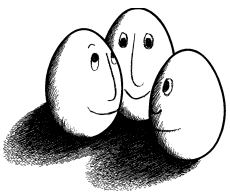
Diplomarbeit

Preis- und Trendvorhersagen auf Energiemarktdaten

Oliver Heering



Diplomarbeit
Fakultät Informatik
Technische Universität Dortmund

Dortmund, 4. September 2009

Betreuer:

Prof. Dr. Katharina Morik
Dipl.-Inform. Marco Stolpe

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

ACF	Autocorrelation Function
ACF	Inverse Autocorrelation Function
AE	Absolute Error
AIC	Akaike's Information Criterion
AICC	Akaike's Information Corrected Criterion
ANOVA	Analysis of Variance
AR	Autoregressive
ARIMA	Autoregressive, Integrated Moving Average
ARMA	Autoregressive, Moving Average
CLS	Conditional Least Sum of Squares
DWT	Discrete Wavelet Transform
EEX	European Energy Exchange
FFT	Fast Fourier Transform
FWT	Fast Wavelet Transform
HQ	Hannan-Quinn Criterion
IID	Independent, Identically Distributed
MA	Moving Average
ML	Maximum Likelihood
MRA	Multiresolution Analysis
MSE	Mean Squared Error
OLS	Ordinary Least Squares
OTC	Over the Counter
PACF	Partial Autocorrelation Function
Phelix	Physical Electricity Index
QMF	Quadrature Mirror Filter
RMSE	Root Mean Squared Error
SBC	Schwarz-Bayes Criterion
STFT	Short-Time Fourier Transform
SVM	Stützvektormethode (Support Vector Machine)
ULS	Unconditional Least Sum of Squares
YW	Yule-Walker

"Die Wahrheit liegt im Verborgenen. Wir müssen sie zu schätzen lernen."
(Joseph Honerkamp)

1 Einleitung

Seit es Börsenkurse gibt ist es ein Wunsch der Menschen, diese vorhersagen zu können. Eine zeitlich geordnete Reihe von Signalen, wie es Börsenkurse sind, nennt man allgemein *Zeitreihe*. Zeitreihen liefern uns eine Einsicht in die Vergangenheit eines Prozesses. Da aber die meisten Zeitreihen nicht völlig zufällig sind, sondern eine gewisse Struktur enthalten, die von vielen verschiedenen, manchmal auch nur wenigen Faktoren abhängen kann, erhofft man sich, mit Hilfe der Vergangenheit eines Prozesses Aussagen über seine Zukunft treffen zu können. Sowohl in der Statistik, wie später auch im Bereich des maschinellen Lernens und der künstlichen Intelligenz wurden zahlreiche Verfahren entwickelt, die aus gesammelten Daten Modelle berechnen, mit denen solche Prognosen möglich wurden.

Da Prognosen bei realen Daten nie hundertprozentig exakt sind, versucht man, durch immer bessere Methoden die Performanz der Vorhersagen zu verbessern, den Fehler also zu minimieren. Allerdings werden nicht ständig komplett neue Algorithmen entworfen. Vielmehr wird der Vorverarbeitung der Daten große Aufmerksamkeit geschenkt. Es lassen sich eine Vielzahl unterschiedlichster Merkmale, *Features* genannt, schon vor dem eigentlichen Lernalgorithmus aus den Daten extrahieren. Diese Merkmale sind teilweise wesentlich aussagekräftiger als die Rohdaten und nicht selten auch platzsparender. Eine andere Möglichkeit, Verfahren zu verbessern, besteht darin, sie zu modularisieren und Teile ihrer Berechnungen durch dem Problem besser angepasste auszutauschen. Ein Beispiel hierfür stellt die *Stützvektormethode* (SVM) dar, bei der der Kern der Berechnungen, in seiner ursprünglichen Form ein Skalarprodukt, ausgetauscht werden kann. So lässt sich dasselbe Lernverfahren in einem anderen, meist höherdimensionalen Raum mit geeigneteren Eigenschaften durchführen.

Diese Arbeit beschäftigt sich mit der Analyse und Prognose von Strompreisen des Day-Ahead Auktionshandels. Diese entsprechen Börsenkursen herkömmlicher Aktienbörsen, jedoch wird an Energiemärkten mit Produkten gehandelt, die mit Energieerzeugung zu tun haben (Strom, Kohle, Öl, Gas, Emissionsrechte). Beim Handel mit diesen Erzeugnissen erfolgt die Lieferung verständlicherweise nicht unmittelbar, weshalb als Lieferzeit stets zukünftige Zeiträume betrachtet werden. So bieten Käufer beispielsweise auf eine Stromlieferung einer einzelnen Stunde des Folgetags im sogenannten *Day-Ahead Auktionshandel* oder auf eine Lieferung Öl für das gesamte Jahr 2010 am sogenannten *Terminmarkt*. Näheres dazu in Kapitel 2.

Damit ein Lernverfahren gute Ergebnisse erzielen kann, ist eine ausreichende Menge an historischen Daten nötig. Die Tatsache, dass Energiebörsen wie die *European Energy Exchange* [EEX] erst seit wenigen Jahren (verglichen mit Aktienbörsen) existieren, hat zur Folge, dass die Historie an Daten längst nicht so umfangreich ist wie zum Beispiel an

Wertpapierbörsen. An der EEX wurde beispielsweise erst 2007 ein Spotmarkt für Gas eingerichtet. Hinzu kommt, dass die meisten Zeitreihen, welche von der EEX bezogen werden können, nur in täglicher Auflösung vorliegen, pro Tag also lediglich ein Wert zur Verfügung steht. Die längste Historie weist die stündlich aufgelöste Zeitreihe des Strom Day-Ahead Auktionshandels auf. Mit Strom wird auf dem Spotmarkt an der EEX schon seit dem Jahr 2000 gehandelt. Das Hauptaugenmerk wird in dieser Arbeit daher auf der Prognose der Strompreise des Spot Day-Ahead Auktionshandels. Es werden unterschiedliche Prognoseverfahren vorgestellt, angewendet und miteinander verglichen. Besonderes Augenmerk wird dabei auf die Vorverarbeitung der Zeitreihen gelegt, wobei insbesondere Methoden aus der Wavelet-Theorie zur Anwendung kommen. Wavelets zerlegen die Zeitreihe ähnlich wie die Fouriertransformation, die in dieser Arbeit allerdings nicht näher erläutert wird, in eine Folge von Koeffizienten. Manipulation oder Selektion der Koeffizienten mit anschließender Rücktransformation erzeugt neue Zeitreihen, die für die Lernverfahren oder zumindest für weitere Vorverarbeitungsschritte möglicherweise besser geeignet sind als die ursprünglichen Daten. Zur Vorhersage der Strompreise werden auch Wertereihen mit direktem oder indirektem Einfluss wie beispielsweise Temperatur- und Kalenderdaten hinzu genommen, um ihren Einfluss auf die Ergebnisse der Lernverfahren zu studieren.

In dieser Arbeit werden Methoden evaluiert, von denen angenommen wird, dass sie die Prognosefehler deutlich verringern können. Dies sind keine gänzlich neu entwickelten Methoden, sondern vielmehr Vorverarbeitungsschritte wie die Anwendung von Wavelet-Algorithmen und Erweiterungen bewährter Verfahren wie das Verwenden neuer Kernfunktionen bei der Stützvektormethode. Zahlreiche Studien belegen die Überlegenheit der einzelnen Methoden gegenüber herkömmlichen Verfahren, siehe dazu Kapitel 5, was natürlich zu der These führen könnte, dass diese Methoden auch im vorliegenden Fall der Energiebörsendaten von Nutzen sind. Diese Arbeit kombiniert nun einige der Methoden und wendet sie auf den oben erwähnten Zeitreihen des Energiehandels an, mit dem Ziel, geeignete Kombinationen von Vorverarbeitungsschritten und Lernverfahren zu finden, die den Prognosefehler weiter reduzieren. Ein Vergleich mit der statistischen Zeitreihenanalyse und -prognose nach Box und Jenkins soll außerdem herausfinden, wie die Verfahren des maschinellen Lernens gegenüber den klassischen Zeitreihenmethoden abschneiden.

Im weiteren Verlauf dieser Arbeit werden zunächst die zu analysierenden Daten in Kapitel 2 näher vorgestellt. In Kapitel 3 werden schließlich die grundlegenden Lern- und Prognoseverfahren aus Statistik und maschinellem Lernen vorgestellt, welche später auf die Energiemarktdaten angewendet werden. Dieses Kapitel liefert außerdem eine Einführung in die übliche Vorgehensweise bei Aufgabenstellungen des maschinellen Lernens und in das Gebiet der Zeitreihenanalyse. Der Vorverarbeitung und Transformation der Daten widmet sich Kapitel 4. Darin wird insbesondere auf die diskrete Wavelet Transformation eingegangen, die in dieser Arbeit in einigen Versuchsreihen zum Einsatz kommt. Einen Überblick über den aktuellen Forschungsstand mit Bezug auf die in dieser Arbeit verwendeten Methoden liefert Kapitel 5. Experimente mit den Softwarepaketen RapidMiner, ehemals Yale und gretl (Gnu Regression, Econometrics and Time-series Library), geben schließlich in Kapitel 6 Aufschluss über die Performanz und Effektivität der unter-

schiedlichen Verfahren und Vorverarbeitungsschritte. Als neuartig anzusehen ist dabei die Verwendung von Wavelet-Kernfunktionen mit der Stützvektormethode auf Energiebörsendaten. Eine Zusammenfassung der Ergebnisse der Experimente sowie ein Ausblick schließen diese Arbeit letztlich ab.

2 Energiemarktdaten

Die Idee zu dieser Diplomarbeit entstand als Folge eines Gedankenaustauschs zwischen Mitarbeitern des Lehrstuhls 8 für künstliche Intelligenz der Technischen Universität Dortmund und Mitarbeitern der TradeSpark GmbH & Co. KG. TradeSpark entwickelt und vertreibt ein Marktinformationssystem für an europäischen Energiebörsen handelnde Unternehmen. Als solches Unternehmen ist TradeSpark im Besitz von Lizenzen, die das Sammeln und zum Teil auch Weiterverkaufen von Energiebörsendaten erlaubt. Bevor die Daten in Kapitel 2.1.2 erläutert werden, soll jedoch zunächst die Energiebörse „European Energy Exchange“, die als Quelle der in dieser Arbeit behandelten Zeitreihen diente, vorgestellt werden.

2.1 Energiebörse European Energy Exchange

Die Energiebörse European Energy Exchange (EEX), dessen Betreibergesellschaft, die EEX AG, in Leipzig sitzt, ist der Hauptaktionsort für den Handel am deutschen Energiemarkt. Die, gemessen an Teilnehmern und Handelsvolumen, größte kontinentaleuropäische Energiebörse entstand im Jahr 2002 aus der Fusion der bis dahin in Frankfurt am Main ansässigen EEX mit der Strombörse Leipzig Power Exchange (LPX). Heute kaufen und verkaufen über 200 Handelsteilnehmer aus 19 verschiedenen Ländern dort Strom, Gas, Kohle und Emissionszertifikate [Verivox].

Die meisten Stromlieferanten beziehen nur einen sehr geringen Teil ihrer Elektrizität über die Strombörse EEX. Das an der Börse gehandelte Jahresvolumen an Strom macht nur etwa 15 Prozent der insgesamt in Deutschland verbrauchten Strommenge aus. Die verbleibenden 85 Prozent werden über direkte Lieferverträge gehandelt. Für diese Lieferverträge werden jedoch die an der Energiebörse ausgehandelten Preise als Referenzen und Standardwerte herangezogen.

Stromversorger mit eigenen Kraftwerken sind von diesen Preisen natürlich weitaus weniger abhängig, da sie nur wenig oder gar keine Elektrizität zukaufen müssen. Die tatsächlichen Kosten der Stromversorger und -händler hängen außerdem davon ab, zu welchem Zeitpunkt und zu welchem Preis wie viel Strom eingekauft wurde. Hier lassen sich die Unternehmen ungern in die Karten schauen.

Die EEX veröffentlicht selbstverständlich alle Kurse auch im Internet, allerdings nur nach gezielter Eingabe eines einzelnen Datums. Die Firma TradeSpark GmbH & Co. KG besitzt jedoch eine Lizenz, die es dem Unternehmen ermöglicht, auf historische Da-

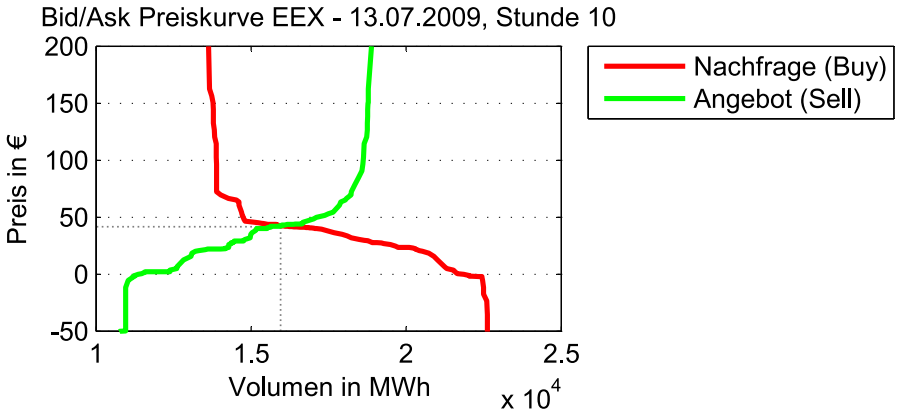


Abbildung 2.1: Bid/Ask Preiskurve eines einzelnen Spot-Stundenkontraktes. Gezeigt werden Käufer- und Verkäufergebote zu unterschiedlichen Mengen und Preisen. Der Schnittpunkt beider Kurven ergibt den Clearing-Preis für den Stundenkontrakt. Im Beispiel 42,45 €/MWh bei einem Volumen von 15.960,0 MWh.

ten via FTP zuzugreifen. Die Daten für diese Arbeit wurden uns freundlicherweise von TradeSpark zur Verfügung gestellt.

2.1.1 Produkte an der EEX

Der für Deutschland und weite Teile Mitteleuropas ausschlaggebende Referenzpreis wird „Phelix“ (Physical Electricity Index) genannt. Es wird unterschieden zwischen dem „Base“ und dem „Peak“ Index, wobei der Base-Index sich auf die Grundlast bezieht, die Energieversorgung an allen 24 Stunden eines Tages, wohingegen der „Peak-Load“ nur die Tagesstunden der Hauptarbeitszeit umfasst (Spitzenlast). Üblicherweise ist der Peak-Index höher als der Base-Index, da in der Regel tagsüber auch der Stromverbrauch höher, der Strom also teurer ist. Obwohl der Preis nicht nur abhängig von der Nachfrage ist, ist diese doch einer der Haupt-Einflussfaktoren.

Die EEX betreibt unterschiedliche Formen des Spothandels. Die Daten, welche in dieser Arbeit untersucht werden, entstammen dem „Day-Ahead“ Spot-Auktionshandel. Hier wird ausschließlich der Strom für den nächsten Tag gehandelt. Dabei wird die simultane Platzierung von unterschiedlichen Kaufs- und Verkaufsmengen zu jeweils verschiedenen Preisen für jede Stunde ermöglicht. Abbildung 2.1 zeigt die Gebote für einen einzelnen Stundenkontrakt und die anschließende Einigung im Schnittpunkt der Angebots- und Nachfragekurve (*Clearing*).

Die ausgehandelten Preise beziehen sich natürlich stets auf eine gehandelte Menge an Strom. Man spricht vom gehandelten „Volumen“, welches in Megawattstunden angegeben wird. Da der Terminmarkt der EEX für diese Arbeit keine Rolle spielt, soll an dieser Stelle auf eine Erläuterung verzichtet werden. Auch dem Handel mit Gas, Kohle und Emissionszertifikaten widmet sich diese Arbeit nicht. Eine ausführliche Abhandlung über die verschiedenen Handelsformen findet man in [Lehrmann 2002].

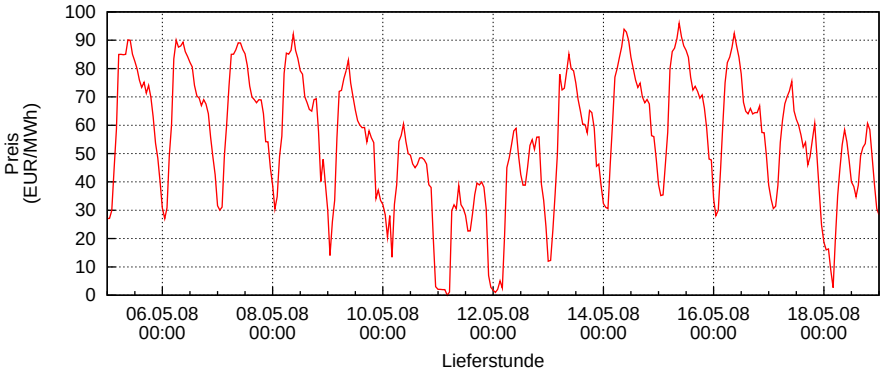


Abbildung 2.2: Auszug aus dem historischen Verlauf der Spotpreise des deutschen Energiemarktes, gehandelt an der Energiebörse European Energy Exchange. Die besonderen Charakteristika der Zeitreihe sind deutlich erkennbar. Jeder „Zacken“ stellt einen Tag dar. Die 5 Wochentage Montag bis Freitag weisen in der Regel ein höheres Preisniveau auf als die Wochenenden.

2.1.2 Daten der EEX

Die vorliegende Arbeit widmet sich in erster Linie den Strompreisen des Day-Ahead Spot Auktionshandels. Gehandelte Volumen werden nicht weiter betrachtet. Da sich der Day-Ahead Auktionshandel lediglich auf den nächsten Tag bezieht, werden Vorhersagen von mehr als 24 Stunden in die Zukunft in dieser Arbeit ebenfalls nicht betrachtet. Am Ende eines jeden Handelstages veröffentlicht die EEX sämtliche Handelsergebnisse des Tages. Es stehen also pro Tag jeweils 24 neue Werte zur Verfügung. Die Historie der EEX reicht bei diesen Daten bis zum 16. Juni 2000 zurück. Diese Arbeit beschränkt sich auf die Betrachtung der Preisdaten von Beginn 2006 an. Siehe dazu auch Kapitel 6.2. Es stehen also Daten von über 3 vollen Jahren für Analysezwecke zur Verfügung.

Abbildung 2.2 zeigt einen Auszug aus den historischen Preisverläufen des Day-Ahead Auktionshandels in Deutschland. Deutlich erkennbar sind die einzelnen Tage in Form von lokalen Maxima. Grund für diese Form ist, dass der Strompreis in den Abend- und Nachtstunden deutlich günstiger ist als tagsüber. Sieben Maxima entsprechen sieben Tagen, also einer Woche. Von den sieben Maxima treten 5 deutlich hervor, nämlich die Wochentage Montag bis Freitag. Die Wochenenden sind vom Preisniveau her deutlich günstiger.

3 Lern- und Prognoseverfahren

Im Falle von Zeitreihen spricht man häufig von einer *Regressionsanalyse*. Gesucht wird hierbei ein Modell, welches in Abhängigkeit einiger unabhängiger, erklärenden Variablen einen Output erzeugt. Dieser Output soll natürlich so nahe wie möglich am tatsächlichen Verlauf der Zeitreihe liegen. Im einfachsten Fall ist dieses Modell beispielsweise eine Funktion, die zu einem reellwertigen Input x einen reellwertigen Output y erzeugt. Dieses Modell kann dann dazu verwendet werden, zu Werten von x, für die noch kein y-Wert bekannt ist, einen solchen zu berechnen, die Zeitreihe also fortzuführen. Man spricht von einer *Prognose*. Da die meisten Prognoseverfahren ihre Modelle anhand von Daten aus der Vergangenheit erstellen, verwendet man häufig den Begriff des *Lernens*.

In diesem Kapitel werden die später verwendeten Lernverfahren vorgestellt. Dabei wird unterschieden zwischen den Verfahren der klassischen, statistischen Zeitreihenanalyse basierend auf dem Box-Jenkins Modell in Kapitel 3.1 einerseits und den Verfahren des maschinellen Lernens (Kapitel 3.2), insbesondere der Stützvektormethode (Kapitel 3.3) andererseits. Eine ausführliche und umfassende Einführung in das Gebiet der maschinellen Lernverfahren aus statistischer Sicht liefert [Hastie et al. 2009]. Dort werden auch weitere Verfahren vorgestellt, deren Behandlung den Rahmen dieser Arbeit sprengen würde. Es sei erwähnt, dass maschinelle Lernverfahren und Statistik eng miteinander verbunden sind und nicht losgelöst voneinander betrachtet werden können.

3.1 Zeitreihenanalyse in der Statistik

Im Folgenden sollen die klassischen Verfahren der statistischen Zeitreihenanalyse, welche als Spezialform der Regressionsanalyse angesehen werden kann, vorgestellt werden. Da dieses Gebiet jedoch zu groß ist, um in dieser Arbeit umfassend behandelt werden zu können, wird das Hauptaugenmerk auf die 1970 von G. E. P. Box und G. M. Jenkins entwickelten Methoden geworfen [Box und Jenkins 1970], [Anderson 1976]. Eine allgemeine, umfassende Übersicht über die verwendeten Methoden der Zeitreihenvorhersage in den letzten 25 Jahren gibt [De Gooijer und Hyndman 2006]. Box und Jenkins haben als erste eine stochastische Alternative zum damals verbreiteten, deterministischen Trendmodell vorgestellt. Die Methoden sind häufig unter der Bezeichnung „ARMA“ (Autoregressive Moving Average) anzutreffen und beschreiben lineare Modelle, in die Rauschterme und gewichtete frühere Werte der Zeitreihe mit einfließen. ARMA-Modelle werden bis heute als Prognosemodelle von etlichen Banken und Wirtschaftsinstituten eingesetzt, welche teilweise extra für diesen Zweck spezielle Software entwickelt haben.

Da die Analysen von Zeitreihen aufgrund der oftmals komplizierten Zusammenhänge der Zeitreihen keinem strengen Muster folgen, braucht es eine Menge Erfahrung und umfassende Kenntnis der Methoden, um sinnvolle Analysen mit aussagekräftigen Ergebnissen zu erstellen. Daher kann das folgende Kapitel keine „Kochrezepte“ zur Zeitreihenanalyse liefern, sondern beschränkt sich auf die Vorstellung der wesentlichen Methoden. Zuvor werden allerdings einige nötige Begriffe und Definitionen eingeführt.

Die eigentliche Verwendung der Methoden für die Prognose ist erst am Ende dieses Kapitels zu finden, was aber daran liegt, dass zuvor erst einige Begriffe und Hilfsmittel eingeführt werden müssen. Die Kenntniss dieser Begriffe und Hilfsmittel ist notwendig für den interaktiven Prozess der Modellfindung bei der Prognose.

3.1.1 Begriffe der Zeitreihenanalyse

Um zu verstehen, was Zeitreihen mathematisch gesehen eigentlich sind, sind zunächst einige Definitionen aus dem Bereich der Zeitreihenanalyse der Statistik hilfreich. Diese geben außerdem Aufschluss über wichtige Kennzahlen und Eigenschaften einer Zeitreihe, welche bei der späteren Analyse von Bedeutung sind. Die Definitionen sind größtenteils [Schlittgen 2001] und [Schlittgen und Streitberg 1997] entnommen, finden sich aber in ähnlicher Form in quasi sämtlichen Werken zur statistischen Zeitreihenanalyse wieder.

Definition 3.1.1. (Stochastischer Prozess) Ein stochastischer Prozess ist eine Folge (Y_t) von Zufallsvariablen. Der Index t, $t \in \mathbb{N}, \mathbb{N}_0$ oder $\mathbb{Z}$ wird in der Regel als Zeit aufgefasst.

Definition 3.1.2. (Zeitreihe) Eine Zeitreihe ist eine Folge $y_1, \ldots, y_n$ von Realisationen eines Ausschnittes des stochastischen Prozesses (Y_t). Oft wird (Y_t) oder $Y_1, \ldots, Y_n$ selbst als Zeitreihe bezeichnet. Eine einzelne Zeitreihe nennt man *univariat*. Existieren für denselben Zeitindex $i \in \{1, \ldots, n\}$ mehrere y_i Werte, ist y_i also ein Vektor, so spricht man von einer *multivariaten* Zeitreihe.

Definition 3.1.3. (Weißes Rauschen) Weißes Rauschen, auch *White-Noise-Prozess* genannt, ist eine Folge von unabhängigen, identisch verteilten („*iid*“, independent, identically distributed) Zufallsvariablen. In der Statistik und auch in dieser Arbeit wird weißes Rauschen meist mit (ε_t) bezeichnet.

Definition 3.1.4. (Autokovarianz) Autokovarianz untersucht den Zusammenhang zwischen Realisationen einer Zufallsvariablen zu unterschiedlichen Zeitpunkten t_1 und t_2. Die Autokovarianzfunktion ist definiert als

$$\gamma(t_1, t_2) = E[(Y_{t_1} - \mu_{t_1})(Y_{t_2} - \mu_{t_2})]$$

Dabei sind die μ_{t_i} die Erwartungswerte der Zeitreihe zum Zeitpunkt t_i. Im endlichen, diskreten Fall werden die Erwartungswerte über einfache Mittelwertberechnungen durchgeführt. Die Autokovarianzfunktion bildet die Grundlage der Berechnung der Autokorrelation.

Definition 3.1.5. (Lag) Mit einem *Lag* τ ist eine Zeitdifferenz zwischen zwei Zeitpunkten t_1 und t_2 gemeint. Es gilt also

$$\tau = t_1 - t_2$$

Definition 3.1.6. (Backshift-Operator) Der Backshift-Operator, teilweise auch *Lag-Operator* genannt, ist ein Hilfsmittel zur Beschreibung von linearen Filtern. Er verschiebt eine Zeitreihe um genau eine Zeiteinheit in die Vergangenheit:

$$BY_t = Y_{t-1}$$

Obwohl er ein Operator ist, der ein Argument, in diesem Fall eine Zeitreihe benötigt, lässt sich mit ihm wie mit einer Zahl rechnen. Potenzen entsprechen mehrmaligem Anwenden, negative Exponenten bedeuten ein Verschieben in die Zukunft.

$$B^d Y_t = B^{d-1}(BY_t) = B^{d-1}Y_{t-1} = \cdots = Y_{t-d}$$

Definition 3.1.7. (Autokorrelation) Die Autokorrelationsfunktion (ACF) ist eine normierte Form der Autokovarianz. Sie ist definiert als

$$\rho(t_1, t_2) = \frac{\gamma(t_1, t_2)}{\sigma_{t_1}\sigma_{t_2}}$$

wobei σ_{t_i} die jeweiligen Standardabweichungen sind. Die Funktion nimmt Werte zwischen 1 (stark korreliert) und -1 (stark anti-korreliert, gegenläufig) an. Ein Wert um die 0 bedeutet „keine Korrelation“.

Definition 3.1.8. (Partielle Autokorrelation) Die partielle Autokorrelationsfunktion (PACF) beschreibt den Zusammenhang zwischen Y_t und $Y_{t+\tau}$ unter Ausschaltung des Einflusses der dazwischen liegenden Werte. Ihre formale Definition lautet

$$\pi_\tau = \mathrm{Corr}(Y_t, Y_{t-\tau} | Y_{t-1}, \ldots, Y_{t-\tau+1}) \qquad \tau \in \{0, \pm 1, \pm 2, \ldots\}$$

Sie wird geschätzt über den Einsatz der *Yule-Walker* Gleichungen oder der *Levinson-Durbin*-Rekursion [Brockwell und Davis 1991]. Auch sie nimmt Werte zwischen -1 und 1 an.

3.1.2 Stationäre Zeitreihen

Damit Vorhersagen einer Zeitreihe statistisch gesehen überhaupt sinnvoll durchgeführt werden können, müssen geeignete Annahmen getroffen werden, wobei die zentralste Forderung sicherlich die der Stationarität ist. Die Analyseverfahren des Box-Jenkins Modells [Box et al. 2008] beispielsweise setzen Stationarität zwingend voraus.

Definition 3.1.9. (Stationarität) Ein stochastischer Prozess (Y_t) heißt *stationär*, falls für alle Zeitpunkte t und Lags τ gilt:

$$\begin{aligned} E(Y_t) &= \mu \\ \mathrm{Var}(Y_t) &= \sigma^2 \\ \mathrm{Cov}(Y_t, Y_{t+\tau}) &= \gamma_\tau \end{aligned}$$

Streng genommen handelt es sich hier nur um sogenannte *schwache* Stationarität. Die *starke* Stationarität fordert, dass die Verteilung der Zeitreihe nicht vom Zeitpunkt abhängt. Für unsere weiteren Untersuchungen in dieser Arbeit genügt jedoch die schwache Stationarität. Bei stationären Zeitreihen kommt es nicht mehr auf die genauen Zeitpunkte t_1 und t_2 an, da Erwartungswert, Standardabweichung und Varianz nicht mehr zeitabhängig sind. Die Berechnung der Autokovarianz beschränkt sich somit auf die Angabe des Lags τ:

$$\gamma_\tau = \frac{1}{n}\sum_{t=1}^{n}(Y_t - \bar{Y})(Y_{t+\tau} - \bar{Y}) = \mathrm{Cov}(Y_t, Y_{t+\tau})$$

Ebenfalls vereinfacht sich die Berechnung der Autokorrelationsfunktion zu

$$\rho_\tau = \frac{\gamma_\tau}{\sigma_Y^2} = \frac{\gamma_\tau}{\gamma_0}$$

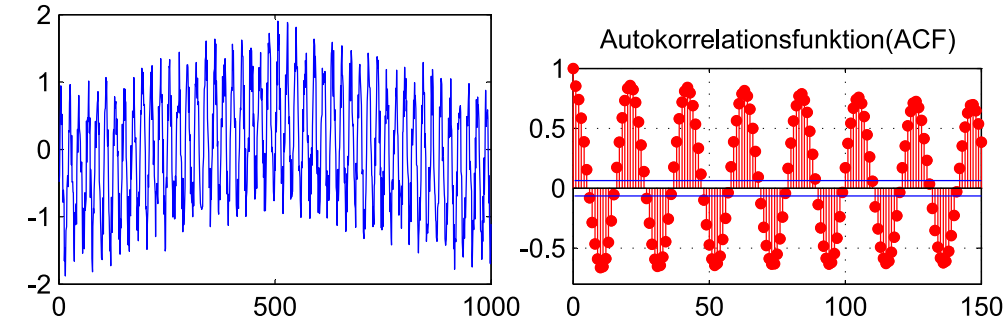
3.1.3 Erster Eindruck

Der erste Schritt bei jeder Analyse einer Zeitreihe, unabhängig von den im weiteren Verlauf benutzten Analysemethoden, sollte darin bestehen, sich die grafische Darstellung der Reihe anzuschauen. Anhand dieser Grafik lassen sich einige Charakteristika oftmals gut mit bloßem Auge erkennen. Zwar entspricht dies noch keinem wissenschaftlichen Ergebnis, jedoch gibt die Betrachtung in der Regel Hinweise auf weitere durchzuführende Vorverarbeitungs- und Analyseschritte. Beispielsweise sind lineare Trends oder deutliche zyklische Schwankungen oftmals schon direkt am Verlauf der Zeitreihe zu erkennen. In diesen Fällen kann man meist davon ausgehen, eine nichtstationäre Zeitreihe zu behandeln. Die Wichtigkeit des ersten optischen Eindrucks sollte bei einer ernsthaften Analyse nicht unterschätzt werden.

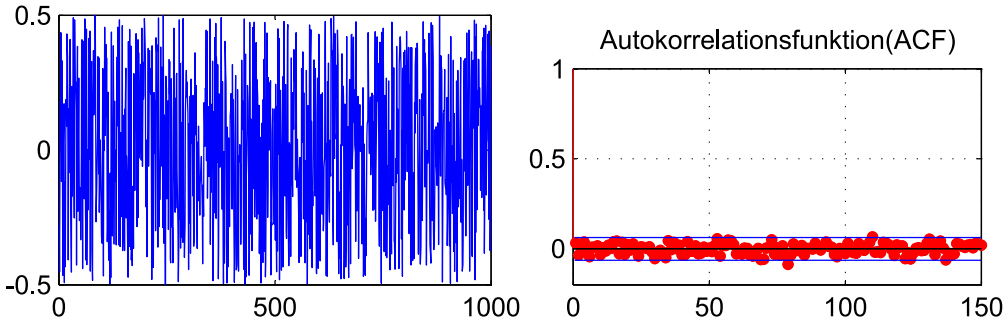
Die grafische Analyse und auch der Blick auf das in Kapitel 3.1.4 vorgestellte Autokorrelogramm kann Aufschluss über Stationarität des Prozesses und damit der Zeitreihe geben. Es existieren allerdings auch eine Reihe statistischer Tests für diesen Zweck [Kendall und Stuart 1966]. Prominentes Beispiel ist hierbei der sogenannte *augmented Dickey-Fuller* Test [Dickey und Fuller 1979], auch bekannt als *Einheitswurzel*-Test (*Unit-Root* Test).

3.1.4 Das Autokorrelogramm

Als eines der wichtigsten Werkzeuge bei der Zeitreihenanalyse dient das Autokorrelogramm. Es kann als Nachweis periodischer beziehungsweise saisonaler Schwankungen, Kurzzeit-Korrelation oder auch alternierender Reihen dienlich sein [Chatfield 1982]. Im Autokorrelogramm werden die Funktionswerte der Autokorrelationsfunktion (*ACF*), die Autokorrelationskoeffizienten einer Zeitreihe gegen unterschiedliche Lags τ abgetragen. Die ACF kann als Maß der Ähnlichkeit der Reihe zu sich selbst um den Lag τ verschoben angesehen werden. Abbildung 3.1 zeigt zwei verschiedene Zeitreihen und ihre



(a) Selbstähnliche Zeitreihe



(b) Weißes Rauschen

Abbildung 3.1: Autokorrelationsfunktionen von unterschiedlichen Zeitreihen. Abbildung 3.1a zeigt eine Zeitreihe mit hohem Grad an Selbstähnlichkeit. Da die Zeitreihe aus Sinuskurven besteht, ergibt sich je nach Lag mal ein hoher positiver und mal ein hoher negativer Autokorrelationskoeffizient. Abbildung 3.1b zeigt unkorreliertes weißes Rauschen.

jeweiligen Autokorrelogramme. Da eine Zeitreihe zu sich selbst am ähnlichsten ist, wenn sie nicht verschoben wurde, hat die Autokorrelationsfunktion bei 0 stets den höchstmöglichen Wert 1.

Da die im Folgenden behandelten Modelle stets von stationären Zeitreihen ausgehen, wäre ein Autokorrelogramm wie das in Abbildung 3.1b dargestellte wünschenswert. Viele in der Praxis auftretenden Zeitreihen sind allerdings in höchstem Maße nichtstationär und müssen zunächst durch geeignete Techniken in einen stationären Zustand gebracht werden. Diese Techniken werden im Folgenden beschrieben.

3.1.5 Das Komponentenmodell

Da die meisten in der Praxis vorkommenden Zeitreihen in der Regel nichtstationär sind, geht man üblicherweise dazu über, eine Zeitreihe in mehrere Komponenten zu zerlegen. Die so zerlegte Zeitreihe kann auf diese Weise von nichtstationären Einflüssen bereinigt werden.

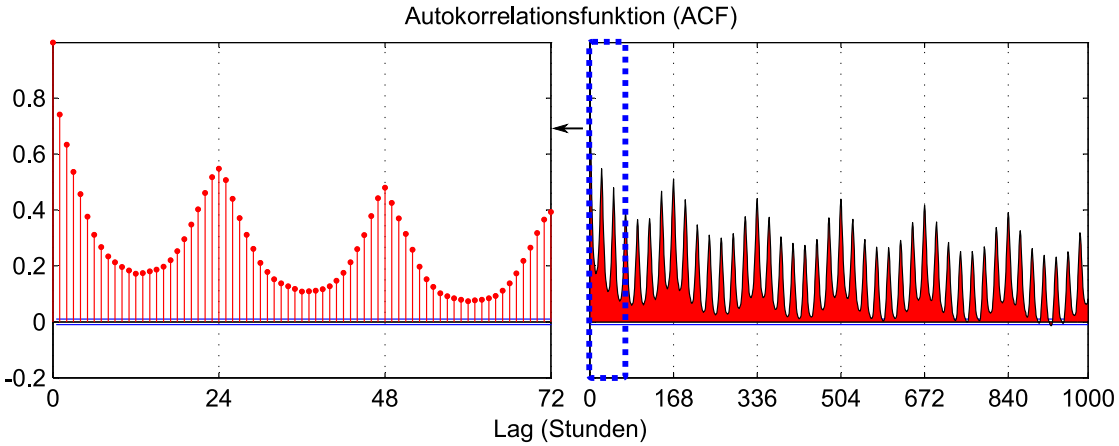


Abbildung 3.2: Autokorrelationsfunktion der Spotpreise des Day-Ahead Auktionshandels an der EEX. Die Abbildung links ist dabei ein vergrößerter Ausschnitt aus der rechten Abbildung. Klar zu erkennen sind die jeweiligen lokalen Maxima in Abständen von je 24 Stunden. Bei Betrachten der ACF in höherer Auflösung (rechts) fallen außerdem Spitzen im regelmäßigen Abstand von 168 Stunden auf. Dies entspricht genau einer Woche. Die Zeitreihe ist sich also im Abstand von einem Tag und einer Woche selbstähnlich.

Definition 3.1.10. (Komponentenmodelle) Es existieren die folgenden beiden prinzipiellen Formen der Zerlegung von Zeitreihen:

$$Y_t = T_t + S_t + \varepsilon_t \qquad \text{(additives Modell)} \qquad (3.1)$$

$$Y_t = T_t \cdot S_t \cdot \varepsilon_t \qquad \text{(multiplikatives Modell)} \qquad (3.2)$$

Dabei ist Y_t die Zeitreihe, T_t wird als langfristige Trendkomponente und S_r als Saisonkomponente bezeichnet. „Trend“ meint hiermit einen störenden, systematischen Effekt mit einer Grundtendenz, wohingegen die Saisonkomponente ein zyklisches, zeitabhängiges Verhalten darstellt. ε_t stellt den nicht weiter erklärbaren Rest dar. Ein Modell muss nicht zwingend jede dieser Komponenten enthalten. Auch sind Modelle mit mehr Komponenten als die hier vorgestellten nicht selten. Die Wahl des Modells richtet sich meist nach dem Erscheinungsbild der Zeitreihe. Sind beispielsweise die Saisonausschläge über die gesamte Zeit ähnlich, so wird man in der Regel das additive Modell vorziehen. Bei zunehmenden Ausschlägen der Saisonkomponente und vorhandenem Trend ist möglicherweise das multiplikative Modell angebrachter.

3.1.6 Trend- und Saisonbereinigung

Hat man festgestellt, oder weiß man gar, dass die untersuchte Zeitreihe einen langfristigen Trend enthält, so gilt es diesen Trend zu modellieren, um einerseits die Zeitreihe von ihm zu befreien und andererseits bei der Prognose entlang des Trends in die Zukunft extrapolieren zu können. Deterministische Trends werden dabei häufig durch Polynome

dargestellt. Die durch eine Atemkurve bewirkten systematischen Effekte auf eine Herzfrequenzkurve beispielsweise werden in der Medizin durch ganzrationale Funktionen vierten Grades modelliert [Schlittgen 2001]:

$$Y_t = \beta_0 + \beta_1 t + \beta_2 t^2 + \beta_3 t^3 + \beta_4 t^4 + \varepsilon_t \qquad (t = 1, \ldots, N)$$

Dabei werden die Polynomkoeffizienten mit Hilfe der *kleinste Quadrate (KQ)* Methode geschätzt. Prinzipiell kann aber jede in irgendeiner Art und Weise geglättete Approximation der Zeitreihe als Trend dienen. Manchmal ist es allerdings gar nicht von Vorteil, dem Trend einen funktionalen Verlauf zu unterstellen, die Trendfunktion sollte dann eher als *glatte Komponente* betrachtet werden. Das können lokale Mittelwerte, sogenannte *gleitende Durchschnitte* sein, bei dem arithmetische oder gewichtete Mittelwerte einzelner Reihensegmente berechnet werden. Aber auch Splines sind oftmals gute Kandidaten für glatte Komponenten. Im Vergleich zu einzeln angepassten, lokalen Polynomen weisen sie an den Übergangsstellen einen glatten Verlauf auf, sind also differenzierbar. In dieser Arbeit wird ein Ansatz vorgestellt, bei dem eine durch Wavelets konstruierte, geglättete Kurve als Trend dient, vom Prinzip sehr ähnlich den gleitenden Durchschnitten.

Liegt keine deterministische Trendkomponente vor, so kann es sein, dass die nichtstationäre Zeitreihe durch einen nichtstationären stochastischen Prozess, beispielsweise einen *Random Walk* erzeugt wurde. In diesem Fall eignet sich die Bildung von sukzessiven Differenzen zur Vorverarbeitung (B entspricht hierbei dem Backshift-Operator aus Definition 3.1.6):

$$Z_t = Y_t - Y_{t-1} = (1 - B)Y_t \tag{3.3}$$

Man nennt einen Prozess $Y(t)$ vom Typ I(1) (I für *integriert*), falls Z_t aus Gleichung 3.3 stationär ist. Entsprechend lassen sich die Typen I(k) mit $k \in \mathbb{N}_+^0$ definieren. Das I entspricht im übrigen dem I der ARIMA-Prozesse. Auch hier steht das I für eine implizite Differenzbildung im Unterschied zu ARMA-Prozessen, bei denen nicht differenziert wird, siehe Kapitel 3.1.10.

3.1.7 Transformationen

Weitere Möglichkeiten, eine Zeitreihe in einen stationären Zustand zu bringen besteht in der Anwendung von Transformationen. Dabei unterscheidet man zwischen gleichartiger Modifikation eines jeden einzelnen Reihenwerts einerseits und der Kombination mehrerer Reihenwerte zu einem neuen Wert andererseits.

Beispiele für letzteres haben wir bereits in Kapitel 3.1.6 kennen gelernt. Gleitende Durchschnitte sind nichts weiter als lineare Filter, die die Zeitreihe transformieren. Allgemein definiert man zu einem Filter $(a_u)_{u=-r,\ldots,s}$ die Transformation

$$z_t = \sum_{u=-r}^{s} a_u y_{t-u} = \sum_{u=-r}^{s} a_u B^u y_t \tag{3.4}$$

wobei B den Backshift-Operator darstellt. Die Folge (z_t) heißt der *Output* des Filters. Filter sind uns bereits in Kapitel 4.2 bei der Berechnung der Wavelet-Koeffizienten begegnet.

Transformationen der zuerst genannten Kategorie kommen insbesondere dann zum Tragen, wenn die Zeitreihe mit ihrem Niveau zunehmend streut. Man spricht in diesem Fall von einer Varianzinstationarität im Zusammenhang mit einer Mittelwertinstationarität. Häufig werden in solchen Situationen nichtlineare Transformationen angewendet, beispielsweise durch Bilden des Logarithmus, wie es häufig exponentielle Wachstumsraten in der Biologie erfordern. Eine Obermenge dieser und anderer Transformationen sind die sogenannten *Box-Cox Transformationen* [Sakia 1992], definiert als

$$Z_t = \begin{cases} \frac{(Y_t+c)^\lambda - 1}{\lambda} & \text{für } \lambda \neq 0 \\ \ln(Y_t + c) & \text{für } \lambda = 0 \end{cases} \tag{3.5}$$

in die sich auch die logarithmischen Transformationen einfügen. c ist hierbei ein Parameter, der lediglich sicherstellen soll, dass alle Werte größer Null sind. In der Praxis verwendet man statt Gleichung 3.5 häufig die vereinfachte Form

$$Z_t = Y_t^\lambda \tag{3.6}$$

mit „glatten“ Werten für λ: $\ldots, -1, -0.5, 0, 0.5, 1, \ldots$. Des Weiteren hat die Anwendung der Box-Cox Transformation einen symmetrisierenden Effekt auf die Zeitreihe [Taylor 1985].

3.1.8 Autoregressive Modelle

Aus dem Begriff „autoregressiv“ wird bereits deutlich, dass es sich hier um Regressionsmodelle handelt, deren Regressor die Zeitreihe selbst ist. Wie zuvor erwähnt, wird im Folgenden davon ausgegangen, dass die zu untersuchenden Zeitreihe stationär ist. Die Idee ist nun, den Wert einer Zeitreihe zum Zeitpunkt t aus vorangegangenen Zeitpunkten zu erklären. Dies wird als Linearkombination formuliert:

$$Y_t = \varepsilon_t + \sum_{i=1}^{p} \alpha_i Y_{t-i} \tag{3.7}$$

Die Anzahl p der vorangegangenen Werte wird als *Ordnung* bezeichnet. Man spricht von einem *autoregressiven Prozess p-ter Ordnung*, kurz AR[p]. Die ε_t stellen weißes Rauschen dar. Die Regressionskoeffizienten α_i werden durch spezifische Schätzmethoden bestimmt, wie beispielsweise *Maximum Likelihood* (ML), *Conditional Least Sums of Squares* (CLS), *Unconditional Least Sums of Squares* (ULS) [Kay und Marple Jr 1981], *Yule-Walker* (YW) [Priestley 1981] oder dem *Burg*-Schätzer [Burg 1967].

Für spätere Zusammenhänge sei noch auf folgend Definition hingewiesen.

Definition 3.1.11. (Stationärer AR-Prozess) Ein AR[p]-Prozess heißt *stationär*, wenn die zugehörige *charakteristische Gleichung*

$$1 - \alpha_1 z - \alpha_2 z^2 - \ldots - \alpha_p z^p = 0$$

ausschließlich Lösungen $|z_i| > 1$ besitzt.

Zur Bestimmung der Ordnung des AR-Prozesses kann man die Autokorrelationsfunktion nutzen. Da die Eigenschaften, die man der ACF entnehmen müsste, mit dem Auge schwer zu erkennen sind, zieht man die partielle Autokorrelationsfunktion π_τ hinzu. Es lässt sich zeigen, dass für ein AR[p] Modell gilt [Schlittgen und Streitberg 1997]:

$$\pi_p \neq 0 \quad \text{und} \quad \pi_\tau = 0 \quad \text{für } \tau > p \tag{3.8}$$

Alternativ probiert man AR[p] Modelle aufsteigender Ordnungen durch und entscheidet anhand von statistischen Informationskriterien wie SBC, AIC, AICC oder dem HQ-Kriterium.

Als letzter Schritt sollte die Anpassungsgüte des Modells untersucht werden. Bei einem geeigneten AR[p]-Modell sollten sich die Residuen wie weißes Rauschen verhalten. Dazu stehen Methoden wie die Prüfung der Residuen auf Normalverteilung, die ACF und PACF der Residuen oder auch statistische Tests wie der *Box-Pierce-Test* [Box und Pierce 1970] oder der *Ljung-Box-Test* [Ljung und Box 1978] zur Verfügung.

3.1.9 Moving-Average Modelle

Die sogenannten *Moving-Average*-, kurz MA[q]-Prozesse verfolgen einen etwas anderen Ansatz. Hier wird die Zeitreihe nicht aus sich selbst, sondern aus den vorangegangenen Störungen erklärt. Formal aufgeschrieben bedeutet das:

$$Y_t = \varepsilon_t - \sum_{i=1}^{q} \beta_i \varepsilon_{t-i} \tag{3.9}$$

Definition 3.1.12. (Invertierbarer MA-Prozess) Ein MA[q]-Prozess heißt *invertierbar*, wenn die zugehörige *charakteristische Gleichung*

$$1 - \beta_1 z - \beta_2 z^2 - \ldots - \beta_q z^q = 0$$

ausschließlich Lösungen $|z_i| > 1$ besitzt.

Die Schätzung der Koeffizienten β_i erfolgt wie bei AR-Prozessen über Methoden wie ML, CLS, ULS[1] oder der von Brockwell und Davis vorgestellten *Innovationsmethode*, ein spezieller Schätzalgorithmus für MA-Prozesse [Brockwell und Davis 2002]. Yule-Walker und Burg-Schätzer können allerdings nicht verwendet werden. Die Schätzung gestaltet sich allerdings schwieriger als bei AR-Prozessen, was aus der Tatsache begründet ist, dass die ε_t, mit denen die β_i linear verknüpft sind, nicht direkt beobachtet werden können und stattdessen nur die Y_t zur Verfügung stehen.

Um die Ordnung des MA-Prozesses zu bestimmen, nutzt man wieder die ACF beziehungsweise PACF. Für $\tau > q$ gilt nämlich $\rho_\tau = 0$ und π_τ klingt exponentiell ab.

[1] siehe Abschnitt 3.1.8

	AR[p]	MA[q]	ARMA[p,q]
ACF	klingt ab	bricht nach Lag q ab	klingt ab
PACF	bricht nach Lag p ab	klingt ab	klingt ab

Tabelle 3.1: Typisches Verhalten der ACF und PACF bei autoregressiven Moving-Average Prozessen, entnommen aus [Schlittgen 2001].

3.1.10 ARMA und ARIMA

Verknüpft man nun AR- und MA-Prozesse, so erhält man die sogenannten *ARMA[p,q]*-Prozesse, formal also

$$Y_t = \varepsilon_t + \sum_{i=1}^{p} \alpha_i Y_{t-i} - \sum_{j=1}^{q} \beta_j \varepsilon_{t-j} \tag{3.10}$$

Üblicherweise schreibt man

$$\sum_{i=0}^{p} \alpha_i Y_{t-i} = \sum_{j=0}^{q} \beta_j \varepsilon_{t-j} \tag{3.11}$$

oder kurz

$$\alpha(B) Y_t = \beta(B) \varepsilon_t \tag{3.12}$$

Zur Zentrierung der Zeitreihe wird häufig Y_t durch $Y_t - \mu$ ersetzt. Die Eigenschaften „stationär“ und „invertierbar“ sind auch auf ARMA-Prozesse anwendbar. Dabei zeigt sich die Dualität der beiden Modelle:

Satz 3.1.13. *Jeder stationäre AR[p]-Prozess lässt sich als MA[∞]-Prozess modellieren. Jeder invertierbare MA[q]-Prozess lässt sich als AR[∞]-Prozess modellieren.*

Als Analysewerkzeug zur Bestimmung der Ordnungen p und q dienen auch hier wieder ACF und PACF. Beispielhaftes Verhalten beider Korrelogramme zeigt Tabelle 3.1. Weitere Hilfsmittel sind außerdem die *inverse* ACF (IACF), welche die ACF des Prozesses $\beta(B)Y_t = \alpha(B)\varepsilon_t$ ist [Chatfield 1979], Vektorkorrelationen [Paparoditis und Streitberg 1992] und erweiterte Stichprobenkorrelationen (ESACF) [Tsay und Tiao 1984].

Zur Schätzung der Koeffizienten stehen die bekannten Verfahren mit Ausnahme des Yule-Walker und des Burg-Schätzers zur Verfügung und auch hier existiert ein spezifisches rekursives Schätzverfahren [Hannan und Rissanen 1982].

Bildet man vor der Analyse einer nichtstationären Zeitreihe zunächst Differenzen, so kommt man zu den ARIMA[p,d,q]-Modellen für trendbehaftete Zeitreihen

$$\alpha(B) \underbrace{(1-B)^d}_{\text{Trendbereinigung}} Y_t = \beta(B)\varepsilon_t \tag{3.13}$$

bei denen zusätzlich zu den Parametern p und q der ARMA-Modelle noch der Grad d der Differenzbildung zu bestimmen ist. Ob Differenzbildung überhaupt notwendig ist, lässt

sich an der ACF erkennen. Bei einem Verlauf, der nur langsam von +1 absteigt, liegt mit hoher Wahrscheinlichkeit ein Trend vor. Bei der Bestimmung der drei Parameter sollte man allerdings q mindestens so groß wie d wählen.

Reihen mit saisonalen Komponenten können wie folgt modelliert werden:

$$\alpha(B)(1-B)^d \underbrace{(1-B^s)^D}_{\text{Saisonbereinigung}} Y_t = \beta(B)\varepsilon_t \tag{3.14}$$

Dabei entspricht der Term $(1-B^s)^D$ dem Eliminieren der saisonalen Komponente mit Periode s durch geeignete, D-fache Differenzbildung. Als Verbesserung dieser Modelle seien noch die *Saisonalen ARIMA-Modelle* (SARIMA) von Box und Jenkins angeführt, die aber in dieser Arbeit nicht weiter untersucht werden.

3.1.11 Prognosen

Bei der Zeitreihenprognose geht es nun darum, die Reihe in die Zukunft fortzusetzen. Formal möchte man $\hat{Y}_{N,h}$ bestimmen, also den geschätzten Wert der Reihe im Abstand h zu den N bisher beobachteten Werten. Es lässt sich zeigen, dass die unter Anwendung der linearen ARMA Modelle resultierende lineare Prognose zumindest asymptotisch *optimal* ist [Schlittgen 2001]. Dazu wird zunächst $\hat{Y}_{N,1}$ bestimmt und dieser Wert dann im nächsten Schritt zu den bisher beobachteten Werten hinzugenommen. Stationäre ARMA-Prozesse gehen für $h \to \infty$ gegen ihren Erwartungswert. Weißes Rauschen wird im übrigen aus naheliegenden Gründen am besten durch den Wert Null prognostiziert.

3.2 Maschinelles Lernen

Bevor spezielle Verfahren des maschinellen Lernens erläutert werden, soll ein kurzer Abriss über die typische Verfahrensweise bei maschinellen Lernaufgaben in das Themengebiet einführen. Diese Verfahrensweisen bilden die Grundlage aller in RapidMiner durchgeführten Experimente in Kapitel 6.

3.2.1 Grundbegriffe des maschinellen Lernens

Die Kenntnis einiger Begriffe aus dem Bereich des maschinellen Lernens sind für das Verständnis dieser Arbeit zwingend notwendig und sollen daher im Folgenden kurz vorgestellt werden.

Definition 3.2.1. (Feature) Ein Feature oder auch Attribut ist ein einzelnes Merkmal einer Beobachtung beziehungsweise eines Datensatzes. Beispielsweise sind das Gehalt und das Geschlecht eines Mitarbeiters zwei unterschiedliche Features einer Person.

Definition 3.2.2. (Label) Ein Label ist ein ausgezeichnetes Feature eines Datensatzes. Es stellt die Information dar, die von überwachten Lernverfahren gelernt und später bei

ungelabelten Datensätzen vorhergesagt werden soll. Unüberwachte Lernverfahren wie Clustering benötigen hingegen kein Label.

Definition 3.2.3. (Beispiel) Ein Beispiel (englisch: Example) fasst zusammengehörige Features zu einem Datensatz zusammen. In der Regel werden mehrere Beispiele für ein Lernverfahren benötigt, denn nur unterschiedliche Beispiele machen maschinelles Lernen überhaupt erst möglich. Beispiele sind häufig ganze Zeilen einer (Datenbank-)Tabelle. Diese Definition wird auch von RapidMiner verwendet.

Definition 3.2.4. (überwachtes Lernen) Hiermit sind Lernverfahren gemeint, die in ihrer Lernphase aus historischen Daten unter Vorgabe des jeweils korrekten Ergebnisses ein Modell berechnen. Dieses Modell soll dazu in der Lage ist, eigenständig Ergebnisse zu bestimmen, die die wahre Verteilung der Daten möglichst präzise approximiert.

Definition 3.2.5. (unüberwachtes Lernen) Lernverfahren auf ungelabelten Daten bezeichnet man als unüberwachte Lernverfahren. Clusteringverfahren beispielsweise ordnen nach Vorgabe einer bestimmten Anzahl von Klassen jedes Beispiel selbstständig in eine Klasse ein. Dies funktioniert ohne vorangestellten Lernschritt auf bereits klassifizierten Daten.

3.2.2 Analysezyklus maschineller Lernverfahren

Maschinelle Lernverfahren extrahieren automatisiert Informationen aus Daten. Diese Information kann die Zugehörigkeit zu einer Klasse von Daten, oder aber auch die Prognose eines zukünftigen Wertes einer Zeitreihe sein, wie es in dieser Arbeit angewendet wird. Das Ergebnis einer maschinellen Lernaufgabe ist in jedem Fall ein Modell, welches es ermöglicht, neue, ungesehene Daten automatisiert korrekt zu klassifizieren beziehungsweise zu prognostizieren. Nachdem man sich mit den Daten vertraut gemacht hat, werden in der Regel folgende Schritte durchlaufen:

1. Entscheidung für eine Modellklasse fällen
2. Parameter der Modellklasse festlegen
3. Modell auf Trainingsmenge trainieren
4. Trainiertes Modell auf Testmenge validieren
5. Falls Ergebnis nicht den Wünschen oder Vorgaben entspricht: Parameter ändern und weiter bei 2., möglicherweise sogar zurück zu 1.

3.2.3 Parameteroptimierung

Die meisten Lernverfahren besitzen eine Vielzahl an Parametern, die das Verhalten der Algorithmen beeinflussen. Die Festlegung auf bestimmte Parameterwerte legt somit auch das entstehende Modell fest. Aus diesem Grunde entspricht die Parameteroptimierung eigentlich der Modellselektion. Um Verwirrungen vorzubeugen wird im Folgenden jedoch

von Parameteroptimierung die Rede sein, wenn die Optimierung von Modellparametern gemeint ist. Nur bei korrekter Wahl dieser Parameter liefern die Verfahren gute Ergebnisse. Ein manuelles Finden der besten Parameterwerte ist jedoch häufig schwierig, wenn nicht sogar unmöglich. Folgende Ansätze können bei der Suche nach den besten Parametern helfen. Allen Verfahren ist gemeinsam, dass sie verschiedene Parameterkombinationen „durchprobieren“ und am Ende die besten Parameterwerte zurück liefern. Dabei wird in jeder Runde ein kompletter Trainings- und Validierungsschritt unter Anwendung der gewählten Parameter durchlaufen. Die Performanz dieser Durchläufe, ein frei wählbares Fehlermaß, ist letztlich der Indikator für die optimale Parameterbelegung, wobei „optimal“ als „im Rahmen der Möglichkeiten der Parameteroptimierung“ gesehen werden muss. Da viele Parameter durch kontinuierliche Werte dargestellt werden, ist es unmöglich, sämtliche Werte dieser Parameter auszuprobieren.

Gitter-Parameteroptimierung: Auch „Grid Parameter Optimization“ genannt. Für jeden Parameter werden Minimum, Maximum und die Anzahl m der zu testenden Parameterwerte bestimmt. Der zu untersuchende Parameterbereich wird nun linear, quadratisch oder logarithmisch in m Punkten untersucht. Die Gitter-Parameteroptimierung eignet sich gut, um den Suchraum der Parameterwerte grob abzutasten. In wiederholter Anwendung kann man so den Suchraum weiter einschränken.

Quadratische Parameteroptimierung: Identisch zur Gitter-Parameteroptimierung, jedoch werden nach der Auswertung aller Parameterkombinationen die besten Parameter mittels quadratischer Optimierung ermittelt.

Evolutionäre Parameteroptimierung: Bestimmt unter Einhaltung von Schranken für die Parameterwerte eine zufällige Population von verschiedenen Parameterkombinationen (Individuen) und konstruiert neue Parameterkombinationen anhand einer evolutionären Strategie durch Mutation und Rekombination [Droste et al. 1998]. Als Fitness eines Individuums wird die Performanz des Lernmodells unter Anwendung der entsprechenden Parameterwerte gewählt. Kann von Vorteil sein, wenn man die Schranken der Parameter bereits relativ eng setzen kann. Dafür kann das Verfahren aber leider auch leicht in lokalen Optima stecken bleiben.

Basierend auf der Gitter-Parameteroptimierung wurde im Rahmen dieser Arbeit die **rekursive Gitter-Parameteroptimierung** implementiert und zum Einsatz gebracht. Sie übernimmt automatisiert das wiederholte Einschränken des Suchraums um die bisher besten gefundenen Parameterwerte bis zu einer definierbaren Tiefe. Dies spart viel Handarbeit und ermöglicht ein grobes Abtasten des Suchraums, sowie die Suche in die Tiefe mit beliebiger Genauigkeit (ausgenommen: ganzzahlige Parameter).

3.2.4 Validierung

Welches Lernverfahren man auch immer anwendet, man ist stets an der Güte des resultierenden Modells interessiert. Zum einen benötigt man natürlich vergleichbare Kennzahlen über das Lernergebnis. Dafür existieren eine ganze Reihe von statistischen Kennzahlen, in der Regel Fehlermaße wie beispielsweise der häufig verwendete mittlere quadratische

Fehler (MSE). Zum anderen möchte man auch sicherstellen, dass das Modell möglichst gut generalisiert, also auch auf bisher ungesehenen Daten gute Ergebnisse liefert.

Im Falle von überangepassten Modellen, welche auf den Trainingsdaten zwar sehr wenige, auf ungesehenen Daten jedoch viele Fehler machen, spricht man vom *Overfitting*. Diese Modelle generalisieren in der Regel schlecht und sind somit für die Praxis nur bedingt zu gebrauchen. Um Overfitting zu begegnen, ist es notwendig, neben der Trainingsmenge, also der Menge an Daten, mit denen das Modell trainiert wird, noch eine genügend große Menge an Validierungsdaten zu haben. Es sollte einleuchten, dass das vom Lernverfahren erstellte Modell üblicherweise auf den Trainingsdaten gute Ergebnisse liefert, diese aber nicht zwingend auch auf bisher ungesehenen Daten ebenso bringen muss.

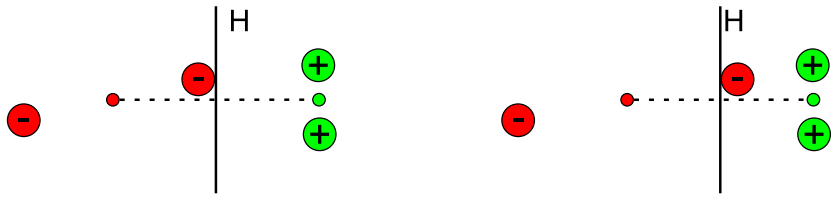
Im Zusammenhang mit der Parameteroptimierung verwendet man häufig sogar drei disjunkte Datenmengen [Witten und Frank 2005]. Die Trainingsmenge dient wie gehabt zum Trainieren des Modells. Die Schleife der Parameteroptimierung verwendet nun die Validierungsdaten, um die Güte der verwendeten Parameter abzuschätzen. Nach Beendigung der Parameteroptimierung wird das Modell schließlich auf den Testdaten evaluiert.

Nicht selten ist die Menge an zur Verfügung stehenden Daten allerdings begrenzt, so dass nicht beliebig viele Datensätze für eine Validierung benutzt werden können. Auch möchte man nicht ständig auf denselben Daten testen. Das Verfahren der *Kreuzvalidierung* partitioniert die Daten zufällig in m (häufig $m = 10$) gleichgroße Mengen (oft als geschichtete Stichprobe „stratified sampled“). Nun wird m Mal je eine dieser Mengen als Testdaten verwendet und auf den restlichen $m - 1$ das Modell trainiert. Die Performanz des Modells ergibt sich schließlich aus dem Durchschnitt der Performanz der einzelnen Durchläufe. Kreuzvalidierung ist eine weit verbreitete und anerkannte Methode, Behauptungen über das Lernergebnis zu untermauern und eignet sich besonders als innerer Validierungsoperator bei der Parameteroptimierung.

3.3 Stützvektormethode

Ein Lernverfahren mit breitem Einsatzgebiet, welches aus der statistischen Lerntheorie hervorgegangen ist, ist die Stützvektormethode (*Support Vector Machine*, SVM). Als lineares, binäres Klassifikationsverfahren bestimmt es die optimal trennende Hyperebene nach dem Maximum-Margin Prinzip, also diejenige Hyperebene mit größtmöglichem Abstand zu beiden Klassen. Dies macht die Stützvektormethode zu einem robusten, gut generalisierenden Lernverfahren. Durch den Austausch ihrer Kernfunktion ermöglicht die Methode außerdem auch nichtlineare Klassifikation durch implizit im Kern durchgeführte Transformation in einen höherdimensionalen Raum [Schölkopf und Smola 2002].

Dieses Kapitel behandelt zunächst die mathematischen Grundlagen der Stützvektormethode sowie ihre allgemeine Funktionsweise. In Kapitel 3.3.3 wird erläutert, wie durch den Austausch der Kernfunktion nichtlineare Klassifikation realisiert wird. Dies alles dient letztlich dem Verständnis bei der Definition der Stützvektor-Regression in Kapitel 3.3.4, welche eine zentrale Rolle in dieser Arbeit spielt. Die meisten der im Rahmen



(a) Schlecht plazierte Hyperebene

(b) Falsch plazierte Hyperebene

Abbildung 3.3: Hyperebene durch den Mittelpunkt der Verbindungslinie der Zentroiden (kleine Punkte) beider Klassen. Auf diese Weise können schlecht plazierte (a) oder gar falsch plazierte (b) Hyperebenen entstehen

dieser Arbeit durchgeführten Experimente verwenden die Stützvektor-Regression zur Bestimmung von Prognosemodellen.

Als vertiefende Literatur sei neben dem Standardwerk [Vapnik 2000] vor allem [Burges 1998] und [Cristianini und Shawe-Taylor 2006] empfohlen.

3.3.1 Herleitung

Bleiben wir zunächst beim linearen Fall. Sei also eine endliche Menge $X = \{\vec{x}_1, \ldots, \vec{x}_n\} \subseteq \mathbb{R}^p$ von n Beispielen gegeben. Die Menge $Y := \{-1; +1\}$ definiert die Labelmenge. Jedes Beispiel $\vec{x}_i \in X$ ist eindeutig einem Label $y_i \in Y$ zugeordnet, woraus sich die beiden Mengen (Klassen) $C_+ := \{(\vec{x}, y) | \vec{x} \in X, y = +1\}$ und $C_- := \{(\vec{x}, y) | \vec{x} \in X, y = -1\}$ ergeben. Man nehme an, C_+ und C_- seien linear separierbar. Es existiert also eine lineare Hyperebene, die beide Klassen trennt. Zunächst ist diese Hyperebene allerdings nicht eindeutig bestimmt, siehe Abbildung 3.4 links. Man könnte sich überlegen, einfach die Mittelpunkte (Zentroide) beider Klassen zu verbinden und die Hyperebene zu nehmen, die senkrecht durch die Mitte dieser Verbindungslinie geht. An Abbildung 3.3 wird allerdings schnell deutlich, dass die so bestimmte Hyperebene schlecht generalisiert. In Abbildung 3.3a ist die Hyperebene beispielsweise so nah an C_-, dass ungesehene (negative) Beispiele, die nur etwas weiter rechts liegen, falsch klassifiziert würden. Abbildung 3.3b klassifiziert bereits von Beginn an falsch, obwohl eine lineare Trennung möglich wäre.

Die Aufgabenstellung lautet nun also, die optimal separierende Hyperebene zu finden. Optimal bedeutet in diesem Zusammenhang, dass die Hyperebene den größtmöglichen Abstand zu beiden Klassen besitzt, sie also einen maximalen Rand hat (Maximum Margin Methode).

Definition 3.3.1. (Optimal separierende Hyperebene) Eine separierende Hyperebene H heißt optimal, wenn ihr Abstand d zum nächsten positiven und negativen Beispiel maximal ist (siehe Abbildung 3.4).

Diejenigen Beispiele, die die Lage der Hyperebene bestimmen, nennt man Stützvektoren. Man sieht direkt, dass alle anderen Beispiele keine Auswirkungen auf die resultierende

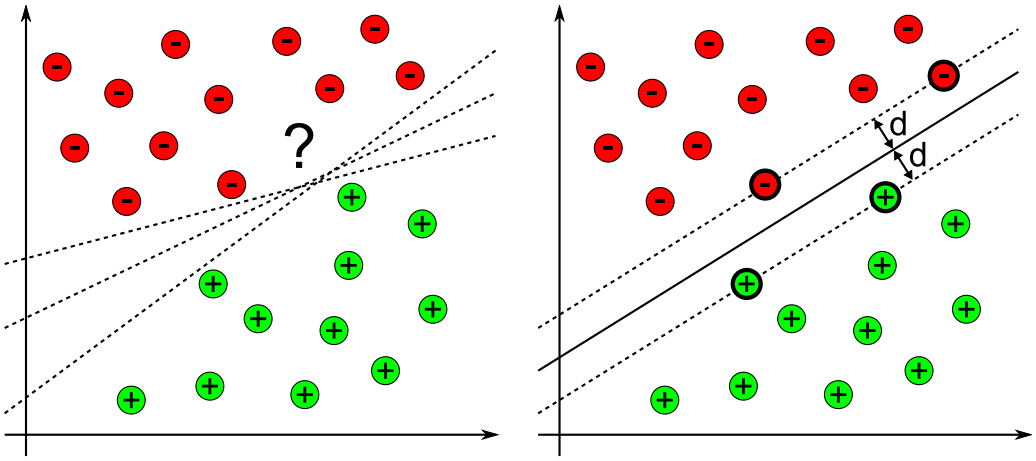


Abbildung 3.4: Trennende Hyperebenen im $\mathbb{R}^2$. Diese sind zunächst nicht eindeutig bestimmt (links). Im Bild rechts ist die optimal trennende Hyperebene eingezeichnet. Die Beispiele, die die Ebene definieren (Stützvektoren), sind fett umrandet.

Hyperebene haben. Selbst wenn diese Beispiele fehlen würden oder noch viele weitere Beispiele existierten, die nicht näher zur Hyperebene liegen würden als die Stützvektoren, würde das Verfahren dieselbe Hyperebene liefern.

Formal ist also eine Hyperebene H der Form $\langle \vec{w}, \vec{x} \rangle + b = 0$ gesucht, wobei $\vec{w} \in \mathbb{R}^p$ den Normalenvektor darstellt und $b \in \mathbb{R}$ ein Skalar ist. Ist diese Hyperebene gefunden, so lassen sich später ungesehene Beispiele $\vec{x}$ leicht auf folgende Weise klassifizieren.

$$\hat{y} = \text{sign}(\langle \vec{x}, \vec{w} \rangle + b) \tag{3.15}$$

Der Abstand der Hyperebene zum Ursprung ist $\frac{|b|}{||\vec{w}||}$, wobei $||\vec{w}||$ der euklidischen Norm von $\vec{w}$ entspricht. Angenommen, H sei eine (beliebige) trennende Hyperebene und alle Beispiele erfüllen die Ungleichungen (ggf. Skalieren von $\vec{w}$ und b)

$$\langle \vec{x}_i, \vec{w} \rangle + b \geq +1 \qquad \text{für } y_i = +1 \tag{3.16}$$

$$\langle \vec{x}_i, \vec{w} \rangle + b \leq -1 \qquad \text{für } y_i = -1 \tag{3.17}$$

welche sich zusammenfassen lassen zu

$$y_i(\langle \vec{x}_i, \vec{w} \rangle + b) - 1 \geq 0 \quad \forall i \tag{3.18}$$

Dann definieren die obigen Ungleichungen 3.16 und 3.17 zwei Halbräume, in deren Zwischenraum sich keine Beispiele befinden. Diejenigen Beispiele, für die in den Ungleichungen 3.16 und 3.17 Gleichheit herrscht, liegen genau auf den Hyperebenen $H_+ : \langle \vec{x}_i, \vec{w} \rangle + b = +1$ und $H_- : \langle \vec{x}_i, \vec{w} \rangle + b = -1$, die die Halbräume begrenzen. H_+ hat Abstand $\frac{|1-b|}{||\vec{w}||}$ zum Ursprung, H_- den Abstand $\frac{|-1-b|}{||\vec{w}||}$. Demnach sind beide Hyperebenen im Abstand $\frac{2}{||\vec{w}||}$ parallel zueinander. Die optimal separierende Hyperebene muss also dazwischen im Abstand von $\frac{1}{||\vec{w}||}$ zu H_+ und H_- liegen. Es genügt also, $\frac{1}{||\vec{w}||}$ zu maximieren, beziehungsweise $||\vec{w}||^2$ zu minimieren, natürlich unter den Bedingungen 3.18.

Leider ist dieses Optimierungsproblem in seiner ursprünglichen Form schlecht mit numerischen Methoden zu lösen. Die Anwendung der Lagrange-Methode verschafft hier Abhilfe. Allgemein lassen sich mit dieser Methode Nebenbedingungen der Art

$$g_i(x) \leq 0 \qquad \text{und} \qquad h_j(x) = 0$$

behandeln, indem sie zur zu optimierenden Funktion hinzugefügt werden. Wir erhalten so Nebenbedingungen auf den sogenannten Lagrange-Multiplikatoren, welche deutlich einfacher zu handhaben sind. Außerdem hat die Umformulierung zur Folge, dass die Trainingsbeispiele ausschließlich in Skalarprodukten auftauchen, was später den Kernel-Trick für nichtlineare Trennungen erst möglich macht. Sei $f(x)$ nun eine zu minimierende Funktion. Dann lautet das Minimierungsproblem nach Lagrange somit

$$\min f(x) + \sum_i \alpha_i g_i(x) + \sum_j \mu_j h_j(x) \quad \text{mit } \alpha_i, \mu_i \geq 0 \forall i, j \tag{3.19}$$

Die α_i und μ_i heißen auch *Lagrange-Multiplikatoren*. Im Falle der Stützvektormethode entspricht die Minimierung von $f(x)$ der Minimierung von $||\vec{w}||^2$. Die Nebenbedingungen $g_i(x)$ entsprechen den Nebenbedingungen 3.18, die $h_i(x)$ sind hingegen nicht erforderlich, da in der ursprünglichen Formulierung des Stützvektor-Optimierungsproblems keine Gleichheits-Nebenbedingungen auftreten. Es ergibt sich für uns die Lagrange-Funktion

$$L(\vec{w}, \vec{\alpha}) = f(\vec{w}) - \sum_i \alpha_i g_i(\vec{w}) \tag{3.20}$$

mit $\vec{\alpha} = (\alpha_1, \ldots, \alpha_n)$ und n der Anzahl der Beispiele. Dabei muss gelten, dass die $\alpha_i \geq 0$ sind. Setzt man nun f und die g_i ein, so erhält man das primale Problem der Stützvektormethode:

Definition 3.3.2. (Primales Problem) Die Funktion

$$L_P(\vec{w}, b, \vec{\alpha}) = \frac{1}{2}||\vec{w}||^2 - \sum_{i=1}^{n} \alpha_i \left(y_i(\langle \vec{x}_i, \vec{w} \rangle + b) - 1\right) \tag{3.21}$$

soll bezüglich $\vec{w}$ und b minimiert und bezüglich der α_i maximiert werden.

Bilden der partiellen Ableitungen nach $\vec{w}$ und b und Nullsetzen derselben führen zu den *Karush-Kuhn-Tucker* (KKT) Bedingungen:

$$\sum_i \alpha_i y_i \vec{x}_i = \vec{w} \tag{3.22}$$

$$\sum_i \alpha_i y_i = 0 \tag{3.23}$$

$$\alpha_i \geq 0 \qquad \forall i = 1, \ldots, n \tag{3.24}$$

$$\alpha_i \left(y_i(\langle \vec{x}_i, \vec{w} \rangle + b) - 1\right) = 0 \qquad \forall i = 1, \ldots, n \tag{3.25}$$

Durch Umformen des primalen Problems und Einsetzen der KKT-Bedingungen gelangt man schließlich zur Formulierung des SVM-Optimierungsproblems als

Definition 3.3.3. (Duales Problem) Maximiere

$$L_D(\vec{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \langle \vec{x}_i, \vec{x}_j \rangle \tag{3.26}$$

unter den Bedingungen

$$\alpha_i \geq 0 \ \forall i = 0, \ldots, n \quad \text{und} \ \sum_{i=1}^{n} \alpha_i y_i = 0$$

Aus den KKT-Bedingungen folgt, dass es für jedes Beispiel $\vec{x}_i$ genau ein α_i gibt mit

$\alpha_i = 0$	falls $\vec{x}_i$ im richtigen Halbraum liegt
$\alpha_i > 0$	falls $\vec{x}_i$ auf der Hyperebene H_+ oder H_- liegt

Offensichtlich sind die Beispiele mit $\alpha_i > 0$ genau die Stützvektoren. Wenn die optimalen α_i letztendlich bekannt sind, kann man $\vec{w}$ und b mit Hilfe der KKT-Bedingungen 3.22 und 3.25 (für ein beliebiges i mit $\alpha_i > 0$) bestimmen.

Wie aber gelangt man nun an die optimalen α_i? Nachdem wir nun ein Problem der quadratischen Programmierung haben, bieten sich z.B. folgende Verfahren an.

- Numerische Verfahren (*quadratic problem solver*)
- *Sequential Minimal Optimization* (*SMO*, [Platt 1998] und [Keerthi und Shevade 2003])
- Evolutionäre Algorithmen (*EvoSVM*, [Mierswa 2006])

Details zur Funktionsweise der Verfahren würden den Rahmen dieser Arbeit sprengen und sind der entsprechenden Literatur zu entnehmen. Als Schranke für die Lösung des quadratischen Optimierungsproblems gilt hier $O(N^3)$.

3.3.2 SVM mit weicher Trennung

Da in der Praxis linear trennbare Daten selten sind, muss die Stützvektormethode natürlich auch mit Fehlklassifikationen umgehen können. So lange Beispiele zu entfernen, bis die übrigen linear trennbar werden, macht das Verfahren allerdings exponentiell. Stattdessen führt man Strafterme ξ_i für jedes Beispiel ein [Cortes und Vapnik 1995].

$\xi_i = 0$	für korrekt klassifizierte Beispiele
$\xi_i > 0$	falls Beispiel falsch klassifiziert oder zwischen H_+ und H_- liegt

Aus dem ursprünglichen Problem, $||w||^2$ zu minimieren, wird nun also das folgende Minimierungsproblem:

Definition 3.3.4. (Relaxiertes Optimierungsproblem)

Sei $C \in \mathbb{R}$, $C > 0$ fest.
Minimiere

$$||\vec{w}||^2 + C \sum_{i=1}^{n} \xi_i$$

unter den Nebenbedingungen

$$\begin{aligned} \langle \vec{x}_i, \vec{w} \rangle + b &\geq +1 - \xi_i \qquad \text{für } y_i = +1 \\ \langle \vec{x}_i, \vec{w} \rangle + b &\leq -1 + \xi_i \qquad \text{für } y_i = -1 \end{aligned}$$

beziehungsweise zusammengefasst:

$$y_i(\langle \vec{x}_i, \vec{w} \rangle + b) \geq 1 - \xi_i \quad \forall i$$

Das duale quadratische Optimierungsproblem für SVMs mit weicher Trennung gleicht dem für „harte“ Trennungen, wobei die Bedingungen $\alpha_i \geq 0$ durch $C \geq \alpha_i \geq 0$ ersetzt werden. Die Variable C ist vom Anwender zu wählen, wobei größere Werte für C die Fehler stärker bestraft.

3.3.3 Kernfunktionen

Liegen nun Daten vor, die zwar nicht linear, aber möglicherweise auf andere Art und Weise trennbar sind, so werden die Daten üblicherweise in einen anderen, gegebenenfalls sogar unendlichdimensionalen Raum $\mathcal{H}$ transformiert. Dabei wird nicht wie sonst eine Transformationsfunktion $\Phi : \mathbb{R}^d \mapsto \mathcal{H}$ benutzt, sondern ausgenutzt, dass die Beispiele in der Formulierung 3.26 des SVM Optimierungsproblems ausschließlich in einem Skalarprodukt auftauchen. Man führt nun eine Kernfunktion

$$K(\vec{x}_i, \vec{x}_j) = \langle \Phi(\vec{x}_i), \Phi(\vec{x}_j) \rangle \tag{3.27}$$

ein, die implizit die Transformation und die Berechnung des Skalarproduktes durchführt, ohne dass die Transformationsfunktion Φ notwendigerweise bekannt sein muss. Dieser „Kernel-Trick“ ist nicht neu, funktioniert aber bei der Stützvektormethode hervorragend. Im Trainingsalgorithmus muss demnach nur jede Stelle, an der $\langle \vec{x}_i, \vec{x}_j \rangle$ auftaucht, durch $K(\vec{x}_i, \vec{x}_j)$ ersetzt werden. Die Stützvektormethode trennt nach wie vor linear, jedoch in einem anderen, meist höherdimensionalen Raum (siehe Abbildung 3.5).

Die Benutzung von Kernfunktionen stellt auch bei der Klassifizierung ungesehener Beispiele kein Problem dar. Angenommen, Φ sei die der Kernfunktion entsprechenden Trans-

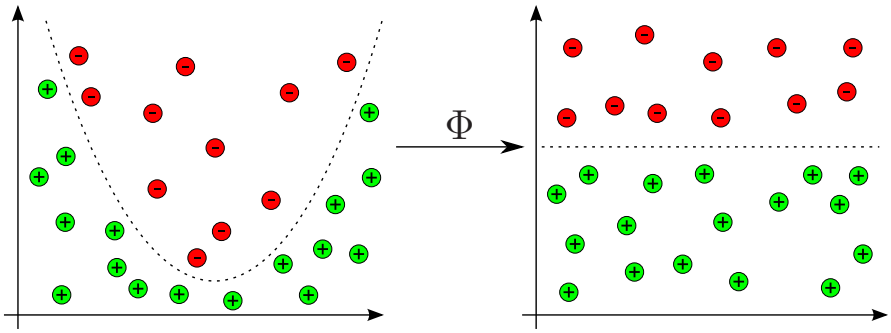


Abbildung 3.5: Die Kernel-Methode führt implizit eine Transformation via Φ in einen anderen, höherdimensionalen Raum durch, in welchem die SVM linear trennen kann.

formationsfunktion. Anwendung der KKT-Bedingung 3.22 auf die Klassifikationsfunktion 3.15 ergibt

$$\begin{aligned}\hat{y} &= \text{sign}(\langle \vec{x}, \vec{w} \rangle + b) \\ &= \text{sign}(\langle \vec{x}, \sum_i \alpha_i y_i \vec{x}_i \rangle + b) \\ &= \text{sign}(b + \sum_i \alpha_i y_i \langle \vec{x}_i, \vec{x} \rangle) \\ &\widehat{=} \text{sign}(b + \sum_i \alpha_i y_i K(\vec{x}_i, \vec{x}))\end{aligned} \tag{3.28}$$

Einige typische Beispiele für Kernfunktionen sind

- **Linear:** $K(\vec{x}_i, \vec{x}_j) = \langle \vec{x}_i, \vec{x}_j \rangle$
- **Polynomiell:** $K(\vec{x}_i, \vec{x}_j) = \langle \vec{x}_i, \vec{x}_j \rangle^d$
- **Radial-Basisfunktion:** $K(\vec{x}_i, \vec{x}_j) = e^{-\gamma ||\vec{x}_i - \vec{x}_j||^2}$
- **Neuronale Netze (Sigmoid-Funktion):** $K(\vec{x}_i, \vec{x}_j) = \tanh(\alpha \langle \vec{x}_i, \vec{x}_j \rangle + b)$
- **Gauss:** $K(\vec{x}_i, \vec{x}_j) = e^{-\frac{||\vec{x}_i - \vec{x}_j||^2}{2\sigma^2}}$
- sowie Produkte und Summen von anderen Kernfunktionen (siehe [Schölkopf et al. 1998])

Als Kernfunktion kann allerdings nicht einfach jede beliebige Funktion herhalten. In der Literatur findet man zwangsläufig die Bedingung, dass der Raum $\mathcal{H}$, in den die Funktion Φ abbildet, ein Hilbertraum sein muss. Hilberträume sind vollständige Vektorräume mit einem Skalarprodukt (Innenprodukt). Es lässt sich zeigen, dass man eine Transformationsfunktion Φ konstruieren kann, welche die Gleichung 3.27 erfüllt, falls die Kernmatrix $(K(\vec{x}_i, \vec{x}_j))_{i,j=i\ldots n}$ symmetrisch positiv definit ist, wenn also gilt $x^T K x > 0$ für alle $x \in \mathbb{R}^n$, $x \neq 0$. Eine etwas andere Formulierung der Zulässigkeit einer Funktion als Kernfunktion ist in den Bedingungen von Mercers Theorem zu finden [Mercer 1909].

Auf diese Weise lassen sich nun also fast beliebige Mengen trennen, wobei es einem Anwender natürlich auch frei steht, eigene Kernfunktionen zu entwerfen. Wir werden später sehen, wie sich eine Kernfunktion, die sich Eigenschaften der Wavelets zu Nutze macht, auf Zeitreihendaten verhält und die Ergebnisse mit denen anderer Kernfunktionen vergleichen.

3.3.4 Stützvektor-Regression

Die Stützvektormethode eignet sich nicht nur zur Klassifikation zweier Klassen. Neben der Erweiterung auf eine beliebige Anzahl von Klassen wird die Stützvektormethode auch als Regressionsverfahren verwendet [Smola 1996]. Hier wird zwar keine *trennende* Hyperebene gesucht, dafür aber eine, die möglichst dicht an den Beispielen liegt. Es sei nach wie vor eine Menge $X = \{\vec{x}_1, \ldots, \vec{x}_n\} \subseteq \mathbb{R}^p$ von n Beispielen gegeben. Die Menge Y entspricht nun allerdings $\mathbb{R}$. Die Trainingsdaten bestehen also aus Paaren $\{(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n)\} \subseteq \mathbb{R}^p \times \mathbb{R}$. Ein ausführliches Tutorial bietet [Smola und Schölkopf 2004], in dieser Arbeit werden lediglich die Grundideen und die wichtigsten Ergebnisse präsentiert. Wie man die Stützvektor-Regression auch zur Prognose benutzen kann, zeigt [Muller et al. 1997].

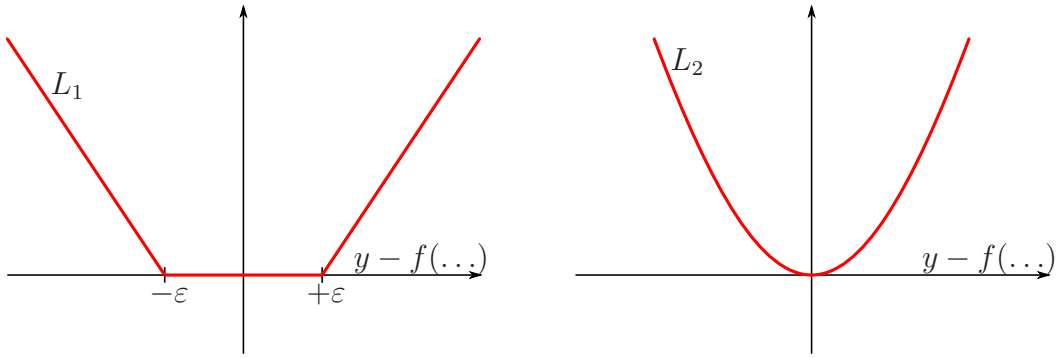
Definition 3.3.5. (ε-Stützvektor-Regression (ε-SVR)) Gegeben Trainingsdaten $\{(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n)\} \subseteq \mathbb{R}^p \times \mathbb{R}$. Dann sucht die ε-SVR eine approximierende Funktion $f(x)$, die höchstens um ε von den gegebenen y_i abweicht und gleichzeitig so „flach“ wie möglich ist.

Flach in Definition 3.3.5 bedeutet dabei, dass die Steigung der Kurventangente in jedem Punkt minimal ist. Dies lässt sich z.B. durch Minimierung der Norm $||\vec{w}||^2 = \langle \vec{w}, \vec{w} \rangle$ erreichen, so dass sich daraus das folgende konvexe Optimierungsproblem ergibt:

$$\begin{aligned} &\text{minimiere} && \frac{1}{2}||\vec{w}||^2 \\ &\text{unter den Bedingungen} && \begin{cases} y_i - \langle \vec{w}, \vec{x}_i \rangle - b & \leq \varepsilon \\ \langle \vec{w}, \vec{x}_i \rangle + b - y_i & \leq \varepsilon \end{cases} \end{aligned} \tag{3.29}$$

Natürlich muss eine solche Funktion nicht zwangsläufig existieren. Daher sollen Fehler zugelassen werden. Analog zu Kapitel 3.3.2 werden Strafterme ξ_i und ξ_i' eingeführt, um die ansonsten unmöglich erfüllbaren Bedingungen handhabbar zu machen. So kommen wir zu folgender Formulierung des Optimierungsproblems [Vapnik 2000]:

$$\begin{aligned} &\text{minimiere} && \frac{1}{2}||\vec{w}||^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i') \\ &\text{unter den Bedingungen} && \begin{cases} y_i - \langle \vec{w}, \vec{x}_i \rangle - b & \leq \varepsilon + \xi_i \\ \langle \vec{w}, \vec{x}_i \rangle + b - y_i & \leq \varepsilon + \xi_i' \\ \xi_i, \xi_i' & \geq 0 \end{cases} \end{aligned} \tag{3.30}$$



(a) ε-insensitive **(b)** quadratisch

Abbildung 3.6: Beispiele zweier Loss-Funktionen für die Stützvektor-Regression

Die Konstante C stellt dabei den Kompromiss zwischen der *Flachheit* der Funktion dar und bis zu welchem Grad Abweichungen größer als ε vom Zielwert toleriert werden.

Die Berechnung der Strafterme entspricht der Einführung einer *Loss-Funktion*

$$L_k(y, f(\vec{x}, \alpha)) = \begin{cases} 0 & \text{falls } y - f(\vec{x}, \alpha) \leq \varepsilon \\ (y - f(\vec{x}, \alpha) - \varepsilon)^k & \text{sonst} \end{cases} \tag{3.31}$$

Je nach Wahl von k und ε erhält man natürlich andere Loss-Funktionen, beispielsweise die ε-insensitive ($k = 1$, $\varepsilon > 0$, siehe Abbildung 3.6a) oder die quadratische ($k = 2$, $\varepsilon = 0$, siehe Abbildung 3.6b) Loss-Funktion.

Wie bei der Stützvektor-Klassifikation kommt man auch bei der SV-Regression wieder zu einer Formulierung als duales Problem.

Definition 3.3.6. (Duales Problem der Stützvektor-Regression) Man beachte, dass durch die Einführung von jeweils ξ_i und ξ_i' pro Beispiel auch je zwei α-Werte bestimmt werden müssen. Maximiere

$$L_D(\vec{\alpha}, \vec{\alpha}') = \sum_{i=1}^{n} y_i(\alpha_i - \alpha_i') - \varepsilon \sum_{i=1}^{n}(\alpha_i + \alpha_i') - \frac{1}{2}\sum_{i,j=1}^{n}(\alpha_i - \alpha_i')(\alpha_j - \alpha_j')\langle \vec{x}_i, \vec{x}_j \rangle \tag{3.32}$$

unter den Bedingungen

$$\sum_{i=1}^{n}(\alpha_i - \alpha_i') = 0 \quad \text{und} \quad 0 \leq \alpha_i, \alpha_i' \leq C$$

Entsprechend Gleichung 3.28 werden bei der Stützvektor-Regression ungesehene Beispiele $\vec{x}$ wie folgt interpretiert

$$f(\vec{x}) = \sum_{i=1}^{n}(\alpha_i - \alpha_i')\langle \vec{x}_i, \vec{x} \rangle + b$$

beziehungsweise im nichtlinearen Fall unter Verwendung von Kernfunktionen $K(\cdot,\cdot)$

$$f(\vec{x}) = \sum_{i=1}^{n}(\alpha_i - \alpha_i')K(\vec{x}_i, \vec{x}) + b \tag{3.33}$$

Dass das Verfahren der Stützvektor-Regression durchaus gegen andere Verfahren wie polynomielle und rationale Approximation, lokale polynomielle Techniken, Radial-Basis Funktionen und Neuronale Netze antreten kann und oftmals sogar bessere Ergebnisse erzielt, zeigt [Mukherjee et al. 1997]. Daher betrachten wir es in dieser Arbeit auch als viel versprechenden Kandidaten eines performanten und robusten Lernverfahrens.

3.3.5 Strukturelle Risikominimierung

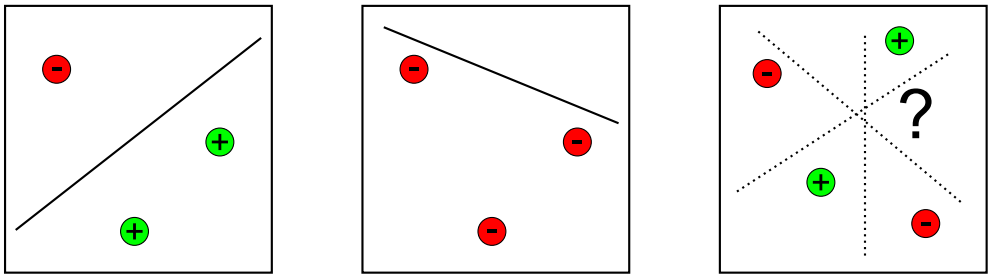
Beschäftigt man sich etwas mehr mit der Theorie und den Hintergründen zur Stützvektormethode, so taucht immer wieder der Begriff der *strukturellen Risikominimierung* auf. Was hat es damit auf sich? Um dies zu verstehen, muss sich zunächst überlegen, was eigentlich mit *Risiko* gemeint ist. Gemeint ist das Risiko, mit seinem Lernverfahren einen Fehler zu machen. Denkt man an einfache Lernverfahren, so wird dieser Fehler in der Regel empirisch, d.h. anhand der vorhandenen Trainings- und Testdaten gemessen. Bei linearen Modellen wird beispielsweise häufig die Berechnung der *Residual Sum of Squares* verwendet.

$$\mathrm{RSS} = \sum_{i=1}^{n}(y_i - f(\vec{x}_i))^2$$

Wenn mehrere Modelle mit minimalem empirischen Fehler existieren muss allerdings nach anderen Kriterien das „Beste“ ausgewählt werden. Die Stützvektormethode führt an dieser Stelle ein verbessertes Kriterium, die Suche nach der Maximum-Margin Hyperebene, ein. Zugleich sorgt sie allerdings auch dafür, dass die Komplexität des Modells so gering wie möglich gehalten wird. Als Kontrastbeispiel stelle man sich einen k-Nearest Neighbors Klassifizierer mit $k = 1$ vor. Dieser macht auf den Trainings- und Testdaten keinen Fehler, das empirische Risiko $R_{\mathrm{emp}}(\vec{\alpha})$ beträgt also Null, jedoch zu Lasten einer hohen Modellkomplexität. $\vec{\alpha}$ ist in diesem Zusammenhang der Vektor der Parameter des Modells, welche die Trainingsphase ermittelt.

Interessiert ist man bei einem Lernverfahren eigentlich am *wahren* Risiko $R(\vec{\alpha})$, dem durchschnittlichen Fehler bzgl. der wahren Verteilung aller, auch der ungesehenen Daten. Leider ist „wahr“ in der Statistik oft gleichbedeutend mit „unbekannt“, so dass lediglich das empirische Risiko berechnet werden kann. [Vapnik 2000] hat nun eine obere Schranke für das wahre Risiko bzgl. des empirischen Risikos und der sogenannten VC-Dimension aufgestellt.

Satz 3.3.7 (Risikoschranke nach Vapnik)**.** *Gegeben eine unbekannte Wahrscheinlichkeitsverteilung $P(\vec{x}, y)$, nach der Daten ausgewählt werden. Die Abbildungen $\vec{x} \to f(\vec{x}, \vec{\alpha})$*



(a) 3 Punkte **(b)** 3 Punkte **(c)** 4 Punkte

Abbildung 3.7: Abbildung (a) und (b) zeigen *eine* Menge von 3 Punkten im $\mathbb{R}^2$ und wie sie bei jeder möglichen Markierung von einem linearen Klassifizierer getrennt werden können (andere Markierungen als die beiden dargestellten überlege man sich analog). Abbildung (c) zeigt stellvertretend für *alle* Anordnungen von 4 Punkten im $\mathbb{R}^2$, dass man stets eine Markierung der Punkte finden kann, so dass der lineare Klassifizierer die beiden Mengen nicht mehr fehlerfrei trennen kann.

werden dadurch gelernt, dass $\vec{\alpha}$ bestimmt wird. Mit einer Wahrscheinlichkeit von $1-\mu$ ist das Risiko $R(\vec{\alpha})$ nach dem Sehen von n Beispielen beschränkt und es gilt

$$R(\vec{\alpha}) \leq R_{\text{emp}}(\vec{\alpha}) + \underbrace{\sqrt{\frac{\eta\left(\log\left(\frac{2n}{\eta}\right)+1\right)-\log\left(\frac{\mu}{4}\right)}{n}}}_{\text{VC-Confidence}}$$

Diese Schranke hat interessante Eigenschaften. Zum einen ist sie unabhängig von $P(\vec{x}, y)$. Es wird lediglich gefordert, dass Trainings- und Testdaten bezüglich *derselben* Wahrscheinlichkeitsverteilung ausgewählt werden. Zum anderen ist es in der Regel nicht möglich, die linke Seite direkt auszurechnen. Falls η bekannt ist, kann allerdings die rechte Seite leicht bestimmt werden.

Die Variable η wird auch Vapnik-Chervonenkis Dimension (VC Dimension) genannt. Sie beschreibt in gewisser Weise die Kapazität eines Lernverfahrens.

Definition 3.3.8. (VC Dimension) Sei $f(\vec{x}, \vec{\alpha})$ ein Klassifikationsmodell mit Parametern $\vec{\alpha}$ auf Datenpunkten $\vec{x}$. $f(\vec{x}, \vec{\alpha})$ „*zerschmettert*“ Datenpunkte $\{\vec{x}_1, \ldots, \vec{x}_n\}$, wenn für jede mögliche Belegung der Labels y_i der Datenpunkte $\vec{x}_i$ eine Belegung der Parameter $\vec{\alpha}$ existiert, so dass f die Datenpunkte fehlerfrei trennt. Die *VC Dimension* eines Modells f ist das größte η, so dass es eine Menge von η Datenpunkten gibt, die von f zerschmettert werden kann.

Zur Veranschaulichung der Bedeutung der VC Dimension nehme man den $\mathbb{R}^2$ als Beispiel. Ein linearer Klassifizierer soll positiv und negativ markierte Punkte im $\mathbb{R}^2$ trennen. Bei einer gegebenen Anzahl von Punkten muss es nun mindestens eine Verteilung der Punkte im $\mathbb{R}^2$ geben, so dass der Klassifizierer jede mögliche Belegung der Markierungen linear

trennen kann. Ist dies der Fall, so ist die VC Dimension des Klassifizierers mindestens gleich der Kardinalität der Punkte. Abbildungen 3.7a und 3.7b zeigen diese Situation für 3 Punkte. Natürlich können 3 Punkte auch so angeordnet werden, dass eine lineare Trennung unmöglich wird, hier genügt aber die Tatsache, dass es eine Anordnung *gibt*, bei der die lineare Trennung stets möglich ist. Abbildung 3.7c zeigt den Fall bei 4 Punkten. Egal, wie die 4 Punkte angeordnet sind, es lässt sich stets eine Belegung der Markierungen finden, so dass der Klassifizierer nicht mehr linear trennen kann. Die VC Dimension dieses Klassifizierers im $\mathbb{R}^2$ ist also mindestens 3 und kleiner als 4, demnach also genau gleich 3. Allgemein lässt sich zeigen:

Satz 3.3.9. *Die VC Dimension der Hyperebenen im $\mathbb{R}^p$ ist $p + 1$.*

Die VC Dimension ist nicht abhängig von der Anzahl der Parameter, eine Funktion mit nur einem Parameter kann beispielsweise unendliche VC Dimension haben. Somit ist die VC Dimension eine Kennzahl für die Kapazität des Lernverfahrens. Für die Stützvektormethode lässt sich zeigen:

Satz 3.3.10. *Gegeben Beispiele $\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^p$ mit $||\vec{x}_i|| < D \quad \forall i$. Für die VC Dimension der durch den Vektor $\vec{w}$ gegebenen optimalen Hyperebene H gilt:*

$$\text{VCdim}(H) \leq \min\{D^2||\vec{w}||^2, p\} + 1$$

Die Stützvektormethode minimiert somit nicht nur das empirische Risiko, sondern auch das strukturelle. Es wird die einfachste Hypothese gewählt, die noch an die Daten anpassbar ist.

4 Datentransformationen und Vorverarbeitung

In den letzten Kapiteln wurden Lernverfahren für Zeitreihen vorgestellt. In [Rüping und Morik 2003] wurde jedoch gezeigt, dass man das Lernergebnis deutlich verbessern kann, wenn man der Vorverarbeitung der Daten mehr Aufmerksamkeit schenkt. Zwei ausgewählte Vorverarbeitungsschritte, die in den Experimenten dieser Arbeit genutzt werden, sollen hier exemplarisch erläutert werden. Die Wavelet-Analyse wird in den Versuchsreihen dabei nicht nur zur Vorverarbeitung, sondern ebenso zur Nachbearbeitung der Ergebnisse verwendet. Außerdem stellt sie die grundlegende Theorie der später eingesetzten Wavelet-Kernfunktionen dar.

4.1 Fensterung

Methoden wie die Stützvektormethode benötigen Datensätze, die aus mindestens einer unabhängigen und genau einer abhängigen Variable (das **Label**) bestehen. Letztere soll dabei in Abhängigkeit der unabhängigen Variablen (**Features**) bestimmt werden. Das resultierende Modell soll letztendlich dazu in der Lage sein, allein aus den Features das korrekte Label zu bestimmen. Bei der Zeitreihenprognose entspricht das Label einem Wert einer Zeitreihe. Im einfachsten Fall dient als einziges Feature die Zeit, ist doch die Zeitreihe nichts anderes als eine Funktion der Zeit. Funktionen, die nicht ausschließlich von der Zeit abhängen, lassen sich auf diese Weise allerdings nur schlecht modellieren, weshalb man hier einen Trick anwendet, die sogenannte Fensterung, um künstlich mehr Features zu erzeugen und somit die Muster erkennende Eigenschaft der Lernverfahren auszunutzen.

Bei der klassischen Fensterung, auch „Phase Space“ Repräsentation [Mei-Ying und Xiao-Dong 2005] oder „*Windowing*“ genannt, werden zusammenhängende Ausschnitte der Zeitreihe der Breite w betrachtet. Zusammen mit einem weiteren Wert der Reihe im Abstand von h (Horizont) zum Fenster ergibt jedes Fenster ein Beispiel für das anzuwendende Lernverfahren. Verschiebt man das Fenster um s (Schrittweite) Elemente nach vorn, erhält man ein weiteres Beispiel. Insgesamt ergeben sich so $n - (w + h) + 1$ Beispiele. Der Wert im Abstand von h zum Fenster eines Beispiels wird dabei für Verfahren wie die Stützvektormethode als Label gekennzeichnet. Es wird also beispielsweise bei einem Horizont von 1 unter Betrachtung der vorangegangenen w Werte der jeweils nachfolgende vom Verfahren „gelernt“. Bei nicht zu kleiner Fenstergröße w liefert dieses Verfahren Modelle, die sich für Vorhersagen eignen [Takens et al. 1981].

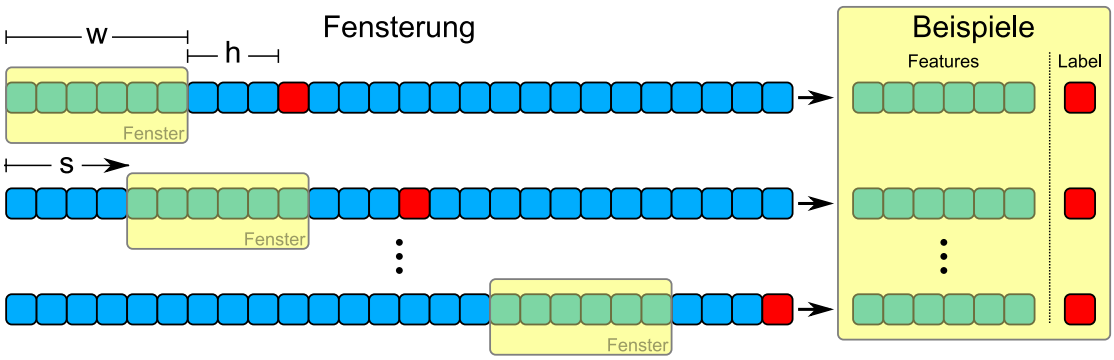


Abbildung 4.1: Fensterung einer Zeitreihe. Fensterbreite w, Horizont h, Schrittweite s

Abbildung 4.1 zeigt nicht nur die Prozedur anschaulich, sondern verdeutlicht auch gleich einen der Vorteile des Verfahrens. Obwohl gut auf Zeitreihen anwendbar, sind die erzeugten Beispiele losgelöst von der Definition der Zeitreihe. Einerseits spielt die Reihenfolge der Features, im Gegensatz zu einer Zeitreihe, für das Lernverfahren keine Rolle mehr. Andererseits sind natürlich auch andere Features als die unmittelbaren Werte der Reihe denkbar. Zusatzinformationen über die dargestellten Ausschnitte können dem Lernverfahren deutlich helfen [Rüping und Morik 2003]. Denkbar sind beispielsweise binäre Kennzeichen, ob sich das Zeitfenster auf einen Ferientag bezieht oder eine Kodierung der Jahreszeit, des Monats oder des Wochentags. Dies ergibt bei Betrachtung der in dieser Arbeit behandelten Zeitreihen unmittelbaren Sinn, so dass diese Art der Fensterung in späteren Experimenten zum Einsatz kommen wird.

4.2 Wavelet-Analyse

Zur Vorverarbeitung von Zeitreihen können zahlreiche Verfahren verwendet werden. Oftmals wird beispielsweise ein Wechsel in den Frequenzraum mit Hilfe der *Fourier-Transformation* durchgeführt [Bracewell und Kahn 1966], [Brigham und Yuen 1978]. Die Zeitreihe wird dabei in einen Vektorraum transformiert, in dem die Basisvektoren Sinusfunktionen sind. Dabei wird ausgenutzt, dass sich jedes periodische Signal aus periodischen, harmonischen Schwingungen verschiedener Phase und Amplitude und genau definierter Frequenz zusammensetzen lässt. Das Ergebnis, die *Fourier-Transformierte*, liefert zu einer Frequenz ihre jeweilige Amplitude im Originalsignal. Somit erhält man das *Frequenzspektrum* eines Signals. Eine ausführliche Abhandlung dieses Themas liefert beispielsweise [Brigham 1988]. Der gravierende Nachteil der Fourier-Transformation ist allerdings, dass die Anteile der Frequenzen an der gesamten Zeitreihe gebildet werden. Eine Lokalisierung auf bestimmte Zeitpunkte oder -intervalle gibt es nicht. Es lässt sich also nicht feststellen, welche Frequenzen in welcher Intensität zu welchem Zeitpunkt vorherrschen.

Um dieses Defizit zu mildern hat [Gabor 1946] die Fourier-Transformation dahingehend modifiziert, dass lediglich kleine Ausschnitte der Zeitreihe analysiert werden (die sogenannte *Fensterung*). Mit Hilfe dieses, *Short-Time Fourier Transform* (STFT) genannten, Verfahrens konnte das Auftreten bestimmter Frequenzen erstmals in der Zeit lokalisiert werden. Siehe zum Beispiel [Griffin und Lim 1984] für eine Anwendung dieses Verfahrens. Hierbei kann es allerdings, je nach Wahl der Implementierung der Fensterung, zu Problemen an den Randstellen der Ausschnitte kommen. Der größte Nachteil stellt allerdings der Kompromiss zwischen Auflösung im Zeit- und im Frequenzbereich dar. Ein breites Fenster liefert hierbei eine hohe Frequenzauflösung bei schlechter Lokalisierung, wohingegen eine schmale Fensterbreite zwar die Genauigkeit der zeitlichen Lokalisierung verbessert, jedoch nicht mehr alle vorhandenen Frequenzen erfassen kann.

Das Problem ist die statische Wahl der Fensterbreite und genau diesen Missstand versucht die Wavelet-Analyse zu beheben [Strang 1993]. Sie vereint die Vorteile der Lokalisierung mit den Vorteilen einer Analyse in verschiedenen Auflösungen. Zur historische Entstehung dieser Theorie sei an dieser Stelle auf [Hubbard 1997] verwiesen, welche ausführlich und auf anschauliche und amüsante Art die Theorie der Wavelets und ihre Entstehung, Anwendung und mathematische Grundlagen beschreibt. In dieser Arbeit beschränken wir uns auf die grundlegenden Ergebnisse, die insbesondere S. Mallat zu verdanken sind, dessen Buch [Mallat 2009] neben [Meyer 1993] eins der vollständigsten zu diesem Thema ist. Als Klassiker unter der Wavelet Literatur sei außerdem [Daubechies 1994] empfohlen. Ingrid Daubechies konstruierte als erste stetige, orthogonale Wavelets mit kompaktem Träger (zur Definition eines kompakten Trägers siehe Kapitel 4.2.1), ohne die bei Computerberechnungen zwangsläufig Rundungsfehler entstehen würden.

4.2.1 Was sind Wavelets?

Wavelets sind Funktionen „kleiner Wellen“. Im Vergleich dazu bestehen die Basisvektoren der Fourier-Transformation aus unendlichen Sinuswellen, also aus „großen Wellen“. Wavelets sind eine natürliche Erweiterung der Fourier-Analyse. Dabei wird das Signal zunächst grob abgetastet, um einen Gesamteindruck zu erhalten. Danach werden die Details mit immer größerer Auflösung analysiert. Man spricht hierbei von Mehrfachauflösung oder auch einem „mathematischen Mikroskop“, denn das Stauchen und Dehnen der Wavelets erlaubt das Erfassen von großen und kleinen Schwingungen.

Mathematisch gesehen muss nach [Percival und Walden 2000] eine Wavelet-Funktion $\psi(d)$ folgende Bedingungen erfüllen:

$$\int_{-\infty}^{\infty} \psi(d)d = 0 \tag{4.1}$$

$$\int_{-\infty}^{\infty} \psi^2(d)d = 1 \tag{4.2}$$

Gleichung 4.1 fordert hierbei, dass die Funktion eine Welle darstellen muss, sich seine positiven und negativen Anteile also aufheben müssen. Gleichung 4.2 bedeutet, dass die Funktion fast überall nahe 0 sein muss. Die Bedingung, dass die Funktion nur auf einem endlichen Intervall ungleich Null sein darf, also einen kompakten Träger hat, wird zwar von einigen üblichen Wavelet Funktionen erfüllt, ist jedoch nicht ausdrücklich gefordert. Eine Funktion wie die Sinusfunktion würde zwar Gleichung 4.1 erfüllen, jedoch in Gleichung 4.2 ein unendliches Integral erzeugen und sich nie zu 1 normalisieren lassen. Einige beispielhafte Wavelet Funktionen zeigt Abbildung 4.2. Auf die Skalierungsfunktionen in der Abbildung wird später noch genauer eingegangen. Welche Wavelet Funktion letztlich genommen wird, hängt von vielen unterschiedlichen Faktoren ab. Als Faustregel gilt allerdings, dass die Form des Wavelets dem Originalsignal möglichst ähnlich sein sollte [Percival und Walden 2000]. Die Namen der Wavelets sind in der Regel zusammengesetzt aus dem Namen ihres Entdeckers und einer Ordnung, wobei höhere Ordnungen einen glatteren Verlauf des Wavelets bedeuten. Das älteste von ihnen ist das *Haar*-Wavelet aus Abbildung 4.2a, dargestellt durch eine Treppenfunktion auf dem Intervall $[0; 1]$.

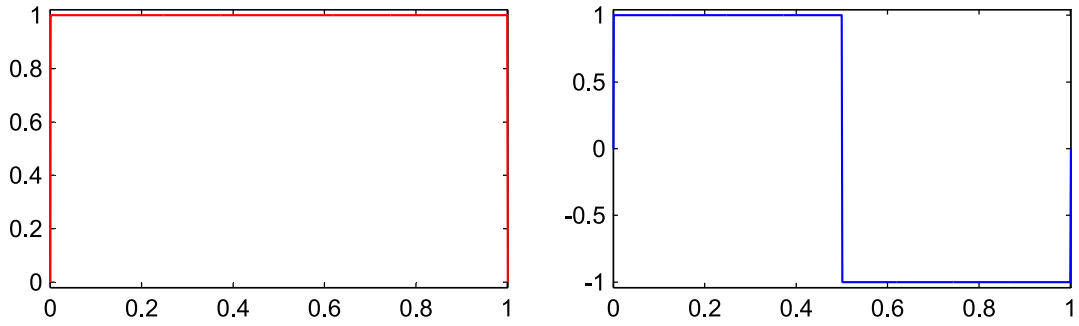
4.2.2 Wavelet Transformation

In ihrer ursprünglichen Definition berechnet man die Wavelet-Transformation eines Signals, indem man das Originalsignal nacheinander mit einer Schar von Wavelets, die alle vom sogenannten „Mutter-Wavelet“ abgeleitet sind, vergleicht. Das Mutter-Wavelet wird dabei verschoben und gestreckt, um die unterschiedlichen Zeitfenster und Auflösungen widerzuspiegeln. Die Multiplikation von Originalsignal und Wavelet mit anschließender Integralbildung ergibt schließlich einen Wavelet-Koeffizienten. Dadurch, dass das Wavelet bis auf einen kleinen Bereich überall nahe Null ist, fließt in die Rechnung hauptsächlich der entsprechende Zeitraum des Signals ein. Im kontinuierlichen Fall wird das Signal $f(t)$ also überführt in eine Funktion c zweier Variablen a (Skala) und b (Zeit), welche die Streckung beziehungsweise Stauchung (*Dilatation*) und die Verschiebung (*Translation*) darstellen:

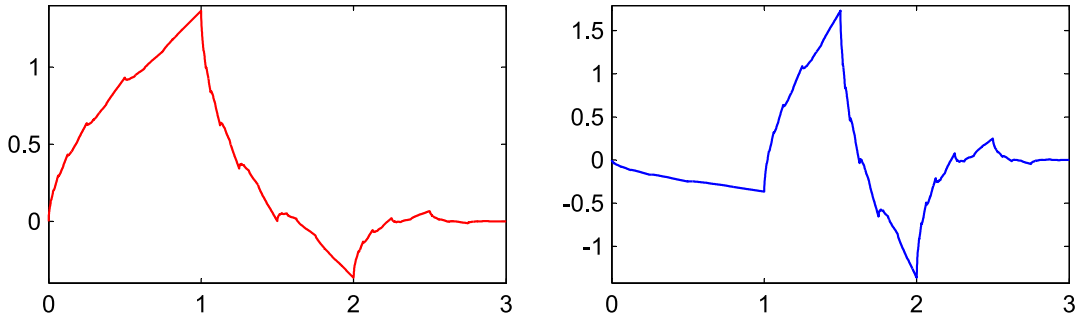
$$c(a, b) = \int_{-\infty}^{\infty} f(t)\psi(at + b)dt \tag{4.3}$$

Die Parameter a und b sind hierbei reelle Zahlen, es wird also beliebig gestaucht und verschoben. Dies wird als *kontinuierliche Wavelet-Transformation* bezeichnet. Diese Art der Berechnung ist in der Praxis allerdings aufwändig und enthält außerdem unendliche Redundanzen. Bei der Datenkompression beispielsweise möchte man Redundanzen nach Möglichkeit verhindern. Die Verwendung von Orthogonaltransformationen gestattet nun einerseits die originalgetreue Rekonstruktion des Signals und verhindert andererseits die Redundanz. Im Gegensatz zur kontinuierlichen Wavelet-Transformation werden bei der *diskreten Wavelet-Transformation* (DWT) außerdem nur Zweierpotenzen als Streckungs- und ganze Zahlen als Verschiebungsfaktoren benutzt [Jensen und La Cour-Harbo 2001]. Die Wavelets haben also die Gestalt

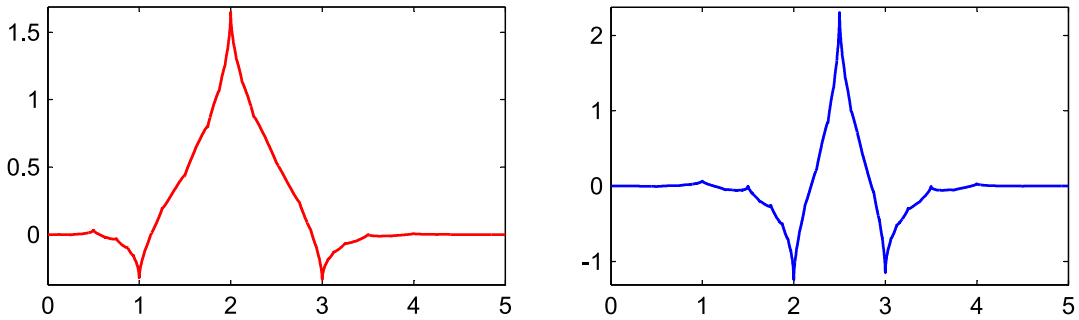
$$\psi(2^k t + l)$$



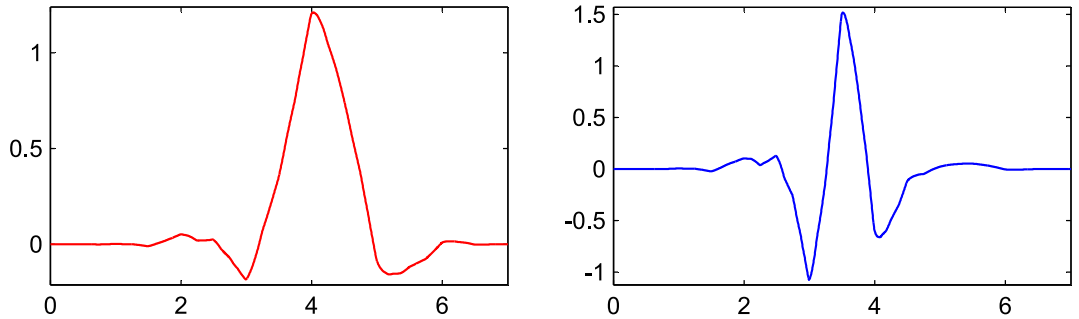
(a) Haar



(b) Daubechies-4



(c) Coiflet-1



(d) Symlet-4

Abbildung 4.2: Einige typische Wavelet Funktionen ψ (rechts) mit ihren entsprechenden Skalierungsfunktionen ϕ (links).

mit ganzzahligem k und l. Bilden die Wavelets sogar eine *Orthonormalbasis*, so vereinfacht sich die Gleichung 4.3 zu einem einfachen Skalarprodukt, was die Berechnungen enorm vereinfacht.

4.2.3 Der Pyramiden-Algorithmus

Um die Berechnungen noch weiter zu vereinfachen wird außerdem eine Idee der 80er Jahre aus der Bildverarbeitung verwendet, bei der nicht mehr jedes Wavelet mit dem Originalsignal verknüpft wird, sondern iterativ mit sich immer wieder halbierenden, geglätteten Signalen. Dies ist der sogenannte „Pyramiden-Algorithmus“ [Vishwanath 1994].

Im ersten Schritt des Pyramiden-Algorithmus zerlegt man das Signal dabei in zwei Teile, einen groben Verlauf und eine Detail-Kurve, jeder Teil nur halb so lang wie das Originalsignal. Der grobe Verlauf wird erhalten, indem das Ursprungssignal geglättet wird, z.B. durch Betrachten des Signals in halber Auflösung. Die Details entsprechen den Fluktuationen, also den Änderungen, die zum groben Verlauf aufaddiert wieder das Ursprungssignal ergeben. Technisch gesehen entspricht dies der Verwendung eines Tiefpaßfilters für den groben Verlauf und eines Hochpaßfilters für die Details.

Der so entstehende grobe Verlauf übernimmt im zweiten Schritt des Pyramiden-Algorithmus die Rolle des Originalsignals. Auf die gleiche Weise wie im ersten Schritt wird nun ein weiterer grober Verlauf und eine weitere Detail-Kurve erzeugt, die jeweils nur noch ein Viertel des Ursprungssignals einnehmen. Der grobe Verlauf aus dem ersten Schritt wird verworfen, da er ja durch die beiden Teile des zweiten Schrittes rekonstruiert werden kann. Dies kann fortgeführt werden bis grober Verlauf und Details jeweils nur noch aus genau einem Koeffizienten besteht. Hier leuchtet auch unmittelbar ein, dass sich dieses Verfahren nur für Signallängen von 2^n mit $n \in \mathbb{N}$ eignet.

4.2.4 Das Haar Wavelet

Zur Berechnung des Signals in halber Auflösung verwendet man in der Regel eine zum Wavelet passende Skalierungsfunktion ϕ, oft auch als „Vater-Wavelet“ bezeichnet. Im einfachen Fall des Haar-Wavelets (siehe Abbildung 4.2a) hat die Skalierungsfunktion beispielsweise die Form

$$\phi_{\text{Haar}}(t) = \begin{cases} 1 & \text{für } 0 \leq t < 1 \\ 0 & \text{sonst} \end{cases}$$

Unter Verwendung des Vorfaktors $\frac{1}{2}$ bildet diese Skalierungsfunktion hier also Mittelwerte. Technisch geschieht das über einen der Skalierungsfunktion angepassten Tiefpaßfilter. Die Skalierungsfunktion definiert außerdem die Anfangsauflösung des zu analysierenden Signals. Die entsprechende Wavelet-Funktion des Haar-Wavelets hat die Form

$$\psi_{\text{Haar}}(t) = \begin{cases} 1 & \text{für } 0 \leq t < \frac{1}{2} \\ -1 & \text{für } \frac{1}{2} \leq t < 1 \\ 0 & \text{sonst} \end{cases}$$

Sie kodiert die Differenzen der Wavelet-Transformation, die eigentlichen Wavelet-Koeffizienten. Abbildung 4.3 stellt den Ablauf des Algorithmus grafisch dar. Ausgehend vom Originalsignal werden in jedem Schritt eine Hälfte Detailkoeffizienten und eine Hälfte Skalierungskoeffizienten erzeugt, wobei letztere im nächsten Schritt derselben Prozedur unterzogen werden, bis am Ende nur noch ein Skalierungskoeffizient übrig bleibt. Es ist allerdings stets möglich, den Algorithmus schon vorher abzubrechen. Das Ergebnis einer diskreten Wavelet Transformation ist in Abbildung 4.4 zu sehen. Hier wurde das Ausgangssignal, eine Elektrokardiogram-Kurve, mit Hilfe des Haar Wavelets in 4 Schritten zerlegt. In jedem Schritt entsteht eine Menge von Detail- und Skalierungskoeffizienten, wobei nur die letzte Menge der Skalierungskoeffizienten beibehalten wird. Die Anzahl aller auf diese Weise entstehender Koeffizienten entspricht genau der Anzahl der Werte des Originalsignals. In Kapitel 4.2.8 wird eine weitere Darstellungsmöglichkeit vorgestellt, die in dieser Arbeit praktische Relevanz hat, allerdings $n(m+1)$ Werte erzeugt, wobei m die Anzahl der Skalen, also die Anzahl der Schritte der Transformation ist.

Da dieser Algorithmus für einen Schritt lineare Laufzeit hat und sich nach jedem Schritt das zu analysierende Signal halbiert, beträgt seine Laufzeit $O(n \log n)$. Je nach Verwendung des Mutter-Wavelets ist jedoch die tatsächliche Komplexität unterschiedlich. Methoden zur effizienten Implementierung finden sich in [Rioul und Duhamel 1992].

4.2.5 Mehrfachauflösung

Problematisch dabei ist, dass es für viele Wavelet-Funktionen keine explizite Funktionsvorschrift gibt, sondern dass diese iterativ berechnet werden. Insbesondere betrifft dies die vielfach eingesetzten orthogonalen Daubechies Wavelets. Die Praxis sieht allerdings so aus, dass weder die Wavelet- noch die Skalierungsfunktionen benötigt werden. Stéphane Mallat schlug dabei mit der Theorie der Mehrfachauflösung die Brücke von den orthogonalen Wavelets zu den bei der Signalverarbeitung verwendeten Filtern. Dazu werden die Vektorräume $\ldots, V_{-2}, V_{-1}, V_0, V_1, V_2, \ldots$ betrachtet. V_0 soll dabei von der Skalierungsfunktion und all ihren ganzzahligen Translationen aufgespannt werden. V_j entsteht aus V_{j-1} durch Skalierung um Faktor 2. Folgende vier Bedingungen müssen dabei eingehalten werden:

1. Die Skalierungsfunktion muss zu allen durch ganzzahlige Translation aus ihr hervorgegangenen Funktionen orthogonal sein.
2. Bei einer vorgegebenen Auflösung enthält das Signal alle Information über die niedrigeren Auflösungen.

$$\ldots \subseteq V_{-2} \subseteq V_{-1} \subseteq V_0 \subseteq V_1 \subseteq V_2 \subseteq \ldots$$

3. Das einzige Objekt, das allen Räumen V_j gemein ist, ist die Funktion 0

$$\lim_{j \to -\infty} V_j = \bigcap V_j = 0$$

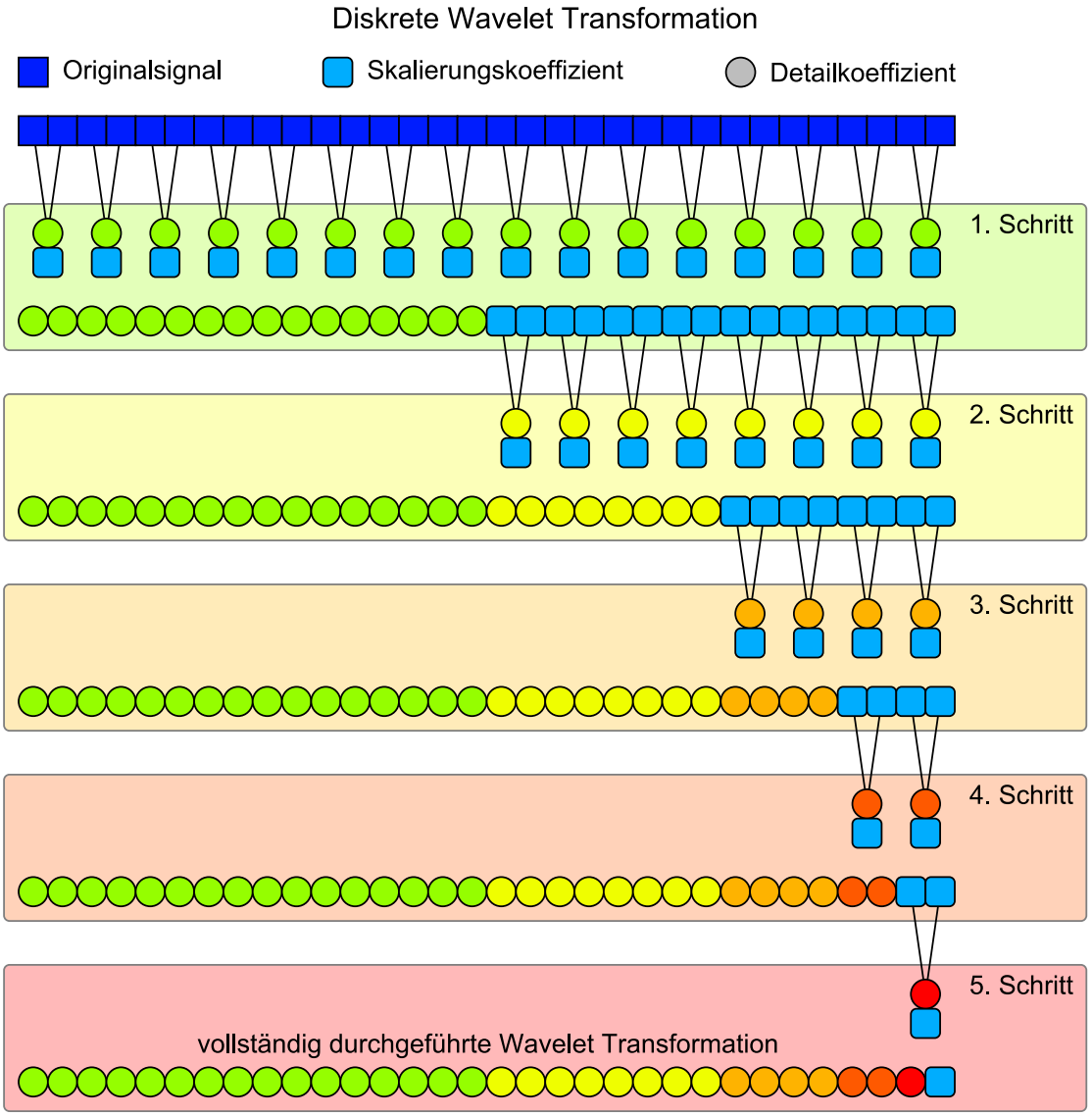


Abbildung 4.3: Die diskrete Wavelet Transformation berechnet jeweils aus Paaren des Ausgangssignals je einen Skalierungs- und einen Wavelet- beziehungsweise Detailkoeffizienten. Die Detailkoeffizienten werden übernommen und dieselbe Prozedur an den entstehenden Skalierungskoeffizienten, dem Signal in halber Auflösung, durchgeführt. So bleiben am Ende $n-1$ Detailkoeffizienten und ein Skalierungskoeffizient übrig. Vorzeitiges Abbrechen lässt entsprechend mehr Skalierungskoeffizienten und weniger Detailkoeffizienten übrig.

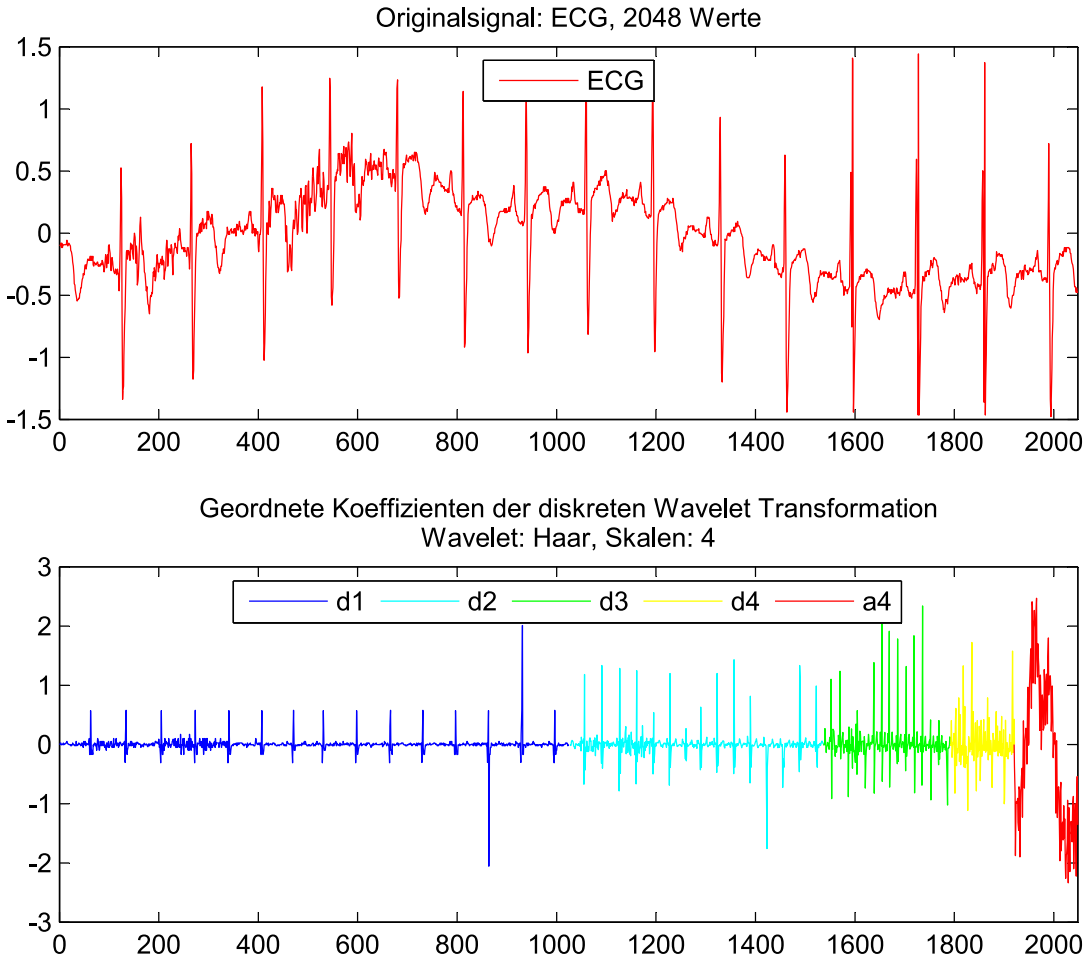


Abbildung 4.4: ECG-Signal, oben im Ursprungszustand, unten nach der Anwendung der diskreten Wavelet Transformation mit dem Haar Wavelet in 4 Skalen. Die Koeffizientenfolgen d1 bis d4 entsprechen den Detailkoeffizienten der vier Skalen, a4 enthält die übrigen Approximations- beziehungsweise Skalierungskoeffizienten. Das Signal ist dabei auf ein sechzehntel seiner ursprünglichen Größe geschrumpft, wobei es jedoch durch die in den Detailkoeffizienten gespeicherten Differenzen vollständig rekonstruierbar ist. Die Detailkoeffizienten sind üblicherweise von kleinem Betrag, so dass sich die Folge der Koeffizienten gegenüber dem Ursprungssignal oftmals besser komprimieren lässt.

4. Jedes Signal kann mit beliebiger Genauigkeit approximiert werden.

$$\lim_{j\to\infty} V_j = L^2\mathbb{R}$$

Dabei ist $L^2\mathbb{R}$ der Raum der zweifach integrierbaren Funktionen über $\mathbb{R}$.

Bei Erfüllung dieser Bedingungen müssen durch die Wavelet-Funktionen aufgespannte Räume W_j existieren, die orthogonal zu V_j sind und genau die Differenzen zwischen V_j und V_{j+1} repräsentieren:

$$W_j \oplus V_j = V_{j+1}$$

Obwohl es zunächst schwierig erscheint, geeignete Kandidaten als Skalierungs- und Waveletfunktion zu finden, zeigte Mallat in [Mallat 1989], wie sich beide über die Fourier-Transformierten nahezu beliebiger Filter konstruieren ließen. Genauere Details und Herleitungen würden an dieser Stelle den Rahmen sprengen, lassen sich aber ausführlich in [Strichartz 1993] und natürlich in [Mallat 1989] nachlesen.

Die Theorie der Mehrfachauflösung hat zwei entscheidende Ergebnisse zur Folge. Dadurch, dass ein Funktionenraum V_j stets im Raum der doppelten Auflösung V_{j+1} enthalten ist, lassen sich Skalierungs- und Waveletfunktion als Linearkombination der Skalierungsfunktion in doppelter Auflösung schreiben.

$$\phi(t) = 2\sum_{n=-\infty}^{\infty} a_n \phi(2t-n) \tag{4.4}$$

$$\psi(t) = 2\sum_{n=-\infty}^{\infty} d_n \phi(2t-n) \tag{4.5}$$

Der vorangestellte Faktor 2 dient dabei lediglich der Normierung. Im Falle der Haar-Skalierungsfunktion erkennt man die Koeffizienten a_n direkt:

$$\phi(t) = \frac{1}{2}(2\phi(2t)) + \frac{1}{2}(2\phi(2t-1))$$

Also ist $a_0 = a_1 = \frac{1}{2}$ und alle anderen $a_n = 0$. Die Wavelet-Funktion genügt folgender Funktionalgleichung

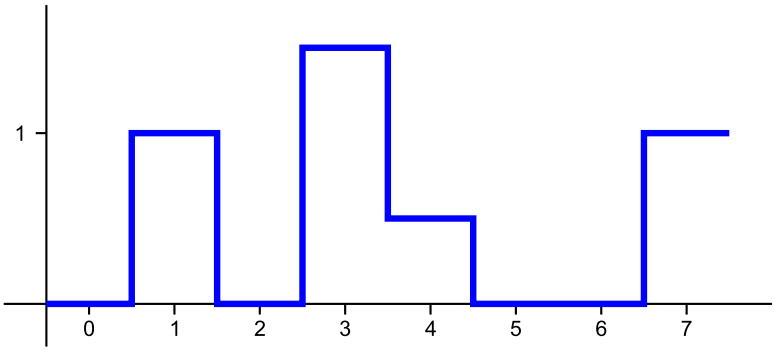
$$\psi(t) = \phi(2t) - \phi(2t-1)$$

Somit ist $d_0 = \frac{1}{2}$ und $d_1 = -\frac{1}{2}$. Bemerkenswerterweise gilt außerdem $a_0 = -d_1$ und $a_1 = d_0$. Die Theorie der Mehrfachauflösung zeigt, dass dieser Zusammenhang bei allen orthogonalen Filtern besteht. Man spricht hierbei von einem *Quadrature Mirror Filter* (QMF). Dabei ergeben die Koeffizienten in umgekehrter Reihenfolge bei alternierendem Vorzeichen den neuen Filter. Eine weitere Folge besteht, wie eingangs erwähnt, darin, die Wavelet-Transformation anwenden zu können, ohne überhaupt auf die Wavelet- oder Skalierungsfunktion zurückgreifen zu müssen. Die Theorie der Mehrfachauflösung zeigt, dass diese Funktionen als Grenzwerte von Iterationen gegeben sind, weshalb es lediglich geeigneter Filter bedarf. Die Filter entsprechen genau den a_n und d_n. Mit diesen Filtern wird das Signal dann gefaltet, um sowohl das approximierte Signal als auch die

Wavelet-Koeffizienten zu erhalten. Diese Faltung ist der algorithmische Trick der *schnellen Wavelet Transformation*. Weitere Informationen über den Zusammenhang zwischen Filtern und Wavelets liefert unter anderem [Strang und Nguyen 1997]. Ein mathematisches Tutorial lässt sich in [Kaiser 1994] finden.

4.2.6 DWT am Beispiel

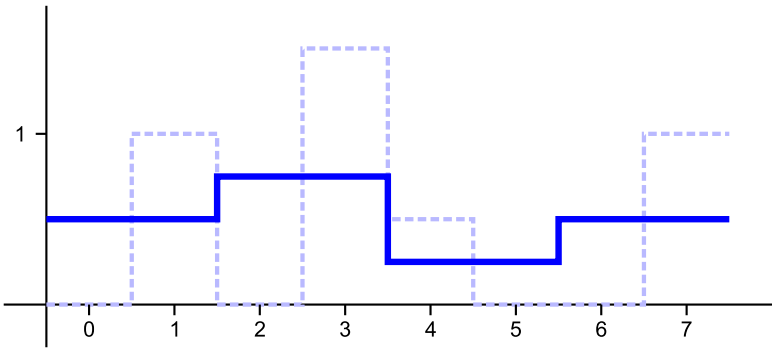
Das folgende Beispiel verdeutlicht die Funktionsweise der diskreten Wavelet-Transformation. Das Originalsignal $b = [0, 1, 0, 1.5, 0.5, 0, 0, 1]$ sei wie folgt dargestellt:



Die Filterkoeffizienten der Haar-Skalierungsfunktion lauten $a_{-1} = a_0 = \frac{1}{2}$. Die Faltung des Originalsignals mit dem Skalierungsfilter $(a \star b)$ liefert somit:

$$\begin{array}{lcllcl} (a \star b)_0 & = & a_{-1}b_1 + a_0 b_0 & = & 0,5(1+0) & = & 0,5 \\ (a \star b)_1 & = & a_{-1}b_3 + a_0 b_2 & = & 0,5(1,5+0) & = & 0,75 \\ (a \star b)_2 & = & a_{-1}b_5 + a_0 b_4 & = & 0,5(0+0,5) & = & 0,25 \\ (a \star b)_3 & = & a_{-1}b_7 + a_0 b_6 & = & 0,5(0+1) & = & 0,5 \end{array}$$

Entsprechend sieht nun das geglättete Signal wie folgt aus:



Die beiden Koeffizienten des Hochpaßfilters der Waveletfunktion $d_{-1} = -\frac{1}{2}$ und $d_0 = \frac{1}{2}$ ergeben gefaltet mit dem Signal die ersten vier Waveletkoeffizienten:

$$\begin{array}{lcllcl} (d \star b)_0 & = & d_{-1}b_1 + d_0 b_0 & = & -0,5 \cdot 1 + 0,5 \cdot 0 & = & -0,5 \\ (d \star b)_1 & = & d_{-1}b_3 + d_0 b_2 & = & -0,5 \cdot 1,5 + 0,5 \cdot 0 & = & -0,75 \\ (d \star b)_2 & = & d_{-1}b_5 + d_0 b_4 & = & -0,5 \cdot 0 + 0,5 \cdot 0,5 & = & 0,25 \\ (d \star b)_3 & = & d_{-1}b_7 + d_0 b_6 & = & -0,5 \cdot 0 + 0,5 \cdot 1 & = & -0,5 \end{array}$$

Man erkennt hierbei deutlich, dass die Waveletkoeffizienten auf die entsprechenden Skalierungskoeffizienten aufaddiert (jeweils einmal mit positivem und negativem Vorzeichen) wieder das Originalsignal ergeben. In der Praxis werden die Filterkoeffizienten allerdings normiert, um die Energie des ursprünglichen Signals zu erhalten. Eine genaue Herleitung der Filterkoeffizienten findet sich in [Percival und Walden 2000].

Einige Gründe, weshalb die DWT ein effektives Analysewerkzeug ist, sind die folgenden.

1. Die DWT stellt eine Zeitreihe in Form von Koeffizienten dar, welche mit einer spezifischen Zeit und einer bestimmten dyadischen Skala verknüpft sind. Aus diesen Koeffizienten lässt sich die ursprüngliche Zeitreihe originalgetreu wieder herstellen. Diese Rekonstruktionseigenschaft ist essenziell für die Anwendung von Wavelet-Methoden, so auch in dieser Arbeit.
2. Die DWT kann von einem Algorithmus berechnet werden, der schneller ist als die Fast Fourier Analyse.

Es existieren noch weitere Gründe, jedoch sind diese im Rahmen dieser Diplomarbeit nicht von Interesse. Von besonderer Bedeutung ist die soeben erwähnte Rekonstruktionseigenschaft. Eine inverse Transformation der unveränderten Wavelet-Koeffizienten stellt demnach das Originalsignal wieder her. Wie jedoch eine Modifikation der Koeffizienten vor der inversen Transformation von Nutzen sein kann, zeigt das folgende Kapitel.

4.2.7 Wavelet-Glättung

Wavelet-Koeffizienten spiegeln in gewisser Weise die Änderungen der Zeitreihe in verschiedenen Skalen wider. Einerseits kann man mit ihnen das Ursprungssignal originalgetreu wiederherstellen, allerdings hat man dadurch zunächst einmal keinen Gewinn. Andererseits kann man hoffen, dass die Koeffizienten klein genug sind und man so Speicherplatz einsparen kann. Man weiß, dass kleine Detailkoeffizienten auch nur kleinen Änderungen im Ursprungssignal entsprechen. Daher benutzt man häufig das sogenannte *Thresholding*-Verfahren, bei dem Koeffizienten unterhalb eines gewissen Wertes, dem Threshold, auf Null gesetzt werden. Die so entstehende Koeffizientenfolge lässt sich zum einen deutlich besser komprimieren, zum anderen weist das aus dieser Folge zurückgewonnene Signal einen geglätteten Charakter auf. Insbesondere Rauschen, beispielsweise Störeinflüsse von gemessenen Sensordaten, lässt sich auf diese Weise gut heraus filtern, indem Thresholding auf die Koeffizienten der niedrigeren Auflösungen, welche hochfrequente Änderungen darstellen, angewendet wird. Eine Reduktion oder Eliminierung

dieser hochfrequenten Änderungen führt so zu einer, teilweise deutlichen, Glättung. Abbildung 4.5 zeigt die Wavelet-Glättung am Beispiel. Die Wavelet-Glättung wird in dieser Arbeit an zwei Stellen verwendet. Zum einen dient sie in einem Experiment als Vorverabeitungsschritt. So soll herausgefunden werden, ob Modelle, die auf einer geglätteten Zeitreihe trainiert wurden, bessere Ergebnisse erzielen als solche, die auf ungeglättete Reihen trainiert wurden. Zum anderen dient die Wavelet-Glättung allerdings auch der Nachbearbeitung. Erstellt man mit den Lernverfahren eine prognostizierte Zeitreihe und glättet diese im Nachhinein, so ist eine Reduzierung des Approximationsfehlers möglich.

4.2.8 Multiskalenanalyse

Eine andere Art der Darstellung der diskreten Wavelet Transformation ist häufig unter dem Begriff der Multiskalenanalyse anzutreffen. Anstatt die reinen, bei der Zerlegung auftretenden Koeffizienten zu präsentieren wird hierbei aus den Koeffizienten jeder Skalierung einzeln das Ursprungssignal rekonstruiert. Im Prinzip setzt man dabei ähnlich wie beim Thresholding alle anderen Koeffizienten auf Null und führt eine inverse Wavelet Transformation durch. Dieses Vorgehen wiederholt man für die Detailkoeffizienten aller Auflösungen und ebenso für die Approximationskoeffizienten, so dass letzten Endes $m+1$ Wertereihen der Länge n entstehen, wobei m die Anzahl der Analyseskalen und n die Anzahl der Werte des Originalsignals ist. Abbildung 4.6 zeigt eine Multiskalenanalyse am Beispiel der bereits vorgestellten ECG-Daten. Dabei wurde das Signal 6 Mal mit dem Daubechies-10 Wavelet abgetastet und die entsprechenden Koeffizientenfolgen einzeln für die Rekonstruktion verwendet. Die besondere Eigenschaft dieser Darstellung ist, dass die Summe aller bei der Multiskalenanalyse entstehenden Wertereihen wieder das Ursprungssignal ergibt. Die Folge a_m stellt bei dieser Zerlegung den allgemeinen Verlauf des Ursprungssignals dar. Einige wissenschaftliche Arbeiten benutzen diese Art der Zerlegung, um auf den einzelnen Komponenten getrennt Prognosemodell zu erstellen, deren Prognosewerte aufsummiert eine Prognose des Originalsignals darstellen, siehe dazu Kapitel 5. Ob dieses Vorgehen im Falle der Strompreise des Day-Ahead Auktionshandels ebenso gut funktioniert, wird später untersucht werden.

4.2.9 Wavelet Kernfunktion

Obwohl sie weder Datentransformation, noch Vorverarbeitung ist, soll in diesem Zusammenhang eine Kernfunktion zur Verwendung in der Stützvektormethode näher beleuchtet werden. Diese spielt insbesondere in den späteren Experimenten eine Rolle, indem sie auf ihre Tauglichkeit bei der Vorhersage der Strompreise des Day-Ahead Auktionshandels untersucht wird. Zunächst wird jedoch eine wichtige Definition benötigt.

Definition 4.2.1. (Translationsinvariante Kernfunktion) Ist eine Kernfunktion nicht direkt von ihren beiden Parametern, sondern nur von der relativen Differenz beider Parameter abhängig, gilt also

$$K(\vec{x}, \vec{y}) = K(\vec{x} - \vec{y})$$

so nennt man die Kernfunktion *translationsinvariant* [Smola et al. 1998].

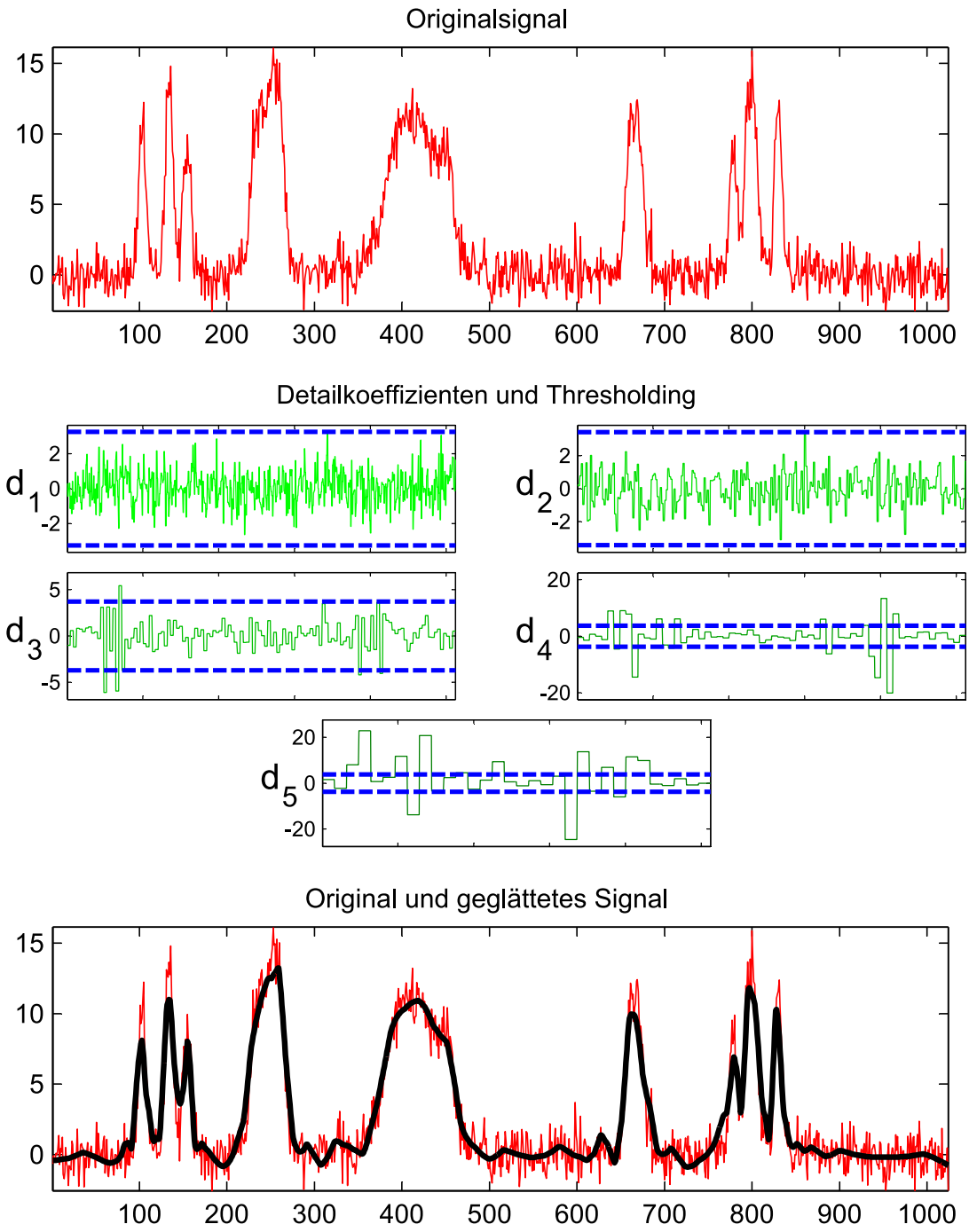


Abbildung 4.5: Wavelet Glättung (Thresholding) am Beispiel eines verrauschten Signals, welches mit dem Symlet-4 Wavelet in 5 Auflösungen analysiert wurde. Die gestrichelten blauen Linien in den Diagrammen der Detailkoeffizienten geben die Threshold-Grenzen an. Innerhalb der Grenzen liegende Koeffizienten wurden bei der Rekonstruktion des Signals auf 0 gesetzt.

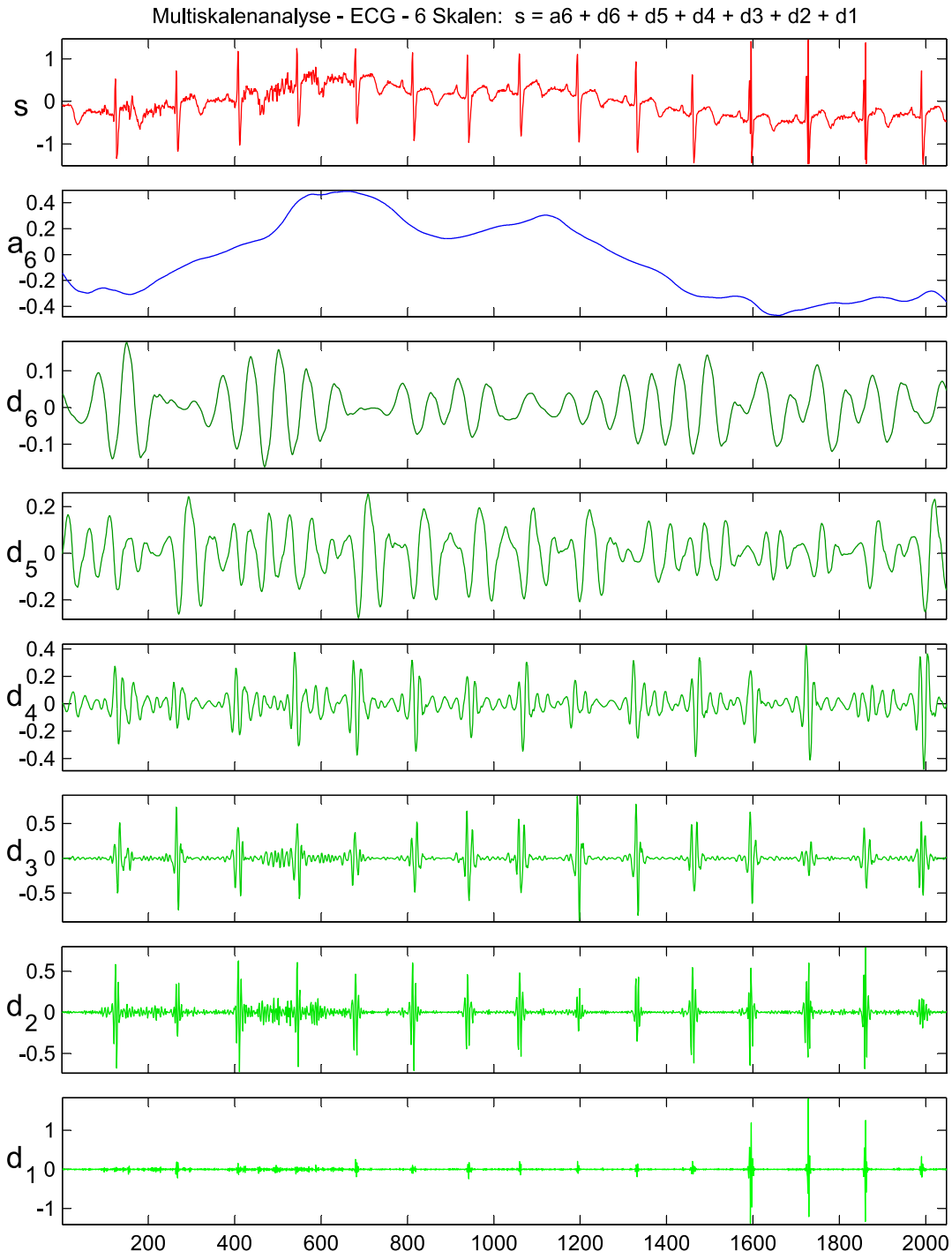


Abbildung 4.6: Zerlegung des ECG-Signals in seine Wavelet-Bestandteile unter Verwendung des Daubechies-10 Wavelets. Die Folgen d_1 bis d_6 entsprechen den aus den Detailkoeffizienten rekonstruierten Signalen, Folge a_6 (Approximationsfolge) entstammt den Approximationskoeffizienten.

In Kapitel 3.3.3 wurde bereits erwähnt, dass eine Funktion $K(\vec{x}, \vec{y}) : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}$ die Bedingungen aus Mercers Theorem erfüllen müsse, um eine zulässige (*admissible*) SVM-Kernfunktion darstellen zu können. Translationsinvariante Kernfunktionen lassen sich allerdings nur schwer in das Produkt zweier Funktionen aufteilen, weshalb in diesem Fall folgende Äquivalenz gelten muss:

Satz 4.2.2. *Eine translationsinvariante Kernfunktion $K(\vec{x}, \vec{y}) = K(\vec{x} - \vec{y})$ ist genau dann eine zulässige SVM-Kernfunktion, wenn die Fourier-Transformation*

$$F[K](\omega) = (2\pi)^{-N/2} \int_{\mathbb{R}^N} e^{-i(\omega \cdot \vec{x})} K(\vec{x}) d\vec{x} \tag{4.6}$$

nicht negativ ist. Dabei entspricht N der Dimension der Beispiele.

Eine auf der Wavelet-Theorie basierende Kernfunktion wird in [Zhang et al. 2004] vorgestellt:

Satz 4.2.3. *Sei $\psi(x)$ eine „Mutter"-Wavelet-Funktion und seien $a \in \mathbb{R}$ und $c, c' \in \mathbb{R}^N$ Dilatations- und Translationsparameter. Seien weiter $\vec{x}, \vec{x}' \in \mathbb{R}^N$. Dann existieren sowohl Skalarprodukt-Wavelet-Kernfunktionen der Form*

$$K(\vec{x}, \vec{x}') = \prod_{i=1}^{N} \psi\left(\frac{x_i - c_i}{a}\right) \psi\left(\frac{x_i' - c_i'}{a}\right) \tag{4.7}$$

als auch translationsinvariante Wavelet-Kernfunktionen der Form

$$K(\vec{x}, \vec{x}') = \prod_{i=1}^{N} \psi\left(\frac{x_i - x_i'}{a}\right) \tag{4.8}$$

Es kann gezeigt werden, dass einerseits Skalarprodukt-Wavelet-Kernfunktionen die Bedingungen aus Mercers Theorem und andererseits translationsinvariante Wavelet-Kernfunktionen Satz 4.2.2 erfüllen, so dass in beiden Fällen eine zulässige SVM-Kernfunktion vorliegt. Entsprechend Gleichung 3.33 ergibt sich nun für die Stützvektor-Regression die approximierte Funktion

$$\hat{f}(\vec{x}) = \sum_{i=1}^{n} (\alpha_i - \alpha_i') \prod_{j=1}^{N} \psi\left(\frac{x^j - x_i^j}{a_i}\right) + b \tag{4.9}$$

sowie für die Klassifikation gemäß Gleichung 3.28

$$\hat{y} = \operatorname{sign}\left(b + \sum_i \alpha_i y_i \prod_{j=1}^{N} \psi\left(\frac{x^j - x_i^j}{a_i}\right)\right) \tag{4.10}$$

Hierbei entspricht x_i^j der j-ten Komponente des Trainingsbeispiels $\vec{x}_i$ und N der Dimension der Beispiele. Zhang, Zhou und Jiao setzen in ihrer Arbeit aus Gründen der

Einfachheit alle Parameter a_i gleich, so dass nur ein einziger Parameter a für ihre Experimente zu wählen ist. Dieser Parameter lässt sich mit einem Kreuzvalidierungsverfahren bestimmen [Wahba und Wold 1975], [Joachims 2000], [Kearns und Ron 1999].

In [Szu et al. 1992] wird nun folgende Morlet-Wavelet-Funktion vorgeschlagen:

$$\psi(x) = \cos(1.75x)e^{-\frac{x^2}{2}} \tag{4.11}$$

Streng genommen ist das Morlet Wavelet eine komplexwertige Funktion, deren reeller Teil wie folgt definiert ist [Goupillaud et al. 1984].

$$g_r(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} \cos(2\pi\nu x) \tag{4.12}$$

Aus diesem Grund wird in den Experimenten zwischen dem tatsächlichen **Morlet-Wavelet** mit zusätzlichem Parameter ν, sowie dem auf dem Morlet-Wavelet beruhenden Wavelet $\psi(x)$ unterschieden. Letzteres wird von nun an **Szu-Wavelet** genannt.

Abbildung 4.7 stellt das Szu-Wavelet im Vergleich zum sogenannten „Mexican Hat“ Wavelet ($\psi_{\text{MH}}(x) = c(1-x^2)e^{-\frac{x^2}{2}}$ mit $c = \frac{2}{\sqrt{3\sqrt{\pi}}}$) dar. Beide Wavelets ähneln sich nicht nur stark in ihrer Erscheinung, sondern sind bis auf den ersten Term auch identisch.

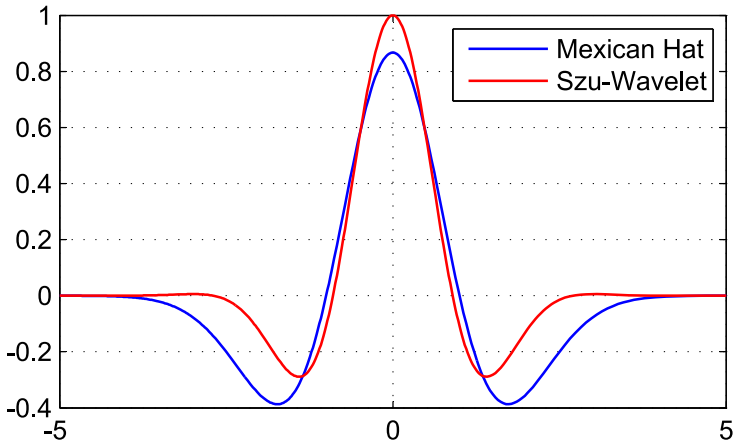


Abbildung 4.7: Mexican-Hat- und Szu-Wavelet $\psi(x)$ im Vergleich.

Eingesetzt in Gleichung 4.8 ergibt sich die endgültige Kernfunktion des Szu-Wavelets folglich zu

$$K(\vec{x}, \vec{x}') = \prod_{i=1}^{N} \left(\cos\left(1.75\frac{x_i - x_i'}{a}\right) e^{\left(-\frac{||x_i - x_i'||^2}{2a^2}\right)} \right) \tag{4.13}$$

Der Beweis der Zulässigkeit dieser SVM-Kernfunktion, sowie weitere Details sind in [Zhang et al. 2004] aufgeführt. Der Artikel zeigt, dass der vorgestellte Wavelet-Kernel im Vergleich zum Gauss-Kern besser approximiert bei gleichzeitig kürzerer Trainingsdauer, weswegen er im Rahmen dieser Arbeit zu Vergleichszwecken implementiert und

eingesetzt wurde. Die soeben beschriebene Kernfunktion wird außerdem in [Tolambiya und Kalra 2007] für ein Wavelet-Stützvektor Bildkompressionsverfahren verwendet. Der Vollständigkeit halber wurden auch die Mexican Hat Funktion, sowie das Morlet Wavelet mit Parameter ν als Kernfunktion implementiert und in die Experimente mit einbezogen.

Es sind noch weitere Wavelet-basierende Kernfunktionen denkbar, beispielsweise zeigt [Zhang et al. 2005] eine Wavelet-Kernfunktion ähnlich einer Radial-Basis-Funktion. Die vorliegende Arbeit beschränkt sich allerdings auf die in [Zhang et al. 2004] vorgestellte Kernfunktion unter Verwendung der drei Wavelet-Funktionen.

5 Themenbezogene Arbeiten

Im Folgenden soll ein Abriss über bisherige Arbeiten, die im Rahmen dieser Diplomarbeit von Interesse sind, gegeben werden. Auch wenn sie zum Teil andersartige Daten behandeln, so unterstreichen sie doch das Potenzial und die Mächtigkeit der in den vorangegangenen Kapiteln vorgestellten Methoden.

Die Stützvektormethode findet unter anderem in [Cao und Tay 2001] Anwendung bei der Vorhersage von Börsenkursen am Beispiel des US-amerikanischen S&P 500 Index der Jahre 1993 bis 1995. Um die Ergebnisse der Regression zu verbessern, werden in der Arbeit weitere, aus dem Indexkurs berechnete Kennzahlen, wie prozentuale, relative Differenzen und gleitende Durchschnitte hinzugenommen. Im Vergleich mit neuronalen Netzen zeigt sich hier die Überlegenheit der Stützvektormethode im Kontext der Regression. Zum selben Fazit kommt [Xie et al. 2006], die neben den neuronalen Netzen noch ARIMA Modelle in den Vergleich mit einbeziehen, vorhergesagt werden hier allerdings Ölpreise. Gute Regressionsergebnisse erzielt man mit der SVM ebenfalls bei der Preisvorhersage am Elektrizitätsmarkt [Gao et al. 2007] und bei der Vorhersage der Elektrizitätslasten [Chen et al. 2004]. Die Anwendung der SVM in zuletzt genannter Arbeit erzielte dabei den ersten Platz in einem vom EUNITE Netzwerk veranstalteten Wettbewerb, in dem es darum ging, die Maximallast der nächsten 31 Tage vorherzusagen.

In [Contreras et al. 2003] und [Conejo et al. 2005] werden Vorhersagen aufgrund von ARIMA-Modellen für den Strompreis des spanischen Festlandes erstellt. In zweitgenannter Arbeit wird jedoch nicht direkt aus den historischen Werten prognostiziert. Stattdessen wird zunächst eine Wavelet Multiskalenanalyse durchgeführt und die so entstandenen Reihen getrennt mit speziell angepassten ARIMA Modellen prognostiziert. Die Wavelet Rücktransformation ergibt schließlich die endgültige Prognose. Es zeigt sich, dass dieses Verfahren im Vergleich zu direkt an die Zeitreihe angepassten ARIMA Modellen bessere Ergebnisse liefert. Grund dafür ist das weniger chaotische Verhalten der Wavelet-transformierten Zeitreihe.

Dass sich eine Wavelet Transformation als Vorverarbeitungsschritt eignet, zeigt ebenfalls [Patnaik 2005]. Dort werden Magnetresonanzbilder des menschlichen Gehirns auf mögliche Krankheiten hin klassifiziert. Unterschieden wird dabei allerdings lediglich zwischen „gesund“ und „krank“. Die aufgenommenen Bilder werden zunächst in Wavelet-Koeffizienten überführt, die entstehenden Detailkoeffizienten einem Kantenverbesserungsverfahren unterzogen und das Ergebnis schließlich rücktransformiert. Als alternative Vorverarbeitung wird eine Independent Component Analyse (ICA) durchgeführt. Beim Vergleich beider Vorverarbeitungsschritte mit anschließender SVM-Klassifikation zeigt sich die Überlegenheit der Wavelet-Vorverarbeitung gegenüber der ICA. Von zentraler Bedeutung sind die Detailkoeffizienten auch in [He und Starzyk 2006]. Hier wird die

Energie der Koeffizienten als Feature für die Erkennung von bevorstehenden Energieversorgungsproblemen zur Qualitätssicherung verwendet.

Die Wavelet-Theorie war indes auch Vorbild für die sogenannte *Multi-resolution SVM* aus [Shao und Cherkassky 1999]. Dabei wurde die Grundidee, ein Signal in verschiedenen Auflösungen „abzutasten“ auf die Stützvektormethode übertragen. Mehrere Kernfunktionen in verschiedenen „Auflösungen“ wurden hierbei simultan benutzt, um die Zielfunktion zu approximieren, wobei unterschiedliche Werte für die Parameter der Kernfunktionen als „Auflösungen“ fungieren. Die Komplexität, und damit auch die Laufzeit, des Algorithmus erhöht sich allerdings drastisch, was das Verfahren für die Praxis unattraktiv macht.

Dass Wavelet-Verfahren auch für die Vorverarbeitung von Anwendungen der Stützvektormethode beliebt sind, zeigen eine Reihe weiterer Arbeiten. In [Malathi und Marimuthu] wird beispielsweise die diskrete Wavelet Transformation im Vorfeld durchgeführt, um daraufhin mit einer SVM die Fehlerstelle einer Stromübertragungsleitung zu schätzen. Es wurden 5 verschiedene Wavelets, sowie lineare und Radial-Basis Kernfunktionen der SVM verglichen, einen klaren Sieger gab es bei dem Vergleich allerdings nicht. Ähnlich wie in [Conejo et al. 2005] zerlegt auch [yong Liu und Fan 2006] eine Zeitreihe von Futures-Preisen anhand der Wavelet Multiskalenanalyse zunächst in mehrere Zeitreihen. Für diese werden dann getrennt SVM-Modelle trainiert, wobei Spline- und RBF-Kernfunktionen zum Einsatz kommen. Die Summe der vorhergesagte Zeitreihen ergibt letztlich die endgültige Vorhersage. Dasselbe Vorgehen findet sich ebenfalls in [Guo et al. 2008] wieder. Ferner verwendet [Lin et al. 2005] die Wavelet-Transformation, um Audio-Signale im Frequenzbereich unter Hinzunahme weiterer akustischer Merkmale mit Hilfe der SVM zu klassifizieren und [Xi und Lee 2003] wendet die zweidimensionale Wavelet-Transformation an, um Gesichter zu erkennen.

Eine etwas andere Symbiose von Stützvektormethode und Wavelets stellt die in [Zhang et al. 2004] eingeführte, bereits in Kapitel 4.2.9 erwähnte Wavelet SVM (WSVM) dar. Hier wurde eine neuartige Kernfunktion mit Ursprung in der Wavelet-Theorie entworfen, ihre Zulässigkeit bewiesen und an praktischen Daten im Vergleich zur Gauss-Kernfunktion evaluiert. Der Wavelet-Kern zeigte bessere Approximationseigenschaften als der Gauss-Kern. Dieselbe Wavelet-Kernfunktion benutzt auch [Dan et al. 2005] in einer Studie über digital modulierte Signale. Hier unterschied die SVM zwischen 10 verschiedenen Modulationsklassen mit Hilfe einer sogenannten *Directed Acyclic Graph SVM* (DAGSVM). Diese benutzt intern mehrfach eine binäre Klassifikations-SVM, um Beispiele in mehr als zwei Klassen einzuordnen. In einem Vorverarbeitungsschritt, der zuerst in [Pittner und Kamarthi 1999] vorgestellt wurde, wurden außerdem Features aus den Koeffizienten einer im Vorfeld durchgeführten Wavelet-Transformation erzeugt. Aus jeder Reihe von Wavelet-Koeffizienten wurde durch Quadrieren und Summenbildung genau ein Feature erzeugt. Die eigentlichen Koeffizienten wurden nicht weiter gebraucht, was zu einer drastischen Kompression des Signals führte.

6 Anwendung und Auswertung der Methoden

In den folgenden Experimenten werden nun die oben vorgestellten Verfahren auf den stündlichen Strompreisen des Day-Ahead Auktionshandels der EEX angewendet und auf ausgewählten Zeitreihenausschnitten miteinander verglichen. Im Vordergrund steht hierbei die Vergleichbarkeit der Resultate, weshalb in allen Experimenten ein nahezu identischer Versuchsaufbau gewählt wird, zumindest in den Experimenten, die die Stützvektormethode verwenden. Außerdem werden die Probleme bei der Planung und Durchführung der Experimente geschildert, sowie Lösungsansätze präsentiert und Ergebnisse diskutiert.

Kapitel 6.1 erläutert zunächst, wie Prognosemodelle evaluiert und bewertet werden können und weist dabei auf eine besonders simple Prognose hin, die sich in Experimenten als problematisch erweist. Auf die Verwendung des verfügbaren Datenmaterials und die Aufteilung in Trainings- und Testdaten wird in Kapitel 6.2 näher eingegangen. Kapitel 6.3 schließlich befasst sich mit Prognoseverfahren unter Verwendung der Stützvektormethode. In diesem Zusammenhang wird dort auch die Experimentierumgebung RapidMiner, sowie im Rahmen dieser Diplomarbeit entwickelte Erweiterungen vorgestellt. Anschließende Versuchsreihen in Kapitel 6.4 zeigen Möglichkeiten auf, Verfahren aus der Wavelet-Theorie in die Vor- und Nachbearbeitung der Lernalgorithmen zu integrieren. Besonderes Augenmerk auf praxisnahe Vorhersagen legt Kapitel 6.5. Die Methoden der klassischen Zeitreihenanalyse kommen in Kapitel 6.6 unter Verwendung der Open Source Software gretl zum Einsatz. Die zusammenfassende Interpretation der Ergebnisse bildet letztlich den Schluss dieses Teils der Arbeit.

Vereinbarungen

Der Leser sei darauf aufmerksam gemacht, dass im Folgenden die Begriffe „Prognose“ und „Vorhersage“ aus stilistischen Gründen äquivalent benutzt werden. Des Weiteren ist im Kontext dieser Arbeit offensichtlich stets die Stützvektor-Regression gemeint, auch wenn von „Stützvektormethode“ oder „SVM“ die Rede ist. Da Prognosekurven den wahren Verlauf einer Zeitreihe annähern, sei alternativ auch der Begriff der Approximation und des Approximationsfehlers gestattet.

6.1 Zur Bewertung von Prognosen

Da kein Lernverfahren perfekt ist, werden auch bei der Prognose von Zeitreihen stets Fehler gemacht. Spricht man von der Performanz einer Prognose, so meint man übli-

cherweise genau diese Fehler. Sie zu minimieren ist dabei das Ziel einer jeden Prognoseaufgabe. Diese Arbeit verwendet folgende Fehler- beziehungsweise Performanzmaße, wobei (y_i) die Reihe der tatsächlichen und $(\hat{y}_i)$, jeweils mit $i \in \{1, \dots n\}$, die Reihe der prognostizierten Werte darstellt.

Definition 6.1.1. (Absoluter Fehler) Auch **ME** (*mean error*) genannt. Der absolute Fehler misst die absolute Abweichung der Prognose vom wahren Wert.

$$\mathrm{ME}(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i|$$

Definition 6.1.2. (Quadratischer Fehler) Auch **MSE** (*mean squared error*) genannt. Im Vergleich zum absoluten Fehler wird hier die Differenz zum wahren Wert quadriert. Größere Fehler werden hierbei stärker hervorgehoben.

$$\mathrm{MSE}(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Anschließendes Ziehen der Wurzel führt zum sogenannten **RMSE** (*root mean squared error*), welcher dieselbe Größenordnung wie der absolute Fehler und die Reihe der tatsächlichen Werte hat. Unterscheidet sich der RMSE signifikant vom absoluten Fehler, lässt dies darauf schließen, dass es Beispiele gibt, deren Vorhersagefehler signifikant größer ist als der durchschnittliche Vorhersagefehler.

In den späteren Untersuchungen ist ein weiteres Fehlermaß von entscheidender Bedeutung, welches nicht die Abweichung zum wahren Wert, sondern die Abweichung von der Tendenz des wahren Verlaufs misst.

Definition 6.1.3. (Prediction Trend Accuracy) Im Folgenden als **PTA** bezeichnet beschreibt dieser Wert den Anteil an korrekten Trendprognosen, also ob der aktuelle Wert höher, niedriger oder gleichbleibend im Vergleich zum vorherigen ist. Für einen Zeitpunkt wird dabei einerseits die Differenz zwischen wahrem Wert zu diesem Zeitpunkt und wahrem vorherigen Wert $(y_i - y_{i-1})$ und andererseits die Differenz zwischen prognostiziertem Wert zum Zeitpunkt und dem wahren vorherigen Wert $(\hat{y}_i - y_{i-1})$ gebildet. Haben diese beiden Differenzen dasselbe Vorzeichen, gilt dies als korrekte Trendprognose.

$$\mathrm{PTA}(y, \hat{y}) = \frac{\text{Anzahl korrekter Trendprognosen}}{n}$$

Ein Wert von 1 bedeutet, dass der Trend in allen Fällen korrekt prognostiziert wurde, der Wert 0 hingegen sagt, dass stets in die entgegengesetzte Richtung vorhergesagt wurde. Einfaches Raten führt im Erwartungswert auf eine PTA von 0.5.

Möglicherweise problematisch an der Implementierung der Berechnung des PTA könnte sein, dass hier auch dann eine korrekte Trendprognose erkannt wird, wenn bereits eine der beiden gebildeten Differenzen Null ist, sich also der Trend nicht ändert. Mathematisch formuliert gilt es als korrekte Trendprognose, falls gilt: $\mathrm{diff}_{\mathrm{wahr}} \cdot \mathrm{diff}_{\mathrm{Prognose}} \geq 0$. Wieso dies problematisch sein kann, zeigt sich im Zusammenhang mit der naiven Prognose in Kapitel 6.1.2.

6.1.1 Vergleich von Performanzmaßen

Die meisten Lernverfahren müssen über entsprechende Parameter an die Lernaufgabe angepasst werden. Verschiedene Belegungen von Parameterwerten erzeugen dabei unterschiedliche Modelle mit unterschiedlichen Eigenschaften und unterschiedlicher Performanz. Da man stets an der bestmöglichen Performanz interessiert ist, gilt es, die Parameter des Modells zu optimieren. Mit Hilfe der Performanzmaße bewertet die Parameteroptimierung die ausgewählten Parameterwerte und entscheidet so jeweils über „bessere“ und „schlechtere“ Parameterbelegungen. Üblicherweise wird dabei ein Hauptkriterium, beispielsweise der negative RMSE verwendet[1].

Im Rahmen dieser Arbeit wurde ein neues, kombiniertes Vergleichskriterium verwendet. Nachdem unter Verwendung ausgewählter Parameterwerte ein Modell gelernt und validiert wurde, werden sowohl der RMSE als auch die PTA berechnet. Anstatt nun einfach den negativen RMSE als Vergleichswert zu benutzen, wird dieser zuvor durch die PTA dividiert. Auf diese Weise werden falsche Trendprognosen „bestraft“ und der Vergleichswert künstlich verkleinert (negativer RMSE geteilt durch positive PTA im Bereich zwischen 0 unr 1). So werden Parameterwerte, die einen ähnlichen RMSE erzeugen, jedoch im Schnitt häufiger den korrekten Trend prognostizieren, bevorzugt.

6.1.2 Die naive Prognose

Eins der einfachsten denkbaren Prognosemodelle prognostiziert stets den letzten bekannten Wert. Bei Betrachtung eines größeren Zeitraums kann es aufgrund der ungenauen Darstellung auf beispielsweise einem Computerbildschirm passieren, dass dies zunächst nicht auffällt. Abbildung 6.1 zeigt die Preiskurve einer zufällig ausgewählten Woche und die naive Prognose dieser Preiskurve. Da dies lediglich 168 Werte sind, wird an diesem Beispiel unmittelbar klar, dass es sich hier um eine solche Prognose handelt. Da die meisten der in dieser Arbeit vorgestellten Verfahren, welche auf der Stützvektormethode basieren, eine Fensterung mit Horizont 1 verwenden, wäre dies ein durchaus mögliches Prognoseergebnis. Auf den ersten Blick scheint diese Prognose nicht übermäßig schlecht, möglicherweise sogar recht gut zu sein, jedoch ist sie in folgender Weise unbefriedigend. Das vorliegende Datenmaterial, also die Reihe der Spot-Auktionspreise, ergänzt sich ein Mal täglich um die jeweils 24 Stundenpreise des nächsten Tages. Ein Prognosemodell, welches auf der Fensterung mit Horizont 1 basiert kann somit realistisch betrachtet nur den Preis der Stunde 1 (von 0 Uhr bis 1 Uhr) des nächsten Tages bestimmen. Eine Möglichkeit, weiter reichende Prognosen zu berechnen, besteht darin, den soeben prognostizierten Wert in einem weiteren Prognoseschritt als bekannt voraus zu setzen und basierend auf dieser Prognose inkrementell einen weiteren Wert zu prognostizieren. Würde man nun aber immer den zuletzt bekannten Wert prognostizieren, so ergäbe sich eine konstante Funktion als Prognosekurve. Diese inkrementellen Prognosen werden im Verlauf dieser Arbeit angewendet, daher gilt es, das Modell der naiven Prognose zu vermeiden.

[1] Das Vergleichskriterium wird stets maximiert.

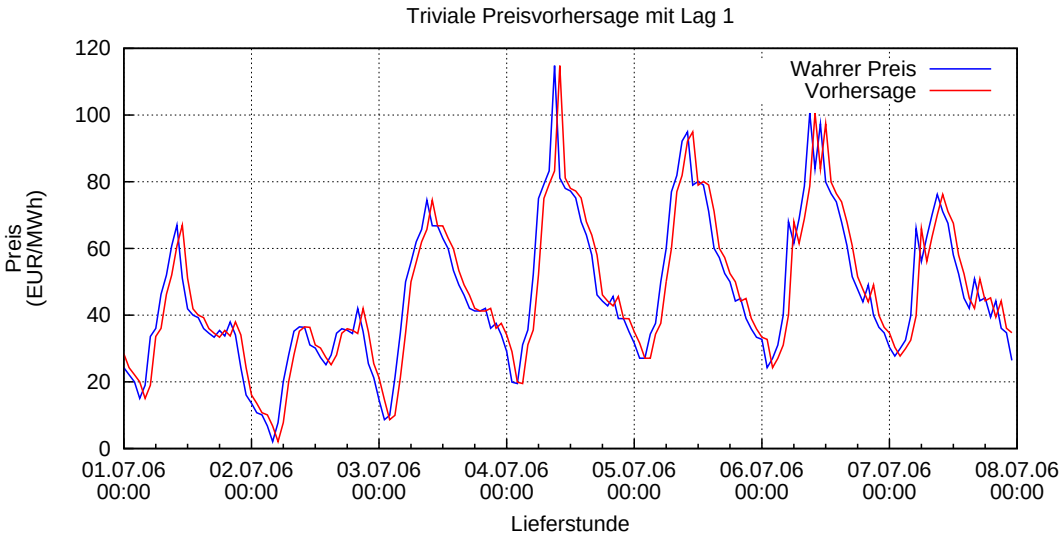


Abbildung 6.1: Naive Prognose mit Lag 1 am Beispiel der Preiskurve der ersten Juli-Woche des Jahres 2006. Hierbei wird für jede Lieferstunde stets derselbe Preis wie zur vorigen Lieferstunde prognostiziert. Der RMSE beträgt auf dieser kurzen Reihe 8.438, der absolute Fehler bei 6.218 ± 5.705, die PTA liegt bei 0.994 (laut RapidMiner. Mehr dazu im Text).

Abbildung 6.1 verdeutlicht eine weitere Besonderheit. RapidMiner errechnet hier eine PTA von 0.994. Zunächst stellt sich die Frage, wie dies sein kann, da beide Kurven sicherlich häufiger gegenläufige Trends aufweisen als nur in weniger als einem von hundert Fällen. Man darf jedoch nicht vergessen, dass die Trends der Vorhersage-Reihe stets bezogen auf die wahren Werte sind. Bei einer naiven Prognose beläuft sich der Unterschied der Prognose zum vorangegangenen wahren Wert aber auf genau Null. RapidMiner zählt einen solchen Fall allerdings stets als korrekte Trendprognose, unabhängig davon, wie der Trend der wahren Werte lautet. Ob dies ein sinnvolles Vorgehen ist, sei dahin gestellt. Bei einer Prognose, die nicht der naiven Prognose entspricht, fällt dies jedoch weniger ins Gewicht, weshalb an dieser Stelle nicht nachgebessert wurde. Weiterhin verwundert, dass ein PTA-Wert kleiner als 1 berechnet wird. Das liegt daran, dass bei der Differenzbildung eine Reihe entsteht, die um einen Wert kürzer ist als die ursprüngliche Reihe, RapidMiner aber bei der Mittelwertbildung durch die Länge der ursprünglichen Reihe dividiert. Bei Zeitreihen mit mehreren hundert oder tausenden von Werten ist dieser Fehler jedoch vernachlässigbar.

6.2 Aufteilung der Daten

Als problematisch stellte sich Menge an zur Verfügung stehenden Daten heraus, indem beobachtet wurde, dass Experimente, die das optimale Modell eines festgelegten Lernverfahrens, also die optimalen Parameter des Verfahrens, bestimmen sollten, mehrere

Wochen liefen, ohne in dieser Zeit aussagekräftige Resultate erzeugt zu haben. Grund dafür war die große Menge an Daten, die als Trainingsbeispiele für das Lernverfahren dienen sollten. Zwar ist es stets der Wunsch, auf einer möglichst großen Menge an Daten sein Verfahren zu trainieren, jedoch hätten die optimalen Parameter des Verfahrens auf diese Weise nicht bestimmt werden können. Eine Anpassung der Parameter führt allerdings in der Regel zu deutlich besseren Ergebnissen. Aus diesem Grund wurde an einigen Stellen eine zufällige Auswahl an Daten getroffen (*Sampling*) und mit diesen gearbeitet. Durch Wiederholung dieses Vorgangs und anschließende Mittelwertbildung der Ergebnisse sollte dennoch ein großes Spektrum der Daten erfasst werden und zur Optimierung der Parameter des Lernverfahrens beitragen.

Um Ergebnisse der verschiedenen Methoden beziehungsweise Modelle miteinander vergleichen zu können, wurde allen Verfahren dieselbe Menge an Trainingsdaten zugrunde gelegt. Zwar stehen Spot-Auktionspreise ab dem Jahr 2000 zur Verfügung (insgesamt ca. 79800 Werte), aber aus oben genanntem Grund werden zum Training und zur Validierung der Modelle lediglich die Daten der Jahre 2006 bis 2008 (einschließlich) genutzt (siehe Abbildung 6.2). Dies ist für die Stützvektormethode insbesondere unter Verwendung der Wavelet-Kernfunktionen immer noch sehr viel, jedoch würde unter einer weiteren Einschränkung der Trainingsmenge die Diversität des Datenmaterials zu sehr leiden. Aus diesem Grund wird im Parameteroptimierungsprozess das erwähnte Sampling-Verfahren eingesetzt, um die Menge der Daten weiter zu reduzieren, dabei aber Diversität zu erhalten. Näheres dazu im Kapitel 6.3.1. Die Zeitreihenprognosen nach Box und Jenkins sind von dieser Einschränkung nicht betroffen, da die entsprechenden Versuchsreihen ausschließlich auf der relativ kurzen Testreihe 1 (siehe unten) durchgeführt werden.

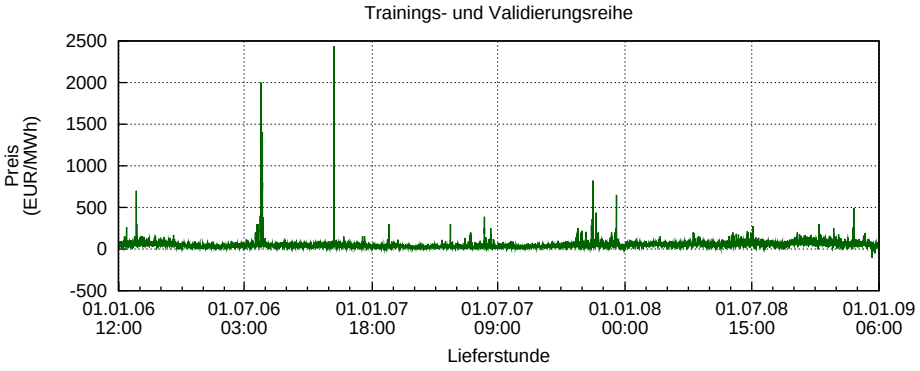
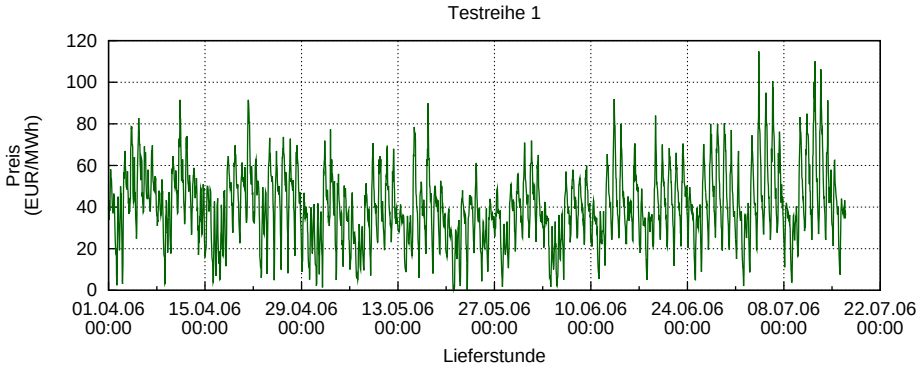


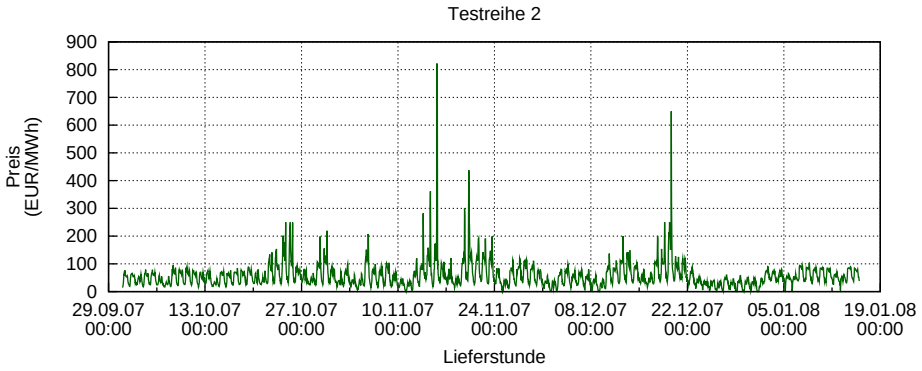
Abbildung 6.2: Preiskurve, die zum Training der Modelle verwendet wird.

Um zuverlässige Aussagen über die Performanz der Methoden treffen zu können, sollten sowohl Trainings- als auch Validierungs- und Testdaten disjunkt zueinander sein [Witten und Frank 2005]. Die Anwendung der Kreuzvalidierung während der Optimierung der Parameter sorgt hierbei für eine disjunkte Trennung von Trainings- und Validierungsdaten. Anders bei den Testdaten. Der Verlauf der Strompreise weist in Abschnitten ein stark unterschiedliches Verhalten auf. Manche Wochen und sogar Monate verlaufen gleichförmig und ohne stärkere Schwankungen, wohingegen in anderen Zeiträumen die Preise auch mal um ein Vielfaches in die Höhe schnellen und starke Ausreißer

enthalten. Die Performanz der Modelle soll unabhängig voneinander auf den verschiedenen Intervalltypen evaluiert werden, weshalb 4 unterschiedliche Testreihen festgelegt wurden (Abbildung 6.3 und 6.4). Auf ihnen wurde untersucht, wie sich die Verfahren auf gemäßigten oder stark schwankenden Intervallen verhalten, sowie wie ihre durchschnittliche Performanz über mehrere Jahre hinweg ist. Es sei allerdings angemerkt, dass sich die Testdaten teilweise mit den Trainingsdaten überschneiden. Da jedoch das Training der Stützvektormodelle nur auf einem Teil der Trainingsdaten stattfindet, wird die Glaubwürdigkeit der Ergebnisse nicht zu sehr beeinträchtigt. Die Testreihe „2009“ wurde im Vergleich dazu in keinem Schritt der Optimierung verwendet. Sie stellt die ideale Testmenge dar und enthält die aktuellsten Daten.

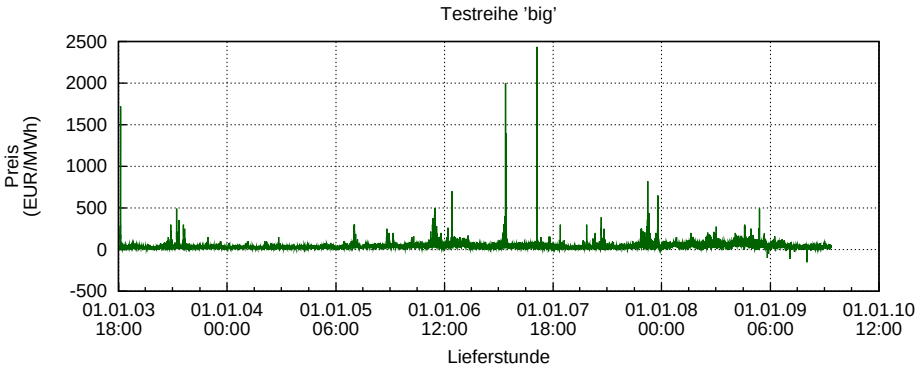


(a) Testreihe ohne starke Schwankungen, 107 Tage

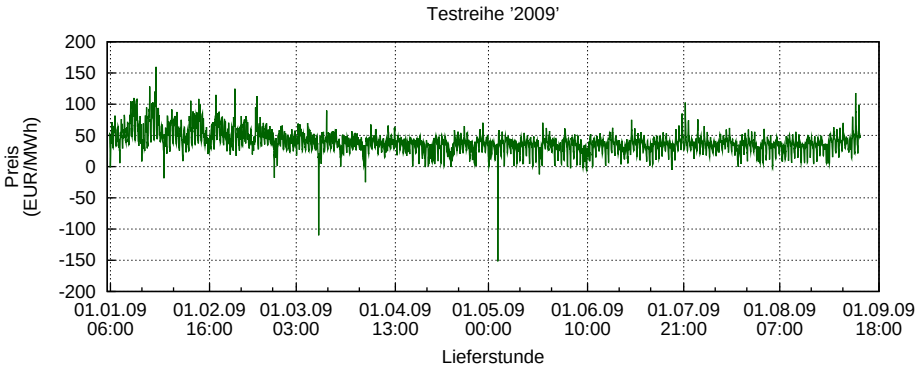


(b) Testreihe mit stärkeren Schwankungen, 107 Tage

Abbildung 6.3: Ausschnitte aus der Reihe der Strompreise des Day-Ahead Auktionshandels der EEX. Sie dienen als Testreihen für die durchgeführten Experimente. Testreihe 1 steht dabei für Intervalle, in denen keine stärkeren Schwankungen auftreten, Testreihe 2 für Zeiträume mit intensiveren Schwankungen.



(a) Testreihe 2003-2009



(b) Testreihe nur 2009

Abbildung 6.4: Ausschnitte aus der Reihe der Strompreise des Day-Ahead Auktionshandels der EEX. Sie dienen als Testreihen für die durchgeführten Experimente. Anhand Testreihe „big“ kann die durchschnittliche Performanz einer Prognose über mehrere Jahre hinweg gemessen werden, Testreihe „2009ënthält als einzige Reihe ausschließlich Daten, die an keiner Stelle in den Optimierungsprozess eingeflossen sind.

6.3 Prognosen mit der Stützvektormethode

In dieser Arbeit stehen Lernverfahren mit Hilfe der Stützvektormethode im Vordergrund. Zusammen mit dem leicht erweiterbaren Data-Mining Programm RapidMiner ist es hier besonders einfach, unterschiedliche Lernverfahren und Vorverarbeitungsschritte miteinander zu kombinieren und auf seinen Daten zu testen. Im Rahmen dieser Diplomarbeit wurde RapidMiner um einige allgemeine, wie auch um einige speziell angepasste Module (Operatoren) erweitert, um die Ideen dieser Arbeit realisieren zu können.

Das nachfolgende Kapitel gibt einen kurzen Einblick in die Funktionsweise von RapidMiner und stellt das in dieser Arbeit vornehmlich verwendete Prozess-Setup vor. In diesem Zusammenhang werden auch einige Designentscheidungen begründet. Kapitel 6.3.2 stellt die wesentlichen, neu entwickelten Operatoren vor. Ab Kapitel 6.3.4 werden dann die Experimente vorgestellt, und ihre Ergebnisse diskutiert. Die Fragestellung, die sich durch dieses Kapitel zieht, lautet im Wesentlichen: „Was gewinnt man durch Hinzunahme von weiteren Einflussfaktoren bei der Preisprognose und sind Wavelet-Kernfunktionen für diesen Zweck zu gebrauchen?“. Daher werden zunächst Experimente an „nackten“, untransformierten Reihen durchgeführt und diese sukzessive mit weiteren Features und vorverarbeitenden Maßnahmen angereichert.

6.3.1 Versuchsaufbau mit RapidMiner

RapidMiner [Mierswa et al. 2006] ist das primär genutzte Werkzeug für Aufgaben des maschinellen Lernens am Lehrstuhl VIII für künstliche Intelligenz an der Technischen Hochschule Dortmund und wurde ebenfalls dort entwickelt. Lernaufgaben, in RapidMiner *Prozesse* genannt, lassen sich hier leicht mit Hilfe einzelner Komponenten, den *Operatoren*, modellieren. Die Operatoren werden dazu in eine Baumstruktur, den *Operatorbaum* eingefügt, welcher den zeitlichen Ablauf des Prozesses darstellt. Abbildung 6.5 zeigt den Operatorbaum, der gleichzeitig das Grundgerüst der folgenden Experimente darstellt.

Nahezu jeder Operator verfügt über konfigurierbare Parameter, die das Verhalten des Operators steuern. Die meisten Operatoren werden genau ein Mal durchlaufen, es gibt jedoch Ausnahmen. Der Operator „XValidation“ beispielsweise stellt eine Kreuzvalidierung dar. Seine inneren Operatoren werden so oft angewendet, wie es der Parameter „number_of_validations“ vorgibt. Die Ergebnisse eines Operators werden an den jeweils nachfolgenden Operator weiter gereicht. Nach Durchlaufen des letzten Operators bekommt der Benutzer das Endergebnis präsentiert. Das können Modellbeschreibungen, aber auch ganze Datentabellen sein. Im Falle von Tabellen lassen sich diese auch direkt am Bildschirm mit einer Vielzahl von verfügbaren Plots grafisch darstellen und als Bild speichern. Die Plots in dieser Arbeit sind allerdings nicht mit RapidMiner erstellt, da RapidMiner derzeit nur Bitmap- jedoch keine Vektorgrafiken speichern kann (lediglich Bitmaps in Vektorgrafik-Containern).

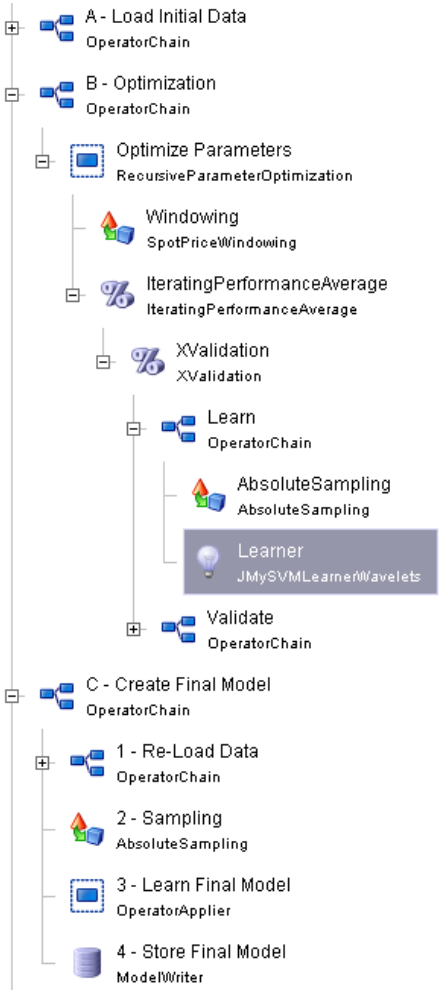


Abbildung 6.5: RapidMiner Operatorbaum. Der hier dargestellte Prozess ist das Grundgerüst aller auf der SVM basierenden Experimente in dieser Arbeit. Jedem Operator ist ein frei wählbarer Name (Text oben) und eine eindeutige Operatorklasse, die seinen Typ bestimmt (Text unten) zugeordnet.

6.3.2 Neue Operatoren

Sind einem die grundlegenden RapidMiner Operatoren bekannt und schaut man sich den Prozess aus Abbildung 6.5 etwas genauer an, so fallen einige neue Operatoren auf, die im Folgenden beschrieben werden.

RecursiveParameterOptimization In Kapitel 3.2.3 wurde bereits ein neuer Operator zur Parameteroptimierung erwähnt. Ganz wie bei der Gitter-Parameteroptimierung mit linearer Aufteilung des Parameter-Suchbereichs wird auch hier der Suchbereich linear in n gleich große Unterbereiche aufgeteilt. Die Mittelpunkte der Unterbereiche stellen die Kandidaten für optimale Parameterwerte in der ersten Runde dar. Unter ihnen wird derjenige Parameterwert gewählt, der zum besten Performanzmaß führt. Nun wird rekursiv der Bereich um diesen Parameterwert weiter eingeschränkt und im näheren Umfeld auf dieselbe Weise nach besseren Parameterwerten gesucht. Unterschreitet einer der zu durchsuchenden Unterbereiche eine gewisse Mindestgröße, so bricht der Algorithmus für diesen Parameter an dieser Stelle ab. Abbildung 6.6 stellt diesen Optimierungablauf für zwei ausgewählte Parameter einer SVM mit RBF-Kernfunktion dar.

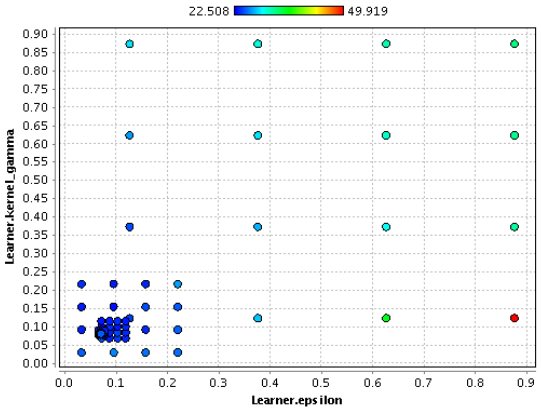


Abbildung 6.6: Rekursive Parameteroptimierung am Beispiel zweier zu optimierender Parameter „kernel gamma“ und „epsilon“. Die Farbe stellt das RMSE Fehlermaß dar. Je kleiner (blauer) der RMSE, desto besser die Wahl der Parameter.

SpotPriceWindowing Ein spezialisierter Operator zur Fensterung. Er ermöglicht neben der Aufteilung der Zeitreihe in gefensterte Beispiele auch die Zentrierung und Normalisierung der Fensterwerte, sowie die Anreicherung durch weitere, teilweise extern zuladbare Features.

JMySVMLearnerWavelets Ein um Wavelet-Kernfunktionen erweiterter JMySVMLearner Operator.

OperatorApplier Erlaubt, an beliebiger Stelle einen beliebigen anderen Operator, oder eine ganze Operatorkette (OperatorChain) anzuwenden. So ist es möglich, zentrale

Operatoren mit ihren Parametern nur einmal im Operatorbaum zu definieren und diese von beliebiger Stelle aus nutzen zu können. Im Beispiel ist so lediglich nötig, die Parameter des Operators namens „Learner“ einmalig zu ändern, um dieselben Änderungen später beim Lernen des endgültigen Modells (Operator „3 - Learn Final Model“) automatisch mit zu nutzen.

6.3.3 Zur Wahl der Parameter

Da im Allgemeinen Unsicherheit bei der korrekten Wahl der Parameter eines Lernverfahrens herrscht, optimieren sämtliche Experimente ihre wesentlichen Parameter in einer äußeren Schleife. Trotzdem müssen, vor allem bei reellwertigen Parametern noch sinnvolle Grenzen gesetzt werden, um die Suche nicht unnötig zu verlängern.

In vielen wissenschaftlichen Arbeiten, unter anderem in [Hsu et al. 2003], wird auf die Wichtigkeit des Parameters C der SVM hingewiesen (siehe dazu auch Kapitel 3.3). Erstaunlicherweise hatte die Änderung dieses Parameters in den folgenden Experimenten, sowohl in kleinem als auch in großem Maßstab keinen wesentlichen Einfluss auf die Güte der Approximation. Ein Experiment zur Suche nach dem besten Wert für C durch zufällige Wahl zeigt Abbildung 6.7. Man kann klar erkennen, dass zwar eine untere Grenze für den RMSE existiert, jedoch wird diese bei allen getesteten Werten für C auch erreicht, weshalb C in den Experimenten dieser Arbeit auf dem Standardwert 0 belassen wurde. In diesem Fall setzt der JMySVMLearner C eigenständig auf den Wert $n \cdot (\sum_{i=1}^{n} K(x_i, x_i))^{-1}$, wobei $K(\cdot, \cdot)$ die verwendete Kernfunktion und x_i die Beispiele darstellen.

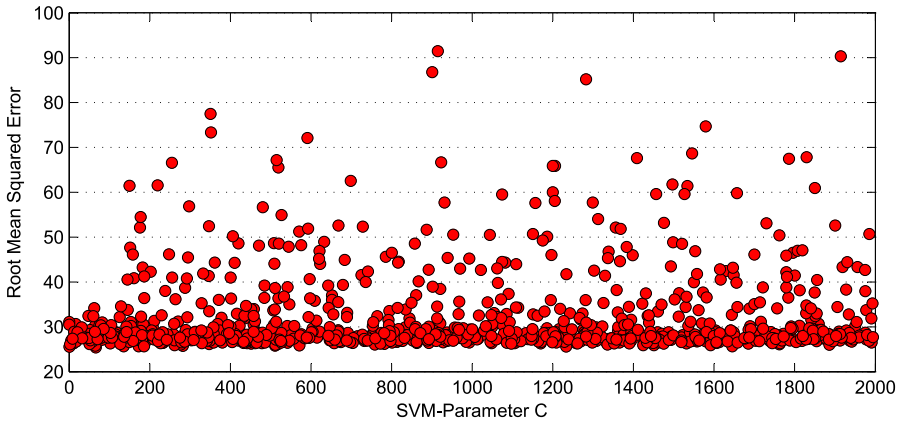


Abbildung 6.7: Auswirkung unterschiedlicher Parameterwerte für C auf den mittleren quadratischen Fehler.

Ebenfalls sollte die zu optimierende Fenstergröße eingeschränkt werden. Erste Experimente optimierten die Fenstergröße in einem sehr naiv gesetzen Rahmen zwischen 2 und 500. Als optimal erwies sich schließlich eine Fenstergröße von 196. Das mit dieser Fenstergröße errechnete Modell generalisierte allerdings äußerst schlecht (siehe Abbildung 6.8). Versuche mit enger gesteckten Grenzen zwischen 2 und 30 zeigten, dass die optimale Fensterbreite sich bei ca. 20, häufig auch 24 einpendelte. Dies entspricht in etwa einem vollständigen Tag. Innerhalb dieser Grenzen wurde in späteren Experimenten nach der optimalen Fensterbreite gesucht.

Die jeweils optimierten Parameter sind Tabelle 6.1 zu entnehmen.

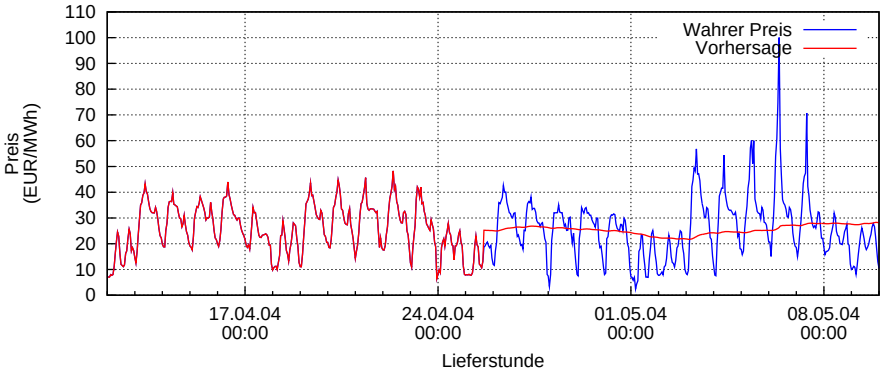


Abbildung 6.8: Probleme mit großer Fensterbreite. Die Parameteroptimierung fand auf der Trainingsmenge die optimale Fensterbreite von 196. Die linke Hälfte des Bildes besteht aus Trainingsdaten, die rechte aus Testdaten. Man sieht deutlich die schlechte Generalisierung bei großen Fensterbreiten.

	Fensterung	SVM	Kernfunktion
linear	Fensterbreite	epsilon	
RBF	Fensterbreite	epsilon	kernel_gamma
Szu	Fensterbreite	epsilon	wavelet_alpha
Morlet	Fensterbreite	epsilon	wavelet_alpha, wavelet_nu
M.Hat	Fensterbreite	epsilon	wavelet_alpha

Tabelle 6.1: Die optimierten Parameter geordnet nach Kategorie und Kernfunktion.

6.3.4 Versuchsreihe: Fensterung ohne Vorverarbeitung

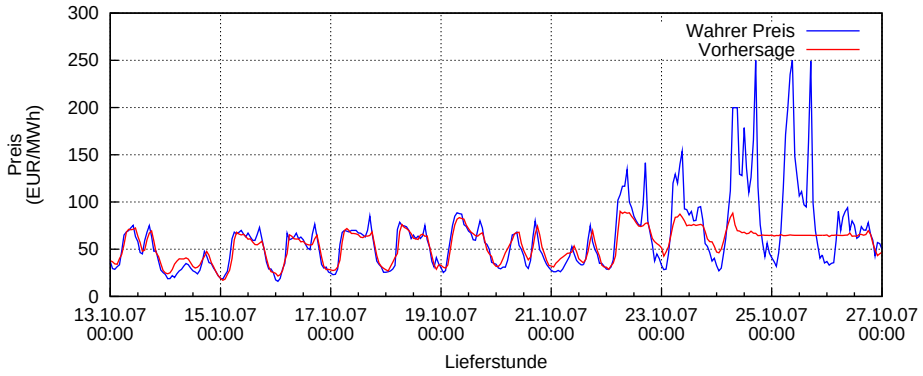
Bei allen folgenden Experimenten basierend auf der Stützvektormethode wird das Verfahren der Fensterung angewendet. Theoretisch wäre es auch denkbar, den Verlauf der Preiskurve als reine Funktion der Zeit zu betrachten und somit die Zeit als einziges Feature und den entsprechenden Preis als Label zu betrachten. Einfache Tests haben jedoch gezeigt, dass dieser Ansatz zum Scheitern verurteilt ist, weshalb ihm in dieser Arbeit keine weitere Aufmerksamkeit geschenkt wird. Stattdessen wird der Ansatz verfolgt, die Stützvektormethode Muster in Form von Zeitfenstern der Vergangenheit lernen zu lassen.

Zunächst interessierte die Performanz der Methode auf den reinen Rohdaten, also der in keiner Weise transformierten (mit Ausnahme der Fensterung) Zeitreihe. Da hier die Absolutwerte der Preise in den Lernbeispielen auftauchen, liegt die Vermutung nahe, dass die Lernverfahren schlecht generalisieren. Unterschiedlichste Preisniveaus lassen ungesehene Beispiele möglicherweise schlecht auf gelernte Beispiele zurückführen. Diese Vermutung bestätigte sich in den Ergebnissen der Experimente. Untersucht wurden 4 unterschiedliche Kernfunktionen, Radial Basis Funktion und die 3 in Kapitel 4.2.9 eingeführten Wavelet-Kernfunktionen, auf den in Kapitel 6.2 vorgestellten Zeitreihenausschnitten. Im Folgenden wird diese Versuchsreihe als Versuchsreihe „**untouched**“ bezeichnet.

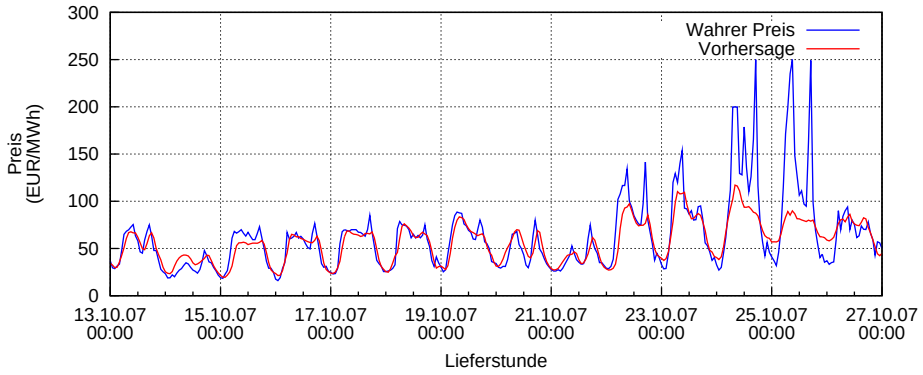
Testreihe	Test 1		Test 2		2009		big	
Fehlermaß	RMSE	PTA	RMSE	PTA	RMSE	PTA	RMSE	PTA
linear	5.752	0.697	25.533	0.694	6.838	0.718	21.150	0.694
RBF	6.889	0.631	33.753	0.643	7.923	0.662	22.622	0.630
Szu	5.910	0.707	36.734	0.644	8.159	0.716	23.600	0.667
Morlet	18.245	0.516	46.212	0.559	18.508	0.518	32.604	0.514
M.Hat	6.005	0.695	37.954	0.639	8.242	0.705	24.341	0.657

Tabelle 6.2: Versuchsreihe „untouched“: SVM mit Fensterung ohne Vorverarbeitung unter Verwendung unterschiedlicher Kernfunktionen und Testmengen

Auffällig ist die durchweg relativ niedrige PTA von nur selten über 0.7. Im Vergleich zu späteren Experimenten wird also deutlich seltener der korrekte Trend der Zeitreihe prognostiziert. Auch die mittleren quadratischen Fehler (RMSE) liegen im Vergleich deutlich höher als bei Experimenten mit Vorverarbeitung und weiteren zusätzlichen Features. Die anfängliche Vermutung, dass unterschiedliche Preisniveaus das gelernte Modell überfordern zeigt sich auch rein optisch bei näherem Hinsehen bestätigt. Abbildung 6.9 zeigt einen Ausschnitt der Approximation mit Hilfe der Szu-Wavelet und der RBF Kernfunktion. Dieser Ausschnitt enthält hohe Preisspitzen und einen allgemeinen Anstieg des Preisniveaus auf über 100 €/MWh tagsüber. Approximiert die SVM bei niedrigem Preisniveau noch augenscheinlich gut, so bricht ihre Performanz bei starken Schwankungen ein. Die Wavelet-Kernfunktionen prognostizieren sogar nur noch einen nahezu konstanten Wert.



(a) Szu-Wavelet Kernfunktion



(b) RBF Kernfunktion

Abbildung 6.9: Versuchsreihe „untouched“: Ausschnitt aus der Prognose der Testreihe 2. Deutlich zu sehen ist die schlechte Performanz der Modelle auf Daten mit stark schwankendem Preisniveau.

Die lineare Kernfunktion scheint mit nicht aufbereiteten Daten am besten zurecht zu kommen, zumindest was die Fehlermaße betrifft. In späteren Experimenten wird jedoch deutlich, dass der lineare Kern gänzlich unbeeindruckt von Vorverarbeitung und zusätzlichen Features ist.

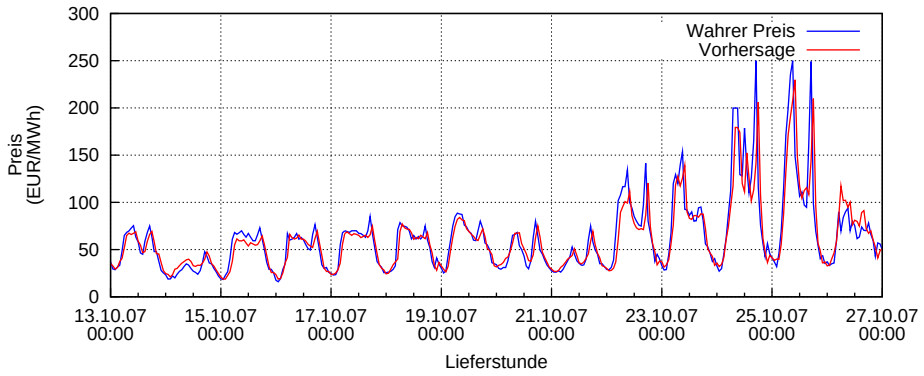


Abbildung 6.10: Lineare Kernfunktion auf Testreihe 2. Augenscheinlich weniger Probleme mit stark schwankenden Preisniveaus, jedoch große Ähnlichkeit zur naiven Prognose durch Vorhersage des letzten bekannten Wertes.

In Abbildung 6.10 ist außerdem zu erkennen, dass die Vorhersage mit dem linearen Kern Anzeichen der naiven Prognose aufweist. Die Vorhersagewerte sind leicht nach rechts versetzt, was bedeutet, dass ein vorangegangener wahrer Wert prognostiziert wurde. Wie in Kapitel 6.1.2 erwähnt muss diese Prognose theoretisch gesehen gar nicht schlechter als andere realistische Prognosen sein, die durch sie induzierten Probleme machen sie jedoch unerwünscht. Es zeigt sich allerdings in späteren Experimenten, dass Prognosen mit dem linearen Kern dieses Verhalten beibehalten.

6.3.5 Versuchsreihe: Zentrierung und Normalisierung

Die vorherige Versuchsreihe hat gezeigt, dass unterschiedliche Preisniveaus zu Problemen bei der Prognose führen können. In vielen Bereichen des maschinellen Lernens und der künstlichen Intelligenz ist es daher angebracht, die zu untersuchenden Daten zunächst in eine normalisierte Form zu bringen. Diese Empfehlung stammt aus dem Bereich der neuronalen Netze und lässt sich auch auf die Stützvektormethode übertragen [Sarle 1997]. Für die Preisverläufe bedeutet das, sie zunächst zu zentrieren und anschließend zu normieren. Daten werden zentriert, indem ein konstanter Wert von ihnen subtrahiert wird. Subtrahiert man den Mittelwert, so sammeln sich die Werte um den Nullpunkt, in unserem Fall die X-Achse. Beim anschließenden Normieren wird jeder Wert durch den maximalen Absolutbetrag der Reihe dividiert. Alle Werte liegen somit im Bereich zwischen -1 und 1.

Würde man die obige Prozedur auf die gesamte Trainingsreihe anwenden, würde es weiterhin größere Bereiche mit einem tendenziell höheren Preisniveau geben als an anderer Stelle. Das liegt an den längerfristigen Schwankungen der Reihe. Erinnert man sich an die Tatsache, dass die Lernverfahren in diesem Kapitel stets auf gefensterten Beispielen trainiert werden, liegt die Idee nahe, jedes Fenster für sich genommen zu zentrieren und normieren.

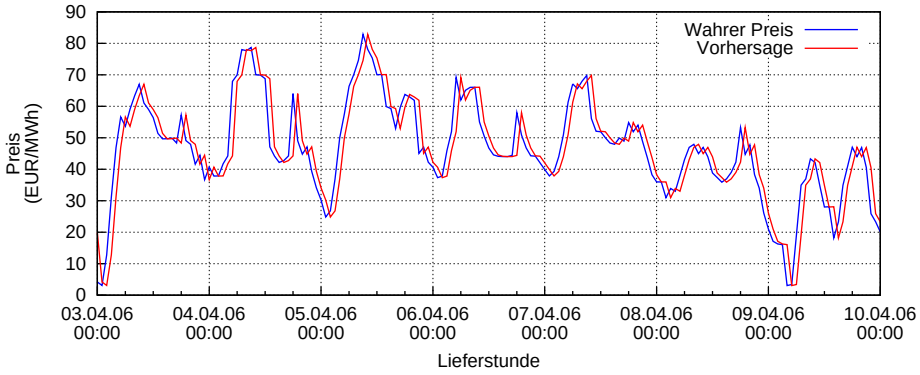
Vermeidung der naiven Prognose

Genau diese Idee führte zur Implementierung eines speziellen Fensterungs-Operator für RapidMiner. Dabei handelt es sich um den bereits in Kapitel 6.3.2 kurz erwähnten **SpotPriceWindowing** Operator. Dieser bietet die Möglichkeit, die Fensterwerte bezüglich des ersten oder letzten Fensterwertes oder bezüglich des Mittelwertes des Fensters zu zentrieren und anschließend zu normieren. Für die restlichen auf der SVM basierenden Experimente wurde diese Art der Transformation als Vorverarbeitungsschritt gewählt.

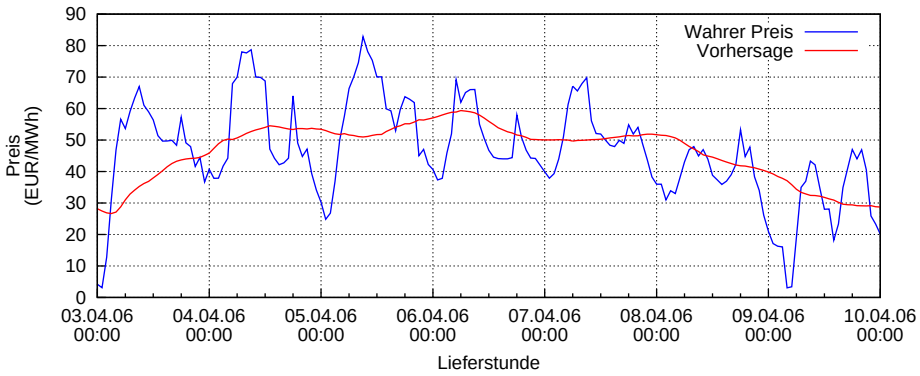
Natürlich müssen vor der Aus- und Bewertung der Prognosen die Beispiele zunächst wieder rücktransformiert werden. Eine allgemeine Aussage über den Approximationsfehler bei unabhängig voneinander zentrierten und skalierten Fenstern wäre ohne vorige Rücktransformation unmöglich. Der SpotPriceWindowing Operator speichert bei der Fensterung entsprechende Meta-Informationen in jedem Fenster, so dass der rücktransformierende Operator (**SpotPriceWindowExamples2OriginalData**) die ursprüngliche Zeitreihe exakt wieder herstellen kann. Die entsprechenden Prognosewerte werden dabei auf dieselbe Weise rücktransformiert, so dass nachträglich gemessene Approximationsfehler vergleichbar werden.

Die von RapidMiner zur Verfügung gestellte Form der Fensterung bietet ebenfalls die Option der Zentrierung (dort „relative Transformation“ genannt). Diese zentriert die Zeitreihenwerte eines jeden Fensters jedoch ausschließlich um den jeweils letzten Wert des Fensters. Der letzte Wert eines Fensters beträgt somit stets 0 und der zu prognostizierende Wert stellt bei der Wahl eines Horizonts von 1 genau die Differenz zum vorherigen Wert dar. Ein Prognosemodell braucht somit stets nur den Wert 0 zu prognostizieren, womit jeweils nach der Rücktransformation der letzte bekannte Wert vorhergesagt wird

(siehe Abbildung 6.11a). Dies entspricht genau der naiven Prognose, die es nach Möglichkeit zu vermeiden gilt. Dass dies nicht immer gelingt, insbesondere bei der Verwendung des linearen SVM-Kerns, wurde in Kapitel 6.3.4 bereits vorweg genommen. Dort sind die Gründe jedoch andere als die einfache Prognose des Wertes 0.



(a) Prognose des Wertes 0, Zentrierung um letzten Wert des Fensters



(b) Prognose des Wertes 0, Zentrierung um Mittelwert des Fensters

Abbildung 6.11: Beide Abbildungen zeigen eine konstante Prognose des Wertes 0 bei einer Fensterung mit Breite 24. Abbildung a) zentriert dabei die Werte eines Zeitfenster um den jeweils letzten Wert herum und erzeugt somit die naive Prognose, wohingegen b) um das arithmetische Mittel des Fensters zentriert.

Zentriert man allerdings die Fenster zuvor um ihren arithmetischen Mittelwert, so führt die Prognose eines konstanten Wertes von 0 zu deutlich schlechteren Ergebnissen (siehe Abbildung 6.11b). Die Prognosen werden zu einem simplen gleitenden Durchschnitt der letzten w Werte, wobei w die Fensterbreite ist. Dies hat zur Folge, dass die Parameterwerte, die zur Berechnung dieses Modells führten, in der Parameteroptimierungsphase mit hoher Wahrscheinlichkeit verworfen und durch bessere ersetzt werden. Umgangssprachlich macht man es dem Lernverfahren also „schwieriger“, ein gutes Modell zu erstellen. Aus diesem Grund wird daher stets die Zentrierung um den arithmetischen Mittelwert gewählt. Diese Versuchsreihe wird im Folgenden „**no features**“ genannt.

Schaut man sich nun die erzielten Approximationsfehler auf den Testdatenreihen an (Tabelle 6.3), sieht man, dass die Normalisierung der Daten teilweise eine deutliche Verbesserung gebracht hat. Vor allem auf stärker schwankenden Zeitreihenausschnitten wie der Testreihe 2 bemerkt man den Unterschied (Abbildung 6.12). Auch die Vorhersage des korrekten Trends hat sich sichtbar verbessert. Nahezu alle Verfahren erzielen über 73% korrekte Trendprognosen auf allen Testdaten.

Testreihe	Test 1		Test 2		2009		big	
Fehlermaß	RMSE	PTA	RMSE	PTA	RMSE	PTA	RMSE	PTA
linear	5.292	0.757	26.120	0.717	6.896	0.743	24.178	0.738
RBF	4.733	0.750	22.381	0.742	6.344	0.734	21.635	0.739
Szu	4.855	0.759	23.897	0.756	6.540	0.713	20.126	0.744
Morlet	5.291	0.736	25.150	0.737	7.247	0.699	23.272	0.733
M.Hat	4.892	0.789	23.236	0.752	6.926	0.738	21.199	0.763

Tabelle 6.3: Versuchsreihe „no features“: SVM mit Fensterung, Zentrierung der Fenster um ihren Mittelwert und anschließender Normalisierung unter Verwendung unterschiedlicher Kernfunktionen und Testmengen

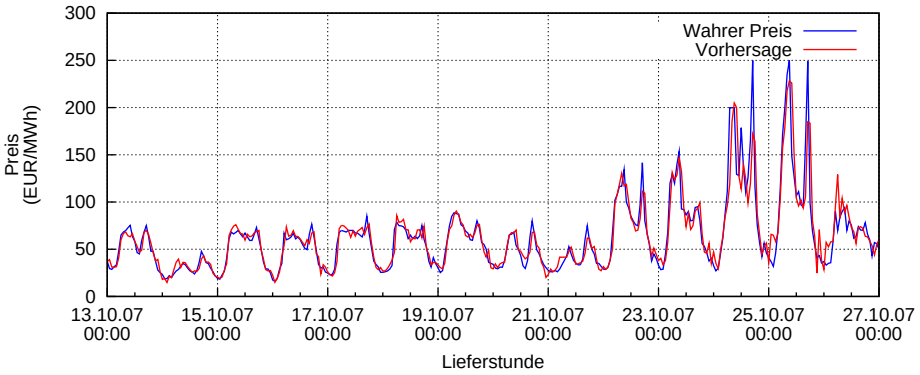


Abbildung 6.12: Versuchsreihe „no features“, Szu Wavelet auf Testreihe 2: Größere Schwankungen im Preisniveau stellen kein prinzipielles Problem mehr dar, lediglich an den Übergangsstellen von hohem zu niedrigem Niveau werden kleinere Approximationsfehler aufgrund der extremen Skalierung verstärkt.

Einzige Auffälligkeit sind Schwankungen auf Zeitfenstern mit höherem Preisniveau, zu sehen in den „Tälern“ der Abbildung 6.12 um den 25. Oktober herum. Dies ist zurückzuführen auf die Normalisierung. Das Lernverfahren arbeitet ja auf Werten zwischen -1 und 1. Bei hohem Preisniveau werden die Prognosen demnach um ein Vielfaches hochskaliert, um das gewünschte Niveau zu erreichen. Etwaige Approximationsfehler werden dabei allerdings mitskaliert und sorgen für einen sehr unruhigen Verlauf an den entsprechenden Stellen. In einem Nachbearbeitungsschritt wird später eine Wavelet-Glättung auf Zeitreihen dieser Art angewendet, um die hochfrequenten Schwingungen herauszufiltern und den Approximationsfehler weiter zu verringern.

Interessant ist, dass alle Kernfunktionen mehr oder weniger gleich gut approximieren. Möglicherweise ist dies ein Indiz dafür, dass die SVM mit den bisher vorgestellten Vorverarbeitungsschritten nicht mehr wesentlich verbessert werden kann. Diese Annahme führt direkt zur nächsten Versuchsreihe: Hinzufügen weiterer Features.

6.3.6 Versuchsreihe: Zusatzfeatures I

Im Bereich des maschinellen Lernens ist es bei Datensätzen mit vielen Attributen üblich, spezielle Feature-Selektionsalgorithmen zu verwenden, um irrelevante Features entfernen bzw. relevante Features feststellen zu können [Mierswa und Morik 2005]. Dies hat neben einer schnelleren Laufzeit vor allem Modelle mit geringerer Komplexität zur Folge. Obwohl die vorgestellten Experimente mit zusätzlichen Features arbeiten, wurde in dieser Arbeit bewusst auf eine automatisierte Feature-Selektion verzichtet. Zum einen ist die Anzahl der Features in den Experimenten klein und gut handhabbar und zum anderen bietet sich so die Möglichkeit, gezielt über den Nutzen einzelner Features zu diskutieren.

Die Preiskurve besitzt einen charakteristischen, zeitabhängigen Verlauf. Tageszeit und Wochentag spielen eine wesentliche Rolle für die Preisentwicklung. So wird der Strom tagsüber aufgrund des höheren Bedarfs in der Regel höher gehandelt als nachts. Auch die Wochenenden unterscheiden sich normalerweise deutlich vom Rest der Woche.

Die Stützvektormethode hat keine Kenntnis über die Herkunft der einzelnen Features der Datensätze. Für die Funktionsweise der SVM spielt es keine Rolle, ob dies nun Zeitreihendaten oder beliebige andere Informationen sind. Die Lernergebnisse wohlgemerkt können durchaus je nach Features variieren. Die Intuition sagt, dass es nicht schaden sollte, die Datensätze mit mehr Features auszustatten, insbesondere, wenn die Features einen direkten Einfluss auf das Preisniveau haben wie es beispielsweise bei Informationen bezüglich Datum und Uhrzeit der Fall ist. Daher soll die folgende Versuchsreihe untersuchen, ob sich die in Kapitel 6.3.5 erzielten Ergebnisse weiter verbessern lassen.

Bei einem gefensterten Datensatz entspricht das Label demjenigen Wert, der im Abstand des Horizontes zum Zeitfenster prognostiziert werden soll (siehe Kapitel 4.1 zur Erinnerung). Das Label eines gefensterten Beispiels bezieht sich also demnach auf eine bestimmte Stunde eines bestimmten Tages. Aus dieser Information werden nun folgende Zusatzfeatures abgeleitet:

day_of_week Ein numerischer Ganzzahlwert zwischen 1 und 7 steht für den Wochentag des Labels. Welcher Wochentag dabei welcher Zahl entspricht, sollte für die SVM völlig unerheblich sein.

hour_of_day Ein ganzzahliger Wert zwischen 0 und 23 steht für die Stunde des Tages, auf die sich das Label bezieht.

is_holiday Ein binärer Wert, der angibt, ob es sich bei dem Tag des Labels um einen gesetzlichen Feiertag handelt oder nicht. Feiertage verhalten sich ähnlich wie Wochenenden, daher scheint eine Kennzeichnung dieser Tage sinnvoll. Aus technischen Gründen wird dieser Wert ebenfalls als Ganzzahl gespeichert und beträgt entweder 0 oder 1.

Ein gefenstertes Beispiel erhält also 3 weitere Features, die nicht unmittelbar zur Zeitreihe gehören. Demnach bleiben sie auch von der Zentrierung und anschließenden Normalisierung unangetastet. Diese Versuchsreihe wird „date-features“ genannt.

Die Ergebnisse des linearen und RBF-Kerns verwundern zunächst nicht. Wie bereits in Kapitel 6.3.4 (Versuchsreihe „untouched“) erwähnt, bleibt die SVM mit dem linearen Kern von Zusatzfeatures weitestgehend unbeeindruckt, die Ergebnisse des RBF-Kerns verbessern sich leicht.

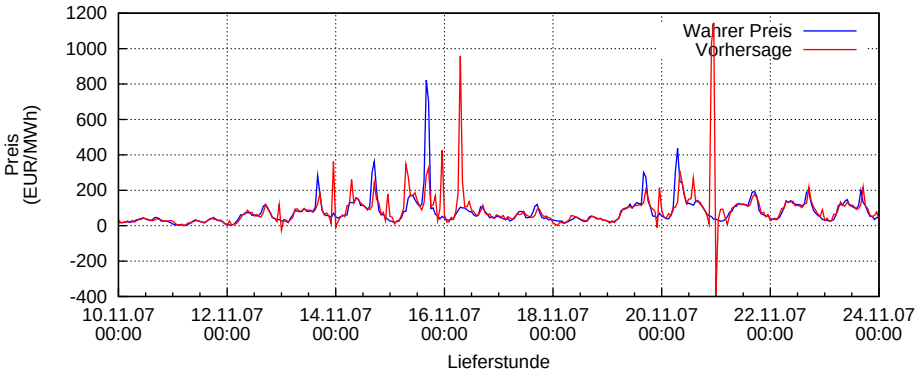
Testreihe	Test 1		Test 2		2009		big	
Fehlermaß	RMSE	PTA	RMSE	PTA	RMSE	PTA	RMSE	PTA
linear	5.240	0.759	25.713	0.714	6.797	0.746	23.994	0.744
RBF	4.344	0.805	21.643	0.779	6.129	0.764	18.853	0.786
Szu	10.153	0.753	50.586	0.737	15.964	0.669	31.307	0.713
Morlet	64.752	0.709	106.505	0.711	60.029	0.643	73.833	0.690
M.Hat	20.264	0.730	58.818	0.712	24.063	0.677	33.562	0.713

Tabelle 6.4: Versuchsreihe „date-features“: SVM mit Fensterung, Zentrierung der Fenster um ihren Mittelwert und anschließender Normalisierung unter Verwendung zusätzlicher, datumsbezogener Features auf unterschiedlichen Kernfunktionen und Testmengen

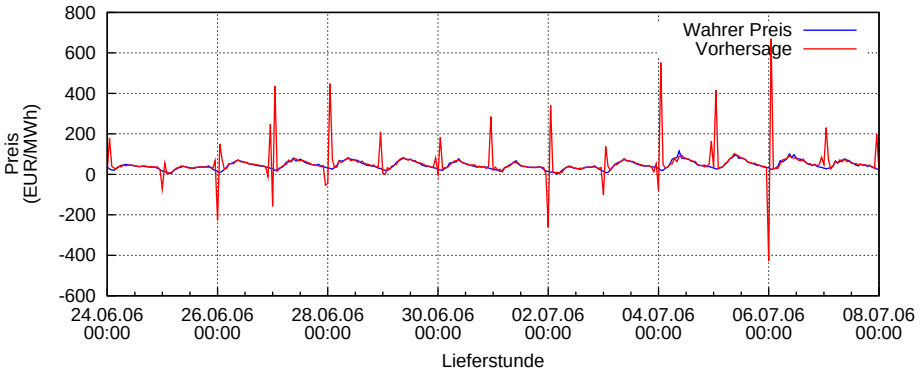
Äußerst merkwürdig verhalten sich allerdings die Experimente mit den Wavelet-Kernfunktionen. Der prognostizierte Trend mag noch im akzeptablen Bereich liegen, jedoch schnellen die mittleren quadratischen Fehler in die Höhe.

Abbildung 6.13 zeigt deutlich, dass sich unerwartet starke Ausreißer in die Prognosereihe einschleichen. Treten diese unter Verwendung des Szu-Wavelet Kerns noch vereinzelt und vornehmlich auf Testreihe 2 auf (Abbildung 6.13a), so sind sie in der Prognosereihe des Morlet-Wavelet Kerns schon auf Testreihe 1 in hoher Regelmäßigkeit vorzufinden. Die Tatsache, dass diese Spitzen stets in der Nacht zum nächsten Tag auftreten, lässt darauf schließen, dass hier die Hinzunahme der weiteren Features ein numerisches Problem für die Wavelet-Kernfunktion darstellt, was möglicherweise an den unskalierten Werten für die Wochentage und Stunden des Tages liegt.

Nach dieser Erkenntnis schwindet die Hoffnung, Wavelet-Kernfunktionen durch Hinzunahme weiterer Features wieder zu stabileren Ergebnissen zu bewegen. Ob die RBF-Kernfunktion jedoch von weiteren Features profitieren kann, ist zunächst noch unklar und soll in der nächsten Versuchsreihe untersucht werden.



(a) Szu-Wavelet auf Testreihe 2



(b) Morlet-Wavelet auf Testreihe 1

Abbildung 6.13: Versuchsreihe „date-features“: Anomalien bei der Verwendung von Wavelet-Kernfunktionen.

6.3.7 Versuchsreihe: Zusatzfeatures II

Als weiterer großer Einflussfaktor auf die Strompreisentwicklung gilt das Wetter [RTE 2009], [Pardo et al. 2002]. Ist es warm, so wird Energie zum Kühlen benötigt. Genauso braucht man Energie zum Heizen an kalten Tagen. Dieser erhöhte Bedarf wirkt sich direkt auf die Preisgestaltung aus. Daher liegt es nahe, entsprechende Kennzahlen in den Lernprozess zu integrieren. Ob die Stützvektormethode dies zu ihrem Vorteil nutzen kann, soll diese Versuchsreihe unter dem Namen „**full-features**“[2] zeigen.

Relative Normtemperatur

Zunächst soll allerdings kurz erläutert werden, welche Arten von Wetterdaten benutzt werden. Grundsätzlich scheint die Unterscheidung zwischen warmen und kalten Tagen sinnvoll zu sein. Diese Unterscheidung ist maßgeblich für einen charakteristischen Verlauf des Preisniveaus über einen Tag.

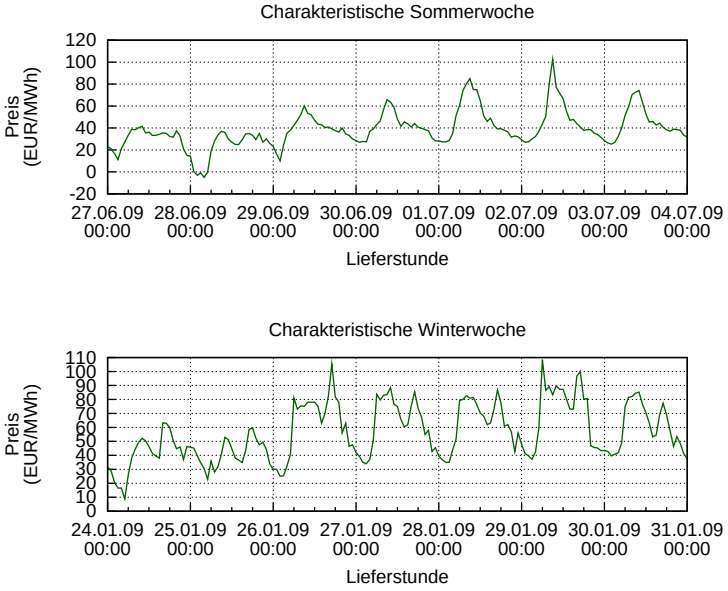


Abbildung 6.14: Charakteristische Tageszyklen des Preisniveaus je einer Woche im Sommer und Winter.

Die genaue Form der Tagesverläufe spielt in dieser Arbeit allerdings keine Rolle, der interessierte Leser sei an dieser Stelle auf [RTE 2009] verwiesen.

In dieser Versuchsreihe soll ein zusätzliches Feature in die Daten integriert werden, welches Auskunft darüber gibt, ob der Tag eher ein sommerlicher, warmer Tag oder eher ein kalter Tag ist. Dazu wurden unter anderem sogenannte Normtemperaturen für jeden Tag des Jahres von der MeteoGroup Deutschland GmbH zur Verfügung gestellt. Aus diesen Normtemperaturen wurde ein Jahresmittel gebildet. Die Abweichung der Norm-

[2] „full“, weil ebenso alle in den Versuchsreihen zuvor vorgestellten Zusatzfeatures integriert werden.

temperatur eines Tages zum Jahresmittel charakterisiert den Tag folglich als eher warm oder eher kalt und wurde als weiteres, reellwertiges Feature namens „**norm_temp_rel**“[3] in die Trainingsmenge integriert.

An dieser Stelle sei darauf hingewiesen, dass die Werte dieses neuen Features über den Tag gesehen identisch bleiben, da lediglich Tagestemperaturen als Normwerte zur Verfügung stehen.

Stundengenaue Temperaturen

Um den Einfluss der Temperatur auch im Laufe eines einzelnen Tages für den Lernalgorithmus nutzbar zu machen, sollten Temperaturwerte in stündlicher Auflösung hinzugenommen werden. Aus Sicht eines Händlers, der an den zukünftigen Strompreisen interessiert ist, sind zum Zeitpunkt seiner Auktionsgebote allerdings lediglich Temperatur*vorhersagen* für den nächsten Tag vorhanden. Idealerweise sollte der Lernalgorithmus genau diese Temperaturvorhersagen als weitere Features erhalten, jedoch, getreu dem Motto „*nichts ist älter als die Zeitung von gestern*“, waren Temperaturvorhersagen nur mit einer zu kurzen Historie verfügbar.

Daher werden ersatzweise die nächstbesten und gleichzeitig auch bestmöglichen Prognosen verwendet: Temperatur*messwerte* in stündlicher Auflösung mit einer bis zum 1. Januar 2003 zurückreichenden Historie. Die Temperaturen liegen jedoch nicht als Werte für ganz Deutschland vor, sondern lediglich als Messwerte 5 deutscher Großstädte (Berlin, Essen, Frankfurt am Main, Hamburg und München). Da der Day-Ahead Auktionshandel an der EEX jedoch keine regionalen Unterschiede macht, wurden die Temperaturen für jede Stunde über alle genannten Städte gemittelt und so eine durchschnittliche Temperatur für ganz Deutschland berechnet.

Wie schon bei den relativen Normtemperaturen sollten auch hier die Temperaturwerte nicht als absolute Zahlen verwendet werden. Stattdessen werden sie um die jeweilige Normtemperatur des Tages zentriert und bilden so als Abweichung von der Normtemperatur das reellwertige Feature „**norm_temp_deviation**“.

Die Experimente dieser Versuchsreihe integrieren nun zusammen mit den zusätzlichen Features der Versuchsreihe „date-features“ insgesamt 5 neue Features in die Beispiele. Die Verbesserung der Ergebnisse durch Hinzunahme der Temperaturfeatures bleibt jedoch aus. Tabelle 6.5 zeigt, dass der lineare und der RBF-Kern im vernachlässigbaren Rahmen unveränderte Ergebnisse im Vergleich zur Versuchsreihe „date-features“ liefern, wohingegen die Wavelet-Kerne wie zuvor inakzeptable Fehler bei der Prognose machen. Die Fehlermaße bei der Verwendung des Mexican Hat Wavelets haben sich zwar numerisch verbessert, die Prognosekurve weist jedoch dieselben Probleme an den Datumsgrenzen auf, wie im vorigen Kapitel am Beispiel des Morlet Wavelets demonstriert. Dieses Phänomen ist mal mehr, mal weniger stark ausgeprägt, lässt sich aber nicht auf bestimmte Charakteristika der Preiskurve zurückführen. Abbildung 6.15 zeigt, dass das Fehlverhalten auch auf einem ansonsten eher unauffälligen Abschnitt der Preiskurve auftritt und nicht nur, wie in den ersten Experimenten, an Tagen mit extremen Preisniveauschwankungen.

[3] Für „relative Normtemperatur“ als Gegensatz zur absoluten Normtemperatur.

Testreihe	Test 1		Test 2		2009		big	
Fehlermaß	RMSE	PTA	RMSE	PTA	RMSE	PTA	RMSE	PTA
linear	5.242	0.759	25.903	0.719	6.814	0.746	23.988	0.742
RBF	4.704	0.802	21.826	0.770	6.602	0.744	19.837	0.772
Szu	22.332	0.752	58.305	0.757	23.699	0.687	35.348	0.727
Morlet	75.615	0.693	102.139	0.724	61.582	0.652	78.290	0.683
M.Hat	18.790	0.750	42.272	0.764	19.400	0.672	29.305	0.720

Tabelle 6.5: Versuchsreihe „full features“: SVM mit Fensterung, Zentrierung der Fenster um ihren Mittelwert und anschließender Normalisierung unter Verwendung zusätzlicher, datums- und temperaturbezogener Features auf unterschiedlichen Kernfunktionen und Testmengen

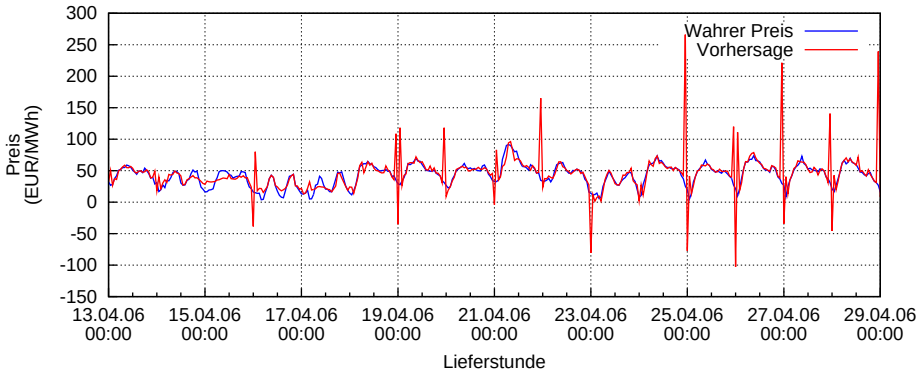


Abbildung 6.15: Versuchsreihe „full-features“: Die Verwendung der Mexican Hat Wavelet-Kernfunktion weist dieselben Anomalien auf, wie zuvor bei der Verwendung der Morlet Wavelet-Kernfunktion.

Da auch hier die Anomalien stets an den Datumsgrenzen auftreten, ist anzunehmen, dass eines der in Versuchsreihe „date-features“ eingeführten neuen Features die Ursache dieses Phänomens darstellt. Als bester Kandidat kommt die Angabe der Stunde des Tages (Feature „hour_of_day“) in Frage. Die nächste Versuchsreihe untersucht daher, ob das Eliminieren bzw. Weglassen dieses Features auch die bei Wavelet-Kernen auftretenden Anomalien beseitigt.

6.3.8 Versuchsreihe: Verzicht

In den Versuchsreihen „date-features“ und „full-features“ lieferte die Stützvektormethode unter der Verwendung von Wavelet-Kernfunktionen Prognosekurven, welche abnormale Peaks in regelmäßigen Abständen von je 24 Stunden enthielten. Als mögliche Ursache kam das zusätzliche Feature „hour_of_day“ in Frage, da dieses am besten zum beobachteten Phänomen passt. Das Feature hat exakt zu den Zeiten, an denen die Anomalien auftreten, den Wert 0, was den Zusammenhang nahe legt.

Die folgende Versuchsreihe untersucht nun, wie sich die Ergebnisse in Bezug auf die zuvor beobachteten Anomalien ändern, wenn das besagte Feature nicht mit in die Daten integriert wird (Tabelle 6.6).

Testreihe	Test 1		Test 2		2009		big	
Fehlermaß	RMSE	PTA	RMSE	PTA	RMSE	PTA	RMSE	PTA
linear	5.259	0.757	26.165	0.713	6.843	0.740	24.081	0.735
RBF	4.700	0.784	21.370	0.760	6.609	0.734	21.614	0.750
Szu	5.007	0.798	22.522	0.776	6.866	0.714	20.742	0.754
Morlet	39.645	0.646	261.697	0.627	25.535	0.636	83.067	0.630
M.Hat	11.562	0.779	61.767	0.752	10.029	0.717	29.008	0.748

Tabelle 6.6: Versuchsreihe „hour of day“: SVM mit Fensterung, Zentrierung der Fenster um ihren Mittelwert und anschließender Normalisierung unter Verwendung zusätzlicher, datums- und temperaturbezogener Features mit Ausnahme der Stunde des Tages (hour of day) auf unterschiedlichen Kernfunktionen und Testmengen

Man erkennt, dass der lineare Kern, wie auch schon in den Experimenten zuvor, nahezu unveränderte Approximationsfehlermaße liefert. Auch der RBF-Kerns scheint unbeeindruckt des fehlenden Features ähnlich gute Ergebnisse zu erzeugen wie in der Versuchsreihe „full-features“. Dies ist ein starkes Indiz dafür, dass die Information über die Stunde des Tages von der SVM mit den getesteten Kernfunktionen nicht sinnvoll genutzt wird oder werden kann.

In der Tat verbessert[4] haben sich die Approximationsfehler des Wavelet-Kerns unter Verwendung des Mexican Hat Wavelets. Sie kommen zwar bei weitem noch nicht an die Approximationsgüte der SVM mit linearem und RBF-Kern heran, doch sind die regelmäßig alle 24 Stunden aufgetretenen Anomalien verschwunden. Dafür treten allerdings andersartige Anomalien in Erscheinung (siehe Abbildung 6.16). Wie man in der Grafik erkennt, verschätzt sich die Prognose an manchen Tagen immens. Auch vereinzelte Peaks sind noch zu beobachten (vgl. Mitte des 28. April in der Grafik), weshalb man vom praktischen Einsatz des Mexican Hat Wavelet-Kerns bei der Zeitreihenprognose von ökonometrischen Daten mit der SVM absehen sollte.

[4] bis auf die Ergebnisse auf Testreihe 2

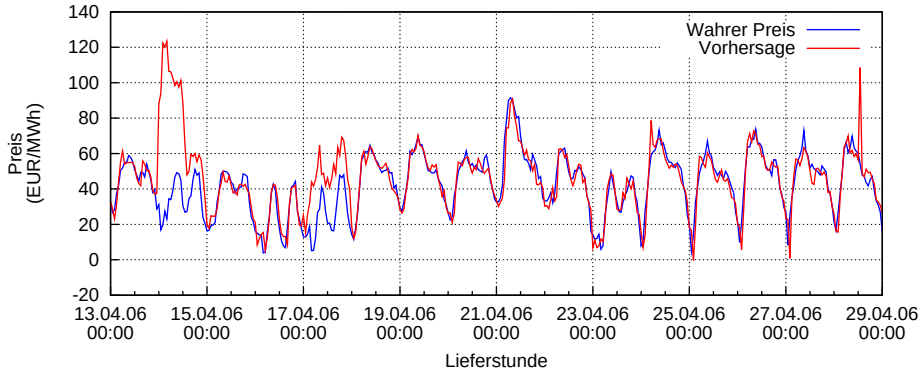


Abbildung 6.16: Versuchsreihe „hour of day“, Mexican Hat: Die mitternächtlichen Peaks sind verschwunden, dafür liegt nun die Prognose an manchen Tagen grob daneben.

Positiv überrascht hat hingegen die Verbesserung, die sich im Zusammenhang mit der Verwendung des Szu-Wavelets gezeigt hat. Er erreicht beinahe persönliche Bestwerte bezüglich der Approximationsfehler. Einzig einige vereinzelte Peaks stören noch das Gesamtbild einer durchaus akzeptablen Prognose (Abbildung 6.17). Nur das völlige Weglassen zusätzlicher Features lässt die Szu-Wavelet Kernfunktion noch bessere Ergebnisse erzielen.

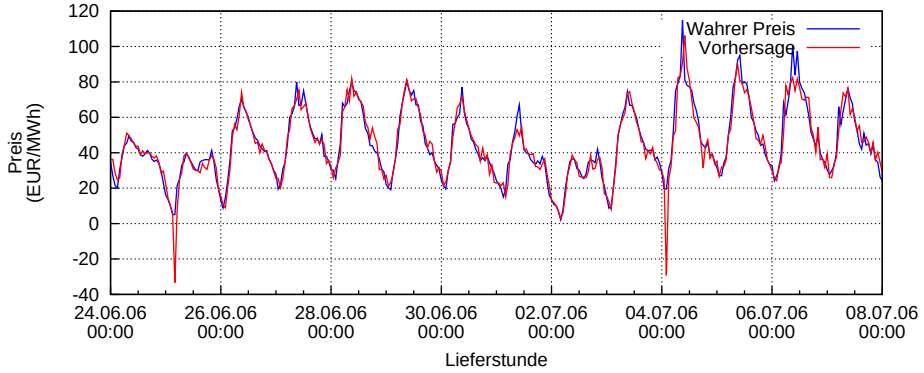


Abbildung 6.17: Versuchsreihe „hour of day“, Szu-Wavelet: Gute Prognoseeigenschaften, lediglich einige wenige Peaks stören das Gesamtbild.

Ganz und gar unbrauchbar sind allerdings immer noch die Prognosen basierend auf Berechnungen mit dem Morlet Wavelet-Kern (Abbildung 6.18). Ohne unmittelbar ersichtlichen Grund weichen die Prognosen um das Tausendfache vom wahren Wert ab. In dieser Form kann keine sinnvolle Prognose getroffen werden. Die Ursache für dieses außergewöhnliche Verhalten bleibt im Rahmen dieser Arbeit unbekannt. Auch Implementierungsfehler können nicht völlig ausgeschlossen werden.

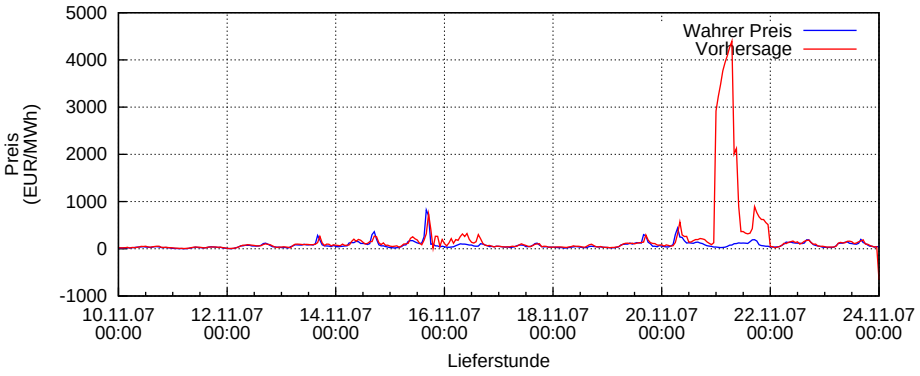


Abbildung 6.18: Versuchsreihe „hour of day“, Morlet Wavelet: Unbrauchbare Prognose durch scheinbar willkürliche, astronomisch große Abweichungen vom wahren Wert.

6.4 Wavelet-Glättung

Die folgenden Versuchsreihen haben im Zentrum ihrer Untersuchungen Verfahren aus der Wavelet Vor- und Nachbearbeitung, insbesondere die Wavelet-Glättung. Daher werden die Ergebnisse, sofern sie SVM-Lernverfahren betreffen, nicht mehr für jede im Kapitel 6.3 diskutierte Kernfunktion durchgeführt und stattdessen ausschließlich der RBF-Kern benutzt.

6.4.1 Versuchsreihe: Wavelet Nachbearbeitung

Obwohl etliche Prognosen, insbesondere die der RBF-Kernfunktion, bereits gute Approximationen darstellen, fallen bei näherem Hinschauen unschöne, gezackte Verläufe auf (Abbildung 6.19).

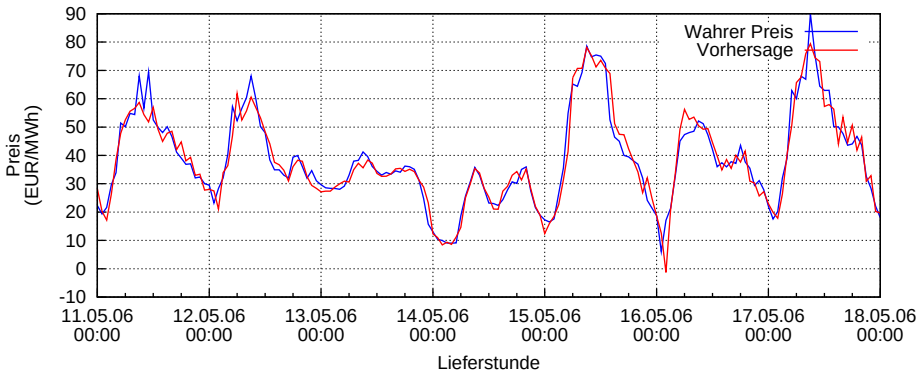


Abbildung 6.19: Prognose mit RBF-Kern. Obwohl die Approximation gut ist, sieht man streckenweise unregelmäßige, gezackte Verläufe.

Es spricht nichts dagegen, eine Prognosekurve noch nachträglich zu bearbeiten, falls so eine bessere Approximation erreicht werden kann. In Kapitel 4.2.7 wurde die Technik der Wavelet-Glättung vorgestellt. Eine passende Glättung ist in der Lage, hochfrequente Schwingungen aus einer Zeitreihe zu entfernen. Der Grad der Glättung richtet sich dabei allein nach den verwendeten Threshold-Parametern. Somit ergibt sich auf natürliche Weise ein Optimierungsproblem.

Das Finden optimaler Parameter mit anschließender Bewertung wurde im Kapitel 6.3 ausgiebig exerziert. In dieser Versuchsreihe nutzen wir einen ähnlichen Aufbau, um die optimalen Thresholding-Parameter zu bestimmen und messen anschließend ihre Performanz auf den Testreihen. Die Optimierung wird an 3 ausgewählten Wavelets (Haar, Daubechies-4 und Daubechies-20) durchgeführt, als zugrunde liegende Prognosereihe wird die Approximation des RBF-Kerns aus der Versuchsreihe „full-features“ aus Kapitel 6.3.7 benutzt.

Es muss erwähnt werden, dass die in RapidMiner implementierten Wavelet-Operatoren, insbesondere der Operator zur diskreten Wavelet-Transformation eine Zeitreihe von der

Länge einer Zweierpotenz erwartet. Dementsprechend wurden alle betreffenden Zeitreihen auf die nächstkleinere Zweierpotenz verkürzt. Das hat unter anderem zur Folge, dass die Werte der Fehlermaße der Originalreihe nicht mehr mit den Werten aus Kapitel 6.3.7 übereinstimmen.

Testreihe	Test 1		Test 2		2009[5]	
Fehlermaß	RMSE	PTA	RMSE	PTA	RMSE	PTA
Original (gekürzt)	4.549	0.796	24.120	0.770	7.306	0.740
Haar	4.416	0.782	24.128	0.777	7.271	0.729
Daub 4	4.018	0.807	23.006	0.779	6.890	0.757
Daub 20	3.951	0.814	22.622	0.795	6.837	0.769

Tabelle 6.7: Versuchsreihe „Wavelet Nachbearbeitung“: Approximationsgüten lassen sich mit Hilfe der Daubechies Wavelets spürbar verbessern.

Wie Tabelle 6.7 zu entnehmen ist, empfiehlt sich eine nachbearbeitende Glättung mit dem Daubechies-20 Wavelet, um den Prognosefehler teilweise deutlich zu senken. Abbildung 6.20 zeigt denselben Ausschnitt wie Abbildung 6.19, jedoch mit einer geglätteten Prognosekurve, basierend auf der Wavelet-Glättung mit dem Daubechies-20 Wavelet.

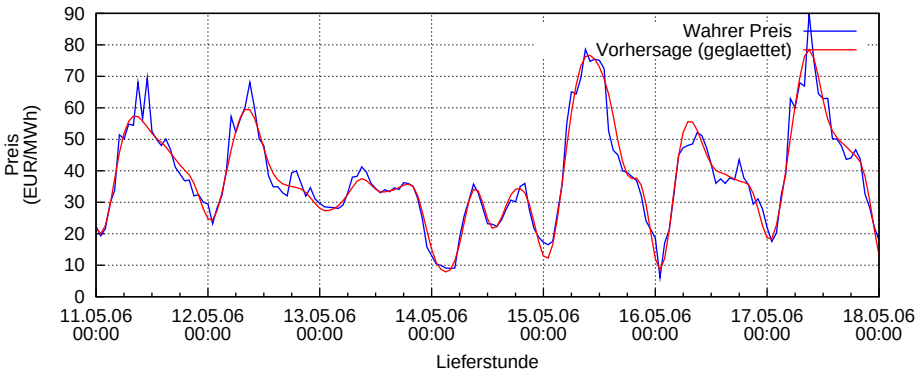


Abbildung 6.20: RBF-Prognose nach Wavelet-Glättung.

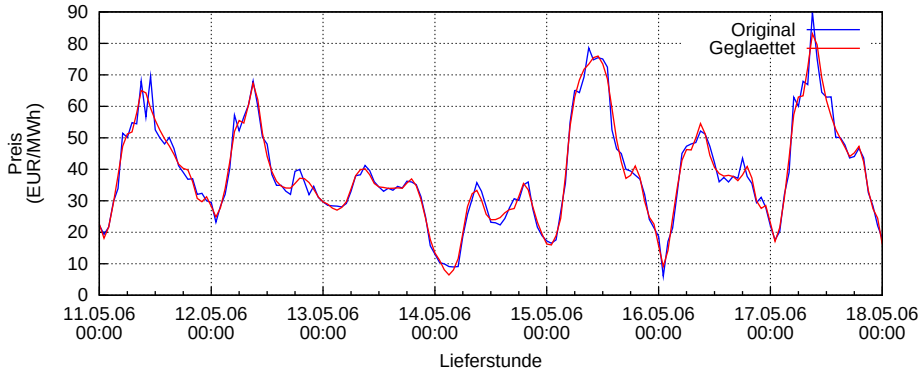


Abbildung 6.21: Geglättete Trainingsdaten.

6.4.2 Versuchsreihe: Lernen auf Wavelet-geglätteter Zeitreihe

In einer anderen Betrachtungsweise kann man auch die Zeitreihe, die als Trainingsmenge dient, vor dem eigentlichen Lernalgorithmus glätten. Dabei besteht die Hoffnung, dass das gelernte Modell durch weniger hochfrequente Schwingungen in der Trainingsmenge besser generalisiert.

Zur Glättung der Trainingsreihe wurde das Daubechies-10 Wavelet gewählt und die Thresholding-Parameter nach Augenmaß bestimmt. Die so entstehende Zeitreihe (siehe Abbildung 6.21) ist ein guter Kompromiss aus Glattheit und Detailtreue. Weitere Experimente sollten ggf. andere Wavelets und Thresholding-Parameter untersuchen, um aussagekräftigere Ergebnisse zu erzielen, dies würde jedoch den Rahmen dieser Arbeit sprengen.

Auf dieser derart geglätteten Zeitreihe wird nun eine SVM mit RBF-Kern trainiert. Dabei wird dieselbe Fensterung wie auch in der Versuchsreihe „full-features“ aus Kapitel 6.3.7 benutzt, die Zeitreihe wird also mit allen verfügbaren, zusätzlichen Features angereichert. Ebenso werden die Parameter der SVM und der Kernfunktion optimiert.

Testreihe	Test 1		Test 2		2009		big	
Fehlermaß	RMSE	PTA	RMSE	PTA	RMSE	PTA	RMSE	PTA
RBF	7.712	0.600	27.587	0.612	8.857	0.583	18.750	0.604

Tabelle 6.8: Versuchsreihe „smoothed learning“: Fehlermaße haben sich im Vergleich zum Lernen auf den Originaldaten deutlich verschlechtert.

Betrachtet man die Prognosekurve, so leuchtet unmittelbar ein, dass das auf einer geglätteten Zeitreihe gelernte Modell nicht mit den Modellen mithalten kann, die auf unveränderter Zeitreihe trainiert wurden. Die stark alternierende, gezackte Kurve erklärt die schlechte Prediction Trend Accuracy, die stellenweise an reines Raten erinnert. Es

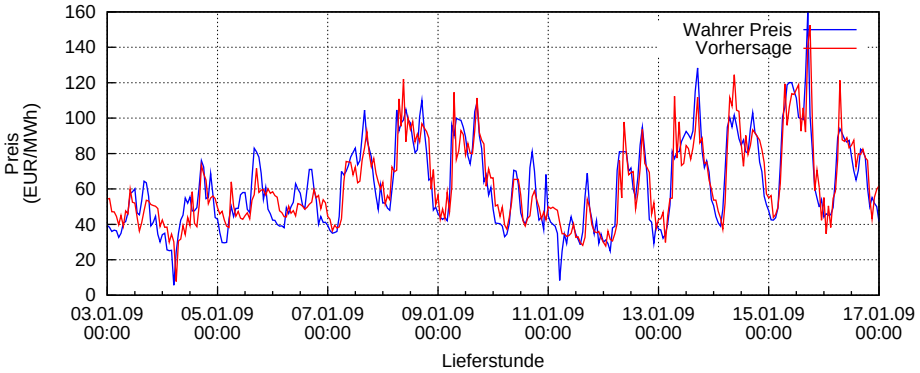


Abbildung 6.22: Prognosekurve einer SVM mit RBF-Kern, die auf Wavelet-geglätteter Zeitreihe trainiert wurde. Auffallend ist das gezackte Erscheinungsbild der Approximationsreihe, welches die schlechte PTA erklärt.

ist anzunehmen, dass hier auch kein nachträgliches Wavelet-Glätten mehr hilft, weshalb diese Versuchsreihe als gescheitert angesehen werden kann.

6.5 Vorhersagen in der Praxis

Die vorherigen Experimente leiden alle unter einer, im Falle der Strompreise des Day-Ahead Auktionshandels, unrealistischen Annahme, nämlich dass alle vorherigen Stundenpreise, bis auf den nächsten, vorherzusagenden bekannt sind. Leider trifft diese Voraussetzung auf den Day-Ahead Auktionshandel im Stromgeschäft nicht zu. Möchte man den Strompreis für 10 Uhr prognostizieren, kann man nicht auf eine Historie bis zum selben Tag um 9 Uhr zurückgreifen. Stattdessen endet die Historie in der letzten Stunde (23 bis 00 Uhr) des Vortages.

Befinden wir uns in der Position des Händlers, so sind die Strompreise des aktuellen Tages und der Vergangenheit bekannt, Gebote für die Preise des nächsten Tages müssen bis jeweils 12 Uhr des aktuellen Tages abgegeben werden [EPEX 2009]. Man ist also an einer 24 Stunden in die Zukunft reichenden Prognose basierend auf den Strompreisen des aktuellen Tages (und ggf. der Vergangenheit) interessiert. Die folgenden Experimente beschäftigen sich mit genau dieser Art von Prognosen auf mehrere unterschiedliche Weisen.

6.5.1 Fensterung mit „window-steps“

Um nach Möglichkeit denselben Versuchsaufbau benutzen zu können, wie er zuvor verwendet wurde, wird zunächst das Verfahren der Fensterung (zur Erinnerung siehe Abbildung 4.1 in Kapitel 4.1) modifiziert. Um verschiedene Horizonte mit einem einzigen Modell vorhersagen zu können, sollte hierbei nicht primär das Fenster über die Zeitreihe gleiten, sondern sich in erster Linie der Horizont in jedem Schritt vergrößern. Erst wenn der Horizont die volle Fensterbreite überschreitet, rückt das Fenster nach. Als zusätzliches Feature wird dabei der momentan verwendete Horizont h zu den Beispielen hinzugefügt. Der Informationsgehalt der Fenster ist derselbe wie bei der normalen Fensterung, wie man sich leicht klar macht. Das Verfahren wird im Folgenden als „window-steps“ Variante der Fensterung bezeichnet. Abbildung 6.23 veranschaulicht dieses Vorgehen am Beispiel einer Fensterbreite von 6. Im Experiment wurde eine Fensterbreite von 24 Stunden gewählt.

Die Hoffnung, mit dieser Vorverarbeitung realistische Prognosen treffen zu können, wurde allerdings in den Experimenten zerschlagen. Da sich ein Großteil der Features (der Fenster-Inhalt) über mehrere Beispiele hinweg nicht ändern, da das eigentliche Fenster ja an derselben Stelle bleibt, erzeugt das erstellte Modell auf den Testdaten für diese Beispiele auch stets denselben Wert. Das Ergebnis ist eine Treppenfunktion wie sie in Abbildung 6.24 dargestellt ist.

6.5.2 Multiple Modelle

Nachdem der Versuch, Prognosen über unterschiedliche Horizonte mit nur einem Modell zu treffen, wenig Aussicht auf Praxisrelevanz zeigt, wird nun die klassische Variante

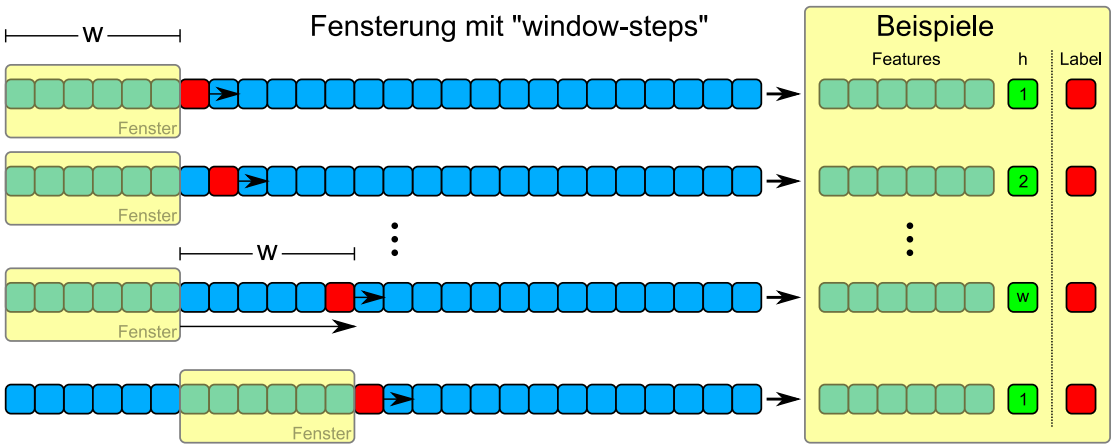


Abbildung 6.23: Modifizierte Form der Fensterung, welche Prognosen mit unterschiedlichen Horizonten in einem Modell zulässt.

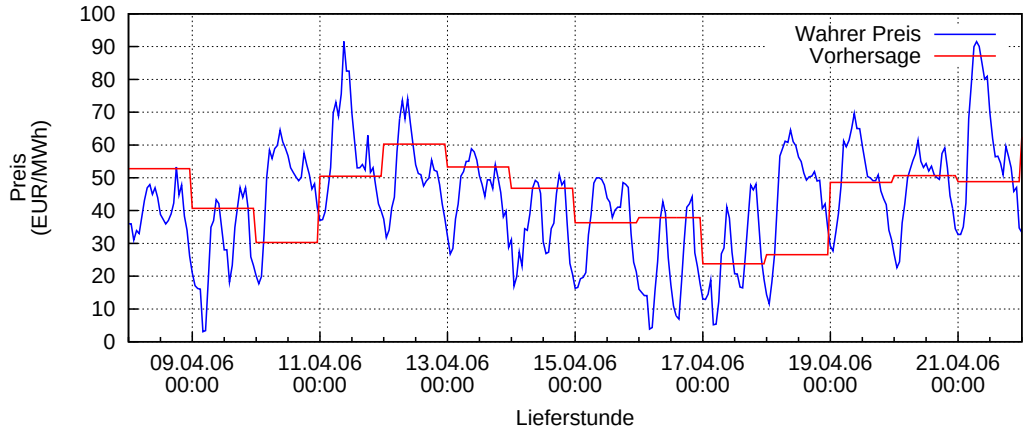


Abbildung 6.24: Versuchsreihe „window-steps“: Prognose mit modifizierter Fensterung.

angewandt. Das bedeutet in diesem Fall, dass für jeden möglichen Horizont (1 bis 24) ein eigenes Modell berechnet wird und somit unabhängige Prognosen für jede der 24 zukünftigen Stunden getroffen werden können.

Auch hier ist die Fensterung der entscheidende Ansatzpunkt. Dabei muss nicht nur für jedes der zu berechnenden Modelle der Horizont auf den gewünschten Wert gesetzt, sondern auch die Schrittweite angepasst werden, in unserem Fall auf Schrittweite 24. Jedes Modell ist somit spezialisiert auf eine bestimmte Stunde des Tages.

Die berechneten Modelle eignen sich nun ausschließlich zur Prognose der entsprechenden zukünftigen Stunde. Wendet man ein solches Modell auf gleichermaßen vorverarbeitete Testdaten an, so erhält man Preisprognosen im Abstand von je 24 Stunden. Fügt man schließlich die Prognosen aller Modelle wie bei einem Reißverschluss zusammen, erhält man eine einzelne Prognosekurve, bei der die Preise jedes Tages ausschließlich

aus den Preisen des vorigen Tages prognostiziert wurden und nicht jede Stunde aus den unmittelbar vorangegangenen Stunden.

Bei einer Prognose von mehr als einer Stunde in die Zukunft erwartet man üblicherweise eine schlechtere Performanz als bei solchen, die jeweils nur die nächste Stunde vorhersagen sollen. In der Tat spiegelt sich diese Erwartung in den Ergebnissen wider, allerdings in einem durchaus erträglichen Ausmaß.

Bei der Durchführung dieses Experiments wurde die Versuchsreihe „full-features“ als Grundlage genommen. Bei der Fensterung wurden also alle verfügbaren, zusätzlichen Features integriert, sowie die Fenster zentriert und normiert. Aufgrund des Aufwands, den die Berechnung 24 unterschiedlicher Modelle, ihre Anwendung auf 4 Testreihen und die oben beschriebene Zusammenfügung der Daten in diesem Fall macht, beschränkt sich diese Versuchsreihe auf die Verwendung des RBF-Kerns, welcher nach den vorangegangenen Experimenten als geeignetste Kernfunktion angesehen werden kann. Zur besseren Unterscheidung wird das in Kapitel 6.3.7 vorgestellte RBF-Modell „Einfach-Modell“ und das in diesem Kapitel beschriebene „Multi-Modell“ genannt.

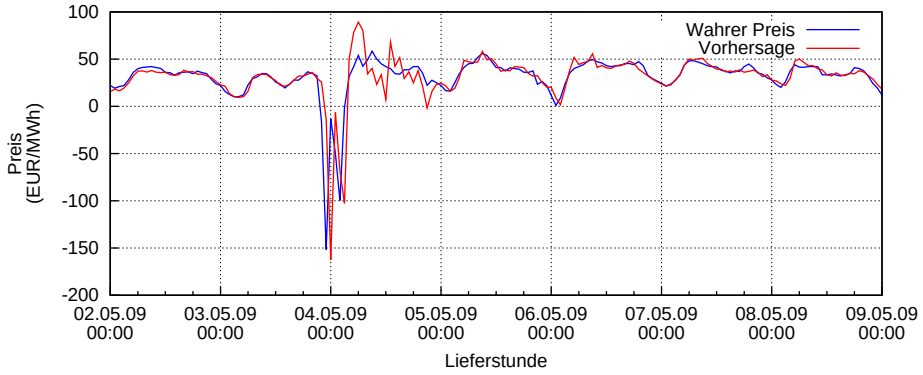
Testreihe	Test 1		Test 2		2009		big	
Fehlermaß	RMSE	PTA	RMSE	PTA	RMSE	PTA	RMSE	PTA
RBF Einfach	4.704	0.802	21.826	0.770	6.602	0.744	19.837	0.772
RBF Multi	5.043	0.823	24.650	0.809	11.190	0.681	20.789	0.749

Tabelle 6.9: Versuchsreihe „multi-model“: SVM unter Verwendung der Fensterung mit unterschiedlichen Horizonten.

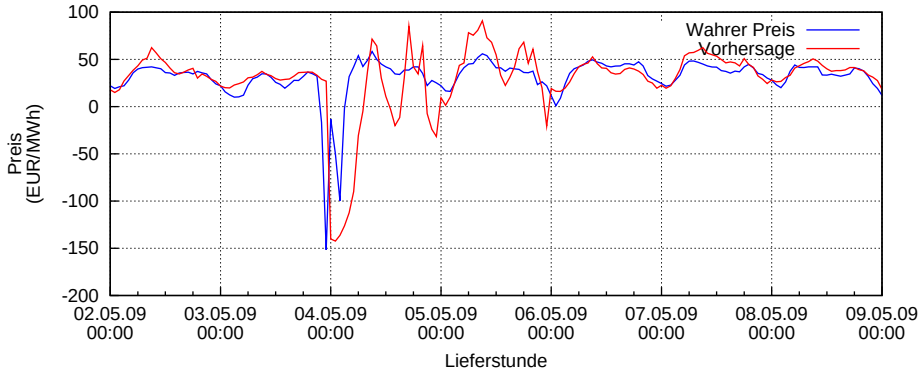
Während auf Testreihe 1 und Testreihe „big“ die Unterschiede in den Fehlermaßen beinahe vernachlässigt werden können, treten sie auf Testreihe 2 und „2009“ deutlicher hervor.

Zunächst erkennt man in Abbildung 6.25b, dass beide Modelle nicht in der Lage sind, irreguläre Schwankungen zu prognostizieren. Das Einfach-Modell stellt sich jedoch innerhalb einer Stunde auf den starken Abwärts-Trend im Bild ein und zieht mit seiner Prognose nach. Hier begegnet uns das Phämomen der naiven Prognose erneut. Da keine bessere Schätzung zur Verfügung steht, wird hier der zuletzt bekannte Wert prognostiziert. Das angeführte Beispiel ist zwar nicht konstruiert, jedoch höchst selten, so dass es nicht weiter verwundert, dass Prognoseversuche an dieser Stelle fehlschlagen.

Das Multi-Modell kann sich erst zu Beginn des 4. Mai auf das tiefe Niveau einstellen, lässt seine Prognose aber schnell wieder nach oben wandern. Die starken Schwankungen in den restlichen Stunden des 4. Mai sind durch die extreme Skalierung des Vortags begründet, wie in Kapitel 6.3.5 bereits erläutert, und sorgen für einen mitskalierten Approximationsfehler. Auch am nachfolgenden Tag sind noch starke Schwankungen zu sehen, was sich ebenfalls durch den verzerrten Mittelwert des Preisniveaus des Vortages begründen lässt.



(a) Versuchsreihe „Multi-Modell“, RBF Kern „einfach“



(b) Versuchsreihe „Multi-Modell“, RBF Kern „multi“

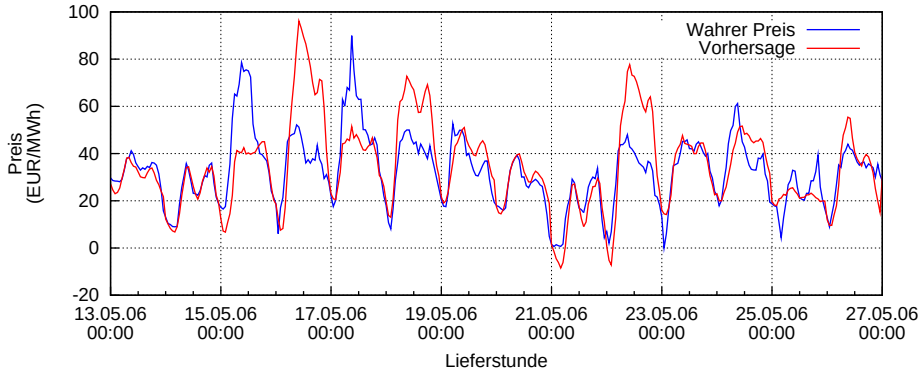
Abbildung 6.25: Versuchsreihe „multi-model“: Kritischer Ausschnitt aus Testreihe „2009“. Vergleich der herkömmlichen Fensterung mit Horizont 1 und Schrittweite 1. (oben) und Anwendung multipler Modelle für unterschiedliche Horizonte (unten). Erläuterungen siehe Text.

6.5.3 Die inkrementelle Vorhersage

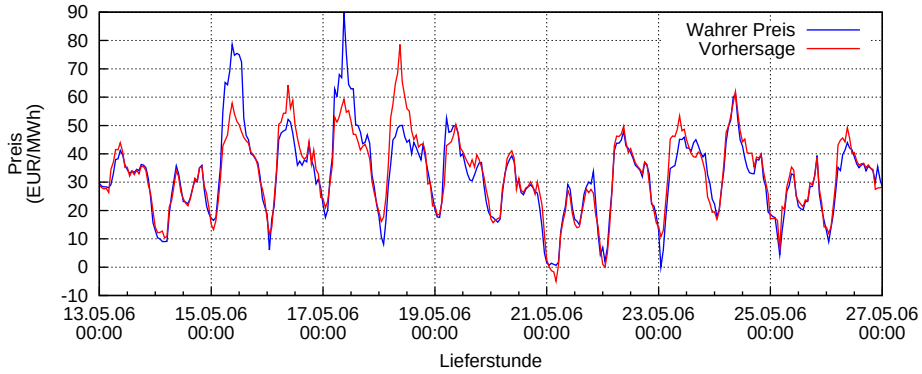
Stehen keine multiplen Modelle zur Berechnung von Prognosen zur Verfügung und möchte man dennoch weiter als bis zur nächsten Stunde in die Zukunft vorhersagen, so bleibt noch die Möglichkeit, die Vorhersagen inkrementell durchzuführen. Dabei wird zunächst der Wert für die nächste Stunde prognostiziert, dieser im Anschluss als „bekannter“ Wert deklariert und die Prognose unter Einbeziehung des soeben prognostizierten Wertes erneut durchgeführt. Dieses Verfahren ist auch in der klassischen Zeitreihenanalyse üblich [Schlittgen 2001]. Theoretisch lassen sich so beliebig weit in die Zukunft reichende Prognosen durchführen, doch sorgen die unweigerlich auftretenden Prognosefehler früher oder später für völlig realitätsfremde Kurvenverläufe.

Als Händler im Day-Ahead Auktionshandel ist man in der Regel nicht an Prognosen interessiert, die weiter als 24 Stunden in die Zukunft reichen[6]. Es liegt also nahe, das

[6] Ausnahme: Wochenenden und Feiertage. Hier gelten Sonderregelungen bzgl. der Gebotsfristen.



(a) Inkrementelle Vorhersage mit RBF-Kern



(b) Vergleich: Prognose mit multiplen Modellen

Abbildung 6.26: (a) Inkrementelle Prognose unter Verwendung eines Modells eines RBF SVM-Kerns, welches auf gefensterten Daten mit Horizont 1 trainiert wurde. (b) Vergleich mit derselben Prognose multipler Modelle mit unterschiedlichen Horizonten (Kapitel 6.5.2).

beste Modell für Vorhersagen mit Horizont 1 zu verwenden, um inkrementell die nächsten 24 Stunden zu prognostizieren.

Ein solcher Operator wurde im Rahmen dieser Versuchsreihe implementiert. Der Operator **IncrementalSeriesForecasting** ist dazu in der Lage, aus einer eingehenden Zeitreihe[7] und einem Prognosemodell eine inkrementelle Vorhersage zu erstellen und soll im Folgenden angewendet werden.

Abbildung 6.26a zeigt, dass die inkrementelle Art der Prognose vom Prinzip her funktioniert. Interessanterweise scheint sie die Form der einzelnen Tageskurven relativ gut vorherzusagen, mit dem durchschnittlichen Preisniveau liegt sie jedoch häufig falsch, was die RMSE-Werte in Tabelle 6.10 bestätigen. Der Vergleich mit dem im letzten Kapitel vorgestellen Verfahren zeigt allerdings auch, dass die Verwendung multipler Modelle unter demselben Problem leidet, wenn auch weniger stark ausgeprägt. Trotzdem ist in allen

[7] allgemeiner: eines beliebigen ExampleSets

Testreihe	Test 1		Test 2		2009		big	
Fehlermaß	RMSE	PTA	RMSE	PTA	RMSE	PTA	RMSE	PTA
RBF Inkr.	11.109	0.638	35.435	0.637	12.663	0.618	25.905	0.631

Tabelle 6.10: Versuchsreihe „incremental forecast“: SVM mit RBF-Kern und inkrementeller Prognose.

Fällen die Verwendung mehrerer Modelle vorzuziehen, nicht zuletzt auch aus Gründen der Parallelisierbarkeit.

6.6 Zeitreihenanalyse mit linearen Modellen

Nach den Versuchsreihen mit RapidMiner, die vornehmlich im Bereich des maschinellen Lernens angesiedelt waren, sollen zusätzlich vergleichend die Methoden der klassischen Zeitreihenanalyse, im speziellen die autoregressiven moving Average Modelle (ARIMA), angewendet werden.

Kapitel 6.6.1 gibt dabei zunächst einen Überblick über gängige und speziell in dieser Arbeit verwendete Software. Die folgenden Versuchsreihen in den Kapiteln 6.6.2 und 6.6.3 erstellen jeweils ARIMA-Modelle und untersuchen ihre Prognoseeigenschaften im Vergleich zu den SVM-Modellen aus den vorangegangenen Experimenten, wobei auch hier Methoden aus der Wavelet-Theorie zum Einsatz kommen.

6.6.1 Verwendete Software

Für die Methoden der klassischen Zeitreihenanalyse steht eine Reihe an Software-Tools zur Verfügung. Neben den für speziell diese Zwecke entwickelten Programmen TRAMO bzw. SEATS „*Time series Regression with ARIMA noise, Missing values and Outliers*“ *und „Signal Extraction in ARIMA Time Series*“ ([Sánchez Gálvez et al.] und X-12-ARIMA [Bureau] besitzt auch namhafte Mathematik- und Statistik-Software wie MATLAB und R Module zur Berechnung autoregressiver Modelle. Da diese Arbeit sich im Hinblick auf die klassische Zeitreihenanalyse ausschließlich mit eben diesen Modellen beschäftigt, scheint es zunächst logisch zu sein, eins der frei erhältlichen Programme TRAMO / SEATS oder X12-ARIMA für Analysezwecke zu benutzen. Leider lassen sich mit ihnen nur Zeitreihen mit maximal 600 Werten analysieren, weswegen sie für die zu untersuchenden Energiemarktdaten ungeeignet sind. Eine Brücke zwischen dem Alleskönner MATLAB und dem spartanischen, aber kostenlosen R schlägt die Software gretl „*Gnu Regression, Econometrics and Time-series Library*“ [Cottrell und Lucchetti]. Kostenlos und leicht zu bedienen erfüllt sie alle Ansprüche an ein statistisches Analysewerkzeug mit benutzerfreundlicher, grafischer Oberfläche und vielfältigen Exportmöglichkeiten aller berechneten Ergebnisse, Tabellen und Plots. gretl bietet außerdem eine Schnittstelle zu R sowie zu TRAMO / SEATS und auch X-12-ARIMA an. Im weiteren Verlauf dieser Arbeit sind alle Ergebnisse aus der statistischen Zeitreihenanalyse, sofern

nicht anders angegeben, mit gretl erstellt. Zur Erstellung der Grafiken wurde gnuplot und MATLAB verwendet.

6.6.2 Versuchsreihe: Klassische ARIMA-Prognose

Zunächst wird eine gewöhnliche Anpassung eines ARIMA-Modells exemplarisch durchgeführt. Sie soll zeigen, welchen Fehler man bei der Anwendung der klassischen Zeitreihenprognose mit autoregressiven Modellen erwarten kann. Vor der eigentlichen Durchführung sind jedoch ein paar Worte zur Versuchsplanung angebracht.

In [Chatfield 1982] wird angeführt, dass Ausreißer in Zeitreihen das Autokorrelogramm stark beeinflussen können. Da sich diese Arbeit nicht mit der Beseitigung von Ausreißern befasst, wird in dieser Versuchsreihe ausschließlich die Testreihe 1 (Abbildung 6.3a) betrachtet. Diese zeichnet sich durch einen unauffälligen Verlauf ohne wesentliche Ausreißer aus. Da außerdem die nachfolgende Versuchsreihe eine Wavelet-Zerlegung durchführt, kürzen wir die Reihe auf die nächstkleinere Zweierpotenz. So bleiben Fehlermaße vergleichbar.

Anders als bei Experimenten mit RapidMiner erlaubt gretl nicht, einmal angepasste Modelle ohne Weiteres auf ungesehene Daten anzuwenden, daher kann in diesem Fall das Modell nicht auf der Grundlage der Trainingsdaten erstellt werden, wenn es auf der Testreihe 1 getestet werden soll. Es wird daher ein ARIMA-Modell an die Testreihe 1 angepasst und auch auf dieser der Approximationsfehler gemessen.

Die Anpassung eines autoregressiven Modells verlangt von der Zeitreihe, dass diese stationär ist. Eine einfache, aber effektive Transformation, die zu diesem Zweck häufig angewendet wird, ist das Differenzieren. Dabei werden die Differenzen eines Wertes der Zeitreihe zu seinem Vorgänger gespeichert. Ein linearer Trend kann auf diese Weise eliminiert werden. Ist der Trend nichtlinear, können weitere Differenzbildungen Abhilfe schaffen. Im Falle der Strompreise des Day-Ahead Auktionshandels ist eine Differenzbildung unumgänglich, da mit der Forderung der Stationarität einher geht, dass die Zeitreihe mittelwertbereinigt ist.

Die Statistik-Software gretl bietet die Funktion `arima` an, die es nicht nur erlaubt, die Parameter eines autoregressiven Modells zu schätzen, sondern auch die Differenzbildung in den Prozess zu integrieren (Autoregressive, *Integrated*, Moving Average). Daher ist in dieser Situation kein zusätzlicher Vorverarbeitungsschritt nötig. Die Integration der Differenzbildung in die Erstellung des Modells hat außerdem den positiven Nebeneffekt, dass man ein Modell für die unveränderte Zeitreihe erhält und nicht für die Reihe der Differenzen, wie es der Fall wäre, wenn man beide Schritte getrennt voneinander ausführen würde.

Zur Bestimmung der AR und MA Ordnungen wird in der Literatur die Betrachtung der ACF und PACF empfohlen [Schlittgen 2001]. Abbildung 6.27 stellt diese dar. Das rasche Abfallen der PACF-Koeffizienten ab dem dritten Lag ist hierbei zunächst ein Indiz für einen AR[3]-Prozess.

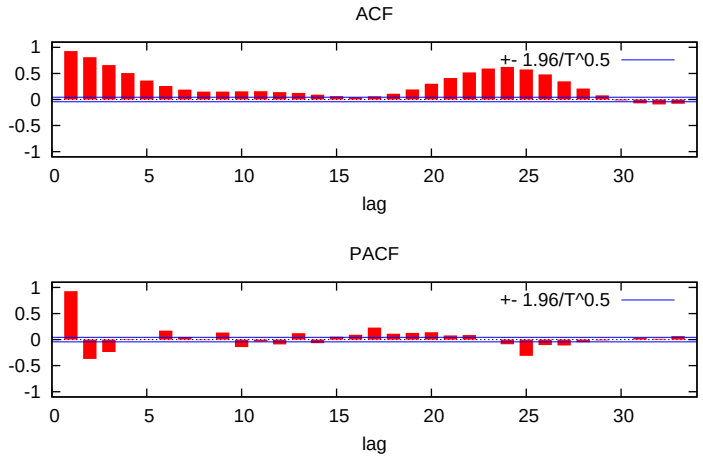


Abbildung 6.27: ACF und PACF der unveränderten Preiskurve

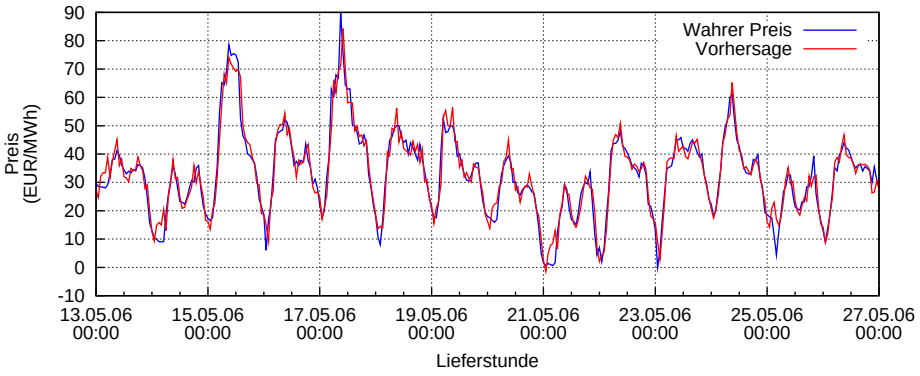


Abbildung 6.28: Regression mit angepasstem ARIMA(3 1 1 ; 1 0 1) Modell.

gretl erlaubt bei der Spezifikation eines ARIMA-Modells die Angabe von saisonalen Ordnungen für AR- und MA-Terme. Eine Saison entspricht dabei im Falle von stündlichen Werten einem vollständigen Tag, also 24 Stunden. Die saisonalen Ordnungen der AR- und MA-Terme sorgen dafür, dass das Modell nicht nur die Werte der vorangegangenen Stunden, sondern auch der vorangegangen Tage mit einbezieht. Aus diesem Grund wird eine saisonale AR-Ordnung von 1 gewählt. Die Hinzunahme einer saisonalen, sowie nichtsaisonalen MA-Ordnung von ebenfalls 1, sowie die Integration der Differenzbildung (nichtsaisonal) führte schließlich zur Spezifikation eines ARIMA(3 1 1 ; 1 0 1) Modells. Die ersten drei Ziffern stehen dabei für die nichtsaisonale AR-Ordnung, Differenzbildung und MA-Ordnung. Das hintere Tripel für die entsprechenden saisonalen Ordnungen. Dieses Modell erzeugte auf der Testreihe 1 einen RMSE von **4.334** und eine Predicted Trend Accuracy (PTA) von **0.817**, etwas besser als die SVM mit RBF-Kern in Versuchsreihe „date-features“ aus Kapitel 6.3.6.

Dass diese Wahl der AR- und MA-Parameter durchaus akzeptabel ist, lässt sich an den Residuen, also den Differenzen von wahrem Wert und Approximation erkennen (siehe

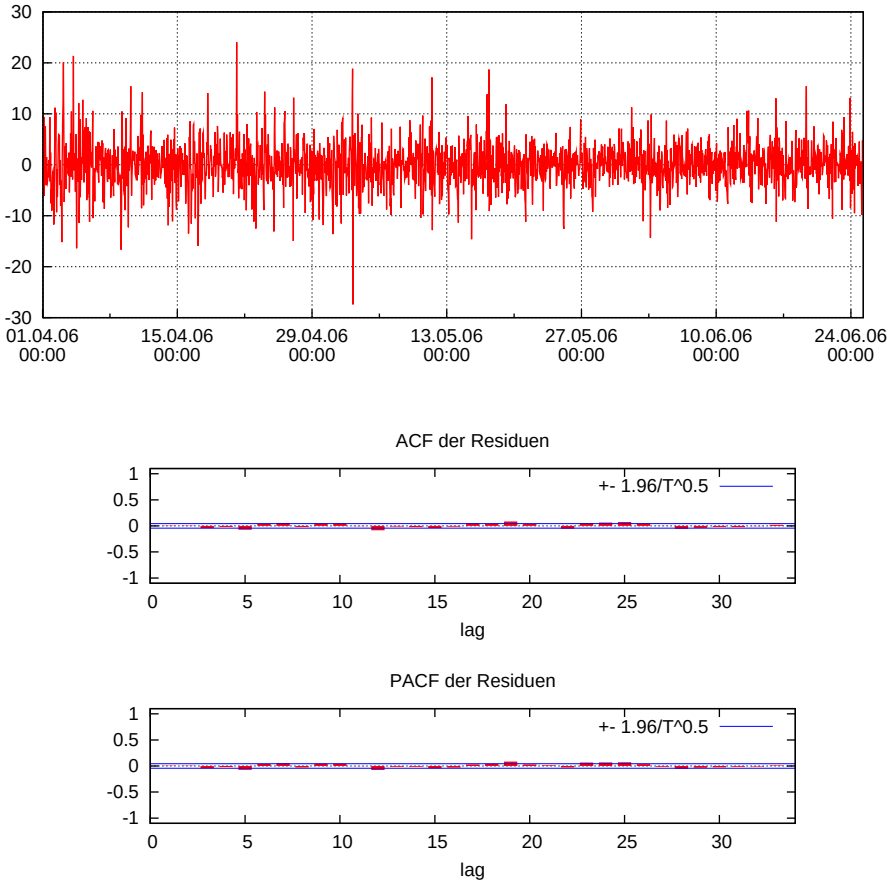


Abbildung 6.29: Unkorrelierte Residuen und ihre Korrelogramme nach Anpassung des ARIMA Modells.

Abbildung 6.29). Diese sind nach der ARIMA-Anpassung unkorreliert, das Modell macht also keine systematischen Fehler mehr, sondern nur noch Fehler in Form von weißem Rauschen.

Betrachtet man die Approximationskurve genauer, so fällt auch hier ein oftmals gezackter Verlauf auf, ebenso bei der Kurve der wahren Werte. In einer wissenschaftlichen Arbeit wurde sich dieses Problems angenommen und auch dort Methoden aus der Wavelet-Theorie eingesetzt. Die folgende Versuchsreihe untersucht die Effektivität dieses Vorgehens auf den in dieser Arbeit untersuchten Daten.

6.6.3 Versuchsreihe: Wavelets und ARIMA

Auch im Bereich der klassischen Zeitreihenanalyse nach Box und Jenkins finden sich Arbeiten, die durch geeignete Vorverarbeitung und Transformation der Daten Prognoseergebnisse von ARIMA-Modellen zu verbessern versuchen. In [Conejo et al. 2005] beispielsweise wird mit Hilfe der Multiskalenanalyse aus der Wavelet-Theorie eine Zeitrei-

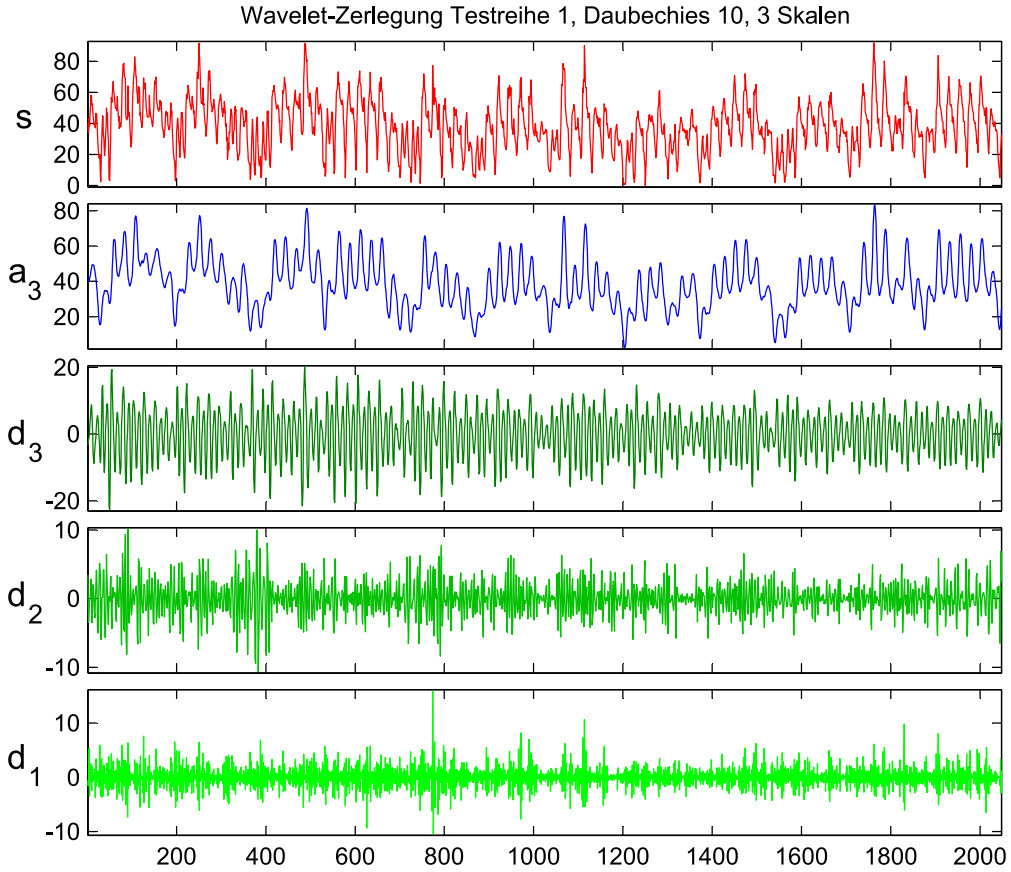


Abbildung 6.30: Zerlegte Testreihe 1 mit Hilfe der Multiskalenanalyse unter Verwendung des Daubechies-10 Wavelets. Hierbei ist s die ursprüngliche Zeitreihe und es gilt: $s = a_3 + d_3 + d_2 + d_1$

he zunächst in ihre Detail- und Approximationskomponenten zerlegt (siehe Kapitel 4.2.8). Den so entstehenden Reihen werden jeweils eigene ARIMA-Modelle angepasst und schließlich separate Prognosen für jede Komponente durchgeführt. Die Rekonstruktionseigenschaft der Wavelet-Zerlegung erlaubt es nun, durch simple Addition der Einzelprognosen eine Gesamtprognose für die ursprüngliche Zeitreihe zu erstellen. Da die soeben beschriebene Arbeit ebenfalls Energiebörsendaten behandelt, soll sie in dieser Versuchsreihe als Vorbild dienen.

Zunächst wird also die Zeitreihe der Preisdaten mit Hilfe der Wavelet Multiskalenanalyse in 3 Skalen zerlegt. Die Verwendung des Daubechies-10 Wavelets sorgt dabei für einen stark geglätteten Verlauf der Kurve der Approximationskoeffizienten a_3, wie Abbildung 6.30 demonstriert.

An die entstehenden vier Reihen werden jeweils eigene ARIMA-Modelle angepasst. Dies wird wie zuvor in Kapitel 6.6.2 durchgeführt. Es ergeben sich folgende Modelle mit ihren mittleren quadratischen Fehlern:

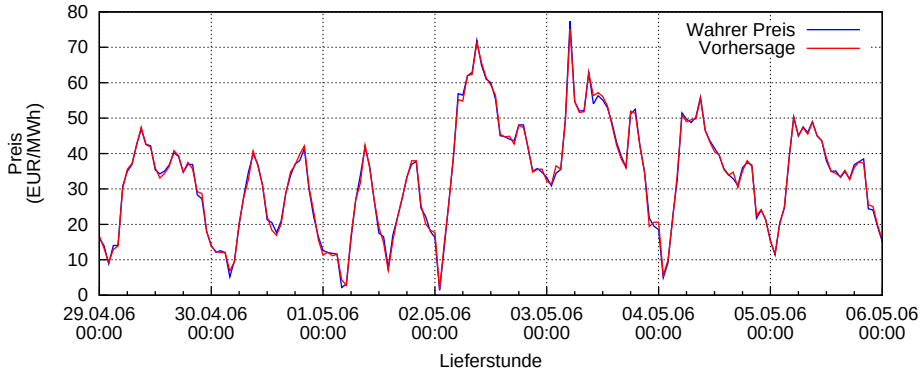


Abbildung 6.31: Rekonstruierte Gesamt-Approximation aus den ARIMA-angepassten Einzel-Approximationen.

	nichtsaisonal			saisonal			
	AR[p]	Diff.	MA[q]	AR[P]	Diff.	MA[Q]	RMSE
a_3	3	1	1	1	0	1	0.32343
d_3	10	0	1	0	0	1	0.49012
d_2	10	0	1	1	0	0	0.44925
d_1	10	0	1	1	0	0	0.44925

Tabelle 6.11: Die ARIMA-Modellparameter der aus der Wavelet-Zerlegung hervorgegangenen Reihen.

Die mittleren quadratischen Fehler in Tabelle 6.11 lassen vermuten, dass sich ARIMA-Modelle an die einzelnen Wavelet Koeffizientenreihen extrem gut anpassen lassen, was bei der ursprünglichen Reihe nicht in diesem Maße der Fall war. Die simple Form der Reihen a_3 bis d_1 trägt dazu bei, dass bei der Regression fast keine Fehler gemacht werden. Addiert man nun alle geschätzten Reihen zusammen, so ergibt sich die Approximationskurve aus Abbildung 6.31 mit einem RMSE von **0.841** und einer PTA von **0.947**.

Dieses überwältigende Approximationsergebnis ist allerdings mit Vorsicht zu genießen. Zwar ist es durchaus zu begrüßen, dass Reihen, die bei einer Wavelet Multiskalenanalyse entstehen, hervorragend durch ARIMA-Prozesse angepasst werden können, jedoch sagt dies im Endeffekt noch nichts über die eigentliche Prognosequalität der Modelle aus, und schon gar nicht, ob die Summe der Einzelprognosen eine gute Gesamtprognose ergibt.

Erfreulicherweise erlaubt gretl leicht, mit bereits berechneten Modellen eine Prognose zu erstellen, die beliebig weit in die Zukunft reicht[8]. Da die vier Einzelmodelle für die Reihen a_3 bis d_1 vorlagen, konnte so schnell eine solche Gesamtprognose erstellt werden 6.32. Ein Fehlermaß für diese Prognose wurde dabei zwar nicht berechnet, jedoch sieht man deutlich, dass die anfängliche Begeisterung über die gute Regressionseigenschaft der Wavelet-Zerlegung kein Garant für eine hochwertige Prognose sein muss.

[8] in unserem Fall genügen jedoch 24 Stunden

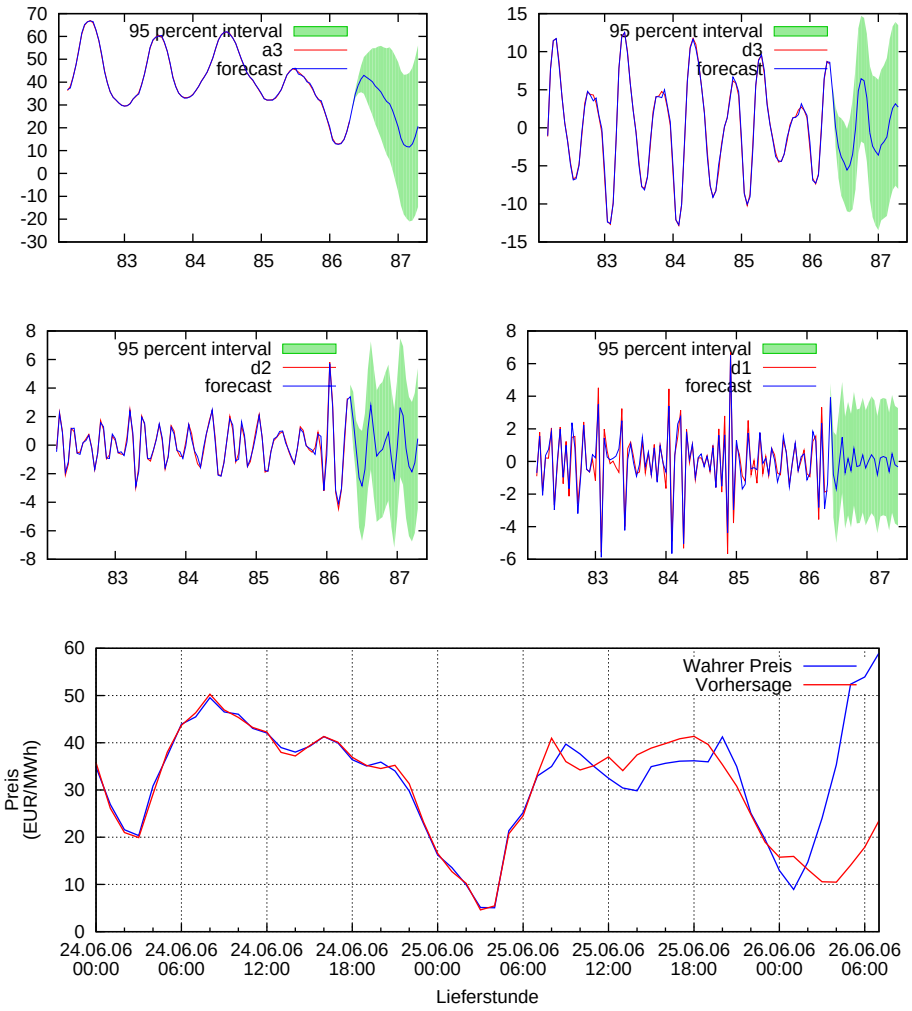


Abbildung 6.32: Die vier Einzelprognosen der ARIMA-Modelle der Reihen a_3, d_3, d_2 und d_1 (4 Abbildungen oben) ergeben addiert die Gesamtprognose (unten, ab Zeitpunkt 25.06. 08:00 Uhr)

7 Zusammenfassung und Ausblick

Die durchgeführten Experimente haben gezeigt, dass alle vorgestellten Verfahren unter gewissen Voraussetzungen dazu in der Lage sind, akzeptable Prognosen auf Energiemarktdaten, in diesem Fall den Strompreisen des Day-Ahead Auktionshandels an der EEX, zu erstellen. Nicht ohne Grund ist der RBF-Kern der Stützvektormethode als äußerst robuster und performanter Kern bekannt. Ähnlich robust, ja geradezu ignorant verhält sich der lineare Kern. Unbeeindruckt von zusätzlichen, praxisrelevanten Features ist seine Performance zwar gut, aber nicht überragend. Wavelet-Kernfunktionen liefern teilweise bessere Ergebnisse, haben aber offenbar Schwierigkeiten mit den zusätzlichen Features. Ob dies an einer Fehlimplementierung oder an anderen numerischen Eigenarten liegt, konnte bis zur Fertigstellung dieser Arbeit nicht geklärt werden. Es sei hinzugefügt, dass die Berechnung der Kernmatrix bzw. der eigentlichen Kernfunktion bei Wavelet-Kernen derart langsam war, dass sogar die Trainingsmenge reduziert werden musste, um noch in doppelter Zeit, die die Berechnungen mit dem RBF-Kern brauchten, ein optimales Modell zu bestimmen.

Die zusätzlichen Features, von denen in Fachkreisen einstimmig und einleuchtenderweise behauptet wird, dass sie einen starken Einfluss auf den Energiebedarf haben, brachten bei den Prognosen nur wenig bis gar keine besseren Ergebnisse, im Falle des „hour_of_day“ Features sorgten sie sogar für drastische Verschlechterungen. Das kann zum einen bedeuten, dass noch weitere Features nötig sind, um die Prognosen deutlich zu verbessern. Kennzahlen aus der Windenergie spielen beispielsweise eine große Rolle bei der Ermittlung des Energiebedarfs und somit bei der Preisgestaltung. Zum anderen ist jedoch auch nicht völlig auszuschließen, dass die Prognosen aus Sicht eines Händlers in der Tat schon sehr gut sind. Die Firma TradeSpark war jedenfalls nach einem flüchtigen Blick auf die in dieser Arbeit erzielten Ergebnisse positiv überrascht.

Verwundert hat, dass in den Experimenten der Stützvektormethode der SVM-Parameter C offenbar keine Rolle spielte. Dies kam allerdings aufgrund der eh schon sehr langen Laufzeit der Experimente durchaus gelegen. Um diese in einem erträglichen Rahmen zu halten, konnte außerdem in den Trainingsphasen nicht die gesamte Trainingsmenge genutzt werden. Stattdessen wurden in jedem Schleifendurchlauf zufällige Samples gezogen. Die Ergebnisse dieser Arbeit können durch umfangreichere und vor allem länger laufende Experimente sicherlich noch gefestigt werden, die grundsätzlichen Aussagen der Ergebnisse sollten sich jedoch dabei nicht ändern.

Absolut praktikabel und leicht durchführbar ist anschließende Wavelet-Glättung nach der Erstellung einer Prognosekurve. Sie liefert in der Tat noch deutliche bessere Approximationen zu Tage. Fraglich ist nur, ob es Wavelets sein müssen, oder ob nicht

ein einfaches gleitendes Mittel dieselbe Wirkung hat. Diese Frage sollte in relativ leicht durchzuführenden weiteren Experimenten schnell geklärt werden können.

Beim Lernen auf einer zuvor geglätteten Trainingsreihe sollte noch getestet werden, wie sich die Prognose verhält, wenn die zu testende Reihe vor dem Testlauf[1] ebenfalls auf dieselbe Weise wie die Trainingsmenge Wavelet-geglättet wird. Zur Berechnung der Performanzmaße müsste dann allerdings wieder die Testreihe im Ursprungszustand benutzt werden.

Dass bei Prognose, die weiter als eine Stunde in die Zukunft reichen, die Verwendung von mehreren spezialisierten Modellen eher zu empfehlen ist als sich inkrementell mit Prognosen in die Zukunft zu bewegen, ist nicht verwunderlich. Verwunderlich ist eher, dass die inkrementelle Prognose trotzdem noch durchaus brauchbare Ergebnisse liefert. In der Praxis wird man aber wohl dennoch zur Methode mit der besseren Performanz greifen. Die Idee der modifizierten Fensterung, um unterschiedliche Horizonte mit ein und demselben Modell vorherzusagen, ist in ihrer vorgestellten Form in der Praxis leider nicht zu gebrauchen. Wenn man es schafft, die Information in den Fenstern mit der Information des zu prognostizierenden Horizontes zu vermischen, wäre dies einen erneuten Versuch wert.

Die Methoden aus der klassischen Zeitreihenanalyse weder schlechter noch wesentlich besser als Prognosen mit der SVM zu sein. Allein die mühsame Bestimmung der AR- und MA-Ordnungen machen sie zu einer unliebsamen Aufgabe, bei der vor allem Erfahrung eine große Rolle spielt. Wären diese Methoden in RapidMiner implementiert, so könnte man hier auch als Unerfahrener bequem mit einer Parameter-Optimierung arbeiten. Die manuelle Suche nach den Ordnungen ist jedoch momentan ein höchst interaktiver Prozess, bei dem man sich mehr Automatismen wünscht.

Die Idee, die Zeitreihe zunächst zu zerlegen und mit angepassten Modellen für jede Komponente Prognosen zu treffen sollte auch auf der Stützvektormethode getestet werden. Es besteht Grund zur Annahme, dass die Muster der Komponenten der Wavelet-Zerlegung von einer SVM gut gelernt werden können. Aus Mangel an Platz kann dieses interessante Experiment jedoch nicht mehr in dieser Arbeit vorgestellt werden.

Weitere große Punkte, die in dieser Arbeit leider keinen Platz zur Erwähnung fanden, sind die Themengebiete „multivariate Zeitreihen“ und Ausreißerentdeckung. Eine multivariate Fensterung würde es beispielsweise ermöglichen, nicht nur die Temperatur zum Zeitpunkt des Labels zu betrachten, sondern auch die vorangegangene Temperatur. Aber auch andere exogene Informationen in Form einer zusätzlichen Zeitreihe könnten so den Lernprozess unterstützen.

Auch könnte man sich denken, dass die Beschränkung auf ein starres Zeitfenster bei der Fensterung nicht immer die optimale Wahl ist. Analog zur Wavelet-Theorie könnte man mehr Informationen über die Historie der Zeitreihe, jedoch in geringerer Auflösung, integrieren, beispielsweise durch Hinzunahme von Tagesmittelwerten für weiter zurückliegende Tage (dynamische Vergangenheit). Die negativen Erfahrungen allerdings, die

[1] genauer: vor der Anwendung des Modells

in dieser Arbeit mit zu großen Fenstergrößen gemacht wurden, könnten allerdings auch diesem Vorhaben einen Strich durch die Rechnung machen.

Abschließend bleibt zu erwähnen, dass die rekursive Parameteroptimierung in RapidMiner eine große Hilfe bei der Bestimmung der besten Prozessparameter war. Zuvor war bei der Suche nach den besten Parameter stets viel Handarbeit angesagt. Mit Hilfe dieses neuen Operators konnten die optimalen Parameter nahezu punktgenau ohne weiteres menschliches Zutun gefunden werden. Die evolutionäre Parameteroptimierung sollte eigentlich ähnlich hilfreich sein, hat in der Praxis jedoch häufig enttäuscht.

8 Danksagung

An dieser Stelle möchte ich mich nochmals herzlich bei der TradeSpark GmbH & Co. KG und der Meteogroup Deutschland GmbH für die Bereitstellung der Daten bedanken.

Literaturverzeichnis

[Anderson 1976] Anderson, O.D. (1976). *Time series analysis and forecasting: the Box-Jenkins approach*. Butterworths London.

[Box und Jenkins 1970] Box, George E. P. und G. M. Jenkins (1970). *Time series analysis : forecasting and control*. Holden-Day series in time series analysis. Holden-Day, San Francisco [u.a.].

[Box et al. 2008] Box, George E. P., G. M. Jenkins und G. C. Reinsel (2008). *Time series analysis : forecasting and control*. Wiley series in probability and statistics. Wiley, Hoboken, NJ, 4. Aufl.

[Box und Pierce 1970] Box, G.E.P. und D. Pierce (1970). *Distribution of residual autocorrelations in autoregressive-integrated moving average time series models*. Journal of the American Statistical Association, 65(332):1509–1526.

[Bracewell und Kahn 1966] Bracewell, R. und P. Kahn (1966). *The Fourier transform and its applications*. American Journal of Physics, 34:712.

[Brigham 1988] Brigham, Elbert Oran (1988). *The fast Fourier transform and its applications*. Prentice-Hall signal processing series. Prentice-Hall Internat., Englewood Cliffs, NJ [u.a.].

[Brigham und Yuen 1978] Brigham, EO und C. Yuen (1978). *The fast Fourier transform*. IEEE Transactions on Systems, Man and Cybernetics, 8(2):146–146.

[Brockwell und Davis 1991] Brockwell, Peter J. und R. A. Davis (1991). *Time series : theory and methods*. Springer series in statistics. Springer, New York [u.a.], 2. ed. Aufl.

[Brockwell und Davis 2002] Brockwell, P.J. und R. Davis (2002). *Introduction to time series and forecasting*. Springer, 2. Aufl.

[Bureau] Bureau, U.S. Census. *X-12-ARIMA*. `http://www.census.gov/srd/www/x12a/`.

[Burg 1967] Burg, J.P. (1967). *Maximum likelihood spectral analysis*. In: *Proc. 37th Meeting Society of Exploration Geophysicists*.

[Burges 1998] Burges, C.J.C. (1998). *A tutorial on support vector machines for pattern recognition*. Data mining and knowledge discovery, 2(2):121–167.

[Cao und Tay 2001] Cao, L. und F. Tay (2001). *Financial forecasting using support vector machines*. Neural Computing & Applications, 10(2):184–192.

[Chatfield 1979] Chatfield, C. (1979). *Inverse autocorrelations*. Journal of the Royal Statistical Society. Series A (General), S. 363–377.

[Chatfield 1982] Chatfield, Christopher (1982). *Analyse von Zeitreihen : eine Einführung*. Teubner, Leipzig, 1. Aufl. Aufl.

[Chen et al. 2004] Chen, B.J., M. Chang und C. Lin (2004). *Load forecasting using support vector machines: a study on EUNITE competition 2001*. IEEE Transactions on Power Systems, 19(4):1821.

[Conejo et al. 2005] Conejo, AJ, M. Plazas, R. Espinola und A. Molina (2005). *Day-ahead electricity price forecasting using the wavelet transform and ARIMA models*. IEEE Transactions on Power Systems, 20(2):1035–1042.

[Contreras et al. 2003] Contreras, J., R. Espinola, F. Nogales und A. Conejo (2003). *ARIMA models to predict next-day electricity prices*. IEEE transactions on power systems, 18(3):1014–1020.

[Cortes und Vapnik 1995] Cortes, C. und V. Vapnik (1995). *Support-vector networks*. Machine learning, 20(3):273–297.

[Cottrell und Lucchetti] Cottrell, Allin und R. Lucchetti. *Gnu Regression, Econometrics and Time-series Library*. `http://gretl.sourceforge.net`.

[Cristianini und Shawe-Taylor 2006] Cristianini, Nello und J. Shawe-Taylor (2006). *An introduction to support vector machines : and other kernel-based learning methods*. Cambridge Univ. Press, Cambridge [u.a.], 10. print. Aufl.

[Dan et al. 2005] Dan, W., G. Xuemai und G. Qing (2005). *A new scheme of automatic modulation classification using wavelet and WSVM*. In: *Mobile Technology, Applications and Systems, 2005 2nd International Conference on*, S. 5.

[Daubechies 1994] Daubechies, Ingrid (1994). *Ten lectures on wavelets*. CBMS NSF regional conference series in applied mathematics ; 61. Society for Industrial and Applied Mathematics, Philadelphia, Pa., 3. print. Aufl.

[De Gooijer und Hyndman 2006] De Gooijer, J.G. und R. Hyndman (2006). *25 years of time series forecasting*. International Journal of Forecasting, 22(3):443–473.

[Dickey und Fuller 1979] Dickey, D.A. und W. Fuller (1979). *Distribution of the estimators for autoregressive time series with a unit root*. Journal of the American statistical association, 74(366):427–431.

[Droste et al. 1998] Droste, S., T. Jansen und I. Wegener (1998). *On the analysis of the (1+ 1) evolutionary algorithm*.

[EEX] EEX. `http://www.eex.com`.

[EPEX 2009] EPEX (2009). *EPEX Spot Operational Rules*. `http://www.eex.com/de/document/53032/EPEX_Product_Power_ENG.PDF`.

[Gabor 1946] Gabor, D. (1946). *Theory of communication: J. Inst.* Electr. Eng,

93:429–457.

[Gao et al. 2007] Gao, C., E. Bompard, R. Napoli und H. Cheng (2007). *Price forecast in the competitive electricity market by support vector machine*. Physica A: Statistical Mechanics and its Applications, 382(1):98–113.

[Goupillaud et al. 1984] Goupillaud, P., A. Grossmann und J. Morlet (1984). *Cycle-octave and related transforms in seismic signal analysis*. Geoexploration, 23(1):85–102.

[Griffin und Lim 1984] Griffin, D. und J. Lim (1984). *Signal estimation from modified short-time Fourier transform*. IEEE Transactions on Acoustics, Speech and Signal Processing, 32(2):236–243.

[Guo et al. 2008] Guo, Xuesong, L. Sun, G. Li und S. Wang (2008). *A hybrid wavelet analysis and support vector machines in forecasting development of manufacturing*. Expert Systems with Applications, 35(1-2):415 – 422.

[Hannan und Rissanen 1982] Hannan, EJ und J. Rissanen (1982). *Recursive estimation of mixed autoregressive-moving average order*. Biometrika, 69(1):81–94.

[Hastie et al. 2009] Hastie, Trevor, R. Tibshirani und J. H. Friedman (2009). *The elements of statistical learning : data mining, inference, and prediction*. Springer series in statistics. Springer, New York, NY, 2. ed. Aufl.

[He und Starzyk 2006] He, H. und J. Starzyk (2006). *A self-organizing learning array system for power quality classification based on wavelet transform*. IEEE Transactions on Power Delivery, 21(1):286–295.

[Hsu et al. 2003] Hsu, C.W., C. Chang, C. Lin et al. (2003). *A practical guide to support vector classification*.

[Hubbard 1997] Hubbard, Barbara Burke (1997). *Wavelets : die Mathematik der kleinen Wellen*. Birkhäuser, Basel [u.a.].

[Jensen und La Cour-Harbo 2001] Jensen, Arne und A. La Cour-Harbo (2001). *Ripples in mathematics : the discrete wavelet transform*. Springer, Berlin [u.a.].

[Joachims 2000] Joachims, T. (2000). *Estimating the Generalization Performance of a SVM Efficiently*. In: *Proceedings of the International Conference on Machine Learning*, San Francisco. Morgan Kaufman.

[Kaiser 1994] Kaiser, Gerald (1994). *A friendly guide to wavelets*. Birkhäuser, Boston [u.a.].

[Kay und Marple Jr 1981] Kay, SM und S. Marple Jr (1981). *Spectrum analysis - a modern perspective*. Proceedings of the IEEE, 69(11):1380–1419.

[Kearns und Ron 1999] Kearns, M. und D. Ron (1999). *Algorithmic stability and sanity-check bounds for leave-one-out cross-validation*. Neural Computation, 11(6):1427–1453.

[Keerthi und Shevade 2003] Keerthi, S. S. und S. K. Shevade (2003). *SMO algorithm for least-squares SVM formulations*. Neural Comput., 15(2):487–507.

[Kendall und Stuart 1966] Kendall, M.G. und A. Stuart (1966). *The advanced theory of statistics : in 3 volumes*. Griffin, London [u.a.].

[Lehrmann 2002] Lehrmann, Edgar (2002). *Informationsmanagement im Handel Strom*. Doktorarbeit, Universität Essen Fachbereich Wirtschaftswissenschaften.

[Lin et al. 2005] Lin, C.C., S. Chen, T. Truong und Y. Chang (2005). *Audio classification and categorization based on wavelets and support vector machine*. IEEE Transactions on Speech and Audio Processing, 13(5 Part 1):644–651.

[yong Liu und Fan 2006] Liu, Fan yong und M. Fan (2006). *A Hybrid Support Vector Machines and Discrete Wavelet Transform Model in Futures Price Forecasting*. S. 485–490.

[Ljung und Box 1978] Ljung, GM und G. Box (1978). *On a measure of lack of fit in time series models*. Biometrika, 65(2):297–303.

[Malathi und Marimuthu] Malathi, V. und N. Marimuthu. *Wavelet Transform and Support Vector Machine Approach for Fault Location in Power Transmission Line*. International Journal of Computer Systems Science and Engineering, 4:4.

[Mallat 1989] Mallat, S.G. (1989). *A theory for multiresolution signal decomposition: the wavelet representation*. IEEE transactions on pattern analysis and machine intelligence, 11(7):674–693.

[Mallat 2009] Mallat, Stéphane G. (2009). *A wavelet tour of signal processing : the sparse way*. Elsevier, Amsterdam [u.a.], 3. ed. Aufl.

[Mei-Ying und Xiao-Dong 2005] Mei-Ying, Ye und W. Xiao-Dong (2005). *Phase Space Prediction of Chaotic Time Series with Nu-Support Vector Machine Regression*. Communications in Theoretical Physics, 43(1):102–106.

[Mercer 1909] Mercer, J. (1909). *Functions of positive and negative type, and their connection with the theory of integral equations*. Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character, S. 415–446.

[Meyer 1993] Meyer, Yves (1993). *Wavelets : algorithms & applications*. Society for Industrial and Applied Mathematics, Philadelphia.

[Mierswa 2006] Mierswa, I. (2006). *Evolutionary learning with kernels: A generic solution for large margin problems*. In: *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, S. 1553–1560. ACM New York, NY, USA.

[Mierswa und Morik 2005] Mierswa, I. und K. Morik (2005). *Automatic feature extraction for classifying audio data*. Machine learning, 58(2):127–149.

[Mierswa et al. 2006] Mierswa, Ingo, M. Wurst, R. Klinkenberg, M. Scholz und T. Euler (2006). *YALE: Rapid Prototyping for Complex Data Mining Tasks*.

In: Ungar, Lyle, M. Craven, D. Gunopulos und T. Eliassi-Rad, Hrsg.: *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, S. 935–940, New York, NY, USA. ACM.

[Mukherjee et al. 1997] Mukherjee, S., E. Osuna und F. Girosi (1997). *Nonlinear prediction of chaotic time series using support vectormachines*. In: *Neural Networks for Signal Processing [1997] VII. Proceedings of the 1997 IEEE Workshop*, S. 511–520.

[Muller et al. 1997] Muller, K.R., A. Smola, G. Ratsch, B. Scholkopf, J. Kohlmorgen und V. Vapnik (1997). *Predicting time series with support vector machines*. Lecture notes in computer science, 1327:999–1004.

[Paparoditis und Streitberg 1992] Paparoditis, E. und B. Streitberg (1992). *Order identification statistics in stationary autoregressive moving-average models: vector autocorrelations and the bootstrap*. Journal of Time Series Analysis, 13(5):415–434.

[Pardo et al. 2002] Pardo, A., V. Meneu und E. Valor (2002). *Temperature and seasonality influences on Spanish electricity load*. Energy Economics, 24(1):55–70.

[Patnaik 2005] Patnaik, LM (2005). *Daubechies 4 wavelet with a support vector machine as an efficient method for classification of brain images*. Journal of Electronic Imaging, 14:013018.

[Percival und Walden 2000] Percival, Donald B. und A. T. Walden (2000). *Wavelet methods for time series analysis*. Cambridge series in statistical and probabilistic mathematics ; 4. Cambridge Univ. Press, Cambridge [u.a.].

[Pittner und Kamarthi 1999] Pittner, S. und S. Kamarthi (1999). *Feature extraction from wavelet coefficients for patternrecognition tasks*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 21(1):83–88.

[Platt 1998] Platt, J.C. (1998). *Fast training of support vector machines using sequential minimal optimization*. MIT Press, Advances in kernel methods, support vector learning, 6:185–208.

[Priestley 1981] Priestley, Maurice B. (1981). *Spectral analysis and time series*. Academic Pr.

[Rioul und Duhamel 1992] Rioul, O. und P. Duhamel (1992). *Fast algorithms for discrete and continuous wavelet transforms*. IEEE Transactions on Information Theory, 38(2 Part 2):569–586.

[Rüping und Morik 2003] Rüping, S. und K. Morik (2003). *Support vector machines and learning about time*. In: *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03)*, Bd. 4.

[RTE 2009] RTE (2009). *Consommation Francaise d'Electricite Caracteristiques et Methode de Prevision*. `http://www.rte-france.com/htm/fr/vie/telecharge/prev_conso_elec.pdf`.

[Sakia 1992] Sakia, RM (1992). *The Box-Cox transformation technique: a review*. The

Statistician, S. 169–178.

[Sarle 1997] Sarle, Warren S. (1997). *Neural Network FAQ. Periodic posting to the Usenet newsgroup comp.ai.neural-nets*. `ftp://ftp.sas.com/pub/neural/FAQ.html`.

[Schlittgen und Streitberg 1997] Schlittgen, R. und B. Streitberg (1997). *Zeitreihenanalyse*. Oldenbourg, 7., unwesentlich veränd. Aufl.

[Schlittgen 2001] Schlittgen, Rainer (2001). *Angewandte Zeitreihenanalyse*. Lehr- und Handbücher der Statistik. Oldenbourg, München [u.a.].

[Schölkopf et al. 1998] Schölkopf, B., C. Burges und A. Smola (1998). *Advances in kernel methods: support vector learning*. The MIT press.

[Schölkopf und Smola 2002] Schölkopf, Bernhard und A. J. Smola (2002). *Learning with kernels : support vector machines, regularization, optimization, and beyond*. Adaptive computation and machine learning. MIT Press, Cambridge, Mass. [u.a.].

[Shao und Cherkassky 1999] Shao, X. und V. Cherkassky (1999). *Multi-resolution support vector machine*. In: *Neural Networks, 1999. IJCNN'99. International Joint Conference on*, Bd. 2.

[Smola 1996] Smola, A.J. (1996). *Regression estimation with support vector learning machines*. Master's thesis, Technische Universität München.

[Smola und Schölkopf 2004] Smola, A.J. und B. Schölkopf (2004). *A tutorial on support vector regression*. Statistics and Computing, 14(3):199–222.

[Smola et al. 1998] Smola, Alex J., B. Schölkopf und K.-R. Müller (1998). *The connection between regularization operators and support vector kernels*. Neural Networks, 11(4):637 – 649.

[Sánchez Gálvez et al.] Sánchez Gálvez, F., R. López Pavón, D. Pérez Cañete, M. A. und N. Morales Villalba. *TRAMO / SEATS*. `http://www.bde.es/servicio/software/econome.htm`.

[Strang 1993] Strang, G. (1993). *Wavelet transforms versus Fourier transforms*. American Mathematical Society, 28(2):288–305.

[Strang und Nguyen 1997] Strang, Gilbert und T. Nguyen (1997). *Wavelets and filter banks*. Wellesley-Cambridge Press, Wellesley, MA, Rev. Aufl.

[Strichartz 1993] Strichartz, R.S. (1993). *How to make wavelets*. American Mathematical Monthly, 100(6):539–556.

[Szu et al. 1992] Szu, H.H., B. Telfer und S. Kadambe (1992). *Neural network adaptive wavelets for signal representation and classification*. Optical Engineering, 31:1907.

[Takens et al. 1981] Takens, F. et al. (1981). *Detecting strange attractors in turbulence*. Dynamical systems and turbulence, 898(1):365–381.

[Taylor 1985] Taylor, J.M.G. (1985). *Power transformations to symmetry*. Biometrika, 72(1):145–152.

[Tolambiya und Kalra 2007] Tolambiya, A. und P. Kalra (2007). *WSVM with Morlet Wavelet Kernel for Image Compression*. In: *IEEE International Conference on System of Systems Engineering, 2007. SoSE'07*, S. 1–5.

[Tsay und Tiao 1984] Tsay, R.S. und G. Tiao (1984). *Consistent estimates of autoregressive parameters and extended sample autocorrelation function for stationary and nonstationary ARMA models*. Journal of the American Statistical Association, S. 84–96.

[Vapnik 2000] Vapnik, Vladimir N. (2000). *The nature of statistical learning theory*. Statistics for engineering and information science. Springer, New York, NY, 2. Aufl.

[Verivox] Verivox. *Wie der Strompreis zustande kommt - Die Energiebörse EEX*. `http://www.verivox.de/power/article.aspx?i=25508`.

[Vishwanath 1994] Vishwanath, M. (1994). *The recursive pyramid algorithm for the discrete wavelet transform*. IEEE Transactions on Signal Processing, 42(3):673–676.

[Wahba und Wold 1975] Wahba, G. und S. Wold (1975). *A completely automatic french curve: Fitting spline functions by cross validation*. Communications in Statistics-Simulation and Computation, 4(1):1–17.

[Witten und Frank 2005] Witten, I.H. und E. Frank (2005). *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, 2. Aufl.

[Xi und Lee 2003] Xi, D. und S. Lee (2003). *Face detection and facial component extraction by wavelet decomposition and support vector machines*. Lecture notes in computer science, S. 199–317.

[Xie et al. 2006] Xie, W., L. Yu, S. Xu und S. Wang (2006). *A new method for crude oil price forecasting based on support vector machines*. Lecture Notes in Computer Science, 3994:444.

[Zhang et al. 2004] Zhang, L., W. Zhou und L. Jiao (2004). *Wavelet support vector machine*. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 34(1):34–39.

[Zhang et al. 2005] Zhang, X., D. Gao, X. Zhang und S. Ren (2005). *Robust wavelet support vector machine for regression estimation*. International Journal of Information Technology, 11(9):35–45.

Erklärung

Hiermit erkläre ich, Oliver Heering, die vorliegende Diplomarbeit mit dem Titel *Preis- und Trendvorhersagen auf Energiemarktdaten* selbständig verfasst und keine anderen als die hier angegebenen Hilfsmittel verwendet, sowie Zitate kenntlich gemacht zu haben.

Dortmund, 4. September 2009