung der Experimente jedoch zeitlich unhandlich gestaltet. Da die Parameterabweichungen bei variierendem Horizont (Label) nur gering waren, wurde in den nachfolgenden Experimenten nur noch die Parameteroptimierung für das Modell mit Horizontgröße=1 durchgeführt. Diese Parametereinstellungen wurden für die SVM in den anschließenden Modellen (pro Prognosewert ein Modell) übernommen. Aus der Erfahrung der M-Serie läßt sich folgende These herleiten:

These 8.2.1. *Die Methode von Rüping ist um ein vielfaches langsamer, wenn für jeden Prognosewert in der Rüping-Methode ein eigenes Modell erstellt werden muß.*

Diese These läßt sich wie folgt rechtfertigen: Der Einsatz der SVM selbst stellt den zeitintensivsten Schritt in den SVM-basierenden Verfahren dar. Bei der Methode Proto ist die Häufigkeit des SVM-Einsatzes durch die Anzahl der Muster und bei der Rüping-Methode durch den Prognosehorizont gegeben. Die Anzahl der verschiedenen Muster ist

im Durchschnitt wesentlich geringer als die Anzahl der zu postulierenden Werte. Folglich ist unter den Bedingungen der These die Methode Proto im Bezug auf den zeitlichen Aspekt zu bevorzugen.

Die entwickelten Verfahren lassen sich problemlos mit bestehenden Methoden kombinieren, so daß im Rahmen dieser Arbeit die Prognose vollständig durchgeführt werden konnte. Große Bereiche wie das Datenpreprocessing mußten geprüft werden, doch bereits bestehende Methoden (z.B. Trendelimination) konnten zur Vorverarbeitung genutzt werden. Sollten Daten spezielles Preprocessing benötigen oder weitere Verfahren in Zukunft die bereits bestehenden ablösen (dies impliziert nur strukturerhaltende Vorverarbeitungsschritte), so können die Methoden weiterhin Anwendung finden.

Es konnte gezeigt werden, daß das Preprocessing-Problem der Fenstergröße auch auf die hier entwickelte Verfahrensweise große Auswirkung hat. Es wird (wie in anderen Arbeiten zuvor) auch hier nur eine problemspezifische (i.A. für spezielle Zeitreihen) Größe angegeben. Dieser Punkt ist in der Realmodellierung als Schwachstelle zu erachten. Auch wenn die Approximation der lokalen Modelle erfolgreich ist, so werden statistische Kennzahlen auf Validierungsdaten unbefriedigende Ergebnisse liefern, wenn die Abfolge der Muster fehlerhaft ist. Auch die praktische Relevanz einer fehlerhaften Objektsequenz ist leicht ersichtlich: Selbst wenn die Musterapproximation von großer Genauigkeit ist, stellt das Modellieren des falschen Musters keinen Nutzen dar. Möchte ein Unternehmen beispielsweise die Kennzahlen für Eisverkäufe im nächsten Monat prognostizieren und dies an der Zeitreihe der ermittelten Kundenzahlen festmachen, dann könnte es vielleicht den Sommer sehr genau durch ein Regressionsmodell darstellen, würde aber in der Praxis versagen, wenn es den nächsten Monat mit den Werten für einen kalten Herbst vorhersagt - es hätte eine falsche Objektsequenz gewählt. In Experimenten hat das entwickelte Modell über einzeln ermittelte NN-Modelle gute Ergebnisse geliefert, das theoretische Bestehen des Problems erzwingt jedoch den expliziten Hinweis.

Verschiedene Einsatzgebiete der Methodik gestatten das Lösen unterschiedlicher Aufgaben. Tabelle 8.7 gibt eine Zusammenfassung der Einsatzgebiete und der dafür notwendigen Methodenschritte an. Prototyp ist hier die Abkürzung für die Strukturprognose und MNN (Modifiziertes Nearest Neighbour) steht für die kurzfristige Prognosetechnik.

Aufgabe	Prototyp	MNN	Vorverarbeitung	Objektsequenz
kurzfristige Prognose	Ja	Ja	Ja	Ja
langfristige Prognose	Ja	Nein	Ja	Ja
Vorhersage Einzelattribute	Ja	Teilweise	Ja	Ja
Zeitreihenrekonstruktion	Ja	Teilweise	Ja	Ja
Rauschelimination	Ja	Nein	Ja	Nein

Tabelle 8.7.: Die Tabelle gibt einen Überblick über die Aufgaben und die bestehenden Lösungsmöglichkeiten. Des Weiteren wird aufgezeigt, ob die Vorverarbeitung sowie das Modellieren der Objektsequenz als Teil der Aufgabenlösung benötigt werden. Die Bezeichnung 'Teilweise' gibt an, daß das Verfahren nur unter bestimmten Bedingungen Anwendung finden kann.

9. Ausblick und Erweiterungen

In diesem Kapitel werden die Ergebnisse und Methoden der Arbeit kurz zusammengefasst. Abschließend wird ein Ausblick auf mögliche Erweiterungen in Form einer Ideensammlung gegeben.

9.1. Zusammenfassung

Der Kern der vorliegenden Arbeit wird von den entwickelten Prognoseverfahren gebildet. Um den Kern ausarbeiten zu können, wurden zuerst verwendete Begrifflichkeiten formalisiert, und das Muster wurde als ein mögliches lokales Modell der Zeitreihe definiert. Die Prognosetechniken wurden mittels einer Abgrenzung zu praxiserprobten Methoden entwickelt. Hierbei hat sich die SVM als geeignetes Regressionsmodell für die zuvor identifizierten Muster herausgestellt. Im Kapitel der Vorverarbeitung wurde gezeigt, daß die SVM nicht einfach auf unbearbeitete Daten angewendet werden kann. Es wurde deutlich, daß sowohl die Normalisierung der Daten als auch die Trendelimination ausschlaggebend für die Ergebnisse der Regression sind. Nachdem die SVM und ihre verfahrensbezogenen Parameter erläutert wurden, wurde gezeigt, wie das Regressionsmodell (bezeichnet als Prototyp) gewonnen werden kann. Aus dem Verfahren ergab sich die anschließende Frage, wie die Prototypen als Folge modelliert werden können. Dieses Problem wurde als Klassifizierungsaufgabe aufgefaßt und mittels Nearest Neighbour gelöst. Anschließend wurde die Zeitreihe erneut geprüft und aufgezeigt, daß ungenutztes Wissen in Form der unvollständigen Instanz vorlag. Über die Betrachtung der Globalität und Lokalität erschloß sich die Menge der korrespondierenden Instanzen als Suchraum. Ein Reihe von Abgrenzungen und Versuchen führte zu einem modifizierten Nearest Neighbour-Ansatz mit euklidischer Distanz, um die Instanzen zu vergleichen. Um der Globalität gerecht zu werden, wurde die Methode um die Balancefunktion erweitert, die die Punktemenge der unvollständigen Instanz als Kriterium nutzt. An geeigneter Stelle wurde in der Arbeit auf Experimente verwiesen, die in einem eigenen Kapitel anschaulich dokumentiert sind. Diverse statistische Definitionen und Ergebnisse weiterführender Experimente, deren Aussagekraft durch die aufgeführten statistischen Werte gegeben ist, können dem Anhang entnommen werden.

9.2. Erweiterungen und Ideensammlung

Nachfolgend sollen drei ausgewählte Erweiterungen und Ideen aufgeführt werden. Sie dienen als Vervollständigung der vorliegenden Arbeit und als Gedankenanregung für zukünftige Arbeiten.

9.2.1. Univariat - Multivariat

Die Methodik wurde auf univariaten Datensätzen durchgeführt. Hierfür wurden multivariate Datensätze merkmalsweise gesplittet, so daß nur univariate Eingaben verarbeitet wurden. Der inverse Schritt ist durchführbar, da die Klasse der Kernfunktion bezüglich Addition und Multiplikation abgeschlossen ist [Schölkopf und Smola 2002]. So können gesplittete Datensätze über Addition bzw. Multiplikation in ihre ursprüngliche multivariate Repräsentation überführt werden. Eine Verarbeitung multivariater Daten in einem Schritt ist jedoch über Multilabel-Eigenschaften von SVM-Implementierungen möglich. Da dies nicht die Ergebnisse, sondern nur die Handhabung vereinfacht, wurde eine Umsetzung nicht im Rahmen dieser Arbeit verwirklicht. Für eine arbeitsintensive Anwendung der Methodik ist es jedoch ratsam.

9.2.2. Automatische Mustersuche

Da wir auf Strukturen angewiesen sind (lokale Muster), wurde die Instanzauswahl in dieser Arbeit manuell durchgeführt. Diese Arbeit setzte sich ebenfalls mit der automatischen Extraktion lokaler Modelle auseinander, da dieser Arbeitsschritt in zukünftigen Anwendungen gegebenenfalls zu automatisieren ist. Ein Verfahren, das den hier verwendeten Musterbegriff unterstüzt, wird in [Geurts 2001] vorgestellt. Die umfangreichen Experimente und Beschäftigungen mit der Thematik des Musters haben zur Erkenntnis geführt, daß Muster über Strukturen zu identifizieren sind. Betrachtet man den Begriff Struktur in Zeitreihen, so läßt sich folgende These aufstellen:

These 9.2.1. *Instanzen als sich wiederholende Strukturen in Zeitreihen müssen mindestens ein Extremum aufweisen, um als Muster identifiziert zu werden.*

Jedes Muster, das als solches identifiziert wurde, wies lokale Extrema auf. Versuche, auf strukturbedingte Muster zu verzichten und abstrakte Musterklassen auf Datenebene wiederzufinden, waren oftmals zum Scheitern verurteilt (Beispieldokumentation im Anhang B). Daher scheint es vielversprechend, Algorithmen zur lokalen Extremasuche auf Zeitreihen zu betrachten. Im Rahmen der Arbeiten von [Gandhi 2003] und [Pratt und Fink 2002] wurden Verfahren zur Extremaextraktion in Zeitreihen vorgestellt. Ein solches Extremum ist ein Merkmal der Instanz. Ist ein Merkmal oder ein Merkmalsvektor, der Muster beschreibt, zu finden, so müßte eine Zeitreihe automatisch nach diesen Merkmalen durchsucht werden, um Instanzen zu identifizieren. Die Ausprägungen der Merkmale wäre das Unterscheidungskriterium der Muster. Weitere Ansätze könnten andere Merkmale zur Musteridentifikation sein (automatische Merkmalsextraktion). Hierbei sind gegebenenfalls Raumtransformationen vorzunehmen, um zu prüfen, ob andere Räume geeignete Merkmale zur Musteridentifikation bieten. Eine umfassende Methodenübersicht zur automatischen Merkmalsextraktion aus Zeitreihen wird in [Mierswa 2004] gegeben. Die meisten Systeme basieren auf dem Ansatz, daß die Musterklassen als Trainingsmenge dem System zu Beginn übergeben werden müssen, d.h. das Ziel muß bekannt sein. Für die hier vorgestellten Verfahren bedeutet das, daß jedoch nur noch eine kleine Menge manuell selektiert werden muß. Die restlichen Instanzen könnten über die Mustersuche automatisch identifiziert werden. Für spezielle Zeitreihen gibt es dazu bereits Umsetzungen. Im Rahmen dieser Arbeit wurden EEG Reihen zu

Demonstrationszwecken analysiert. In [Arnold 2003] wird ein implementiertes System aus dem medizinischen Feld vorgestellt, das die Muster (in der EEG Analyse sind die Klassen vorgegeben und werden mit dem Überbegriff Graphoelemente bezeichnet) auffinden kann. Es darf also erhofft werden, daß der Schritt der manuellen Instanzauswahl Potential zur Automatisierung aufweist.

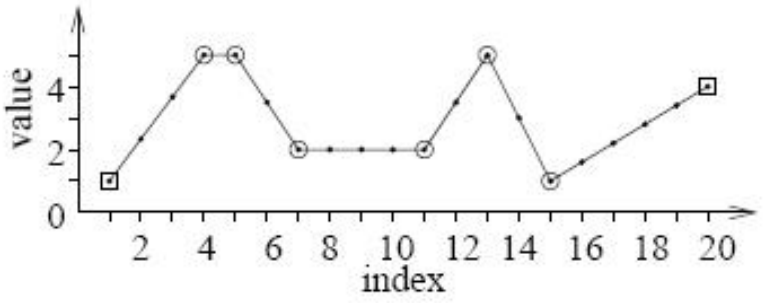


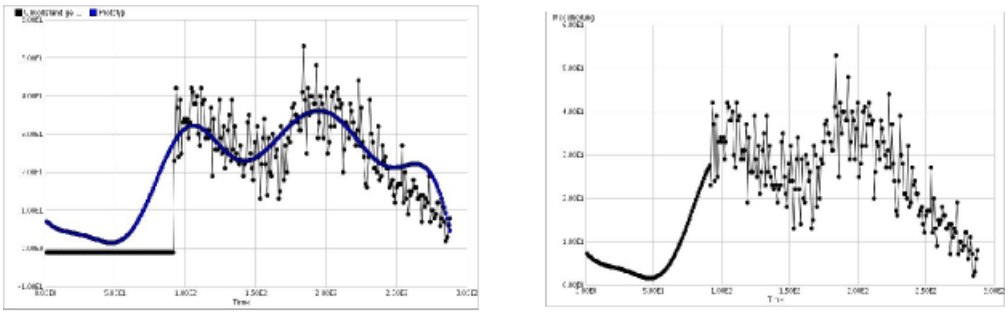
Abbildung 9.1.: Die Abbildung ist der Arbeit von [Gandhi 2003] entnommen und verdeutlicht den Zusammenhang zwischen den Extrema einer Zeitreihe und der intuitiven Mustererkennung.

9.2.3. Erweiterte Anwendungsmöglichkeiten

Der Schwerpunkt der in dieser Arbeit entwickelten Verfahren war die Prognose unbekannter, zukünftiger Werte. Im Rahmen der Arbeit und der intensiven Beschäftigung mit Zeitreihen sind weitere Aufgaben aufgetreten, die sich mittels der hier entwickelten Technik lösen lassen. Zum einen die Aufgabe der Zeitreihenrekonstruktion : Sind Zeitreihen unvollständig (beispielsweise durch den Ausfall von Sensoren), so können die fehlenden Attribute mittels der Methodik modelliert werden. Der nicht erfaßte Indexbereich kann als Prognosehorizont aufgefaßt und mit den bestehenden Methoden modelliert werden. Sind einige Attribute vorhanden (ist der Sensor z.B. erst zur Mitte eines Tages ausgefallen), so kann die kurzfristige Prognosetechnik über die nächste Instanz Anwendung finden. Fehlen vielleicht sogar größere Spannen innerhalb der Aufzeichnung, können die fehlenden Muster mittels Prototyp und Sequenzmodellierung rekonstruiert werden. Im nachfolgenden Beispiel soll eine Anwendung aus dem praktischen Arbeitsteil vorgestellt werden, die sich mittels der entwickelten Methoden rekonstruieren ließ.

Beispiel 9.1 (Signalausfall während kontinuierlicher Messungen)**.** *Das Beispiel ist dem Dodgers-Datensatz aus [*Hettich *und* Bay *2006] entnommen. Während der halbjährlichen Messung traten Sensorausfälle auf. Das Beispiel zeigt eine Instanz, die das Verkehrsaufkommen auf den Stadionstraßen an einem Spieltag der Dodgers repräsentiert. Die ersten Stunden des Tages fiel der Sensor aus, was auf Datenebene durch den Wert* -1 *gekennzeichnet wurde.*

Eine weitere Möglichkeit ist die der Rauschelimination. Bei einer strukturaufweisenden Zeitreihe können die lokalen Muster (wie in dieser Arbeit vorgestellt) mittels SVM-Regression modelliert werden. Im Schritt der Regression darf erhofft werden, daß Rauschen in Form von Ausreißern nicht als Strukturelement in den Prototyp aufgenommen



(a) (b)

Abbildung 9.2.: Die Abbildung (a) zeigt die Originalinstanz, sowie den Prototyp des zugehörigen Musters. In (b) wurden die nichtvorhandenen Werte durch die Funktionswerte des Prototypen remodelliert.

wird. Ersetzt man nach der Prozedur die Zeitreihe durch die Prototypsequenz, so erhalten wir eine rauscheliminierte Zeitreihe. Ausgehend von der Annahme, daß das Regressionsmodell die lokalen Modelle ausreichend gut approximiert, läßt sich folgende These aufstellen:

These 9.2.2. *Der Prototyp, der auf den verrauschten Daten ermittelt wurde, sollte eine möglichst große Übereinstimmung mit dem Prototyp, der auf den Originaldaten gewonnen wurde, aufweisen. Stimmen beide überein, so würden die Ausreißer, die auf graphischer Ebene das Rauschen signalisieren, im Regressionsmodell nicht zum Tragen kommen. Die gewünschte Störgrößenelimination wäre erzielt worden.*

Der Ansatz für die statistische Belegung durch den Prototypvergleich ergab sich in Abgrenzung gegen Beispielexperimente (Vergleiche B.7 und B.8 im Anhang). Unter praktischen Bedingungen existiert der Prototyp, der auf den Originaldaten gewonnen wurde (OP), jedoch nicht. Es ist somit unbekannt, wie gut der ermittelte Prototyp (RP) sich auf einer konkreten Zeitreihe dem OP annähert.

Eine besondere Beobachtung konnte durch das Testen von einzelnen Instanzen sowie geringen Teilmengen der gesamten Instanzmenge und deren Überdeckung durch den Prototyp gemacht werden. War die zugrunde liegende Approximationsfähigkeit des OP bedingt durch komplexe Instanzen schlechter, so waren in der Remodellierung durch den RP statistische Kennzahlen nicht aussagekräftig. Bei guter Überdeckung durch den OP waren die RP-Kennzahlen hingegen ungünstiger. Um die Aussagefähigkeit des RP zu beurteilen, ist wegen eben genannter Beobachtung immer gegen die komplette Instanzmenge der Originalzeitreihe getestet worden.

Doch für die SVM-Regression ist es eine grundlegende Problematik, Rauschen in Form von Datenausreißern von musterinternen Strukturelementen zu unterscheiden (**??**). Das sich hier ergebende Einsatzgebiet zielt auf echtes Rauschen ab und nicht etwa auf Pseudo-Rauschen, das quasi-periodisch ist. Unter dieser Bedingung darf erwartet werden, daß die Ausreißer sich an verschiedenen Strukturabschnitten der einzelnen Instanzen zeigen. In der Folge würde die gemeinsame Struktur der Instanzen den Prototyp definieren und

vereinzelt abweichende Attribute würden von nicht nennenswerter Auswirkung sein. Versuche haben gezeigt, daß bei einer ausreichenden Menge an Instanzen, die für die Bildung der Musterwolke zur Verfügung stehen, der Prototyp nur marginal verändert wird. Signalausfälle können aufgrund ihrer Natur als starkes, partielles Rauschen aufgefaßt werden. Da die Rauschelimination unter konstruierten Testbedingungen untersucht wurde, gibt Experiment 7.3.1 die grundlegende Vorgehensweise an und zeigt die Umsetzung in RM.

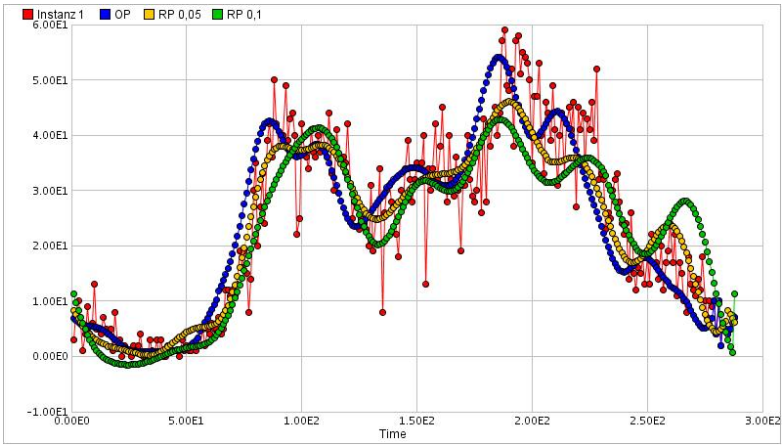


Abbildung 9.3.: Die Abbildung zeigt den OP sowie zwei RPs im Vergleich (50% Trainingsdaten auf D Muster Dodgers-Datensatz [Hettich und Bay 2006]). Zur besseren Veranschaulichung wurde eine Beispielinstanz aus der Originalzeitreihe unterlegt.

9.3. Schlußwort

Abschließend kann festgestellt werden, daß mit den hier entwickelten Methoden sowie dem Anwenden von bestehenden Methoden ein vollständiges Verfahren zur Handhabung von Zeitreihen entwickelt wurde, dessen Stärke in der langfristigen Prognose großer Datenmengen sowie deren einfacher Modellierung liegt. Die Musterdefinition, die an die intuitive menschliche Wahrnehmung angelehnt ist, macht die Analyse der Zeitreihe für den Benutzer zugänglich. Sowohl das verwendete Regressionsmodell (Anhang B.5) als auch der gewählte Kern (Anhang B.4) und letztendlich das Gesamtverfahren selbst (Abschnitt 8.1) konnten sich gegen eine Auswahl bestehender Verfahren behaupten. Die Ideensammlung soll einen Eindruck des Erweiterungspotentials vermitteln, so daß zukünftige Arbeiten in diesem Gebiet vielversprechend erscheinen.

A. Anhang

Einige Begriffe aus der Statistik, die in den Experimenten und dem Anhang verwendet, aber nicht definiert werden, werden hier erklärt. Für eine über die Beschreibung hinausgehende Betrachtung sei als Einstieg auf [Armstrong und Collopy 1992] verwiesen. Die nachfolgenden Definitionen sind in Anlehnung an [Ludwig Fahrmeir und Tutz 2003] entstanden.

Mean Squared Error Die mittlere quadratische Abweichung (MSE) gibt an, welche Abweichung zwischen Schätzfunktion T und dem wahren Wert θ für die Schätzfunktion zu erwarten ist (Erwartungstreue). Die quadratische Abweichung selbst wird mit **Squared Error** (SE) bezeichnet. Abweichungen zwischen dem beobachteten y-Wert und dem prognostizierten y-Wert nennt man **Residuen** $\widehat{\epsilon}_i$ für $i = 1..n$. Im Vergleich von Schätzfunktionen ist die mit kleinerem MSE-Wert zu bevorzugen (sie ist MSE-wirksamer).

$$MSE = E\left(|T - \theta|^2\right)$$

Root Mean Squared Error Der Root Mean Squared Error (RMSE) ist die Wurzel aus dem Mean Squared Error und besitzt die gleiche Einheit wie die Punkte der Ordinate. Über den RMSE Wert (in Verbindung mit dem AE Wert) können Regressionsmodelle miteinander verglichen werden [Cao und Tay 2003].

$$RMSE = \sqrt{E\left(|T - \theta|^2\right)}$$

Absolute Error Wird der Messfehler einer Größe in der Einheit der Messgröße angegeben, so spricht man vom absoluten Fehler (AE). Der absolute Fehler zweier Messgrößen a und b ist

$$AE = |b - a|\,.$$

Wenn der AE der Normalisierung unterzogen wird, so spricht man vom **Normalized Absolute Error** (NAE).

Relative Error Wird der Messfehler relativ zur Messgröße angegeben, spricht man vom relativen Fehler (RE). Der relative Fehler zweier Messgrößen a und b ist

$$RE = \frac{|b - a|}{|a|}.$$

Root Relative Squared Error Der Root Relative Squared Error (RRSE) ist relativ zu dem, was ein einfaches Vorhersagemodell prognostiziert hätte. Als einfaches Vorhersagemodell wird der Mittelwert der Werte gesetzt. Sei P_{ij} der Prognosewert des

Modells i für das Beispiel j für $j = 1...n$ und T_j der wahre Wert des Beispiels j, dann ist der RRSE gegeben durch

$$E_i = \sqrt{\frac{\sum_{j=1}^{n} (P_{ij} - T_j)^2}{\sum_{j=1}^{n} (T_j - \bar{T})^2}} \text{ mit } \bar{T} = \frac{1}{n} \sum_{j=1}^{n} T_j.$$

Das Vorhersagemodell ist ideal, wenn $E_i = 0$ ist.

Korrelation Zwei normalverteilte metrische Merkmale sind genau dann unabhängig, wenn sie auch unkorreliert sind. Testverfahren hierzu bauen naturgemäß auf dem empirischen Korrelationskoeffizienten auf. Der empirische Korrelationskoeffizient für zwei Merkmale X und Y ist gegeben durch

$$r_{XY} = \frac{\sum_{i=1}^{n} (X_i - \bar{X}) (Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n} (X_i - \bar{X})^2 \sum_{i=1}^{n} (Y_i - \bar{Y})^2}}.$$

Je weiter der postulierte Wert r_{XY} von 0 entfernt ist, je unkorrelierter sind die beteiligten Variablen.

Datenaufteilung Die Daten werden geteilt in Trainingsdaten, Testdaten und Validierungsdaten. Die **Trainingsdaten** dienen zum Erlernen der Zielfunktion - in dieser Arbeit der Prototyp. Die **Validierungsdaten** dienen zum Überprüfen der erlernten Zielfunktion, um z.B. Overfitting entgegenzuwirken. Die **Testdaten** dienen zur Bewertung der Zielfunktion nach statistischen Kennzahlen. Trainingsdaten und Testdaten dürfen keine Schnittmenge aufweisen, da die Bewertung der Zielfunktion ansonsten unglaubwürdig ist.

B. Anhang

In diesem Teil des Anhangs sind verschiedenartige Resultate aus Experimenten aufgeführt. Sie sind im Rahmen der Arbeit referenziert worden und werden hier nach den zugrunde liegenden Datensätzen (ohne interne Ordnungsstruktur) aufgeführt.

Datensatz Robot Execution Failures

Datenquelle Robot Execution Failures, LP3 Dataset, Merkmal FZ [Hettich und Bay 2006]. Vorgegebene Musterklasse vier (OK, Slightly Moved, Moved, Lost), 47 Instanzen, 15 Attribute pro Instanz.

Muster	Anzahl Instanzen	C	p	epsilon
OK	19	50	0,05	0,02
Moved	16	500	0,01	0,1
Slightly Moved	10	250	0,005	0,5
Lost	3	500	0,05	0,1

Tabelle B.1.: Die Tabelle zeigt die Ergebnisse der Parameteroptimierung, die für das Regressionsmodell an die SVM übergeben werden. Parameter gamma ist optimiert, aber nicht angegeben.

Muster	RMSE	AE	RE	NAE
OK	4.833 +/- 1.283	3.902 +/- 1.126	0.050 +/- 0.014	1.017 +/- 0.024
Moved	17.227 +/- 21.943	10.172 +/- 9.846	0.209 +/- 0.368	0.956 +/- 0.159
Slightly Moved	6.699 +/- 4.548	4.821 +/- 3.062	0.067 +/- 0.062	1.047 +/- 0.102
Lost	15.174 +/- 8.852	12.504 +/- 7.158	0.244 +/- 0.276	1.253 +/- 0.182

Tabelle B.2.: Die Performancevektoren der einzelnen Muster sind mittels 10facher Kreuzvalidierung ermittelt worden.

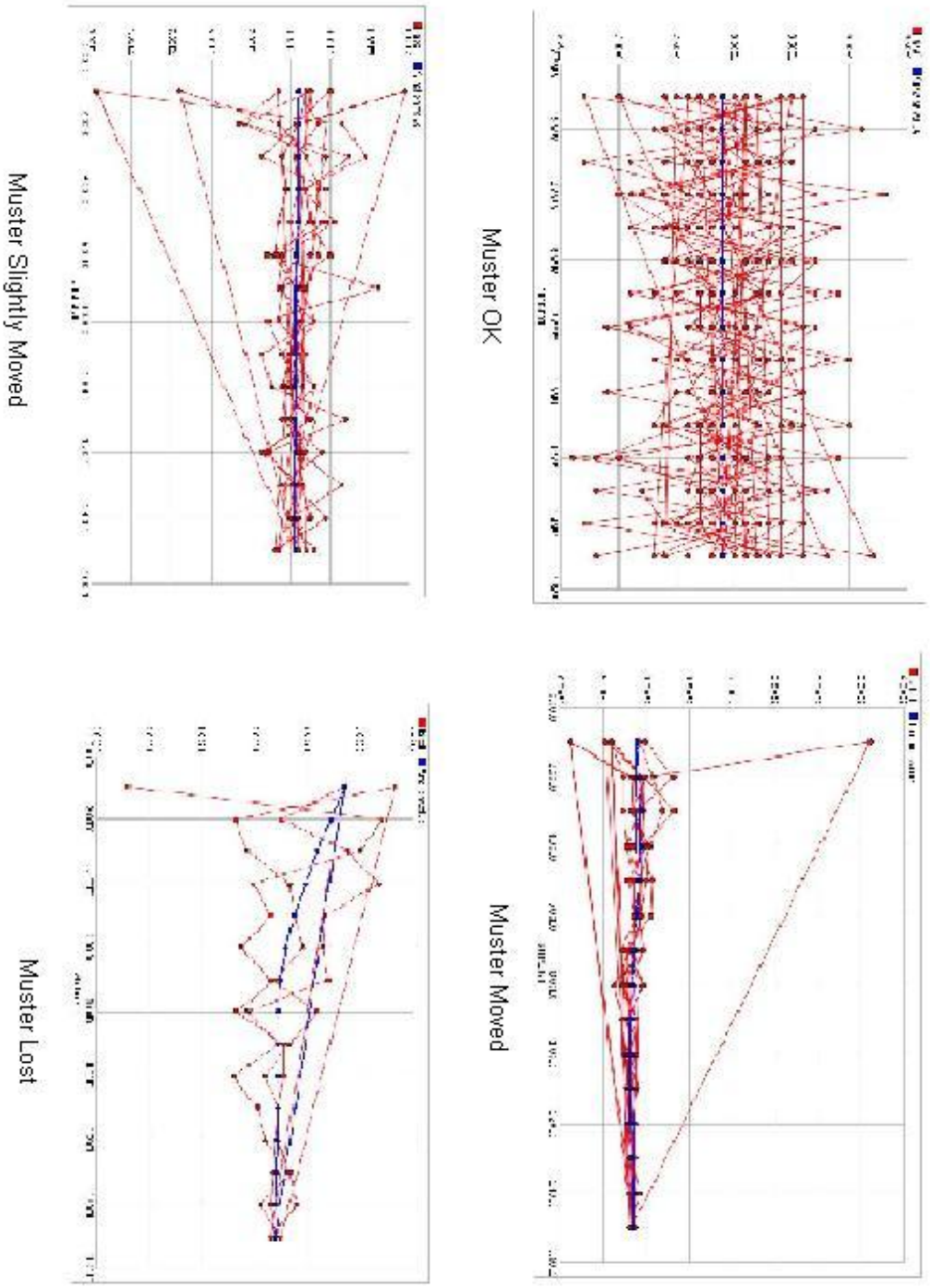


Abbildung B.1.: Der Versuch, Regressionsmodelle ohne manuelle Musterauswahl mittels SVM zu erlernen, war nicht vielversprechend. Einzelne physikalische Werte, die bei konkreten Ereignissen auftreten, weisen aufgrund der komplexen Konstruktion keine Musterstruktur auf. Somit ist kein Prototyp erlernbar, der die einzelnen Werteentwicklungen der Merkmale korrekt approximiert.

Datensatz Dodgers

Datenquelle Dodgers, 175 Instanzen mit jeweils 288 Attributen, Sensorausfall 2800 Attribute (Attributwert =-1, Anzahl ermittelt mit Attribute value filter in RM), 2 Muster: Dodgers-Spiel (81 Instanzen), Kein Spiel (94 Instanzen) [Hettich und Bay 2006]. Wenn nicht anders angegeben, Trainingsdatensatz 90% (davon 10% Validierungsdatensatz) sowie Testdatensatz 10%.

Instanzen	c	p	epsilon	RMSE	AE	NAE
10	500	0,1	0,1	8.208 +/- 0.710	6.045 +/- 0.586	0.527 +/- 0.042
20	500	0,3	0,03	8.299 +/- 0.693	6.035 +/- 0.428	0.541 +/- 0.032
40	500	0,8	0,1	8.293 +/- 0.553	5.977 +/- 0.392	0.530 +/- 0.044
60	500	0,8	0,5	8.473 +/- 0.250	6.123 +/- 0.189	0.534 +/- 0.018
70	500	0,8	0,5	8.432 +/- 0.358	6.103 +/- 0.248	0.532 +/- 0.022

Tabelle B.3.: Die Tabelle zeigt die nur marginalen Veränderungen des Perfomancevektors bei variierender Beispielmenge auf der Musterwolke Dodgers-Spiel (gamma trainiert, aber nicht angegeben).

D-Rang	Kernel	RMSE	AE	NAE	RRSE
1	RBF	8.216 +/- 0.208	6.042 +/- 0.115	0.525 +/- 0.018	0.604 +/- 0.021
2	Linear	10.733 +/- 4.433	8.646 +/- 3.457	0.698 +/- 0.173	0.752 +/- 0.231
3	Sigmoid	14.766 +/- 0.623	12.422 +/- 0.623	1.015 +/- 0.005	1.017 +/- 0.006
4	Poly	366.084 +/- 61.726	258.832 +/- 62.590	21.218 +/- 5.727	22.321 +/- 1.244
N-Rang	Kernel	RMSE	AE	NAE	RRSE
1	RBF	8.147 +/- 0.762	5.646 +/- 0.410	0.505 +/- 0.031	0.628 +/- 0.049
2	Linear	12.600 +/- 1.769	0.088 +/- 1.255	0.876 +/- 0.016	0.959 +/- 0.055
3	Poly	284.763 +/- 149.777	193.363 +/- 106.072	16.786 +/- 9.193	21.753 +/- 11.389
4	Sigmoid	1,947.553 +/- 335.091	903.992 +/- 131.991	80.858 +/- 9.184	49.931 +/- 22.614

Tabelle B.4.: Die Tabelle zeigt die Performancevektoren der Klassen D und N. D-Rang und N-Rang geben die Rankingpositionen gemäß des Hauptkriteriums RMSE wieder. Getestet wurden die verschiedenen Kernel des implementierten LibSVMLearners/epsilonSVR . Alle Kernelparameter wurden optimiert, um vergleichbare Resultate zu garantieren. Aufgrund der SVM-Trainingslaufzeiten wurde eine Stichprobe von 30 Instanzen ausgewählt.

Regression	RMSE	AE	NAE
LibSVML - epsilonSVR	8.473 +/- 0.250	6.123 +/- 0.189	0.534 +/- 0.018
Lineare Regression	11.582 +/- 0.312	9.437 +/- 0.294	0.825 +/- 0.022
EVO SVM	179.353 +/- 3.589	154.048 +/- 4.944	13.707 +/- 0.973
LibSVML - nuSVR	8.200 +/- 0.745	6.033 +/- 0.584	0.528 +/- 0.060

Tabelle B.5.: Die Tabelle zeigt den Vergleich zwischen einem SVM - Prototypen (siehe Kernelranking B.4) und ausgewählten weiteren Regressionsmodellen. Alle Modelle wurden optimiert (Grid + 10fache Kreuzvalidierung) und auf den gleichen Eingaben (Dodgers-Spiele, bereinigt von Sensorausfällen) angewendet, um vergleichbare Ergebnisse zu erzeugen. Tabellarisch wurden nur drei der insgesamt acht Vergleichsparameter dargestellt, da der RMSE als Leitkriterium anzusehen ist.

Fenstergröße	n=4	n=6	n=8
Accuracy A	52,9	58,8	64,7
Accuracy B	43,5	44,9	45,7

Tabelle B.6.: A: Training 90% = 158 Instanzen. Testdaten 10 %= 17 Instanzen. B: Training 80% = 147 Instanzen. Testdaten 20% = 34 Instanzen. Accuracy gibt den Prozentsatz der korrekt vorhergesagten Objekte an. Trotz unterschiedlicher Fenstergrößen enden alle Eingaben im gleichen Zustand und prognostizieren ab wenigen Schritten (Abhängig von n) nur noch die Klasse N (Kein Dogders-Spiel). Je mehr Daten zu prognostizieren sind, umso mehr nähern sich die Accuracy-Werte an. Sie konvergieren für $+\infty$ gegen das prozentuale Auftreten des Musters N (nur N wird korrekt vorhergesagt).

Rauschpegel	RMSE	AE	NAE	RRSE
0,00	8.440 +/- 0.000	6.286 +/- 5.705	0.594	0.622
0,05	8.333 +/- 0.000	6.036 +/- 5.745	0.525	0.615
0,1	8.342 +/- 0.000	6.030 +/- 5.435	0.525	0.617
0,2	8.370 +/- 0.000	6.026 +/- 5.809	0.524	0.618

Tabelle B.7.: Tabelle für variierende Rauschpegel. Test: Prototypen auf Originalinstanz $\mathfrak{m}_1$ (Erstes Dodgers-Spiel). Die erste Instanz wird durch den OP nicht ausreichend approximiert. Dieser Test bestätigt die Vermutung, daß genau in diesem Fall der RP im Test gegen die Instanz vergleichbare Werte zum OP erzielt - unabhängig vom Rauschpegel. Durch Zufall ist sogar eine Verbesserung der statistischen Vergleichswerte möglich (Erklärung: Es könnte genau diese Instanz durch den rauschverzerrten Prototyp exakter approximiert werden).

Rauschpegel	RMSE	AE	NAE	RRSE
0,00	8.400 +/- 0.000	6.256 +/- 5.605	0.544	0.620
0,05	8.436 +/- 0.000	6.123 +/- 5.803	0.533	0.623
0,1	8.743 +/- 0.000	6.398 +/- 5.959	0.557	0.645
0,2	8.763 +/- 0.000	6.554 +/- 5.816	0.570	0.647
0,6	9.764 +/- 0.000	7.852 +/- 5.804	0.683	0.721
1,0	11.101 +/- 0.000	9.113 +/- 6.338	0.793	0.819

Tabelle B.8.: Tabelle für variierende Rauschpegel. In diesem Experiment wurden 100% des Musters D mit Rauschen belegt (Trainingsdaten) und der RP erlernt. 100% der Originalinstanzen dienen als Testmenge des RPs, da nun allgemeingültige Aussagen von Interesse sind. Die statistischen Kennzahlen verdeutlichen, daß der RP die Beispielinstanz mit zunehmendem Rauschpegel schlechter approximiert.

Literaturverzeichnis

[Allen 1983] Allen, James F. (1983). *Maintaining knowledge about temporal intervals*. Commun. ACM, 26(11):832–843.

[Andel 1984] Andel, J. (1984). *Statistische Analyse von Zeitreihen*. Akademie Verlag Berlin, 1 Aufl.

[Armstrong und Collopy 1992] Armstrong, J. und F. Collopy (1992). *Error measures for generalizing about forecasting methods - empirical comparisons*. International Journal of Forecasting 8 (1).

[Arnold 2003] Arnold, Thomas (2003). *Computergestützte Befundung klinischer Elektroenzeophalogramme, MPI Series in Cognitive Neuroscience*. Technischer Bericht 38, Max-Planck-Institut für neuropsychologische Forschung, Leipzig.

[Azoff und Azoff 1994] Azoff, E. Michael und E. M. Azoff (1994). *Neural Network Time Series Forecasting of Financial Markets*. John Wiley and Sons, Inc., New York, NY, USA.

[Bronstein und Semendjajew 1996] Bronstein, I.N. und K. Semendjajew (1996). *Taschenbuch der Mathematik*. B.G. Teubner Stuttgart-Leipzig.

[Burges 1998] Burges, Christopher J. C. (1998). *A Tutorial on Support Vector Machines for Pattern Recognition*. Data Mining and Knowledge Discovery, 2(2):121–167.

[Cao und Tay 2003] Cao, L.J. und F. Tay (2003). *Support vector machine with adaptive parameters in financial time series forecasting*. In: *Neural Networks, IEEE Transactions on*, S. 1506 – 1518. 6 Aufl.

[Chih-Wei Hsu und Lin 2007] Chih-Wei Hsu, Chih-Chung Chang und C.-J. Lin (2007). *A Practical Guide to Support Vector Classification*. Department of computer science.

[Das et al. 1997] Das, Gautam, D. Gunopulos und H. Mannila (1997). *Finding Similar Time Series*. In: *Principles of Data Mining and Knowledge Discovery*, S. 88–100.

[Donner 2007] Donner, Reik, Hrsg. (2007). *Lineare und Nichtlineare Dynamische Modelle*, Technische Universität Dresden. Chair of Transportation Econometrics and Traffic Modelling.

[E. Keogh 2006] E. Keogh, X. Xi, L. Wei & Ratanamahatana (2006). *The UCR Time Series Classification/Clustering Homepage*.

[Faraway und Chatfield 1995] Faraway, J. und C. Chatfield (1995). *Time series forecasting with neural networks: A case study*.

[Gandhi 2003] Gandhi, H. (2003). *Important extrema of time series: Theory and applications*.

[Gautam Das und Smyth 1998] Gautam Das, King-Ip Lin, Heikki Mannila Gopal Renganathan und P. Smyth (1998). *Rule Discovery from Time Series*. In: *Knowledge Discovery and Data Mining*, S. 16–22.

[Geurts 2001] Geurts, Pierre (2001). *Pattern Extraction for Time Series Classification*. Lecture Notes in Computer Science, 2168:115–127.

[Graf und Borer 2001] Graf, A.B.A. und S. Borer (2001). *Normalization in Support Vector Machines*. In Pattern Recognition (DAGM), LNCS 2191. Springer Verlag.

[Han et al. 1999] Han, J., G. Dong und Y. Yin (1999). *Efficient Mining of Partial Periodic Patterns in Time Series Database*. In: *Fifteenth International Conference on Data Engineering*, S. 106–115, Sydney, Australia. IEEE Computer Society.

[Hand 2002] Hand, David J. (2002). *Pattern Detection and Discovery.*. In: *Pattern Detection and Discovery*, Bd. 2447 d. Reihe *Lecture Notes in Computer Science*, S. 1–12. Springer.

[Hettich und Bay 2006] Hettich, S. und S. D. Bay (2006). *The UCI KDD Archive [http://kdd.ics.uci.edu]*. Irvine, CA: University of California, Department of Information and Computer Science.

[Höppner 2002] Höppner, Frank (2002). *Lernen lokaler Zusammenhänge in multivariaten Zeitreihen*. DFG-Projekt Kl 648/1. University of Applied Sciences Braunschweig/Wolfenbüttel, 1 Aufl.

[Keogh und Pazzani 1998] Keogh, Eamonn und M. Pazzani (1998). *An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback*. In: Agrawal, R., P. Stolorz und G. Piatetsky-Shapiro, Hrsg.: *Fourth International Conference on Knowledge Discovery and Data Mining (KDD'98)*, S. 239–241, New York City, NY. ACM Press.

[Ludwig Fahrmeir und Tutz 2003] Ludwig Fahrmeir, Rita Kunstler, Iris Pigeot und G. Tutz (2003). *Statistik*. Springer Verlag, 4 Aufl.

[Mannila und Toivonen 1996] Mannila, Heikki und H. Toivonen (1996). *Discovering Generalized Episodes Using Minimal Occurrences*. In: *Knowledge Discovery and Data Mining*, S. 146–151.

[Mierswa 2004] Mierswa, Ingo (2004). *Automatisierte Merkmalsextraktion aus Audiodaten*. Diplomarbeit, Fachbereich Informatik, Universität Dortmund.

[Mierswa et al. 2006] Mierswa, Ingo, M. Wurst, R. Klinkenberg, M. Scholz und T. Euler (2006). *A 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-06)*.

[Morik 2000] Morik, Katharina (2000). *Einführung in die Künstliche Intelligenz*. Hrsg. G.Görtz,C.Rollinger,J.Schneeberger.

[Morik 2007] Morik, Katharina, Hrsg. (2007). *Wissensentdeckung in Datenbanken - Data Cube*. Universität Dortmund.

[Morik und Köpcke 2005] Morik, Katharina und H. Köpcke (2005). *Features for Learning Local Patterns in Time-Stamped Data*. In: *local pattern detection*, S. 98–114. Springer Verlag.

[Niemann 2006] Niemann, H., Hrsg. (2006). *Klassifikation von Mustern*, Universität Erlangen-Nürnberg. Lehrstuhl für Mustererkennung. (2. überarbeitete und erweiterte Auflage).

[Pawlenko 2005] Pawlenko, Andrej (2005). *Plant Asset Management unterstützt durch empirische Datenanalyse*. Diplomarbeit, Universität Dortmund.

[Petersohn 2005] Petersohn, Helge (2005). *Data Mining: Verfahren, Prozesse, Anwendungsarchitektur*. Oldenbourg Wissenschaftsverlag, 1 Aufl.

[Pollock 2002] Pollock, D. S. G. (2002). *A review of TSW: the Windows version of the TRAMO-SEATS program*. Journal of Applied Econometrics, (vol. 17, issue 3):pages 291–299.

[Pratt 2001] Pratt, K. (2001). *Locating patterns in discrete time series*. Diplomarbeit, Computer Science and Engineering, University of South Florida.

[Pratt und Fink 2002] Pratt, Kevin B. und E. Fink (2002). *Search for Patterns in Compressed Time Series*. International Journal of Image and Graphics, 2(1):89–106.

[Rüping 1999] Rüping, Stefan (1999). *Zeitreihenprognose fur Warenwirtschaftssysteme unter Berücksichtigung asymmetrischer Kostenfunktionen*. Diplomarbeit, Universität Dortmund.

[Rüping 2001] Rüping, Stefan (2001). *SVM Kernels for Time Series Analysis*. In: Klinkenberg, Ralf, S. Rüping, A. Fick, N. Henze, C. Herzog, R. Molitor und O. Schröder, Hrsg.: *LLWA 01 - Tagungsband der GI-Workshop-Woche Lernen - Lehren - Wissen - Adaptivitat*, S. 43–50.

[Rüping und Morik 2003] Rüping, Stefan und K. Morik (2003). *Support Vector Machines and Learning about Time*. In: *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'03)*.

[Schölkopf und Smola 2002] Schölkopf, Bernd und A. Smola (2002). *Learning with Kernels - Support Vector Machines, Regularization, Optimization and Beyond*.

[Singh 1999] Singh, S. (1999). *Noise Impact on Time-Series Forecasting using an Intelligent Pattern MatchingTechnique*.

[Smola 1996] Smola, A. (1996). *Regression Estimation with Support Vector Learning Machines*. Technischer Bericht, Technische Universität München.

[Smola und Schölkopf 1998] Smola, Alexander und B. Schölkopf (1998). *A tutorial on support vector regression*. NeuroCOLT2 Technical Report NC2-TR-1998-030.

[Sven F. Crone und Weber 2006] Sven F. Crone, Jose Guarjado und R. Weber (2006). *The impact of preprocessing on support vector regression and neuronal networks in time series prediction*. CSREA2006.

[TU Clausthal 2007] TU Clausthal, Institut für Mathematik (2007). *http://www.stochastik.tu-clausthal.de/*.

[University 2006] University, Monash (2006). *Time Series Data Liberary, http://www-personal.buseco.monash.edu.au/ hyndman/TSDL/*.

[Vapnik 1998] Vapnik, Vladimir N. (1998). *Statistical Learning Theory*. Wiley Interscience.

[Wegener 1999] Wegener, Ingo (1999). *Theoretische Informatik - eine algorithmenorientierte Einführung*. B.G. Teubner Verlag, 2 Aufl.

[Weiss 1999] Weiss, Gary M. (1999). *Timeweaver: a Genetic Algorithm for Identifying Predictive Patterns in Sequences of Events*. In: Banzhaf, Wolfgang, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela und R. E. Smith, Hrsg.: *Proceedings of the Genetic and Evolutionary Computation Conference*, Bd. 1, S. 718–725. Morgan Kaufmann.

[Wiesbaden 2004] Wiesbaden, Statistisches Bundesamt, Hrsg. (2004). *Time series analysis*. http://www.destatis.de.

[Yang und Zou 2002] Yang, Y. und H. Zou (2002). *Combining time series models for forecasting*.

Abkürzungsverzeichnis

AE	Absolute Error
ANN	Artificial Neural Network
AR	Autoregression
ARMA	Autoregression Moving Average
ED	Euclidean Distance
EEG	Elektroenzephalogramm
FI	Frequent Itemset
FP	Frequent Pattern
GM	Globales Modell
HK	Hauptkriterium
LGS	Lineares Gleichungssystem
LM	Lokales Modell
MLP	Multilayer Perceptron Netz
MNN	Modifiziertes Nearest Neighbour
MSE	Mean Squared Error
NAE	Normalized Absolute Error
NN	Nearest Neighbour
OP	Prototyp der Originalzeitreihe
PG	Prototyp Globalität
RBF	Radial Basis Function Kernel
RE	Relative Error
RM	Rapid Miner
RMSE	Root Mean Squared Error
RP	Prototyp der Zeitreihe mit Rauschpegel
SVM	Support Vector Machine

Erklärung

Hiermit erkläre ich, Anne Antonia Scheidler, die vorliegende Diplomarbeit mit dem Titel *Zeitreihenprognose mittels lokaler Modelle und ihrer Globalisierung* selbständig verfasst und keine anderen als die hier angegebenen Hilfsmittel verwendet, sowie Zitate kenntlich gemacht zu haben.

Dortmund, 28. April 2008