Online Learning für die Analyse von Finanzmarktdaten

Alireza Moradpour and Mirjam Minor

Goethe University, Business Information Systems, Robert-Mayer-Str. 10,
60629 Frankfurt, Germany
{moradpour,minor}@cs.uni-frankfurt.de

Abstract. Die Analyse von Finanzmarktdaten ist ein wertvolles Hilfsmittel beim Treffen von Kaufentscheidungen. Der vorliegende Beitrag stellt einen neuen Ansatz des Online Learnings vor, um die Richtung von Kurszeitreihen zu prognostizieren. Im Unterschied zu klassischen Methoden des Machine Learning benötigt dieser Ansatz keine Trainings- und Testphasen, sondern wird immer auf der Grundlage der aktuellesten Daten kontinuierlich aktualisiert. Wir beschreiben den Ansatz zur Richtungsprognose in Form eines stochastischen Prozesses, der aus lernfähigen Modulen besteht, die endliche Automaten und Zufallsgeneratoren miteinander kombinieren, um Datenausschnitte zu interpretieren und Prognosen zu treffen. In der Lernphase werden sowohl die endlichen Automaten als auch die Zufallsgeneratoren in wechselseitiger Interaktion angepasst. Die experimentelle Evaluierung mit historischen Finanzmarktdaten zeigt, dass unser Verfahren schnell Ergebnisse liefert und besser als der Münzwurf ist.

Keywords: Online learning, Zeitreihenanalyse, High frequency trading

1 Einleitung

Die Analyse von Zeitreihen hat in vielen Bereichen der Wissenschaft (Medizin, Technik, Wirtschaft etc.) eine große Bedeutung. Beispielsweise ist es in den Wirtschaftswissenschaften wichtig zu wissen, wie sich der Aktienmarkt entwickelt. Dazu werden Methoden zur Beschreibung und Analyse von Finanzmarktdaten eingesetzt [6]. Gegenstand vieler Analysen sind Kurszeitreihen. Unter einem Kurs versteht man den auf einem Finanzmarkt gebildeten Preis eines Wertpapiers [6, S. 1].

Die Richtung des Marktes spielt zum Beispiel im High Frequency Trading (HFT) eine grosse Rolle, um automatisch eine Kaufentscheidung zu treffen [1, S. 17]. Häufig analysieren die HFT-Algorithmen Sequenzen von Tick-Daten, die die Preise einzelner Handels-Transaktionen wiedergeben. Tick-Daten sind nicht frei zugänglich, jedoch werden an vielen Handelsplätzen ausgewählte Datenpunkte veröffentlicht, zum Beispiel die Eröffnungspreise zu Handelsbeginn, die Endpreise bei Handelsschluss, die Tageshöchstpreise (high-price) oder Tagestiefpreise (low-price). Zeitreihen über diese frei verfügbaren Daten weisen ähnliche

Eigenschaften wie Tick-Daten auf. Traditionelle Analysemethoden setzen klassische Machine-Learningverfahren [5] ein. Diese Verfahren erzeugen aus einer Menge von Trainingsdaten ein probabilistisches Modell, das genutzt wird, um Vorhersagen über neue Daten zu treffen. Im Unterschied dazu gibt es die neue Forschungsrichtung der Online Learning-Methoden [2], die eine Sequenz von Trainingsbeispielen Schritt für Schritt analysieren und daraus lernen.

In diesem Paper wird ein Ansatz des Online-Learnings vorgestellt, der die Abwärts- und Aufwärtsbewegungen in Kurszeitreihen prognostizieren soll. Eine einfache Definition der Bewegung eines Kurses ist die Vorhersage über die Richtung (ab- oder aufwärts) der Entwicklung einer numerischen Variablen y in Abhängigkeit von einem Zeitindex $t_s \in \{0, 1, 2, \ldots\}$. Dies wird normalerweise durch eine Funktion $y = f(t)$ bestimmt. Man spricht von einer Aufwärtsbewegung, falls $f(t_s) - f(t_s + 1) < 0$ ist, und von einer Abwärtsbewegung, falls $f(t_s) - f(t_s + 1) > 0$.

Wir präsentieren einen neuen Algorithmus zur Zeitreihenanalyse, der vorhersagt, ob es im Zeitindex t_s eine Ab- oder Aufwärtsbewegung bezüglich des Zeitindex t_{s+1} gibt. Der Algorithmus arbeitet mit Zeitfenstern fester Länge n, $n \in \mathbb{N}^+ = \{1, 2, \ldots\}$. Für jeden Zeitschritt t_s nutzt der Algorithmus ein eigenes Modul M_i, $i = t_s \mod n$ zur Interpretation. Ein Modul M_i, $i = 0, 1, \ldots, n-1$ interpretiert in einer Folge nicht überlappender Zeitfenster je einen Wert $w_{t_s} \in \mathbb{R}^+$ an der i-ten Stelle $i = t_s \mod n$ des Zeitfensters. Damit werden also die Rohdaten zu Kursen in n unterschiedliche Zeitreihen zerlegt, deren Datenpunkte disjunkte Mengen bilden, die durch eine Auswahl der Rohdaten mit Schrittgröße n entstehen. Für jede dieser Zeitreihen ist ein eigenes Modul zuständig, das diesen Teil der Rohdaten analysiert. Jedes Modul implementiert einen online-Learner, das heißt es trifft Vorhersagen bezüglich einer Sequenz von Rohdaten, ohne vorher ein Modell trainieren zu müssen. Dabei nutzt ein Modul M_i Informationen von seinem Nachfolgermodul M_{i+1}, es profitiert also vom aktuellen Kenntnisstand von M_{i+1}, um seinen eigenen Fehler durch Anpassung zu reduzieren. Ein Modul besitzt eine Menge endlicher Automaten, die mit einer Menge von Zufallsgeneratoren interagieren, um eine Schätzung über die Bewegungsrichtung der Daten abzugeben. Jedes Modul trifft also eine kumulative Entscheidung, die aus den Entscheidungen der einzelnen Automaten aggregiert wird.

Das Paper ist wie folgt organisiert: In Kapitel 2 werden grundlegende Begriffe definiert. Kapitel 3 stellt den Ansatz zur Richtungsprognose vor. Kapitel 4 enthält die algorithmische Beschreibung des Ansatzes. Die experimentelle Evaluierung ist in Kapitel 5 beschrieben. Kapitel 6 beendet den Beitrag mit einer Schlußfolgerung.

2 Begriffliche Grundlagen

Der Online-Learning-Ansatz nutzt Zufallsgeneratoren. Die Module simulieren darin einen erweiterten stochastischen Prozess, der auf folgenden begrifflichen Grundlagen der Wahrscheinlichkeitstheorie basiert.

In der Wahrscheinlichkeitstheorie wird eine Bernoulli-Verteilung $B(p, 0, 1)$, $p \in [0, 1]$ [4] zur Beschreibung von zufälligen Ereignissen benutzt, bei denen es nur zwei mögliche Versuchsausgänge gibt. Einer der Versuchsausgänge wird mit Eins 1 und Wahrscheinlichkeit p bezeichnet und der komplementäre Versuchsausgang mit Null 0 und Wahrscheinlichkeit $1 - p$. Ein Beispiel hier dafür ist der Münzentwurf.

Weiterhin ist in der Wahrscheinlichkeitstheorie ein Bernoulli-Prozess $BP(n, p)$, $n \in \mathbb{N}^+$, $p \in [0, 1]$ oder eine Bernoulli-Kette [4] ein zeitlich diskreter stochastischer Prozess, der aus n Folgen von unabhängigen Versuchen mit Bernoulli-Verteilung besteht, die unter denselben Umständen und mit demselben Parameter p durchgeführt wurden. Ein Beispiel hierfür ist der n-malige Münzwurf.

In unserem Ansatz betrachten wir einen zweidimensionalen Raum $\mathbb{R}^2$, in dem eine Dimension der Zeitverlauf t ist und in der anderen Dimension die Werte wt_s liegen. Dabei ist von Interesse, wie sich der Wert w in Bezug auf den Zeitverlauf t entwickelt.

3 Richtungsprognose als stochastischer Prozess

Für unsere Richtungsprognose erweitern wir den Bernoulli-Prozess um zwei weiteren Komponenten w_{t_s} und t_s. w_{t_s} gibt den Wert eines Kurses zum Zeitpunkt t_s an.

Wir erhalten den stochastischen Prozess $S(n, B_{t_s}, w_{t_s}, t_s)$, $n \in \mathbb{N}^+$. n ist die Länge des betrachteten Zeitfensters. B_{t_s} ist eine Bernoulli-Verteilung zum Zeitpunkt t_s.

Wir definieren ein Modul M_i, $i = t_s \mod n$, das die Aufgabe hat, eine Richtung in einer Reihe von Werten zu erkennen. Dabei betrachtet Modul M_i den Wert w_{t_s} und Modul M_{i+1} den Nachfolgewert. Ob die Richtung, die M_i prognostiziert, richtig ist, kann erst im Nachhinein beurteilt werden, nämlich sobald der Eingabewert für M_{i+1} bekannt ist.

Das Modul hat folgende Komponenten:

1. In jeder Zeit t_s befindet sich Modul M_i entweder im Zustand "Null" (für Abwärtsrichtung) oder "Eins" (für Aufwärtsrichtung), und dieser Zustand wird in Modul M_i in der Variablen *zustand* gespeichert. Sei $\mathbb{M}$ das Universum aller Module.
   - Der Zustand von Modul M_i zum Zeitpunkt t_s wird durch die Funktion $Z : \mathbb{M} \to \{Null, Eins\}$ wie folgt bestimmt:
   $$Z(M_i) := \begin{cases} Null & falls\, Z(M_{i-1}) = Null\, und\, w_i \leq w_{i-1} \\ Null & falls\, Z(M_{i-1}) = Eins\, und\, w_i < w_{i-1} \\ Eins & falls\, Z(M_{i-1}) = Eins\, und\, w_i \geq w_{i-1} \\ Eins & falls\, Z(M_{i-1}) = Null\, und\, w_i > w_{i-1} \end{cases}$$
   Falls i gleich 0 ist, dann setzen wir $i - 1$ gleich $n - 1$.
   - Wir definieren die Bewegungsstärke für Modul M_i als:
   $Z_{mov}(M_i) := \frac{|w_i - w_{i+1}|}{\max(w_i, w_{i+1})}$

2. r verschiedene (0,1)-Maschinen oder endliche Automaten $\mathcal{A}$ mit $r \in \mathbb{N}^+$. Da zu dem Zeitpunkt, in dem M_i seine Schätzung abgibt, die zukünftigen Kurswerte noch nicht bekannt sind, nutzt der Automat Zufallszahlen als simulierte Eingabewerte.
3. $2r$ Zufallszahlsgeneratoren $\mathcal{G}$ mit $r \in \mathbb{N}^+$. Die Zufallszahlen vertreten die Bewegungsstärke Z_{mov}, da der tatsächliche Wert des Kurses für den nächsten Schritt und damit die Bewegungsstärke noch nicht bekannt ist.
4. Modul M_i gibt eine Richtungsprognose für die Werte des Kurses ab, indem es den Zustand von M_{i+1} mit Hilfe der Automaten $\mathcal{A}$ und der Zufallsgeneratoren $\mathcal{G}$ schätzt. Die Anzahl der durch Modul M_i korrekt geschätzten "Null" und "Eins" Zustände für Modul M_{i+1} über einen Zeitraum wird in den Variablen $kGNull - Zustand$ und $kGEins - Zustand$ gespeichert. Hier setzen wir $i + 1$ gleich 0, falls i gleich $n - 1$ ist.
5. Analog dazu wird die Anzahl der durch M_i falsch geschätzten "Null" und "Eins" Zustände für M_{i+1} in den Variablen $fGNull - Zustand$ und $fGEins - Zustand$ gespeichert. Auch hier setzen wir $i + 1$ gleich 0, falls i gleich $n - 1$ ist.
6. Wir setzen die Qualität von Modul M_i gleich:
   $Q(M_i) := \frac{kGEins-Zustand+kGNull-Zustand}{kGEins-Zustand+fGEins-Zustand+kGNull-Zustand+fGNull-Zustand}$

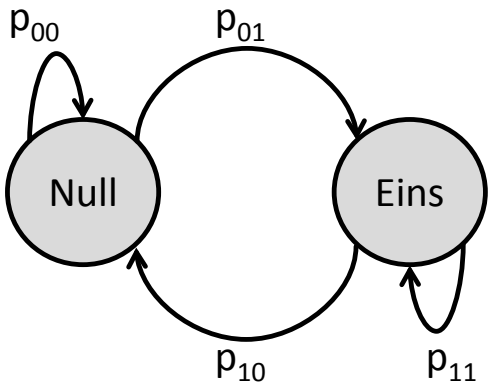

Fig. 1. Endlicher Automat mit Wahrscheinlichkeiten für die Zustandsübergangsfunktionen.

Im folgenden beschreiben wir die endlichen Automaten $\mathcal{A}$, die sich innerhalb eines Moduls M befinden. Die Komponenten eines Automaten A sind wie folgt (vgl. auch Abbildung 1):

1. Ein Automat A in Modul M befindet sich entweder im Zustand "Null" oder "Eins". Dieser Zustand wird in der Variablen *zustand* gespeichert. Es gilt immer $A.zustand = M.zustand$.
2. p_{00} bezeichnet die Wahrscheinlichkeit, dass Automat A, wenn er sich im Zustand "Null" befindet, weiterhin im Zustand "Null" bleibt.
   p_{01} bezeichnet die Wahrscheinlichkeit, dass Automat A, wenn er sich im Zustand "Null" befindet, in den Zustand "Eins" wechselt.
   p_{11} bezeichnet die Wahrscheinlichkeit, dass Automat A, wenn er sich im Zustand "Eins" befindet, weiterhin in diesem Zustand bleibt.
   p_{10} bezeichnet die Wahrscheinlichkeit, dass Automat A, wenn er sich im Zustand "Eins" befindet, in den Zustand "Null" wechselt.

   Es gelten immer die folgenden Beziehungen:
   $p_{01} = 1 - p_{00}$
   $p_{10} = 1 - p_{11}$
3. Wir definieren die Funktion $R(A)$ für Automat A wie folgt:
   $$R(A) := \begin{cases} p_{00} & falls\, A.zustand = Null \\ p_{11} & falls\, A.zustand = Eins \end{cases}$$
   Wir definieren den binären Wertebereich b für $R(A)$ und die Zufallszahl $z \in (0, 1]$ wie folgt:
   $$b(R(A), z) := \begin{cases} -1 & falls\, z \leq R(A) \\ 1 & falls\, z > R(A) \end{cases}$$
4. Wir definieren die Übergangsfunktionen für die Automaten mit Hilfe der Zufallszahl z wie folgt:
   Automat A befindet sich im Zustand "Null", dann definieren wir A^0:
   $$A^0(A, z) := \begin{cases} Null & falls\, z \leq p_{00} \\ Eins & sonst \end{cases}$$
   Automat A befindet sich im Zustand "Eins", dann definieren wir A^1:
   $$A^1(A, z) := \begin{cases} Eins & falls\, z \leq p_{11} \\ Null & sonst \end{cases}$$
5. Um ein Schätzungsschritt des Automaten A zu bewerten, werden seine Schätzergebnisse im Nachhinein betrachtet. Dazu definieren wir die boolesche Funktion $B(A, z)$ bezüglich Automat A und Zufallszahl $z \in (0, 1]$ wie folgt:
   $$B(A, z) := \begin{cases} Falsch & \begin{cases} falls\, A^0(A, z) \neq M_{i+1}.zustand \\ falls\, A^1(A, z) \neq M_{i+1}.zustand \end{cases} \\ Richtig & \begin{cases} falls\, A^0(A, z) = M_{i+1}.zustand \\ falls\, A^1(A, z) = M_{i+1}.zustand \end{cases} \end{cases}$$
   n_A^0 bezeichnet die absolute Anzahl der Aufrufe, in denen Automat A im Zustand "Null" war.

n_A^1 bezeichnet die absolute Anzahl der Aufrufe, in denen Automat A im Zustand "Eins" war.

kn_A^1 ist die Anzahl der korrekt eingeschätzten "Eins"-Zustände durch Automat A.
khn_A^1 ist die Anzahl der korrekt hintereinander eingeschätzten "Eins"-Zustände durch Automat A.
kn_A^0 ist die Anzahl der korrekt eingeschätzten " Null" Zustände durch Automat A.
khn_A^0 ist die Anzahl der korrekt hintereinander eingeschätzten "Null"-Zustände durch Automat A.
fn_A^1 ist die Anzahl der falsch eingeschätzten "Eins"-Zustände durch Automat A.
fhn_A^1 ist die Anzahl der falsch hintereinander eingeschätzten "Eins"-Zustände durch Automat A.
fn_A^0 ist die Anzahl der falsch eingeschätzten "Null"-Zustände durch Automat A.
fhn_A^0 ist die Anzahl der falsch hintereinander eingeschätzten "Null"-Zustände durch Automat A.

Wir setzen $g_0(A)$ und $g_1(A)$ gleich:
$g_0(A) := \frac{kn_A^0}{kn_A^0+fn_A^0}$
$g_1(A) := \frac{kn_A^1}{kn_A^1+fn_A^1}$
Wir setzen Qualität Q von Automat A gleich:
$Q(A) := \frac{g_1(A)+g_0(A)}{2}$

Innerhalb jedes Moduls M gibt es $2r : r \in \mathbb{N}^+$ Zufallszahlsgeneratoren $\mathcal{G}$. Die generierten Zufallszahlen werden von den endlichen Automaten interpretiert, um eine Schätzung für den Zustand des Folgemoduls abzugeben. r Generatoren sind "Null"-Zufallszahlsgeneratoren $\mathcal{G}^0$, deren Ergebnisse im "Null"-Zustand der Automaten benutzt werden. Der Rest von $\mathcal{G}$ sind "Eins"-Zufallszahlsgeneratoren $\mathcal{G}^1$ für den "Eins"-Zustand der Automaten. Jeder Automat eines Moduls nutzt für einen Schätzschritt die Menge an r Zufallsgeneratoren, die zu seinem derzeitigen Zustand passen. Das heißt, das Modul benutzt in jedem Schritt r^2 Schätzungen kumulativ. Ein Zufallgenerator hat eine Menge von Intervallen, von denen das mit der höchsten Qualität als aktives Intervall ausgewählt wird, um damit eine Zufallszahl für den Entscheidungsschritt der Automaten zu generieren. Alle Intervalle werden im nachfolgenden Schritt überprüft und G wird aktualisiert. Jeder Zufallszahlsgenerator G besteht aus folgenden Komponenten:

1. n_{G^0} bezeichnet die gesamte Anzahl der Aufrufe des Zufallszahlsgenerators in der Vergangenheit, falls G ein "Null"-Zufallszahlsgenerator ist, d.h. falls $G \in G^0$.
   n_{G^1} bezeichnet analog die gesamte Anzahl der Aufrufe, falls $G \in G^1$.
2. 10^k, $k \in \mathbb{N}^+$ Intervalle $I = (0, 1]$.

Jedes Intervall I hat folgende Komponenten:

1. Repräsentant re, so dass $re \in (0, 1]$ ist.
2. Eine Boolesche Funktion $B(.)$, um das Intervall I innerhalb eines Zufallszahlsgenerators G in Bezug auf seinen Repräsentanten re, einen Automat A und eine Bewegungsstärke $Z_{mov}(M)$ zu bewerten:
   $B(I, A, Z_{mov}(M)) := \begin{cases} true & falls\ b(R(A), re) = b(R(A), Z_{mov}(M)) \\ false & sonst \end{cases}$
3. Funktion $T: \{Null, Eins\}^2 \to \{-1, 1\}$
   $T(M_i.zustand, M_{i+1}.zustand) := \begin{cases} -1 & M_i.zustand = M_{i+1}.zustand \\ 1 & sonst \end{cases}$
4. Funktion $C: B(.) \to \{0, 1\}$
   $C := \begin{cases} 1 & falls\, B(I, , A, Z_{mov}(M)) = true \\ 0 & sonst \end{cases}$
5. Variable k_a definiert als:
   $k_a := \begin{cases} 0 & falls\, t_s = 0 \\ k_a + 1 & falls\, sign\left(\sum_{i=1}^{r} b(R(A_i), re)\right) = sign\left(T(M_i.zustand, M_{i+1}.zustand)\right) \end{cases}$
6. Variable k_r, die in jedem Schritt wie folgt aktualisiert wird:
   $k_r = \sum_{i=1}^{r} C(B(I, A_i, Z_{mov}(M)))$
7. Wir definieren die Güte g eines Intervalls I wie folgt:
   $g(I) := \frac{\alpha+\beta+\gamma}{3}$, wobei α, β und γ wie folgt definiert sind:

$$\alpha := \begin{cases} \frac{k_a}{n_{G^0}} & falls\ G \in G^0 \\ \frac{k_a}{n_{G^1}} & sonst \end{cases}$$

   Sei die Genauigkeit $l_1(t_s)$ von I zum Zeitpunkt t_s definiert als:
   $l_1(t_s) := 1 - \frac{|re - Z_{mov}(M_i)|}{\max(re, Z_{mov}(M_i))}$
   β ist definiert als Durchschnitt der $l_1(t_s)$ über alle Aufrufe des Intervalls I.

   Sei $l_2(t_s)$ definiert als:
   $l_2(t_s) := \frac{k_r}{r}$
   γ ist definiert als Durchschnitt der $l_2(t_s)$ über alle Aufrufe des Intervalls I.
   (a) Wir definieren das aktive Intervall eines Zufallsgenerators G als:
   $A(G) := \arg\max_{I \in G}\ g(I)$

Nun beschreiben wir die Lernphase unserer Automaten und Zufallszahlsgeneratoren. Wir beginnen mit den Automaten A.

Wir modifizieren $A.p_{00}$ wie folgt, falls $A.zustand =$ Null im Bezug auf die Zufallszahl z ist:

$$A.p_{00} = \begin{cases} A.p_{00} + \frac{1}{\Theta(J_1)}(\alpha * U(0,1)) & falls\, M_{i+1}.zustand = Null \\ A.p_{00} - \frac{1}{\Theta(J_2)}(\alpha * U(0,1)) & falls\, M_{i+1}.zustand = Eins \end{cases}$$

$$J_1 := \begin{cases} khn_A^0 & falls\, B(A, z) = Richtig \\ fhn_A^0 & sonst \end{cases} \text{ und } J_2 := \begin{cases} khn_A^1 & falls\, B(A, z) = Richtig \\ fhn_A^1 & sonst \end{cases}$$

Dabei sind $U(0,1)$ und $\Theta(x)$ wie folgt definiert:

$U(0,1)$ ist eine Funktion, die eine zufällige Zahl $0 < x \leq 1$ mit Gleichverteilung in Intervall $(0,1]$ zurück gibt.

$\Theta(x)$ ist eine Funktion. Wir setzen $\Theta(x) = 2^x$.

α ist gleich $\alpha = \frac{1}{2}(\beta + \gamma)$. β und γ sind wie folgt definiert:

$$\beta := \begin{cases} \frac{fn_A^0}{n_A^0 + (c * kn_A^0)} & falls\, A.Zustand = Null \\ \frac{fn_A^1}{n_A^1 + (c * kn_A^1)} & sonst \end{cases}$$

Dabei ist $c \geq 1$ eine konstante Zahl.

$\gamma := \frac{P_1(t)}{P_2(t)}$, wobei $P_1(t)$ und $P_2(t)$ (t ist der Zeitindex) definiert sind als:

$P_1(t) := \sum_{i=1}^{j} |w_{t-i+1} - w_{t-i}|$

$P_2(t) := \sum_{i=0}^{j} w_{t-i}$.

Analog modifizieren wir $A.p_{11}$ wie folgt, falls $A.zustand = Eins$ in Bezug auf Zufallszahl z ist:

$$A.p_{11} = \begin{cases} A.p_{11} + \frac{1}{\Theta(J_1)}(\alpha * U(0,1)) & falls\, M_{i+1}.zustand = Eins \\ A.p_{11} - \frac{1}{\Theta(J_2)}(\alpha * U(0,1)) & falls\, M_{i+1}.zustand = Null \end{cases}$$

$$J_1 := \begin{cases} khn_A^1 & falls\, B(A,z) = Richtig \\ fhn_A^1 & sonst \end{cases} \text{ und } J_2 := \begin{cases} khn_A^0 & falls\, B(A,z) = Richtig \\ fhn_A^0 & sonst \end{cases}$$

Bei einem Zufallszahlsgenerator G ist dann die Lernphase wie folgt:

$$re = \begin{cases} re + [(Z_{mov}(M) - re) * U(0,1)] & falls\, re \leq Z_{mov}(M) \\ re - [(re - Z_{mov}(M)) * U(0,1)] & sonst \end{cases}$$

Unser Verfahren braucht die Daten also nicht vorzuverarbeiten und hat nicht wie klassische Dataminingverfahren eine Lernphase und eine Testphase. Wir lernen in chronologischer Reihenfolge (inkrementell) und entscheiden auch online (gleichzeitig).

4 Algorithmus zur Richtungsprognose

Hier beschreiben wir die kumulative Entscheidungsphase unseres Algorithmus:

```
do {

    Setze sum = 0;

    r(M_i);

    if(i + 1 == n)
        i = 0;
    else
        i = i + 1;

} while(true);
```

$r(M_i)$ {

Bestimme $Z(M_i)$;

Setze $A_j.zustand = M_i.zustand$, $j = 1, 2, \ldots, r$

Setze Z_j, $j = 1, 2, \ldots r$ gleich $Z_j = \begin{cases} G_j^0 & falls\, A_j.zustand = Null \\ G_j^1 & sonst \end{cases}$

for all $(Z_j, j = 1, 2, \ldots, r)$ {

Setze $sum0 = 0$;

Bestimme $I = A(Z_j)$;

for all $(A_k, k = 1, 2, \ldots, r)$ {

$sum0+ = b(A_k, I.re)$;

}

$sum+ = sum0$;

}

$if(sum > 0)$ {

$if(M_i.zustand == Null)$ Print(``Aufwaertsbewegung``);

$else$ Print(``Abwaertsbewegung``);

}

$if(sum < 0)$ {

$if(M_i.zustand == Null)$ Print(``Abwaertsbewegung``);

$else$ Print(``Aufwaertsbewegung``);

}

}

5 Experimentelle Evaluation

Wir evaluieren unser Verfahren mit empirischen Daten. Die Datengrundlage bilden historische Finanzmarktdaten. Die erste Hypothese besagt, dass unser Verfahren für die Richtungsprognose bessere Ergebnisse liefert als ein Münzwurf. Damit können wir zeigen, dass das Verfahren lernfähig ist. Wir haben für unsere Module die Qualitätsfunktion $Q(.)$ definiert. Wir können anhand dieser Funktion die Genauigkeit der Vorhersagen eines Moduls messen. Ergibt die Messung der Vorhersagen eines Moduls eine durchschnittliche Qualität von über 50%, so gilt das Modul als besser denn der Münzwurf. Ist die Mehrzahl der Module besser als der Münzwurf, so gilt die Hypothese als erfüllt.

Um unser Verfahren zu evaluieren, benutzen wir die google- und yahoo-Finanzmarktdaten [3, 7]. Wir betrachten den Tageshöchstpreis (high-price) und das Tagestief (low-price). Wir bilden Zeitreihen über jeden dieser Tageswerte und beobachten sie für Google vom 17-Aug-2005 bis 10-Jan-2006 und für yahoo von 10.04.1997 bis 02.09.1997. Wir definieren 10 Module und beobachten das Verhalten jedes Moduls. Zur formativen Evaluierung bilden wir zwei Varianten von Modulen, nämlich Module im Abhängigkeitsmodus und im Unabhängigkeitsmodus. Die beiden Varianten unterscheiden sich nur bezüglich der Reinitialisierung eines Moduls. Dafür definieren wir für jedes Modul eine Variable *Zeitschritt*, so dass innerhalb dieser Zeitschritte die Qualität des Moduls eine vorgegebene konstante Zahl α überschreiten muss. Wir haben für unsere Untersuchung *Zeitschritt* gleich 10 und α gleich 0.6 gesetzt. Die Module werden wie folgt reinitialisiert:

- Solange im Abhängigkeitsmodus ein Modul innerhalb einer vorgegebenen Anzahl von Zeitschritten eine Qualität hat, die höher als oder gleich α ist, werden die anderen Module, die eine Qualität kleiner als α haben, nicht erneut initialisiert.
- Im Unabhängigkeitsmodus werden Module unabhängig voneinander erneut initialisiert, falls sie nicht innerhalb einer vorgegebenen Anzahl von Zeitschritten den festgelegten Qualitätswert (gröer gleich α) überschreiten.

Wir stellen unsere Ergebnisse zuerst für ein Datenbeispiel Höchstpreise für Yahoo-Wertpapiere im Abhängigkeitsmodus und dann im Unabhängigkeitsmodus bildlich dar (siehe Abbildung 2 und 3). In beiden Varianten ist die Mehrzahl der Module besser als der Münzwurf. Für Tageshöchstwerte von Yahoo sind sieben von zehn Modulen im Abhängigkeitsmodus besser als der Münzwurf und sogar alle zehn Module im Unabhängigkeitsmodus. Tabelle 1 zeigt die Durchschnittswerte für alle Module des gesamten Experiments. Nach dem optischen Eindruck ist die Qualität der Module im Abhängigkeitsmodus stabiler und hält sich im ganzen Verlauf des Experiments. Die systematische Untersuchung der Stabilität der Module wird Gegenstand unserer zukünftigen Arbeit sein. Insgesamt liefern die Experimente eine klare Bestätigung unserer Hypothese für die gewählten Beispieldaten.

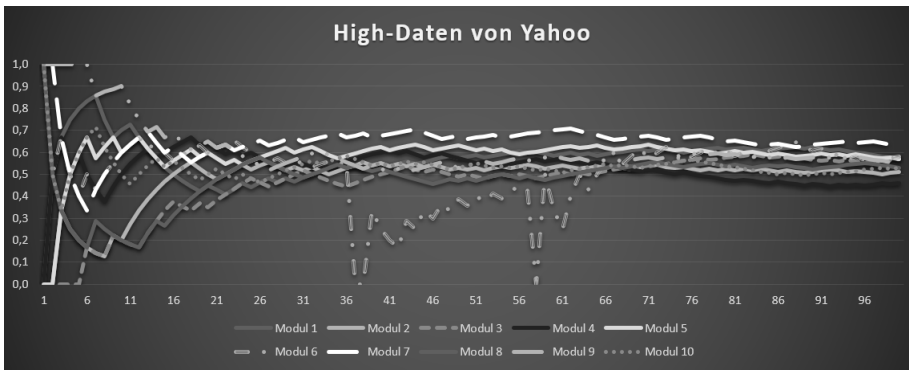

Fig. 2. Qualität der Module für Tageshöchstpreise von Yahoo-Wertpapieren im Abhängigkeitsmodus.

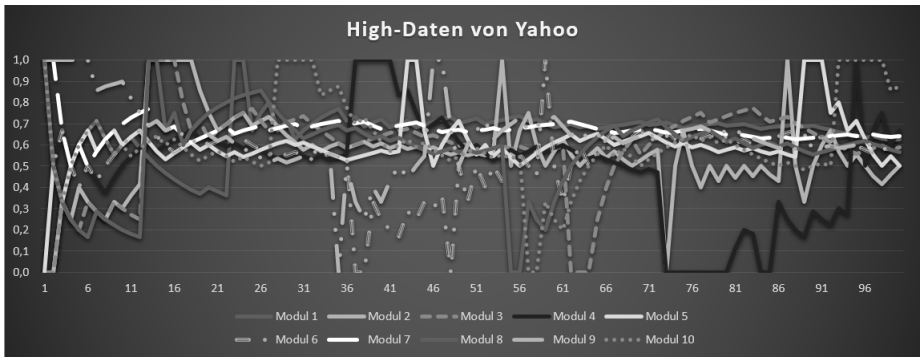

Fig. 3. Qualität der Module für Tageshöchstpreise von Yahoo-Wertpapieren im Unabhängigkeitsmodus.

Table 1. Durchschnittliche Qualität der Module für Tageshöchst- und Tagestiefstspreise.

	Yahoo High		Yahoo Low	
Modul	Abhängig	Unabhängig	Abhängig	Unabhängig
1	0.5746	0.6488	0.5350	0.6519
2	0.4819	0.5625	0.5432	0.5996
3	0.4534	0.5891	0.5559	0.6675
4	0.5302	0.5001	0.5954	0.6322
5	0.5869	0.6050	0.5036	0.5873
6	0.5109	0.5393	0.5527	0.6156
7	0.6510	0.6727	0.6030	0.6566
8	0.4800	0.5821	0.4729	0.5931
9	0.6164	0.6422	0.4659	0.6619
10	0.5349	0.6125	0.6124	0.5404

6 Schlußfolgerung

Im vorliegenden Paper haben wir einen neuen Online-Learning Ansatz zur Analyse von Finanzmarktdaten vorgestellt, der die Richtung in Kurszeitreihen prognostiziert und kein vorgegebenes Modell besitzt. In unserem Ansatz simulieren zufallsbasiert generierte Zahlen $z \in (0, 1]$ zusammen mit (0,1)-Automaten die zukünftige Kursbewegungsstärke. Hier bezieht sich der Begriff Online auf die sequentielle Verarbeitung der Eingabewerte, d.h. beim Aufruf jedes Moduls durch einen Kurswert wird jeweils nacheinander in jedem Schritt $P \rightarrow Q \rightarrow L$ durchgeführt, wobei P, Q und L wie folgt definiert sind
P ist eine Schätzung, ob es im nächsten Zeitschritt eine Ab- oder Aufwärtsrichtung gibt.
Q ist die Qualitätsevaluierung des Moduls und seiner Komponenten nach Bekanntgabe der nächsten Kursbewegungsstärke.
L ist die Anpassung der Interpretation (Verbesserung der Simulation) der (0,1)-Automaten zusammen mit den Zufallszahlen nach Bekanntgabe der nächsten Kursbewegungsstärke.
Da unser Verfahren als Eingabe lediglich Zeitreihen über Numerische Werte benötigt, können wir davon ausgehen, dass es für eine Vielzahl weiterer Anwendungsgebiete genutzt werden kann. Vielversprechende Anwendungsgebiete sind zum Beispiel Richtungsprognosen für medizinische Zeitreihendaten oder Geschäftsdaten wie Umsätze und Provisionen.

References

1. I. Aldridge. *High-frequency trading: a practical guide to algorithmic strategies and trading systems.* John Wiley and Sons, 2013.
2. P. Auer. Online Learning. In C. Sammut and G. I. Webb, editors, *Encyclopedia of Machine Learning and Data Mining*, pages 929–937. Springer US, 2017. DOI: 10.1007/978-1-4899-7687-1_618.
3. Google. https://www.google.com/finance/historical?q=nasdaq%3agoog&ei=hcsdwohiknafsah7sl_aag. letzte Zugriffszeit 24.01.2017.
4. C. H. Hesse. *Angewandte Wahrscheinlichkeitstheorie: Eine fundierte Einführung mit über 500 realitätsnahen Beispielen und Aufgaben.* Vieweg+Teubner Verlag, 2003.
5. K. P. Murphy. *Machine Learning: A Probabilistic Perspective.* The MIT Press, 2012.
6. F. Schmid and M. M. Trede. *Finanzmarktstatistik.* Springer, Berlin, Heidelberg, 2006. DOI: 10.1007/3-540-29795-2_1.
7. Yahoo. https://finance.yahoo.com/quote/yhoo/history?p=yhoo. letzte Zugriffszeit 24.01.2017.