ingenieur
wissenschaften
htw saar

Hochschule für
Technik und Wirtschaft
des Saarlandes
University of
Applied Sciences

STL SYSTEMTECHNIKLABOR

Analyse und Implementierung einer Anwendung zur Ableitung von Handelssignalen anhand von technischen Indikatoren

Bennet Burg, Philip Dausend, Leon Weyand

Technical Report – STL-TR-2026-01 – ISSN 2364-7167

stl.htwsaar.de

Technische Berichte des Systemtechniklabors (STL) der htw saar
Technical Reports of the System Technology Lab (STL) at htw saar
ISSN 2364-7167

Bennet Burg, Philip Dausend, Leon Weyand: Analyse und Implementierung einer Anwendung zur Ableitung von Handelssignalen anhand von technischen Indikatoren
Technical report id: STL-TR-2026-01

First published: March 2026
Last revision: March 2026
Internal review: Andreas Schaffhauser, Michael Messner, André Miede

For the most recent version of this report see: https://stl.htwsaar.de/

Title image source: Philip Dausend

htw saar – Hochschule für Technik und Wirtschaft des Saarlandes (University of Applied Sciences)
Fakultät für Ingenieurwissenschaften (School of Engineering)
STL – Systemtechniklabor (System Technology Lab)
Prof. Dr.-Ing. André Miede (andre.miede@htwsaar.de)
Goebenstraße 40
66117 Saarbrücken, Germany
https://stl.htwsaar.de

ingenieur
wissenschaften
htw saar

Hochschule für
Technik und Wirtschaft
des Saarlandes
University of
Applied Sciences

Wissenschaftlicher Bericht

an der Hochschule für Technik und Wirtschaft des Saarlandes
im Studiengang Praktische Informatik
der Fakultät für Ingenieurwissenschaften

Analyse und Implementierung einer Anwendung zur Ableitung von Handelssignalen anhand von technischen Indikatoren

vorgelegt von
Bennet Burg
Philip Dausend
Leon Weyand

Saarbrücken, 15. März 2026

Selbständigkeitserklärung

Wir versichern, dass wir die vorliegende Arbeit (bei einer Gruppenarbeit: den entsprechend gekennzeichneten Anteil der Arbeit) selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt haben.

Wir erklären hiermit weiterhin, dass die vorgelegte Arbeit zuvor weder von uns noch von einer anderen Person an dieser oder einer anderen Hochschule eingereicht wurde.

Darüber hinaus ist uns bekannt, dass die Unrichtigkeit dieser Erklärung eine Benotung der Arbeit mit der Note „nicht ausreichend“ zur Folge hat und einen Ausschluss von der Erbringung weiterer Prüfungsleistungen zur Folge haben kann.

Saarbrücken, 15. März 2026

Bennet Burg, Philip Dausend,
Leon Weyand

Zusammenfassung

Diese Arbeit untersucht, inwieweit technische Indikatoren zur Ableitung belastbarer Handelssignale geeignet sind und wie sich kurs-, fundamental- und nachrichtenbasierte Informationen in einer Anwendung zur Entscheidungsunterstützung zusammenführen lassen. Ausgangspunkt ist die praktische Frage, ob kurzfristiges Markt-Timing mit klassischen technischen Signalen gegenüber einer Buy-and-Hold-Strategie einen robusten Vorteil erzielen kann.

Methodisch wird zunächst ein naiver Signalansatz analysiert, der RSI, MACD sowie Golden Cross und Death Cross in einem Mehrheitsvotum kombiniert. Anschließend wird eine historische Optimierung der Indikatorgewichte mittels aktienspezifischer Grid Search durchgeführt und auf ungesehenen Daten evaluiert. Aufbauend auf den dabei gewonnenen Erkenntnissen wird ein Regime-Ansatz umgesetzt, bei dem technische Indikatoren nicht primär als Timing-Instrument, sondern zur Klassifikation von Marktzuständen genutzt werden. Dafür werden Trend und Volatilität zu vier Regimen zusammengeführt und in eine abgestufte Exposure-Strategie überführt.

Die Ergebnisse zeigen konsistent, dass sowohl der naive als auch der historisch optimierte Signalansatz Buy-and-Hold auf Testdaten nicht zuverlässig schlagen. Insbesondere weist die starke Leistungsdifferenz zwischen Trainings- und Testphase auf eine geringe Generalisierungsfähigkeit und Overfitting-Risiken hin. Der Regime-Ansatz erzielt hingegen bei den untersuchten Aktien keine systematische Outperformance in der Rendite, reduziert aber den maximalen Drawdown durchgängig und adressiert damit das Risikoziel deutlich besser als die vorherigen Ansätze.

Als praktischer Beitrag wurde mit dem AlgoTrader ein lokaler Prototyp entwickelt, der die relevanten Datenquellen in einer mit NiceGUI umgesetzten Oberfläche integriert. Die Arbeit zeigt damit, dass technische Indikatoren isoliert nur eingeschränkt für präzises Markt-Timing geeignet sind, jedoch einen klaren Mehrwert für die Risikoeinordnung und die strukturierte Entscheidungsunterstützung im Kontext aktienbasierter Handelsstrategien bieten.

Inhaltsverzeichnis

1 Einleitung

Der Aktienhandel hat sich in den letzten Jahrzehnten stark verändert. Einerseits ist die Anzahl der Menschen, die in Aktien investieren, deutlich gestiegen. Laut dem Deutschen Aktieninstitut besaßen im Jahr 2025 rund 14,1 Millionen Menschen in Deutschland Aktien, Aktienfonds oder ETFs, was rund 20 % der Bevölkerung ab 14 Jahren entspricht [7].

Durch Neobroker wie Robinhood[1], Trade Republic[2] und Scalable Capital[3] haben immer mehr Menschen Zugang zu den Finanzmärkten erhalten. Gleichzeitig findet man im Internet immer mehr Wissen über Investitionen, beispielsweise bei Finanzfluss[4], Finanztip[5] oder Investopedia[6]. Diese Entwicklungen haben mutmaßlich zu den heutigen hohen Investitionszahlen beigetragen.

Parallel zu dieser Demokratisierung des Marktzugangs hat sich auch die technische Seite des Investierens weiterentwickelt. Insbesondere die Verfügbarkeit leistungsfähiger Programmiersprachen, offener Schnittstellen zu Marktdaten und benutzerfreundlicher KI-gestützter Werkzeuge senkt die Eintrittsbarrieren für datengetriebene Entscheidungsprozesse erheblich. Strategien, die früher vor allem institutionellen Marktteilnehmern vorbehalten waren, können damit zunehmend auch von privaten und semi-professionellen Akteuren umgesetzt, getestet und bewertet werden.

Vor diesem Hintergrund gewinnt auch der algorithmische Handel als systematischer Ansatz der Kapitalanlage an Bedeutung. Dabei werden Algorithmen eingesetzt, um basierend auf Nachrichtendaten, Volumen- und Preisdaten automatisierte Handelsentscheidungen zu treffen, mit dem Ziel, eine Überrendite gegenüber traditionellen Anlagestrategien zu erzielen [2].

1.1 Zielsetzung

Unsere Arbeit greift diese Entwicklung auf und untersucht, wie ein algorithmisches Handelssystem konzipiert, implementiert und evaluiert werden kann. Dazu entwickeln wir eine Anwendung, die relevante Informationen aus verschiedenen Quellen bündelt und aufbereitet, um eine informierte Entscheidungsfindung zu unterstützen. Gleichzeitig analysieren wir die Grenzen und Möglichkeiten von technischen Indikatoren als Grundlage für Handelssignale und evaluieren unsere Ansätze anhand von Backtests auf historischen Marktdaten.

1.2 Aufbau

Diese Arbeit ist nach der Reihenfolge unseres Vorgehens aufgebaut. Zuerst erläutern wir in Kapitel 2 die Grundlagen, die für das Verständnis der nachfolgenden Kapitel notwendig sind. Anschließend beschreiben wir in Kapitel 3 die technische Umsetzung unserer

[1] `https://robinhood.com/eu/de/`
[2] `https://traderepublic.com/de-de/`
[3] `https://de.scalable.capital`
[4] `https://www.finanzfluss.de/`
[5] `https://www.finanztip.de/`
[6] `https://www.investopedia.com/`

Anwendung, bevor wir in Kapitel 4 die Ergebnisse unserer Analyse und Backtests vorstellen. In Kapitel 6 fassen wir unsere Erkenntnisse zusammen und geben einen Ausblick auf mögliche weiterführende Arbeiten.

2 Grundlagen

Bevor wir uns mit der Implementierung unserer Anwendung beschäftigen, ist es wichtig, die technischen Grundlagen zu verstehen, auf denen unsere Arbeit aufbaut. In diesem Kapitel werden wir die wichtigsten Technologien und Konzepte vorstellen, die für die Entwicklung unserer Anwendung relevant sind. Dazu gehören die Programmiersprache Python, das UI-Framework NiceGUI und der Finanzdatenanbieter Alpha Vantage. Darüber hinaus werden wir auch die Rolle von Nachrichten in der Finanzanalyse erläutern, da sie eine wichtige Quelle für Informationen und Einflussfaktoren auf die Märkte darstellen. Weiterhin werden wir grundlegende Begriffe des Börsenhandels vorstellen, um ein umfassendes Verständnis für die nachfolgenden Kapitel zu schaffen.

2.1 Technische Grundlagen

Zunächst werden wir die technischen Grundlagen vorstellen, die für die Entwicklung unserer Anwendung relevant sind. Dazu gehen wir auf den verwendeten Tech-Stack ein, sowie den genutzten Finanzdatenanbieter Alpha Vantage.

2.1.1 Python

Python[1] ist eine Programmiersprache, die für ihre Einfachheit und Lesbarkeit bekannt ist. Sie bietet eine breite Palette von Bibliotheken und Frameworks, die die Entwicklung von Anwendungen erleichtern. Für unseren Anwendungsfall ist Python besonders geeignet aufgrund der Verfügbarkeit von Bibliotheken für die Datenanalyse. Eine sehr beliebte Bibliothek für die Datenanalyse ist Pandas[2], die leistungsstarke Datenstrukturen und Funktionen für die Manipulation von Daten bietet. Aufgrund der Menge an Daten, die wir verarbeiten müssen, ist es wichtig, eine effiziente und leistungsstarke Sprache zu verwenden, und Python erfüllt diese Anforderungen. Da Python von Haus aus kein UI-Framework bietet, werden wir auf externe Bibliotheken zurückgreifen, um die Benutzeroberfläche zu erstellen. In diesem Fall haben wir uns für NiceGUI entschieden, da es eine einfache und benutzerfreundliche Möglichkeit bietet, Webanwendungen zu erstellen.

2.1.2 NiceGUI

NiceGUI[3][4] ist ein UI-Framework für Python, das es ermöglicht, Benutzeroberflächen im Webbrowser zu erstellen. Es bietet für die gängigsten UI-Elemente vorgefertigte Komponenten, die einfach zu verwenden und anpassbar sind. Die Komponenten erinnern an den Stil der React-Komponentenbibliothek Material-UI[5], welche der Google Material Design Spezifikation folgt. NiceGUI ermöglicht es, interaktive und ansprechende Benutzeroberflächen zu erstellen, ohne dass umfangreiche Kenntnisse in Webentwicklung erforderlich

[1] https://www.python.org/
[2] https://pandas.pydata.org/
[3] https://github.com/zauberzeug/nicegui
[4] https://nicegui.io/
[5] https://mui.com

sind. Es bietet auch Funktionen für die Integration von Daten und die Erstellung von Graphen, was es zu einer idealen Wahl für unsere Anwendung macht.

2.1.3 Alpha Vantage

Alpha Vantage[6] ist ein Online-Dienst, der APIs für den Zugriff auf Finanzmarktdaten bereitstellt. Neben Fundamentaldaten, Echtzeitkursen und historischen Daten bietet der Dienst auch Endpunkte für technische Indikatoren an, die für die Analyse von Finanzmärkten nützlich sind.

2.2 Nachrichten

Ein wichtiger Aspekt bei der Analyse von Aktienkursen sind die Nachrichten, die das Unternehmen betreffen. Diese können einen erheblichen Einfluss auf den Kurs haben, da sie Informationen über die finanzielle Gesundheit, zukünftige Pläne oder auch unerwartete Ereignisse liefern.

Es ist wichtig, diese Nachrichten sorgfältig zu analysieren und in den Kontext der allgemeinen Marktsituation zu setzen, um fundierte Entscheidungen treffen zu können. Die Nachrichten können von verschiedenen Quellen stammen, wie zum Beispiel Pressemitteilungen, Finanzberichte oder auch soziale Medien. Es empfiehlt sich daher, eine breite Palette von Quellen zu nutzen, um eine umfassende Sicht auf die Situation zu erhalten. Einige wichtige Aspekte, die bei der Analyse von Nachrichten berücksichtigt werden sollten, sind die Glaubwürdigkeit der Quelle, die Relevanz der Informationen für das Unternehmen und die potenziellen Auswirkungen auf den Aktienkurs. Es ist auch wichtig, die zeitliche Nähe der Nachrichten zu berücksichtigen, da aktuelle Informationen oft einen stärkeren Einfluss auf den Kurs haben können als ältere Nachrichten. Daher ist es wichtig, dass wir in unserer Anwendung eine Möglichkeit bieten, Zugriff auf aktuelle Nachrichten zu haben, um die Nutzer bei ihrer Analyse zu unterstützen. Die Nachrichten können beispielsweise über die Alpha Vantage API abgerufen werden, mehr dazu in Abschnitt 3.3.

2.3 Börsenbegriffe

Im Folgenden werden wir einige grundlegende Begriffe vorstellen, die im Kontext der Börse und des Aktienhandels relevant sind. Diese Begriffe bilden die Grundlage für das Verständnis der nachfolgenden Kapitel und sind wichtig, um die Funktionsweise von Finanzmärkten und die Analyse von Aktienkursen zu verstehen.

2.3.1 Bullish und bearish

Die Begriffe „bullish“ und „bearish“ werden verwendet, um die allgemeine Stimmung oder Erwartung in Bezug auf die Kursentwicklung eines Vermögenswerts zu beschreiben. „Bullish“, oder auch haussierend, bedeutet, dass die Marktteilnehmer erwarten, dass der Kurs eines Vermögenswerts steigen wird. Der Begriff leitet sich von der Vorstellung eines Bullen ab, der mit seinen Hörnern nach oben stößt, um den Kurs zu erhöhen [5, S. 205]. Auf der anderen Seite bedeutet „bearish“, oder auch baissierend, dass die Marktteilnehmer erwarten, dass der Kurs eines Vermögenswerts fallen wird. Dieser Begriff stammt von der Vorstellung eines Bären, der mit seinen Tatzen nach unten schlägt, um den Kurs zu

[6] https://www.alphavantage.co

senken. Diese Begriffe werden oft verwendet, um die allgemeine Marktrichtung oder die Stimmung der Investoren zu beschreiben [5, S. 123–124].

2.3.2 Fundamentaldaten

Fundamentaldaten beziehen sich auf die finanziellen und wirtschaftlichen Informationen eines Unternehmens, die zur Bewertung seiner finanziellen Gesundheit und seines Potenzials herangezogen werden. Dazu gehören Kennzahlen wie Umsatz, Gewinn, Schulden, Eigenkapital, Cashflow und andere finanzielle Indikatoren. Fundamentaldaten werden oft verwendet, um den inneren Wert einer Aktie zu bestimmen und zu entscheiden, ob sie über- oder unterbewertet ist. Investoren analysieren diese Daten, um fundierte Entscheidungen über den Kauf oder Verkauf von Aktien zu treffen [5, S. 425–427].

2.3.3 Technische Indikatoren

Technische Indikatoren sind mathematische Berechnungen, die auf historischen Kurs- und Volumendaten basieren, um Trends, Muster und potenzielle Kauf- oder Verkaufssignale zu identifizieren. Sie werden von technischen Analysten verwendet, um die zukünftige Kursentwicklung vorherzusagen. Diese Indikatoren können helfen, überkaufte oder überverkaufte Bedingungen zu erkennen und somit potenzielle Wendepunkte im Markt zu identifizieren [5, S. 996].

2.3.4 Überkauft und überverkauft

Die Begriffe „überkauft“ und „überverkauft“ beziehen sich auf die Bewertung eines Vermögenswerts, insbesondere einer Aktie, im Kontext von technischen Indikatoren. Ein Vermögenswert gilt als überkauft, wenn sein Preis in einem bestimmten Zeitraum stark gestiegen ist, die Preissteigerung aber nicht durch fundamentale Faktoren gerechtfertigt ist. Dies kann ein Signal dafür sein, dass der Preis bald fallen könnte, da die Käufer möglicherweise übermäßig optimistisch sind. Auf der anderen Seite gilt ein Vermögenswert als überverkauft, wenn sein Preis stark gefallen ist, aber die fundamentalen Faktoren dies nicht rechtfertigen. In diesem Fall kann es sich um eine Übertreibung der Verkäufer handeln, die auf eine bevorstehende Preissteigerung hinweisen könnten [5, S. 779].

2.3.5 Gleitender Durchschnitt

Ein gleitender Durchschnitt ist ein mathematischer Indikator, der den Durchschnittspreis eines Vermögenswerts über einen bestimmten Zeitraum berechnet. Er wird sowohl in der technischen Analyse als auch in der Fundamentalanalyse verwendet. Er dient dazu, Trends zu identifizieren und potenzielle Kauf- oder Verkaufssignale zu generieren [5, S. 465].

3 Implementierung einer Anwendung zur Handelssignal-Analyse

Um informierte Handelsentscheidungen treffen zu können, ist es wichtig, über eine Vielzahl von Informationen zu verfügen. Um diese Informationen an einem Ort zu bündeln und übersichtlich darzustellen, wurde eine Anwendung namens AlgoTrader entwickelt. In den nachfolgenden Abschnitten beschreiben wir das Grundkonzept der Anwendung, die Architektur sowie die Implementierung der einzelnen Funktionen, und die Besonderheiten bei der Implementierung der Nachrichtenfunktion.

3.1 Grundkonzept

Der AlgoTrader ist eine Anwendung, die den Benutzer dabei unterstützen soll, eine informierte Handelsentscheidung über Aktien zu treffen. Er hat dazu die Möglichkeit, Symbole zu seiner Watchlist hinzuzufügen, wodurch diese von der Anwendung überwacht werden. Der Nutzer erhält so an einer Stelle Informationen zu:

- aktuellen und historischen Kursen,
- technischen Indikatoren zur eigenen Interpretation,
- Nachrichtenartikeln, die das Symbol betreffen, gemeinsam mit dem aktuellen Marktregime als Grundlage für ein kombiniertes Gesamtsignal, und
- Fundamentaldaten, wie Dividenden, Splits sowie Quartalsberichte.

Außerdem hat der Benutzer die Möglichkeit, sich alle Nachrichten zu einer Kategorie (beispielsweise 'Blockchain', 'Technologie' oder 'Finanzen') anzeigen zu lassen, um so einen Überblick über die aktuellen Entwicklungen in diesem Bereich zu erhalten. Er kann sich im Artikel die betroffenen Symbole und dafür jeweils die Relevanz und das Sentiment anzeigen lassen, um so zu entscheiden, ob ein Symbol für ihn relevant ist oder nicht.

Die Anwendung ist für den lokalen Gebrauch gedacht. Da die Daten nicht von einem Server bereitgestellt werden, muss die Anwendung den Abruf, die Speicherung und die Verarbeitung der Daten selbst übernehmen.

3.2 Architektur

Der AlgoTrader ist eine Python-Anwendung. Die grobe Architektur dieser Anwendung ist in Abbildung 3.1 dargestellt. Sie ist in drei logische Module unterteilbar:

- Präsentation: Stellt die Benutzeroberfläche bereit.
- Synchronisation: Ruft die Schnittstellen der Broker auf, um Marktdaten und Nachrichtenartikel abzurufen.
- Persistenz: Speichert Daten wie Kursverläufe, technische Indikatoren und Nachrichtenartikel.

Nachfolgend werden diese Module einzeln betrachtet.

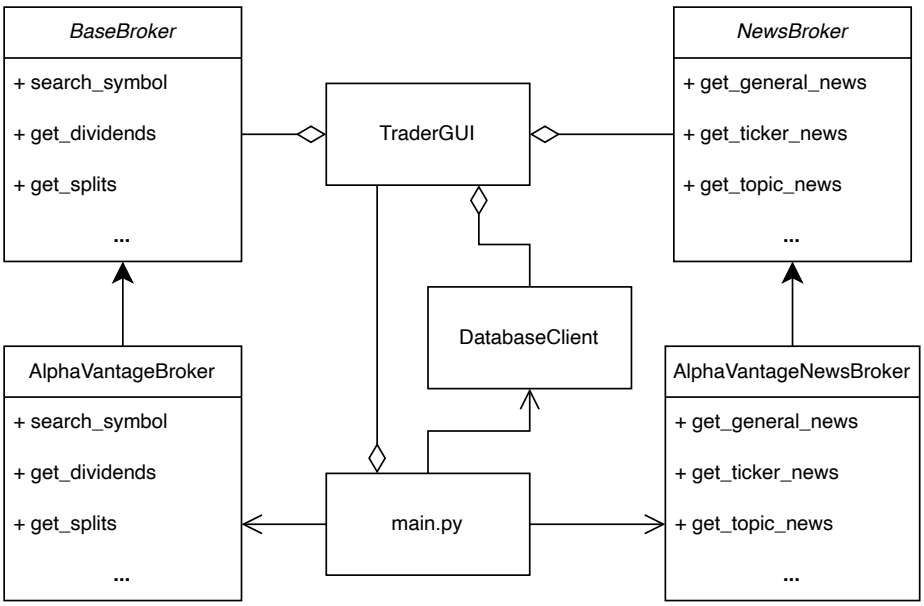

Abbildung 3.1: Architektur der Anwendung

Präsentation

Das Präsentationsmodul stellt die Nutzeroberfläche bereit. Diese wurde mit NiceGUI [1] implementiert. Wir haben uns für dieses Framework entschieden, da es leicht anzuwenden und gut dokumentiert ist.

Die Nutzeroberfläche wurde in mehrere Seiten unterteilt, die über ein Navigationsmenü erreichbar sind und sich an den groben Funktionalitäten der Anwendung orientieren:

- **Startseite**: Zeigt alle Symbole, die sich auf der Watchlist befinden, deren aktuellen Kurs sowie das aktuelle Signal an.
- **Symboldetails**: Zeigt detaillierte Informationen zu einem ausgewählten Symbol an, wie die Zusammensetzung der Signalberechnung, den Kursverlauf, wichtige Termine und relevante Nachrichtenartikel an.
- **Nachrichtenkategorien**: Zeigt alle Nachrichten zu einer ausgewählten Kategorie (beispielsweise 'Blockchain', 'Technologie' oder 'Finanzen') an, zusammen mit betroffenen Symbolen und einer Gesamteinschätzung des Artikelsentiments.
- **Nachrichtendetails**: Zeigt detaillierte Informationen zu einem ausgewählten Artikel an, wie die Relevanz sowie das Sentiment für betroffene Symbole.

Innerhalb dieser Seiten wurden Komponenten erstellt, die wiederverwendbar sind, um Dopplungen zu vermeiden. Gleichzeitig hält dieser Ansatz den Code auf der Seite übersichtlich, da die Logik der Komponenten von der Logik der Seite getrennt ist.

Um die Nutzeroberfläche zu starten, muss die Klasse `TraderGUI` instanziiert werden. Sie erhält als Parameter Instanzen der Datenbroker sowie den Datenbankclient, um die benötigten Daten abrufen und anzeigen zu können. Diese Komponenten werden in den nachfolgenden Abschnitten erklärt.

In unserer Anwendung wird die `TraderGUI`-Klasse in `main.py` mit allen Parametern initialisiert, wie in Listing 3.1 dargestellt.

[1] `https://nicegui.io/`

Listing 3.1: Start der Nutzeroberfläche

```
def main():
  # Initialize database client
  db_client = DatabaseClient("trading_data.db")

  # Initialize stock broker
  stock_broker: brokers.BaseBroker = brokers.AlphaVantageBroker()
  news_broker: newsbrokers.NewsBroker = newsbrokers.
      AlphaVantageNewsBroker()

  # Initialize GUI with database client and stock broker
  gui = TraderGUI(db_client=db_client, stock_broker=stock_broker,
      news_broker=news_broker)
  gui.create_ui()


if __name__ in {"__main__", "__mp_main__"}:
  main()
```

Synchronisation

Wie bereits erwähnt, muss die Anwendung den Abruf, die Speicherung und die Verarbeitung der Daten selbst übernehmen. Um dabei unabhängig von einem bestimmten Datenbroker zu sein, wurden Schnittstellen in abstrakten Klassen definiert, die die benötigten Funktionalitäten bereitstellen.

- **BaseBroker**: Definiert Schnittstellen zum Abruf von Marktdaten, beispielsweise Kursdaten, technische Indikatoren und Fundamentaldaten.
- **NewsBroker**: Definiert Schnittstellen zum Abruf von Nachrichtendaten, beispielsweise Nachrichtenartikel, Schlagzeilen und Stimmungsanalysen.

Die Aufteilung in einen Broker für Aktienmarktdaten und einen Broker für Nachrichtendaten ermöglicht es, die beiden Datenquellen unabhängig voneinander zu behandeln. So können unterschiedliche Anbieter für beide Datenquellen verwendet werden, um beispielsweise die Kosten zu optimieren oder die Datenqualität zu verbessern.

Als konkrete Implementierung für beide Broker wurde im Rahmen dieses Projekts Alpha Vantage gewählt. Wir haben uns für diesen Anbieter aus mehreren Gründen entschieden:

- Kostenlose Erstnutzung: Noch bevor wir für Alpha Vantage bezahlen mussten, konnten wir über einen kostenlosen API-Schlüssel bereits testen, wie wir mit den Schnittstellen arbeiten können.
- Einfache Preisklassen: Alpha Vantage bietet unterschiedliche Preisklassen für unterschiedliche Volumina an API-Aufrufen an, die freigeschalteten Schnittstellen sind zwischen den Klassen jedoch überwiegend gleich. Andere Datenbroker bieten teilweise deutlich komplexere Preismodelle an, wie einen Mix aus einer Premium-Preisklasse sowie Credits für einzelne Aufrufe.
- Gute Dokumentation und Nutzererfahrung: Die Dokumentation von Alpha Vantage ist gut strukturiert und bietet Beispielanfragen an, anhand derer die Antworten der API leicht verständlich sind. Der Aufbau der Endpunkte ist konsistent und unterscheidet sich lediglich in der ausgewählten Funktion sowie deren Parametern.

Ein Vergleich von Alpha Vantage mit einigen anderen Anbietern ist in Tabelle 3.1 dargestellt. Weitere Anbieter, die wir in Betracht gezogen, jedoch nicht weiter verfolgt haben, sind yFinance[2] und Marketstack[3].

Um Alpha Vantage in der Anwendung verwenden zu können, wurden neue Klassen erstellt, die die Schnittstellen der abstrakten Klassen implementieren.

- **AlphaVantageBroker**: Implementiert BaseBroker, um Marktdaten von Alpha Vantage abzurufen.
- **AlphaVantageNewsBroker**: Implementiert NewsBroker, um Nachrichtendaten von Alpha Vantage abzurufen.

Wie oben bereits erwähnt, werden diese Klassen bei der Initialisierung der Nutzeroberfläche instanziiert und als Parameter übergeben, damit die benötigten Daten abgerufen und angezeigt werden können. Dabei muss der API-Schlüssel für Alpha Vantage in der Umgebungsvariable `ALPHA_VANTAGE_API_KEY` hinterlegt sein, damit die Broker vom höheren Rate-Limit profitieren und die Premium-Endpunkte nutzen können.

Um zu vermeiden, dass die Anwendung aufgrund von Rate-Limits der Datenbroker nicht mehr reagiert, wurde die Synchronisation der Daten in die `DataSyncService`-Klasse ausgelagert. Diese Klasse orchestriert den Datenabruf von den Brokern, indem die Anfragen an die Broker mit ausreichender Zeitverzögerung ausgeführt werden. Der Nutzer kann dann in der Nutzeroberfläche die Synchronisation manuell anstoßen, um die Daten zu aktualisieren. So wird vermieden, dass Anfragen wegen Rate-Limits fehlschlagen.

Dieser Ansatz hat den Nachteil, dass die Daten nicht in Echtzeit aktualisiert werden. Allerdings ist so sichergestellt, dass der Nutzer jederzeit Zugriff auf die abgerufenen Daten hat, ohne dass seine Erfahrung durch fehlgeschlagene Anfragen beeinträchtigt wird.

Persistenz

Die abgerufenen Daten von der `DataSyncService`-Klasse müssen gespeichert werden, um wiederholte Anfragen an die Datenbroker zu vermeiden. Dazu wurde eine SQLite-Datenbank verwendet, die über die `DatabaseClient`-Klasse angesprochen wird.

Das Datenbankschema ist in Abbildung 3.2 dargestellt. Es besteht aus mehreren Tabellen, die die verschiedenen Datentypen abdecken:

- `watched_stocks`: Speichert die Symbole, die sich auf der Watchlist befinden.
- `stock_prices`: Speichert den aktuellen Kurs der Symbole.
- `stock_price_history`: Speichert die historischen Tagesendkurse der Symbole.
- `stock_splits`: Speichert Informationen zu Aktiensplits, wie das Datum und die Split-Quote.
- `stock_dividends`: Speichert Informationen zu Dividenden, wie das Datum und die Dividendenhöhe.
- `stock_quarterly_earnings`: Speichert Informationen zu Quartalszahlen, wie das Datum und die wichtigsten Kennzahlen.

[2] `https://github.com/ranaroussi/yfinance`
[3] `https://marketstack.com`

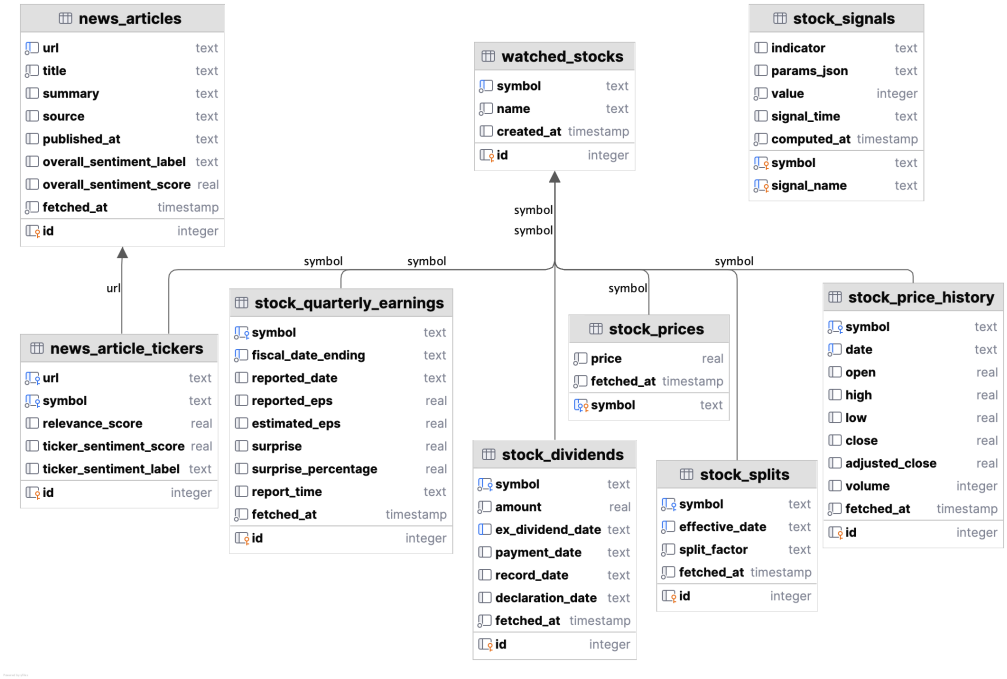

Abbildung 3.2: Datenbankschema der Anwendung

- `stock_signals`: Speichert die aus den Marktdaten berechneten Signale für die Symbole.
- `news_articles`: Speichert die abgerufenen Nachrichtenartikel.
- `news_article_tickers`: Ordnet die Nachrichtenartikel den betroffenen Symbolen zu, zusammen mit der Relevanz und dem Sentiment für jedes Symbol.

Um zu vermeiden, dass die Datenbank zu groß wird, werden bei jedem Abruf die bereits gespeicherten Daten mit den neuen Daten überschrieben. So wird sichergestellt, dass die Datenbank immer die aktuellsten Daten enthält.

Da die `TraderGUI`-Instanz eine Instanz des `DatabaseClient` erhält, kann sie so die zur Darstellung benötigten Daten abrufen und anzeigen.

3.3 Nachrichten

Um eine umfassende Analyse von Aktienkursen zu ermöglichen, ist es wichtig, auch die Nachrichten zu berücksichtigen, die das Unternehmen betreffen. Diese Nachrichten können einen erheblichen Einfluss auf den Kurs haben, da sie Informationen über die finanzielle Gesundheit, zukünftige Pläne oder auch unerwartete Ereignisse liefern.

Eine erste Idee war die Nutzung eines Mail-Newsletters und die Aufbereitung der Informationen aus den Mails, um sie in der Anwendung anzuzeigen. Beispielsweise wurde der Newsletter von MorningCrunch [9] in Betracht gezogen, da er täglich aktuelle Informationen zu verschiedenen Unternehmen liefert. Allerdings könnte die manuelle Aufbereitung der Informationen aus den Mails zeitaufwendig sein und möglicherweise nicht immer zuverlässig funktionieren. Daher haben wir uns entschieden auf die Nutzung von APIs zuzugreifen, die speziell für den Zugriff auf Finanznachrichten entwickelt wurden.

Es gibt verschiedene APIs, die Zugang zu Finanznachrichten bieten, wie zum Beispiel die Alpha Vantage News API [3]. Bei der Implementierung der Nachrichten haben wir

uns auf genau diese API verlassen, da sie neben einer breiten Palette von Finanzdaten auch Informationen über Finanznachrichten bietet [1]. Die API ermöglicht es uns, aktuelle Nachrichten zu bestimmten Aktien oder Märkten abzurufen.

3.3.1 Alpha Vantage Market News & Sentiment API

Die API ermöglicht es, bis zu 1000 Nachrichten pro Anfrage abzurufen, was für unsere Zwecke ausreichend ist. Über den API-Parameter `tickers` können wir die gewünschten Aktien (beispielsweise AAPL für Apple Inc. oder FOREX:USD), und über den Parameter `topics` können wir die Themen der Nachrichten filtern, um nur relevante Informationen zu erhalten. Hierbei gibt es eine Vielzahl an Märkten, die abgedeckt werden, darunter Finanzmärkte, Energie- und Transportunternehmen, Medizin, Technologie, Immobilien und viele mehr. Bei der Sortierung der Nachrichten haben wir uns für die Sortierung `LATEST` entschieden, um immer die neuesten Nachrichten zu erhalten. Das ist per API-Definition der Standardwert, aber es ist wichtig zu erwähnen, dass wir diese Sortierung explizit gewählt haben und keine Änderung dieses Parameters durch den Nutzer erlauben, um sicherzustellen, dass wir immer die aktuellsten Informationen erhalten [3].

Die API liefert verschiedene Informationen zu den Nachrichten zurück, von denen wir nur wenige nicht verwenden. Um die Informationen kompakt zu speichern, verwenden wir einen `DataFrame`. Beispielsweise die Autoren oder das Bild der Headline werden nicht in der Anwendung verwendet, da wir diese Informationen nicht für die Anzeige oder Analyse der Nachrichten benötigen. In Abbildung 3.3 ist ein Beispiel für die Darstellung eines Nachrichtenartikels im UI zu sehen.

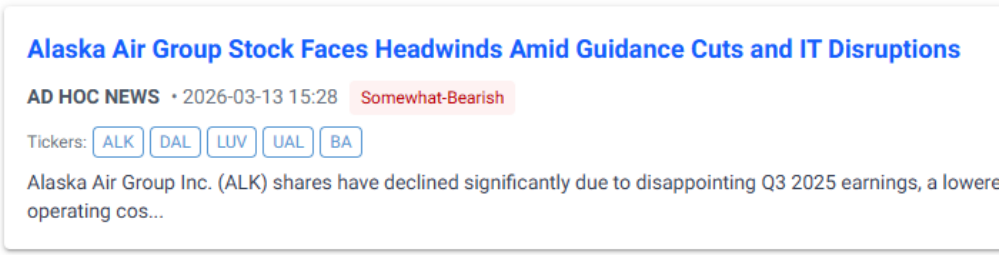

Abbildung 3.3: Beispiel eines Nachrichtenartikels im UI

Die wichtigen Aspekte sind vor allem der Titel, die kurze Zusammenfassung als Teaser und die URL, die zum vollständigen Artikel führt. Darüber hinaus enthält die Antwort der API auch eine Bewertung bezüglich der Stimmung (Sentiment) und der Relevanz der Nachricht für ein Thema oder einen Ticker. Am Anfang der Response steht die Anzahl an Nachrichten, die die Response enhält. Danach folgen die Definitionen der Sentiment- und Relevanzwerte, sodass der Nutzer die Bedeutung des Zahlenwerts schnell einordnen kann. Im folgenden Codeausschnitt ist ein Beispiel für die Antwort der API dargestellt, wobei die Definition der Sentiment- und Relevanzwerte am Anfang der Response zu sehen ist.

Listing 3.2: Beispiel für die API-Antwort von Alpha Vantage

```
1 {
2 "items": "27",
3 "sentiment_score_definition": "x <= -0.35: Bearish; -0.35 < x <=
    -0.15: Somewhat-Bearish; -0.15 < x < 0.15: Neutral; 0.15 <= x <
    0.35: Somewhat_Bullish; x >= 0.35: Bullish",
```

```
"relevance_score_definition": "0 < x <= 1, with a higher score
   indicating higher relevance.",
"feed": [
    {
    "title": "Coinbase Execs Say They Aren't Opposing BTC Tax
       Exemption",
    "url": "https://cointelegraph.com/news/coinbase-deny-lobby-
       bitcoin-tax-exemption",
    "time_published": "20260312T185428",
    "authors": [],
    "summary": "US lawmakers are eyeing tax exemptions for US
       dollar stablecoins, which are pegged and do not change in
       value, but not other cryptocurrencies.",
    "banner_image": "https://s3.cointelegraph.com/uploads
       /2026-03/019ce2e6-a314-7af2-8be8-6b6450429dfe.png",
    "source": "Cointelegraph",
    "topics": [
        {
            "topic": "Economy - Fiscal",
            "relevance_score": "0.158519"
        },
        {
            "topic": "Finance",
            "relevance_score": "1.0"
        }
    ],
    "overall_sentiment_score": -0.016748,
    "overall_sentiment_label": "Neutral",
    "ticker_sentiment": [
        {
        "ticker": "COIN",
        "relevance_score": "0.396574",
        "ticker_sentiment_score": "-0.333047",
        "ticker_sentiment_label": "Somewhat-Bearish"
        },
        {
        "ticker": "CRYPTO:BTC",
        "relevance_score": "0.757528",
        "ticker_sentiment_score": "0.090954",
        "ticker_sentiment_label": "Neutral"
        }
    ]
    }
]
}
```


Der Relevanzwert gibt für die Ticker oder Themen an, wie relevant die Nachricht für das jeweilige Thema oder den jeweiligen Ticker ist. Ein höherer Relevanzwert bedeutet, dass die Nachricht stärker mit dem Thema oder Ticker in Verbindung steht. Der Sentimentwert gibt an, ob die Stimmung der Nachricht bullish, neutral oder bearish ist. Je weiter der Sentimentwert von 0 entfernt ist, desto stärker ist die Stimmung in die jeweilige Richtung. Ein positiver Sentimentwert deutet auf eine bullish Stimmung hin, während ein negativer Sentimentwert auf eine bearish Stimmung hinweist. In unserem UI verwenden wir diese Werte, um die Nachrichten entsprechend zu färben und dem Nutzer eine visuelle Indikation der Stimmung zu geben. So kann der Nutzer schnell sehen, ob es für eine bestimmte Aktie vielleicht bergauf oder bergab geht.

3.3.2 Persistenz der Nachrichten

Im Gegensatz zu den Aktiendaten haben wir uns hier aus verschiedenen Gründen für einen nicht-persistenten Ansatz entschieden. Die Nachrichtendaten sind eine sehr große Menge an Daten, die sich ständig ändern und aktualisieren. Es wäre ineffizient und speicherintensiv, alle diese Daten lokal zu speichern, insbesondere wenn wir nur an den neuesten Nachrichten interessiert sind. Ein Grund für die Persistenz wäre natürlich die Möglichkeit, historische Nachrichten zu analysieren, um als Nutzer Trends oder Muster zu erkennen. Dennoch haben wir uns dazu entschieden, die Nachrichten nur bei Bedarf abzurufen, um die Implementierung zu vereinfachen und Ressourcen zu sparen.

Um trotzdem eine gewisse Historie der Nachrichten zu haben, haben wir uns für einen zweistufigen In-Memory-Ansatz entschieden. Sobald die Nachrichten im GUI gerendert werden, werden sie aus dem oben erwähnten `DataFrame` entnommen und in einem `Dictionary` gecached. Das bedeutet, dass die Nachrichten für ein bestimmtes Thema nur dann abgerufen werden, wenn der Nutzer das Thema auswählt. Wenn der Nutzer das Thema erneut auswählt, werden die Nachrichten jedoch neu abgerufen, um sicherzustellen, dass wir immer die aktuellsten Informationen erhalten. Ein Ausschnitt der Implementierung dieses Caches ist in Listing 3.3 dargestellt.

Listing 3.3: In-Memory-Cache für Nachrichten

```
1     def render([...]) -> None:
2         sentiment_label = article.get('overall_sentiment_label', '
              Neutral')
3         sentiment_color = get_article_sentiment_color(
              sentiment_label)
4
5         \# Cache article and check if seen
6         article_url = article.get('url', '')
7         article_key = None
8         is_seen = False
9         if article_url:
10            article_key = base64.urlsafe_b64encode(article_url.
                  encode()).decode().rstrip('=')
11            if back_url:
12                article['_back_url'] = back_url
13            self.state_manager.cache_article(article_key, article)
14            is_seen = self.state_manager.is_seen(article_key)
15
16        \# Click handler for navigation
17        def on_click():
18            if article_key:
19                self.state_manager.mark_as_seen(article_key)
20                ui.navigate.to(f'/news/{article_key}')
```

Als Schlüssel im Cache verwenden wir die URL der Nachricht, da diese in der Regel eindeutig ist. Um sicherzustellen, dass die Schlüssel im Cache gültige Strings sind und keine Sonderzeichen enthalten, die Probleme verursachen könnten, verwenden wir eine Base64-Codierung der URL.

Stufe zwei ist das Caching der gelesenen Nachrichten in einem `Set`. Wenn ein Nutzer eine Nachricht auswählt, wird die Nachricht in diesem Set gespeichert, sodass der Nutzer sieht, welche Nachrichten er bereits gelesen hat. Dieses Set wird jedoch nicht persistiert, sondern nur während der Laufzeit der Anwendung verwendet. Das heißt, sobald die Anwendung beendet wird, geht die Historie der gelesenen Nachrichten verloren. Wenn jedoch neue Nachrichten abgerufen werden und in der Antwort Nachrichten enthalten

sind, die bereits im Set der gelesenen Nachrichten vorhanden sind, werden diese Nachrichten als gelesen markiert, um dem Nutzer anzuzeigen, dass er diese Nachricht bereits gesehen hat. Auf diese Weise können wir trotz des nicht-persistenten Ansatzes eine gewisse Historie der gelesenen Nachrichten beibehalten. In Listing 3.3 ist zu sehen, dass bei einem Klick auf den Artikel die Nachricht als gelesen markiert wird.

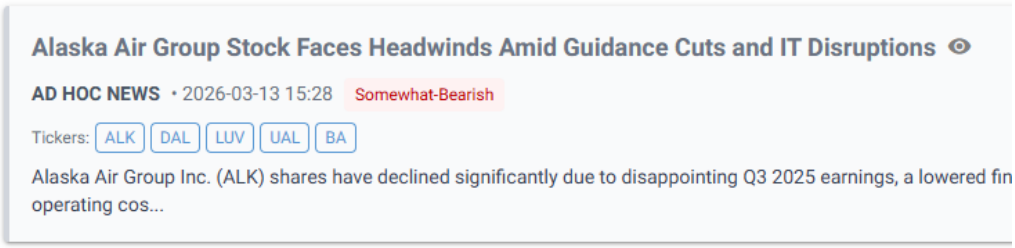

Abbildung 3.4: Beispiel eines gelesenen Nachrichtenartikels im UI

Wenn neue Nachrichten abgerufen werden, kann über die URL der Nachricht überprüft werden, ob die Nachricht bereits als gelesen markiert ist, und entsprechend im GUI dargestellt werden. In Abbildung 3.4 ist ein Artikel zu sehen, der bereits gelesen beziehungsweise angeklickt wurde. Im UI werden gelesene Nachrichten mit einem grauen Hintergrund sowie einem Auge dargestellt, um sie von ungelesenen Nachrichten zu unterscheiden. Auf diese Weise können Nutzer schnell erkennen, welche Nachrichten sie bereits gelesen haben und welche noch ungelesen sind.

Kriterium	Alpha Vantage	FinnHub	TwelveData
Webseite	`alphavantage.co`	`finnhub.io`	`twelvedata.com`
Marktdaten			
Aktualität	15 Minuten verzögert, Echtzeit	Echtzeit	Echtzeit
Zeitraum Historie	Über 20 Jahre	10 Jahre (Basic-Tier)	Über 40 Jahre
Abdeckung	US	US, internationale Börsen als eigene Preisklassen	US, internationale Börsen als eigene Preisklassen
Nachrichtendaten			
Nach Symbol	Ja	Ja	Nein
Nach Kategorie	Ja	Ja	Nein
Enthält Sentiment	Pro Artikel	Gesamtsentiment pro Symbol, nicht pro Artikel	Nein
Fundamentaldaten			
Quartalsberichte	Ja	Ja	Ja
Dividenden	Ja	Ja	Ja
Splits	Ja	Ja	Ja
Nutzungs- und Preismodell			
Gratisvariante	Ja	Ja	Ja
Rate-Limit Gratis	25 pro Tag	60 pro Stunde	8 am Tag
Preismodelle	Premium-Endpunkte und höhere Rate-Limits	Unterschiedliche Module, All-in-One	Unterschiedliche Premium-Tiers, Credit-System
Ideales Modell	50$ pro Monat (75 Anfragen pro Minute)	50$ pro Monat (Marktdaten), 50$ pro Monat und Markt (Nachrichten und Fundamentaldaten)	29$ pro Monat (Grow, 55 API-Credits)
Löschpflicht nach Kündigung	Nein	Ja	Nein

Tabelle 3.1: Vergleich der Datenbroker

4 Analyse

Ein wichtiger Aspekt des Handelns mit Finanzprodukten ist die Strategie, die man verfolgt. So gibt es beispielsweise das Daytrading, bei dem die Händler kleine oder kurzfristige Kursbewegungen ausnutzen, um Gewinne zu erzielen [5, S. 262].

Eine weitere bewährte Strategie ist Buy-and-Hold. Hierbei kauft man ein Finanzprodukt und hält es über einen längeren Zeitraum, um von der langfristigen Wertsteigerung durch das allgemeine Wirtschaftswachstum zu profitieren. Nach Markteffizienztheorie ist es nicht möglich, durch kurzfristige Kursbewegungen Gewinne zu erzielen, da alle Informationen bereits im Kurs enthalten sind [5, S. 27]. Gibt es vielleicht dennoch Ansätze, um kurzfristige Kursbewegungen effizient auszunutzen? In diesem Kapitel werden wir verschiedene Ansätze analysieren, um herauszufinden, ob es bessere Strategien als Buy-and-Hold gibt.

4.1 Ein naiver Ansatz

Unser erster naiver Ansatz besteht darin, drei Kurssignale zu verwenden, um Kauf- und Verkaufssignale zu generieren. Zur Analyse verwenden wir den RSI, den MACD und das Golden Cross beziehungsweise Death Cross.

Zur einfacheren Analyse arbeiten wir mit einem `DataFrame` der Bibliothek `pandas` für Python, welche vor allem für die Datenanalyse und -manipulation verwendet wird [15]. Das dient dazu, die Daten in tabellarischer Form zu organisieren und ermöglicht es uns, die Signale einfach zu berechnen und zu interpretieren. In den folgenden Codebeispielen wird das `DataFrame` als `df` bezeichnet.

Das erste der drei Signale basiert wie bereits erwähnt auf dem Relative Strength Index (RSI). Wenn der RSI über 60 liegt, interpretieren wir dies als überkauft und erwarten eine Kurskorrektur nach unten, was ein Verkaufssignal generiert. Liegt der RSI unter 40, interpretieren wir dies als überverkauft und erwarten eine Kurskorrektur nach oben, was ein Kaufsignal generiert [5, S. 779]. Die Werte zwischen 40 und 60 werden als neutral interpretiert, da sie keine klaren Signale für eine Überkauft- oder Überverkauft-Situation liefern. Die Implementierung dieses Ansatzes sieht wie folgt aus:

Listing 4.1: Implementierung des RSI-Signals

```
1 # -- RSI interpretation --
2 df["rsi_trend"] = "NEUTRAL"
3
4 # Overbought, price went up too much, expects pullback down
5 df.loc[df["rsi"] > 60, "rsi_trend"] = "BEARISH"
6 # Oversold, price went down too much, expects rebound up
7 df.loc[df["rsi"] < 40, "rsi_trend"] = "BULLISH"
```

Ein weiteres Signal ist der Moving Average Convergence/Divergence (MACD) Crossover. Der MACD ist ein Trendindikator auf Basis von exponentiell gleitenden Durchschnitten (EMA) und teilt sich in die MACD-Linie und die Signallinie auf. Die MACD-Linie ist die Differenz zwischen einem schnelleren und einem langsameren exponentiell gleitenden Durchschnitt. Bei Alpha Vantage sind dies standardmäßig der 12-Tage-EMA und der

26-Tage-EMA. Die Signallinie ist ein weiterer exponentiell gleitender Durchschnitt, der standardmäßig über 9 Tage bei Alpha Vantage berechnet wird [1]. Ein Kaufsignal wird genau dann generiert, wenn die MACD-Linie die Signallinie von unten nach oben kreuzt, was auf einen möglichen Aufwärtstrend hinweist. Ein Verkaufssignal entsteht, wenn die MACD-Linie die Signallinie von oben nach unten kreuzt, was auf einen möglichen Abwärtstrend hindeutet [14, S. 294]. Die Implementierung dieses Ansatzes sieht wie folgt aus:

Listing 4.2: Implementierung des MACD-Crossover-Signals

```
# -- MACD interpretation --
df["macd_trend"] = "NEUTRAL"
# MACD-Line crosses above Signal-Line, bullish signal
df.loc[df["macd"] > df["macd_signal"], "macd_trend"] = "BULLISH"
# MACD-Line crosses below Signal-Line, bearish signal
df.loc[df["macd"] < df["macd_signal"], "macd_trend"] = "BEARISH"
```

Als drittes und letztes Signal im naiven Ansatz verwenden wir das Golden Cross beziehungsweise Death Cross. Diese Signale basieren auf dem Schnittpunkt von zwei Simple Moving Averages (SMAs), sprich gleitenden Durchschnitten, nur nicht exponentiell geglättet. In unserer Berechnung haben wir uns für den 50-Tage-SMA und den 200-Tage-SMA entschieden gemäß der gängigen Praxis [10]. Dieses Signal haben wir wie folgt implementiert:

Listing 4.3: Implementierung des Golden-Cross-Signals

```
# -- Trend assessment --
df["trend"] = "NEUTRAL"
# Golden cross, short-term avg crosses above long-term avg
df.loc[df["sma_50"] > df["sma_200"], "trend"] = "BULLISH"
# Death cross, short-term avg crosses below long-term avg
df.loc[df["sma_50"] < df["sma_200"], "trend"] = "BEARISH"
```

4.1.1 Simple Kombination der Signale

In den obenstehenden Codebeispielen in den Listings 4.1, 4.2 und 4.3 haben wir die drei Signale implementiert, um Kauf- und Verkaufssignale zu generieren. Diese Signale können nun kombiniert werden, um einen Gesamttrend zu bestimmen, der als Grundlage für Handelsentscheidungen dienen kann.

Um alle Signale gleichwertig zu behandeln, haben wir uns dazu entschieden, den Trends Zahlenwerte zuzuweisen und diese zu summieren, um eine Gesamtbewertung zu erhalten. Ein Kaufsignal (`BULLISH`) erhält den Wert +1, ein Verkaufssignal (`BEARISH`) erhält den Wert -1 und ein neutrales Signal (`NEUTRAL`) erhält den Wert 0. Sobald die Gesamtsumme positiv ist, interpretieren wir dies als ein Kaufsignal, sobald sie negativ ist, interpretieren wir dies als ein Verkaufssignal, und wenn sie genau 0 ist, interpretieren wir dies als neutral. Die Implementierung dieser Logik sieht wie folgt aus:

Listing 4.4: Implementierung der Signal-Kombination

```
def combine_trends(row):

    scores = {"BULLISH": 1, "NEUTRAL": 0, "BEARISH": -1}
    total = scores[row["trend"]] + scores[row["macd_trend"]] +
        scores[row["rsi_trend"]]
```

```
5
6       if total > 0:
7           return "BULLISH"
8       elif total < 0:
9           return "BEARISH"
10      else:
11          return "NEUTRAL"
12
13  df["overall"] = df.apply(combine_trends, axis=1)
```

Um dieses Gesamtsignal nun in eine Strategie umzusetzen, haben wir uns dazu entschieden, zu kaufen, sobald das Gesamtsignal von neutral oder bearish auf bullish wechselt, und zu verkaufen, sobald das Gesamtsignal von bullish auf neutral oder bearish wechselt. Als Beispiel haben wir die Aktienkurse von Nvidia und Apple analysiert, um die Signale zu generieren und die Strategie zu testen.

Zunächst haben wir den Algorithmus auf die letzten 100 Tagesendpreise der Kurse angewendet, um zu sehen, ob innerhalb eines kurzen Zeitraums bereits bessere Ergebnisse als mit Buy-and-Hold erzielt werden können. Die folgenden zwei Abbildungen zeigen die generierten Signale für Nvidia und Apple. Die grünen Intervalle markieren die Zeiträume, in denen die Strategie ein Kaufsignal generiert hat (`BULLISH`), während die roten Intervalle die Zeiträume markieren, in denen die Strategie ein Verkaufssignal generiert hat (`BEARISH`). Die genauen Preise und Zeitpunkte der Signale können in den Abbildungen nicht genau abgelesen werden, aber die generellen Trends und Signale sind deutlich erkennbar, sowie die Preisunterschiede an den Intervallrändern, die die Renditen der Strategie in diesen Intervallen bestimmen.

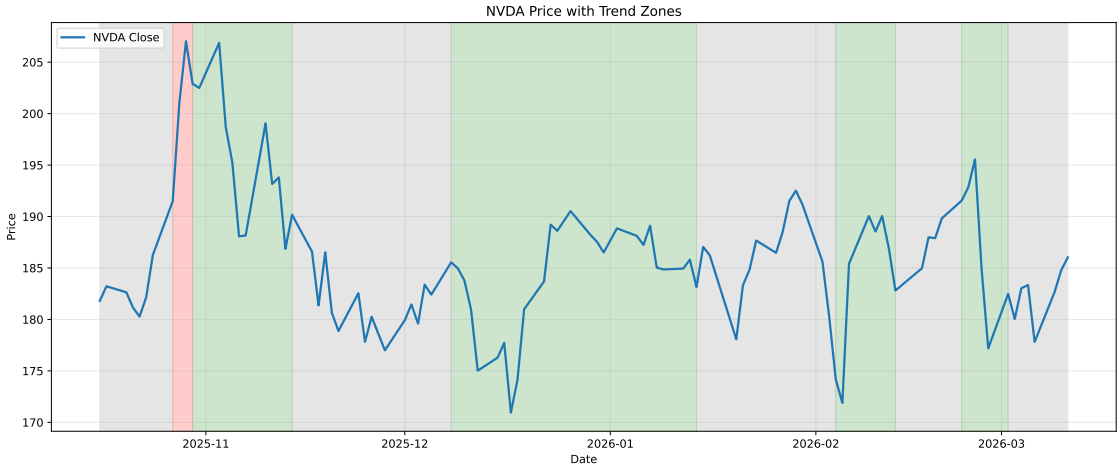

Abbildung 4.1: Generierte Signale für Nvidia

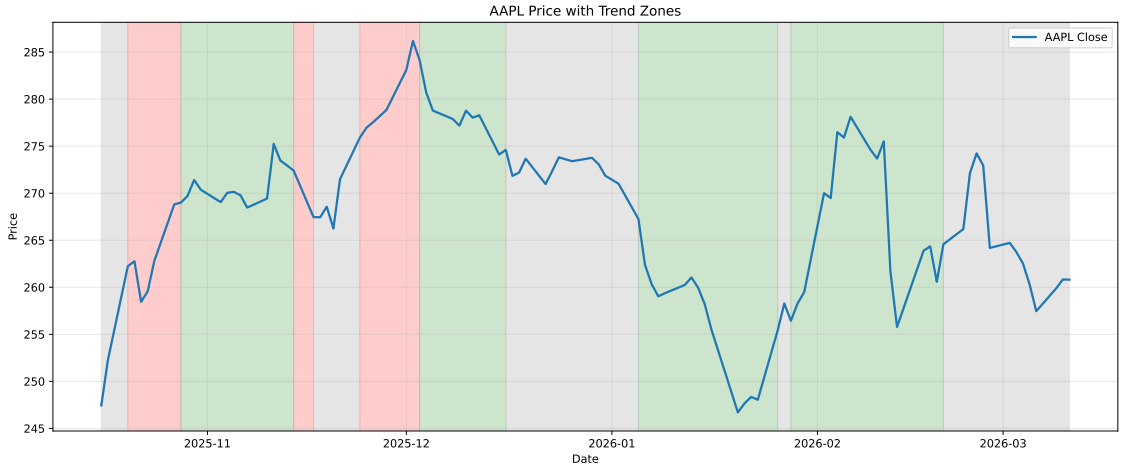

Abbildung 4.2: Generierte Signale für Apple

Bereits in diesem kurzen Zeitraum von etwa 5 Monaten können wir sehen, dass Buy-and-Hold deutlich effektiver gewesen wäre als die Strategie, die auf den generierten Signalen basiert, wodurch sich eine Analyse über einen längeren Zeitraum erübrigt. Wir schauen uns jetzt im Detail an, wie die Strategie im Vergleich zu Buy-and-Hold für die zwei gewählten Aktien abgeschnitten hat. Dazu schauen wir uns die Rendite der Strategie für jedes „grüne“ Intervall an, also die Zeiträume, in denen die Strategie ein Kaufsignal generiert hat, und vergleichen die Summe all dieser Renditen mit der Rendite, die man durch Buy-and-Hold erzielt hätte, also der Rendite von Anfang bis Ende des betrachteten Zeitraums. Da die genauen Werte in den Abbildungen nicht ablesbar sind, haben wir die Preise an den Intervallrändern direkt aus dem `DataFrame` entnommen, um die Renditen der Strategie in den einzelnen Intervallen zu berechnen. Die folgende Tabelle zeigt die Renditen der Strategie im Vergleich zu Buy-and-Hold für Nvidia:

Tabelle 4.1: NVDA Gewinnanalyse für die letzten 100 Datenpunkte (Buy-and-Hold: 2,32 %)

Intervall	Zeitraum	Startpreis	Endpreis	Gewinn (%)
1	30.10.2025 – 14.11.2025	$202,89	$190,17	–6,27 %
2	08.12.2025 – 14.01.2026	$185,55	$183,14	–1,30 %
3	04.02.2026 – 13.02.2026	$174,19	$182,81	+4,95 %
4	23.02.2026 – 02.03.2026	$191,55	$182,48	–4,74 %
Summe				**–7,35 %**

Es zeigt sich deutlich, dass die Strategie in diesem Zeitraum eine negative Rendite von insgesamt –7,35 % erzielt hat. Buy-and-Hold hätte hingegen eine positive Rendite von 2,32 % erzielt, mit einem Tagesendpreis von $181,81 am Anfang des Intervalls und einem Tagesendpreis von $186,03 am Ende des Intervalls. Auch bei Apple zeigt sich ein ähnliches Bild, wie die folgende Tabelle zeigt:

Tabelle 4.2: AAPL Gewinnanalyse für die letzten 100 Datenpunkte (Buy-and-Hold: 5,40 %)

Intervall	Zeitraum	Startpreis	Endpreis	Gewinn (%)
1	28.10.2025 – 14.11.2025	\$269,00	\$272,41	+1,27 %
2	03.12.2025 – 16.12.2025	\$284,15	\$274,61	–3,36 %
3	05.01.2026 – 26.01.2026	\$267,26	\$255,41	–4,43 %
4	28.01.2026 – 20.02.2026	\$256,44	\$264,58	+3,17 %
Summe				**–3,35 %**

Auch hier zeigt sich, dass die Strategie eine negative Rendite von insgesamt –3,35 % erzielt hat, während Buy-and-Hold eine positive Rendite von 5,40 % erzielt hätte, mit einem Tagesendpreis von \$247,45 am Anfang des Intervalls und einem Tagesendpreis von \$260,81 am Ende des Intervalls.

4.1.2 Verbesserung der Signalkombination

Eine Idee zur Verbesserung des Ansatzes ist die Nutzung einer differenzierteren Interpretation der Summe der Signale. Anstatt die Signale einfach zu summieren und eine Schwelle von 0 zu verwenden, haben wir die Summe je nach Größe stärker oder schwächer interpretiert. Damit ergibt sich eine detailliertere Kategorisierung der Signale, die möglicherweise zu besseren Ergebnissen führen könnte. Konkret heißt das: Wenn beispielsweise zwei Signale `BULLISH` sind und eins `NEUTRAL`, interpretieren wir das als starkes Kaufsignal. Wenn nur eines der Signale `BULLISH` ist, interpretieren wir das als schwaches Kaufsignal. Wenn jedoch beispielsweise nur ein Signal `BULLISH` ist und zwei `NEUTRAL`, interpretieren wir das als schwaches Kaufsignal und nicht wie im vorherigen Ansatz als grundsätzliches Kaufsignal, sondern eher als Tendenz.

```
def combine_trends(row):

    scores = {"BULLISH": 1, "NEUTRAL": 0, "BEARISH": -1}
    total = scores[row["trend"]] + scores[row["macd_trend"]] +
       scores[row["rsi_trend"]]

    if total >= 2:
        return "BULLISH"
    elif total == 1:
        return "SOMEWHAT BULLISH"
    elif total == -1:
        return "SOMEWHAT BEARISH"
    elif total <= -2:
        return "BEARISH"
    else:
        return "NEUTRAL"

df["overall"] = df.apply(combine_trends, axis=1)
```

Wir zeigen im folgenden nur die Grafik und den tabellarischen Vergleich für Apple, da sich bei beiden Aktien ein ähnliches Bild zeigt. In der folgenden Grafik sind die grünen Intervalle die Zeiträume, in denen die Strategie ein Kaufsignal generiert hat (`BULLISH`), die hellgrünen Intervalle die Zeiträume, in denen die Strategie ein schwaches Kaufsignal generiert hat (`SOMEWHAT BULLISH`) und die gelben Intervalle die Zeiträume, in denen die Strategie ein schwaches Verkaufssignal generiert hat (`SOMEWHAT BEARISH`):

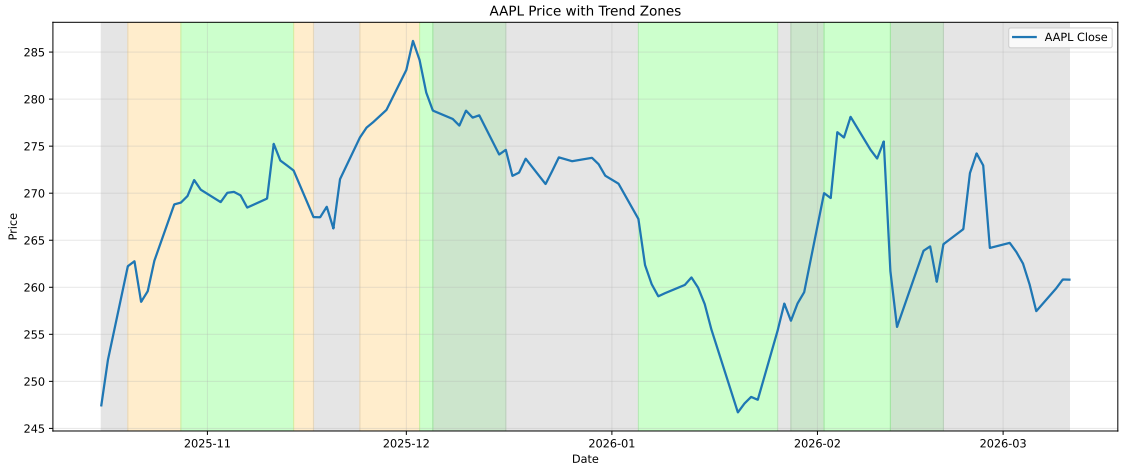

Abbildung 4.3: Generierte Signale für Apple mit differenzierter Interpretation

Um die Effektivität dieser differenzierteren Interpretation zu evaluieren, betrachten wir eine verbesserte Trading-Strategie: Wir kaufen nur bei echten `BULLISH`-Signalen und verkaufen erst, wenn der Trend zu `NEUTRAL`, `SOMEWHAT BEARISH` oder `BEARISH` wechselt. Während `SOMEWHAT BULLISH`-Phasen halten wir die Position. Die folgende Tabelle zeigt die Ergebnisse dieser Strategie für Apple:

Tabelle 4.3: AAPL Holding Strategie für die letzten 100 Datenpunkte (Buy-and-Hold: 5,40 %)

Trade	Kauf	Verkauf	Kaufpreis	Verkaufspreis	Gewinn (%)
1	05.12.2025	16.12.2025	$278,78	$274,61	–1,50 %
2	28.01.2026	20.02.2026	$256,44	$264,58	+3,17 %
Summe					**+1,68 %**

4.1.3 Fazit

Auch damit zeigen sich keine signifikant besseren Ergebnisse im Vergleich zu Buy-and-Hold, weshalb wir diesen Ansatz als naiv bezeichnen. Diese Strategie hat also nicht die Markteffizienztheorie widerlegt, da sie nicht in der Lage war, durch kurzfristige Kursbewegungen Gewinne zu erzielen, während Buy-and-Hold eine durchweg positive Rendite erzielt hat. Es gibt also keinen Grund, diesen Ansatz weiter zu verfolgen, weshalb wir uns im nächsten Abschnitt einem anderen Ansatz zuwenden werden.

4.2 Historische Optimierung technischer Signale

Im vorigen Abschnitt hat sich gezeigt, dass ein einfaches Mehrheitsvotum aus gleichgewichteten Indikatoren Buy-and-Hold kaum schlagen kann. Eine mögliche Erklärung dafür ist, dass Aktien unterschiedlich auf technische Indikatoren reagieren. So kann es zum Beispiel je nach Sektor, Volatilitätsprofil oder Handelsverhalten sein, dass der *RSI* bei einer Aktie zuverlässige Signale liefert, während bei einer anderen der *MACD* deutlich besser funktioniert [6][5, S. 996]. Wir wollen daher untersuchen, ob sich durch eine aktienspezifische Gewichtung der Indikatoren bessere Ergebnisse erzielen lassen. Konkret stellen wir uns die Frage: Lässt sich anhand historischer Daten eine optimale Gewichtung finden, die auch in der Zukunft funktioniert?

4.2.1 Experimentelles Setup

Im Vergleich zum naiven Ansatz, bei dem wir nur Nvidia und Apple betrachtet haben, wollen wir diesmal mehr Aktien aus unterschiedlichen Branchen analysieren, um nicht nur Technologieaktien zu betrachten. Wir haben uns für folgende zehn Aktien entschieden: *AEP, CAT, CVX, DE, DUK, FCX, KO, PEP, PG* und *XOM*.

Zusätzlich zu den drei Indikatoren aus dem naiven Ansatz haben wir als viertes Signal die *BBands* (Bollinger Bands) aufgenommen. Dieses Set aus vier Indikatoren deckt die wichtigsten Kategorien der technischen Analyse ab: Trendfolge, Momentum, Überkauft-/Überverkauft-Signale sowie Volatilität [12]. Jedes Signal wird als diskrete Zahl in $\{-1, 0, +1\}$ kodiert, wobei $+1$ einem Kaufsignal (`BULLISH`), -1 einem Verkaufssignal (`BEARISH`) und 0 einem neutralen Signal entspricht.

Die Daten haben wir in zwei Zeiträume aufgeteilt:

- **Trainingsdaten:** 01.01.2000 – 31.12.2020
- **Testdaten:** ab 01.01.2021

Auf den Trainingsdaten suchen wir die optimale Gewichtung für jede Aktie. Anschließend testen wir, ob diese Gewichtung auf den Testdaten noch funktioniert.

4.2.2 Gewichtetes Ensemble und Grid Search

Die vier Signale kombinieren wir zu einem gewichteten Ensemble-Score:

$$s_t = w_1 \cdot \sigma_{\text{MA},t} + w_2 \cdot \sigma_{\text{MACD},t} + w_3 \cdot \sigma_{\text{RSI},t} + w_4 \cdot \sigma_{\text{BBands},t} \tag{4.1}$$

wobei $w_i \geq 0$ und $\sum_{i=1}^{4} w_i = 1$. Liegt der Score s_t über einem Schwellenwert $\theta = 0{,}20$, gilt das als Kaufsignal; liegt er unter $-\theta$, als Verkaufssignal. Der Bereich dazwischen wird als neutrale Zone gewertet, um bei widersprüchlichen Signalen unnötige Positionswechsel zu vermeiden.

Die optimalen Gewichte suchen wir mit einer **Grid Search**: Wir testen alle möglichen Gewichtskombinationen in 0,1-Schritten, die die Bedingung $\sum w_i = 1$ erfüllen. Das ergibt 286 Kombinationen pro Aktie. Für jede Kombination berechnen wir die Gesamtrendite auf den Trainingsdaten und wählen die Kombination mit der höchsten Rendite aus:

```
# Gewichtsgitter aufbauen (Schrittweite 0.1, Summe = 1)
weight_grid = build_weight_grid(step=0.1)  # 286 Kombinationen

# Fuer jede Aktie: beste Gewichtung auf Trainingsdaten suchen
best_w, best_train_total = None, -inf
for w in weight_grid:
    _, metrics = evaluate_weighting(train_data, w, theta=0.20)
    if metrics['total'] > best_train_total:
        best_train_total = metrics['total']
        best_w = w
```

4.2.3 Ergebnisse

Tabelle 4.4 zeigt die Ergebnisse auf den Trainingsdaten von 2000 bis 2020.

Tabelle 4.4: Grid Search Trainingsdaten (2000–2020): Strategie vs. Buy-and-Hold

Ticker	Strategie	Buy-and-Hold	Strategie > B&H
CAT	986,7 %	1160,2 %	×
XOM	276,4 %	80,6 %	✓
CVX	1106,6 %	313,2 %	✓
DE	5931,1 %	1687,6 %	✓
FCX	6857,8 %	324,2 %	✓
AEP	293,1 %	534,6 %	×
PEP	564,4 %	571,6 %	×
DUK	356,2 %	423,5 %	×
KO	298,1 %	218,6 %	✓
PG	320,9 %	310,7 %	✓
Anteil			**6 von 10 (60 %)**

Tabelle 4.5 zeigt dieselben Aktien auf den Testdaten ab 2021, also dem Zeitraum, den die Strategie nicht gesehen hat.

Tabelle 4.5: Grid Search Testdaten (ab 2021): Strategie vs. Buy-and-Hold

Ticker	Strategie	Buy-and-Hold	Strategie > B&H
CAT	246,4 %	362,8 %	×
XOM	148,7 %	334,1 %	×
CVX	122,7 %	166,3 %	×
DE	94,5 %	139,6 %	×
FCX	79,4 %	139,6 %	×
AEP	68,5 %	93,2 %	×
PEP	57,6 %	30,9 %	✓
DUK	32,6 %	73,2 %	×
KO	25,2 %	75,3 %	×
PG	4,7 %	32,2 %	×
Anteil			**1 von 10 (10 %)**

Auf den Trainingsdaten schlägt die Strategie bei **6 von 10 Aktien** Buy-and-Hold (Tabelle 4.4). Auf den Testdaten gelingt das nur noch bei **1 von 10 Aktien** (Tabelle 4.5). Abbildung 4.4 zeigt die Equity-Kurven aller zehn Aktien auf den Testdaten im Vergleich zu Buy-and-Hold. Bei fast allen Titeln verläuft die Strategie-Kurve unterhalb von Buy-and-Hold, was den Einbruch der Generalisierungsleistung deutlich sichtbar macht.

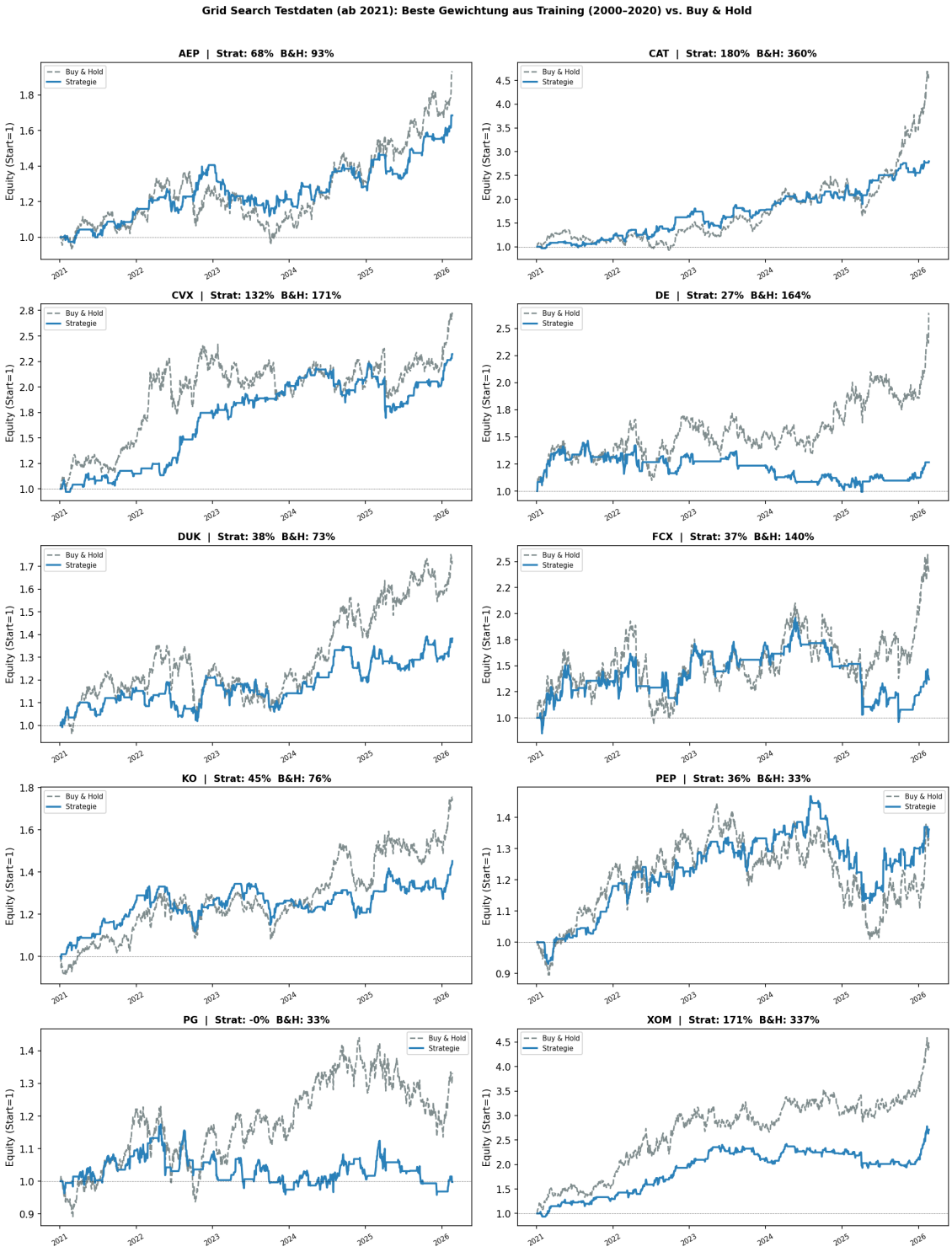

Abbildung 4.4: Equity-Kurven auf den Testdaten (ab 2021): beste Gewichtung aus dem Training (2000–2020) vs. Buy-and-Hold

In den Kurven zeigt sich, dass die Strategie tendenziell in fallenden Phasen investiert ist und steigende Phasen verpasst. Das liegt daran, dass die verwendeten Indikatoren wie RSI und Bollinger Bands erst dann ein Kaufsignal erzeugen, wenn der Kurs bereits gefallen ist und als überverkauft gilt. In einem Aufwärtstrend feuern diese Signale kaum, weshalb die Strategie große Teile der Aufwärtsbewegung verpasst. Gleichzeitig ist sie in Phasen erhöhter Volatilität investiert, weil der Kurs dort häufig die Schwellenwerte der

Indikatoren überschreitet. Das Ergebnis ist eine Strategie, die Verlustphasen mitmacht und Gewinnphasen verpasst.

4.2.4 Fazit

Der Einbruch von 60 % auf 10 % zwischen Training und Test zeigt, dass die gefundenen Gewichtungen auf den Testdaten kaum noch funktionieren. Die Grid Search findet zwar für jeden Trainingszeitraum eine gute Lösung, aber diese passt sich zu stark an die spezifischen Muster der Vergangenheit an und generalisiert schlecht auf neue Daten.

Damit lässt sich festhalten, dass auch eine datengetriebene, aktienspezifische Gewichtung nicht ausreicht, um Buy-and-Hold zuverlässig zu schlagen. Im nächsten Abschnitt wollen wir reflektieren, was das grundsätzlich über technische Indikatoren aussagt.

4.3 Von Signalen zu Regimen

Beide Analysen aus den vorigen Abschnitten kommen zu dem Ergebnis, dass sie nicht in der Lage sind, Buy-and-Hold auf ungesehenen Daten zuverlässig zu schlagen. Weder das einfache Voting aus dem naiven Ansatz noch die historisch optimierten Gewichte. Das wirft die Frage auf, warum technische Indikatoren so schlecht darin sind, präzise Kauf- und Verkaufszeitpunkte zu finden, und ob die Analyse technischer Indikatoren überhaupt beim Aktientrading helfen kann.

4.3.1 Warum technische Indikatoren keine guten Ein- und Ausstiegssignale liefern

Technische Indikatoren wie der MACD, der RSI oder der gleitende Durchschnitt sind alles mathematische Funktionen, die aus vergangenen Kursdaten berechnet werden. Ein Signal entsteht daher immer erst, nachdem eine Kursbewegung bereits begonnen hat. Bis ein Indikator einen Trend erkennt und ein Kaufsignal generiert, ist ein Teil der Bewegung bereits vorbei [6]. Dieses Problem konnten wir bereits bei der naiven Analyse beobachten: Kauf- oder Verkaufssignale kamen meist erst weit nachdem die Trendwende stattgefunden hatte. Dieses Problem lässt sich auch nicht durch eine bessere Gewichtung oder mehr Indikatoren lösen.

Aus diesem Grund ist auch unsere Grid Search gescheitert. Im Training wurden zwar die optimalen Gewichte für den Zeitraum 2000 bis 2020 gefunden, aber diese Gewichte funktionierten auf den Testdaten ab 2021 kaum noch, weil sich die Marktbedingungen verändert hatten. Doch obwohl sich technische Indikatoren nicht eignen, um zuverlässige Handelssignale zu erzeugen, haben wir uns dennoch dazu entschieden, ihre Einzelsignale in der App weiterhin anzuzeigen, damit der Nutzer sie selbst interpretieren kann.

4.3.2 Technische Indikatoren zur Risikoeinschätzung

Auch wenn wir festgestellt haben, dass technische Indikatoren keine zuverlässigen Kauf- und Verkaufssignale liefern können, findet man in der Literatur häufig, dass sie dennoch verwendet werden. Technische Indikatoren eignen sich laut Literatur dazu, den aktuellen Marktzustand einzuschätzen, also ob gerade ein Aufwärts- oder Abwärtstrend vorliegt und ob die Volatilität hoch oder niedrig ist [8]. Diese Informationen werden dabei nicht verwendet, um den genauen Zeitpunkt einer Trendwende auszumachen, sondern um den aktuellen Zustand zu bestimmen.

Diese Idee führt zum Konzept der **Marktregime**. Statt zu versuchen, mit einem Signal genau den richtigen Kaufzeitpunkt zu treffen, teilt man den Markt in verschiedene Zustände ein und passt die eigene Risikobereitschaft daran an. In einem Bullenmarkt mit niedriger Volatilität investiert man mehr, in einem Bärenmarkt mit hoher Volatilität reduziert man die Position oder steigt ganz aus. Das Ziel ist dabei nicht mehr, Buy-and-Hold zu schlagen, sondern Verluste in schlechten Marktphasen zu vermeiden und in guten Phasen verlässlich dabei zu sein. Giner und Zakamulin zeigen, dass Märkte tatsächlich in klar unterscheidbaren Zuständen operieren und dass diese Zustände nicht zufällig verteilt sind [8].

Im nächsten Abschnitt schauen wir uns an, wie wir diesen Ansatz konkret umgesetzt haben.

4.4 Umsetzung des Regime-Ansatzes

Im vorigen Abschnitt haben wir argumentiert, dass technische Indikatoren besser dazu geeignet sind, den aktuellen Marktzustand zu beschreiben als genaue Kauf- und Verkaufszeitpunkte zu finden. In diesem Abschnitt zeigen wir, wie wir diesen Ansatz konkret umgesetzt haben und was die Ergebnisse sagen.

4.4.1 Der Regime-Detektor

Angelehnt an Pillai et al. [12] haben wir einen Detektor entwickelt, der den Markt zu jedem Zeitpunkt in einen von vier Zuständen einteilt. Die zwei Dimensionen sind Trend und Volatilität.

Den Trend bestimmen wir über den gleitenden Durchschnitt der täglichen Renditen der letzten 60 Handelstage. Ist dieser Durchschnitt positiv, gilt die Aktie als im Aufwärtstrend (*Bull*), sonst im Abwärtstrend (*Bear*).

Die Volatilität messen wir mit dem NATR. Wir glätten den NATR über ein 10-Tage-Fenster und vergleichen ihn mit dem 70. Perzentil der letzten 252 Handelstage. Liegt die geglättete Volatilität über diesem Schwellenwert, gilt das Regime als *HighVol*, andernfalls als *LowVol*.

Die Berechnung des Perzentils sieht wie folgt aus [13]:

$$
\begin{aligned}
n = \text{Anzahl Messwerte},\ p = \text{Perzentil},\ k = (n \cdot p) \\
xp = x(\lceil k \rceil),\ \text{wobei } k \notin \mathbb{N} \\
xp = \frac{1}{2} \cdot (x(k) + x_{(k+1)}),\ \text{wenn } k \in \mathbb{N}
\end{aligned}
\tag{4.2}
$$

Wenn der berechnete Wert für k keine ganze Zahl ist, wird der Wert auf die nächst größere ganze Zahl $\lceil k \rceil$ (obere Gaußklammer) aufgerundet. Das bedeutet, dass der k-te Wert in der Liste das Perzentil ist. Wenn k eine ganze Zahl ist, wird der Wert an der Stelle k und der Wert an der Stelle k+1 gemittelt. Dieser Wert ist dann das Perzentil.

Aus den zwei Dimensionen ergeben sich vier Regime:

- **Bull_LowVol**: Aufwärtstrend mit niedriger Volatilität
- **Bull_HighVol**: Aufwärtstrend mit hoher Volatilität
- **Bear_LowVol**: Abwärtstrend mit niedriger Volatilität
- **Bear_HighVol**: Abwärtstrend mit hoher Volatilität

Um schnelle Regimewechsel durch kurzfristige Schwankungen zu vermeiden, haben wir eine Hysterese eingebaut. Ein neues Regime wird erst dann akzeptiert, wenn es drei Tage in Folge bestätigt wird.

4.4.2 Exposure-Strategie im Backtest

Um den Regime-Detektor zu evaluieren, brauchen wir eine Strategie, die die Regime-Informationen konkret umsetzt. Der naheliegendste Ansatz wäre, bei einem Bull-Regime vollständig zu kaufen und bei einem Bear-Regime vollständig zu verkaufen. Pillai et al. zeigen jedoch, dass solche Strategien genau dann scheitern, wenn sich der Markt gerade verändert, also zum Beispiel beim Übergang von einem ruhigen Aufwärtstrend in eine volatile Phase [12].

Butt et al. zeigen, dass es besser funktioniert, die Positionsgröße an das aktuelle Marktrisiko anzupassen, also in sicheren Phasen mehr zu investieren und in riskanten Phasen weniger [4]. Man ist also nicht einfach drin oder draußen, sondern immer mit einer zum Regime passenden Menge investiert. Wir weisen daher jedem der vier Regime eine feste Investitionsquote zu (Tabelle 4.6).

Tabelle 4.6: Exposure-Werte je Regime im Backtest

Regime	**Exposure**	**Begründung**
Bull_LowVol	0,9	Sicherstes Regime, fast vollständig investiert
Bull_HighVol	0,5	Aufwärtstrend, aber erhöhtes Risiko durch Volatilität
Bear_LowVol	0,2	Abwärtstrend, kleine Position für mögliche Erholung
Bear_HighVol	0,0	Schlechtestes Regime, kein Kapital im Markt

Im Bull_LowVol-Regime investieren wir 90 % des Kapitals, in allen anderen Regimen entsprechend weniger. Im Bull_HighVol-Regime halbieren wir die Position auf 50 %, weil trotz positivem Trend die erhöhte Volatilität das Verlustrisiko deutlich steigert. Im Bear_LowVol-Regime halten wir noch 20 %, weil der Abwärtstrend bei niedriger Volatilität weniger aggressiv verläuft und eine Trendwende möglich ist. Im Bear_HighVol-Regime gehen wir komplett aus dem Markt, da gleichzeitig Abwärtstrend und hohe Volatilität auftreten. Das ist typisch für Crashphasen und das einzige Regime, in dem ein vollständiger Ausstieg sinnvoll ist.

4.4.3 Regime-Statistik

Wir haben die Strategie auf denselben zehn Aktien aus dem vorigen Abschnitt getestet. Über den gesamten Zeitraum ab 2000 ist Bull_LowVol das bei weitem häufigste Regime und tritt bei allen Aktien an rund 50 % aller Handelstage auf.

Wichtig ist auch, wie stabil die Regime sind. Abbildung 4.5 zeigt, dass ein Bull_LowVol-Regime im Schnitt 40 bis 48 Tage anhält, bevor es wechselt. Bear_HighVol-Phasen dauern durchschnittlich etwa 28 Tage. Die Hysterese trägt dazu bei, dass kurze Ausreißer nicht sofort einen Regimewechsel auslösen.

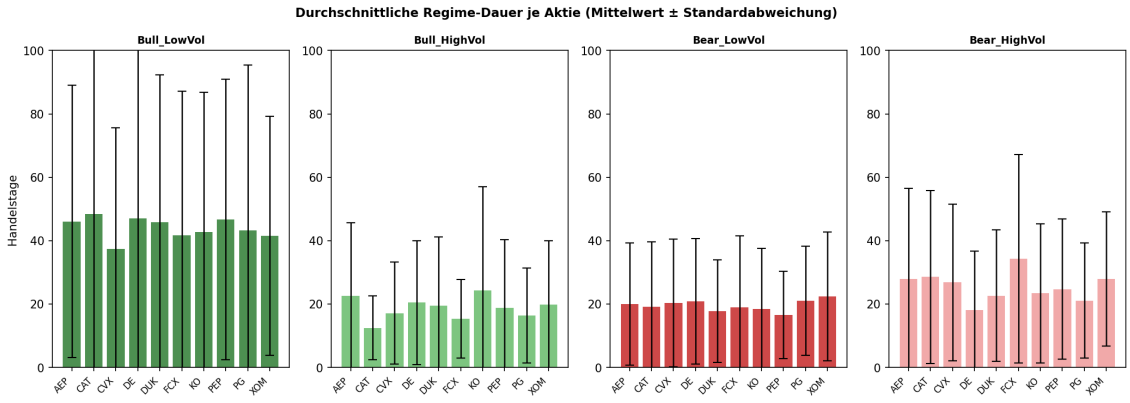

Abbildung 4.5: Durchschnittliche Regime-Dauer je Aktie (Mittelwert $\pm$ Standardabweichung)

Die Grafik zeigt, dass Bull_LowVol im Schnitt die längsten Phasen erzeugt, aber auch die größte Streuung aufweist. Das bedeutet, dass ruhige Aufwärtsphasen zwar lange dauern können, ihre Länge aber stark variiert.

4.4.4 Ergebnisse

Wie in den vorigen Abschnitten vergleichen wir die Strategie wieder mit Buy-and-Hold. Auch wenn das Ziel dieses Ansatzes nicht ist, Buy-and-Hold zu schlagen, ist der Vergleich sinnvoll, um einzuordnen, was die Strategie leistet und was sie kostet.

Beim Return sieht man dasselbe Bild wie zuvor: Buy-and-Hold liegt bei fast allen Aktien deutlich vorne. Die Strategie ist durch die reduzierte Exposure oft nur zu einem Bruchteil investiert und verpasst dadurch einen großen Teil der langfristigen Aufwärtsbewegung.

Diesmal betrachten wir aber zusätzlich den **maximalen Drawdown** (MDD). Der MDD misst den größten prozentualen Verlust vom jeweils höchsten Portfoliowert bis zum darauffolgenden Tiefpunkt. Er berechnet sich als:

$$\text{MDD} = \frac{\text{Tiefstwert} - \text{Höchstwert}}{\text{Höchstwert}} \tag{4.3}$$

Ein MDD von $-50\,\%$ bedeutet also, dass das Portfolio zu einem Zeitpunkt halb so viel wert war wie zuvor auf seinem Höchststand. Der MDD ist besonders geeignet, um eine Risikostrategie zu bewerten, weil er nicht die Rendite misst, sondern wie stark man im schlechtesten Fall verlieren kann.

Tabelle 4.7 zeigt die Ergebnisse des Backtests über den gesamten Zeitraum ab 2000.

Tabelle 4.7: Regime-Strategie vs. Buy-and-Hold: Gesamtzeitraum (ab 2000)

Ticker	Str. Return	B&H Return	Str. MDD	B&H MDD	Bessr. MDD
AEP	119,1 %	1113,4 %	−40,6 %	−62,7 %	✓
CAT	2119,6 %	6574,7 %	−32,6 %	−73,4 %	✓
CVX	126,9 %	870,8 %	−37,6 %	−55,8 %	✓
DE	874,6 %	4485,5 %	−41,9 %	−73,3 %	✓
DUK	127,6 %	625,3 %	−50,2 %	−71,9 %	✓
FCX	2359,0 %	1693,8 %	−54,7 %	−92,5 %	✓
KO	97,5 %	471,1 %	−32,2 %	−40,6 %	✓
PEP	139,7 %	639,9 %	−26,6 %	−40,4 %	✓
PG	148,1 %	1038,9 %	−26,7 %	−38,9 %	✓
XOM	125,5 %	632,3 %	−42,7 %	−62,4 %	✓
Anteil					**10 von 10 (100 %)**

Die Strategie erzielt bei allen zehn Aktien einen besseren maximalen Drawdown als Buy-and-Hold. Das ist der zentrale Erfolg dieses Ansatzes. Bei der absoluten Rendite schlägt die Strategie Buy-and-Hold nur bei einer Aktie (FCX).

Abbildung 4.6: Equity-Kurve der Regime-Exposure-Strategie vs. Buy-and-Hold für CAT (ab 2000)

Am Beispiel von CAT sieht man gut, was der Trade-off bedeutet. Die Strategie nimmt große Teile des langfristigen Aufwärtstrends nicht mit, weil sie in ruhigen Bullenmärkten nur zu 90 % investiert ist und in schlechteren Regimen noch weniger. In Phasen, in denen die Buy-and-Hold-Kurve stark einbricht, ist die Strategie dagegen oft kaum oder gar nicht investiert, was den geringeren maximalen Drawdown erklärt.

4.4.5 Fazit

Die Ergebnisse zeigen, dass die Strategie ihr Ziel erreicht. Bei allen zehn Aktien ist der MDD besser als bei Buy-and-Hold, der Return bleibt dafür in den meisten Fällen zurück. Weniger Exposure bedeutet eben auch weniger Gewinn.

Für Anleger, die nicht den maximalen Gewinn suchen, sondern starke Verlustphasen vermeiden wollen, ist ein Regime-basierter Ansatz damit eine sinnvolle Alternative zu Buy-and-Hold.

4.4.6 Grenzen des Regime-Ansatzes

Der Regime-Detektor arbeitet ausschließlich auf Basis von Preisdaten. Trend und Volatilität werden aus vergangenen Kursen berechnet, weshalb ein Regimewechsel erst erkannt wird, nachdem er sich bereits in den Preisen niedergeschlagen hat.

Dieser Nachteil zeigt sich besonders deutlich an Wendepunkten. Dreht ein Aufwärtstrend in einen Abwärtstrend, bleiben wir zunächst noch investiert, weil der gleitende Durchschnitt der letzten 60 Handelstage noch positiv ist. Erst wenn genug negative Tagesrenditen in das Fenster eingeflossen sind, wechselt das Regime auf Bear. In die andere Richtung gilt dasselbe: Nach einer Erholung ist man zunächst noch draußen, obwohl der Markt bereits wieder steigt.

Die Hysterese von drei Tagen verstärkt diesen Effekt bewusst, um Fehlsignale zu vermeiden, sorgt aber gleichzeitig dafür, dass echte Regimewechsel noch später erkannt werden.

Im nächsten Abschnitt untersuchen wir, ob News-Sentiment als Informationsquelle früher auf Trendwechsel hinweisen kann als preisbasierte Indikatoren.

4.5 News-Sentiment als frühzeitiger Indikator

Der Regime-Detektor aus dem vorigen Abschnitt basiert ausschließlich auf Preisdaten und teilt damit dasselbe grundlegende Problem wie alle technischen Indikatoren. Ein Regimewechsel wird erst erkannt, nachdem er sich bereits in den Kursen niedergeschlagen hat. Nachrichten hingegen sind oft früher verfügbar. Ereignisse wie Quartalszahlen, Übernahmen oder regulatorische Entscheidungen werden öffentlich bekannt, bevor der Markt vollständig darauf reagiert hat. Pillai et al. nutzen genau diesen Unterschied und zeigen, dass die Kombination aus technischen Signalen und News-Sentiment die Robustheit einer Handelsstrategie gegenüber Marktregimewechseln verbessert [12].

4.5.1 Datengrundlage

Für die Sentimentdaten verwenden wir dieselbe Alpha Vantage News & Sentiment API, die auch in der App eingesetzt wird [3]. Der Analysezeitraum beginnt ab 2017, da ein längerer Zeitraum eine deutlich größere Anzahl an API-Aufrufen erfordert hätte und wir den Mehrwert einer solchen Erweiterung als gering eingestuft haben. Die API liefert für jeden Artikel einen tickerspezifischen Sentimentwert im Bereich $[-1, 1]$ sowie einen Relevanzscore. Wir berücksichtigen nur Artikel mit einem Relevanzscore von mindestens 0,5 und bilden für jeden Handelstag den Mittelwert der verbleibenden Sentimentwerte. Liegt dieser Wert über $+0{,}15$, gilt das Tagessignal als *BULLISH*, liegt er unter $-0{,}15$ als *BEARISH*, andernfalls als *NEUTRAL*. An Tagen ohne relevante Artikel wird das Signal auf *NEUTRAL* gesetzt.

4.5.2 Kombinationsstrategie

News-Sentiment und Regime-Detektor eignen sich für unterschiedliche Marktsituationen. News sind besonders wertvoll, wenn konkrete Ereignisse eintreten, also Quartalszahlen, Übernahmen oder regulatorische Entscheidungen, die den Kurs einer Aktie direkt beeinflussen. In solchen Momenten trägt das Sentiment eine Information, die preisbasierte Indikatoren noch nicht erfassen können, weil der Markt erst im Begriff ist zu reagieren.

Gleichzeitig stehen Aktien häufig über längere Zeiträume nicht im Fokus der Medien. Eine Aktie, über die gerade nichts berichtet wird, kann trotzdem stetig steigen, wenn sie sich in einem stabilen Aufwärtstrend mit niedriger Volatilität befindet. Ausschließlich auf

News zu hören würde bedeuten, genau diese ruhigen Gewinnphasen zu verpassen. Der Regime-Detektor ist darauf ausgelegt, solche Situationen zu erkennen.

Die Kombinationsstrategie folgt daher einer klaren Priorität. Liegt ein News-Signal vor, überschreibt es das Regime, da es die aktuellere und direktere Information darstellt. Ist das News-Signal *BULLISH*, investieren wir vollständig, unabhängig vom aktuellen Regime. Ist es *BEARISH*, gehen wir aus dem Markt. Nur bei *NEUTRAL* übernimmt der Regime-Detektor mit den Exposure-Werten aus Tabelle 4.6. Das Signal vom Tag T bestimmt dabei die Position am Tag $T + 1$, um Lookahead-Bias zu vermeiden.

4.5.3 Ergebnisse

Wir testen alle vier Strategien auf denselben zehn Aktien ab 2017. Tabelle 4.8 zeigt den Gesamtertrag und den maximalen Drawdown im Vergleich.

Tabelle 4.8: Gesamtertrag und maximaler Drawdown: alle vier Strategien im Vergleich (ab 2017)

	Buy-and-Hold		**News**		**Regime**		**News+Regime**	
Ticker	**Ret.**	**MDD**	**Ret.**	**MDD**	**Ret.**	**MDD**	**Ret.**	**MDD**
AEP	187,3 %	−32,9 %	49,0 %	−10,0 %	51,7 %	−31,5 %	109,3 %	−22,2 %
CAT	893,5 %	−43,4 %	232,1 %	−18,0 %	264,1 %	−37,0 %	386,1 %	−29,8 %
CVX	131,5 %	−55,8 %	13,2 %	−40,5 %	31,5 %	−37,8 %	82,1 %	−34,4 %
DE	630,8 %	−37,9 %	47,3 %	−25,7 %	283,7 %	−28,7 %	323,6 %	−30,9 %
DUK	137,5 %	−37,4 %	−4,5 %	−22,9 %	15,3 %	−27,8 %	17,8 %	−27,1 %
FCX	386,9 %	−72,6 %	230,4 %	−14,9 %	143,0 %	−45,5 %	234,9 %	−37,3 %
KO	153,3 %	−37,0 %	77,9 %	−17,4 %	37,4 %	−20,0 %	75,7 %	−18,0 %
PEP	106,9 %	−30,3 %	−6,4 %	−31,1 %	26,5 %	−25,3 %	22,5 %	−30,8 %
PG	145,3 %	−23,8 %	29,1 %	−22,3 %	47,5 %	−25,6 %	78,0 %	−27,3 %
XOM	144,3 %	−61,0 %	132,9 %	−16,7 %	62,5 %	−33,2 %	139,0 %	−30,8 %

Beim Gesamtertrag liegt Buy-and-Hold erwartungsgemäß vorne, da die anderen Strategien nie vollständig investiert sind. Beim maximalen Drawdown zeigt sich, dass News+Regime bei acht von zehn Aktien den besten oder gleichwertigen Wert aller vier Strategien erreicht. Bei keiner Aktie ist die Kombination deutlich schlechter als beide Einzelstrategien gleichzeitig, sondern schneidet entweder besser ab als eine der beiden oder liegt zwischen ihnen.

Um die Unterschiede zwischen den Strategien anschaulicher zu machen, betrachten wir DE (Deere & Company) und XOM (ExxonMobil) exemplarisch als Equity-Kurven.

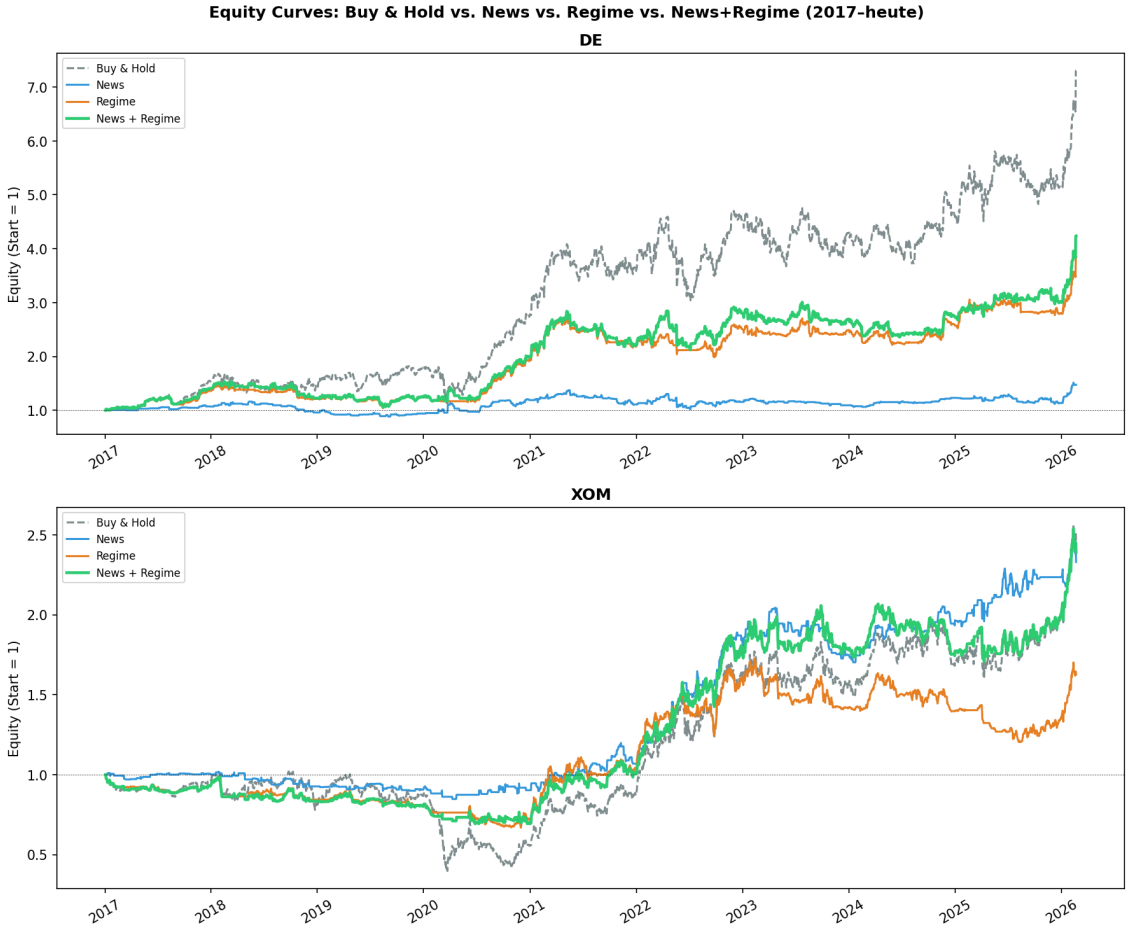

Abbildung 4.7: Equity-Kurven für DE und XOM: Buy-and-Hold, News, Regime und News+Regime im Vergleich (2017–heute)

Bei DE erzielt Regime-only 283,7 % Gesamtertrag, während News-only nur auf 47,3 % kommt. Das Regime erfasst die langen, stabilen Aufwärtsphasen des Titels gut, während News-only genau diese Phasen verpasst, weil DE über weite Strecken schlicht nicht im medialen Fokus steht. News+Regime erreicht mit 323,6 % den höchsten Gesamtertrag aller vier Strategien, allerdings zu einem leicht höheren MDD von −30,9 % gegenüber −28,7 % bei Regime-only.

Bei XOM ist das Gegenteil zu beobachten. News-only erreicht 132,9 % bei einem MDD von lediglich −16,7 %, während Regime-only nur auf 62,5 % kommt. Für einen Energietitel wie XOM tragen Sentimentdaten offenbar deutlich mehr Information als preisbasierte Regime-Signale. News+Regime erzielt 139,0 % bei einem MDD von −30,8 %, was gegenüber News-only einen etwas höheren Drawdown bedeutet, dafür aber auch deutlich mehr Rendite.

Der Unterschied im MDD zwischen News+Regime und News-only lässt sich strukturell erklären. Eine reine News-Strategie investiert nur dann, wenn aktiv bullishe Artikel vorliegen, und ist daher an vielen Tagen gar nicht im Markt. Das hält den MDD klein, bedeutet aber auch, dass ruhige Aufwärtstrends ohne Nachrichtenbegleitung vollständig verpasst werden. Das Regime schließt genau diese Lücke, indem es in stabilen Bull-Phasen auch ohne News investiert. Dadurch steigt die durchschnittliche Investitionsquote, was zwangsläufig auch das Risiko erhöht, in einem Kursrückgang investiert zu sein. News+Regime ist damit kein Ansatz, der den MDD maximiert, sondern ein Mittelweg, der höhere Renditen durch mehr Marktexposure erkauft, dabei aber den Drawdown

gegenüber Buy-and-Hold deutlich reduziert.

4.5.4 Fazit

Die Ergebnisse bestätigen, dass News-Sentiment und Regime-Detektor komplementäre Informationsquellen sind. Bei der Mehrheit der getesteten Aktien verbessert die Kombination den maximalen Drawdown gegenüber beiden Einzelkomponenten. Im nächsten Abschnitt übertragen wir diesen Ansatz auf ein Portfolio aus allen zehn Aktien und simulieren eine monatliche Kapitalentnahme, um die praktische Anwendbarkeit zu bewerten.

5 Evaluation

Die Analysen im vorigen Kapitel haben gezeigt, dass technische Indikatoren als direkte Handelssignale wenig geeignet sind, weil sie in der Praxis eine negative Vorhersagekraft besitzen. Besser eignen sie sich, um den aktuellen Marktzustand zu beschreiben, wie wir es im Regime-Detektor umgesetzt haben. News-Sentiment hingegen reagiert auf fundamentale Ereignisse, bevor der Markt vollständig darauf reagiert hat, und ergänzt das Regime dadurch sinnvoll. Die Kombination aus beiden Signalen liefert nach unserem Backtest bei der großen Mehrheit der getesteten Aktien einen besseren maximalen Drawdown als jede Einzelstrategie.

Ziel der Evaluation ist es nun, diese Strategie in einem realistischeren Szenario zu prüfen. Der Maßstab ist dabei nicht der maximale Gesamtertrag, den ohnehin Buy-and-Hold langfristig dominiert, sondern die Fähigkeit der Strategie, über mehrere Jahre hinweg konstant Gewinne zu erzielen und dabei starke Verlustphasen zu vermeiden.

5.1 Simulationsdesign

Wir simulieren ein Portfolio aus denselben zehn Aktien, das monatlich mit einem Startkapital von 10.000 EUR betrieben wird. Das Kapital wird gleichmäßig auf alle zehn Titel aufgeteilt, so dass jede Aktie einen Anteil von 1.000 EUR erhält. Am Ende jedes Monats wird der erzielte Gewinn entnommen beziehungsweise ein entstandener Verlust wieder aufgefüllt, sodass der nächste Monat stets mit genau 10.000 EUR beginnt. Die Idee dahinter ist, dass ein Anleger aus einem festen Grundkapital regelmäßig Erträge zieht.

Als Handelsstrategie verwenden wir die News+Regime-Kombination aus dem vorigen Kapitel. Ist das News-Signal für einen Titel *BULLISH* oder befindet sich dieser im Regime *Bull_LowVol* bei neutralem News-Signal, wird vollständig investiert. In allen anderen Fällen bleibt der entsprechende Anteil in Cash. Das Signal vom Vortag bestimmt die Position des Folgetages, um Lookahead-Bias zu vermeiden. Zusätzlich werden Transaktionskosten von 0,1 % pro Kauf oder Verkauf berücksichtigt. Als Vergleich dient Buy-and-Hold, bei dem das Kapital dauerhaft vollständig investiert bleibt und am Monatsende ebenfalls auf 10.000 EUR zurückgesetzt wird.

5.2 Ergebnisse

Abbildung 5.1 zeigt die monatlichen Gewinne und Verluste sowie die kumulierten Gewinne beider Strategien über den gesamten Simulationszeitraum von Januar 2017 bis Februar 2026 (110 Monate).

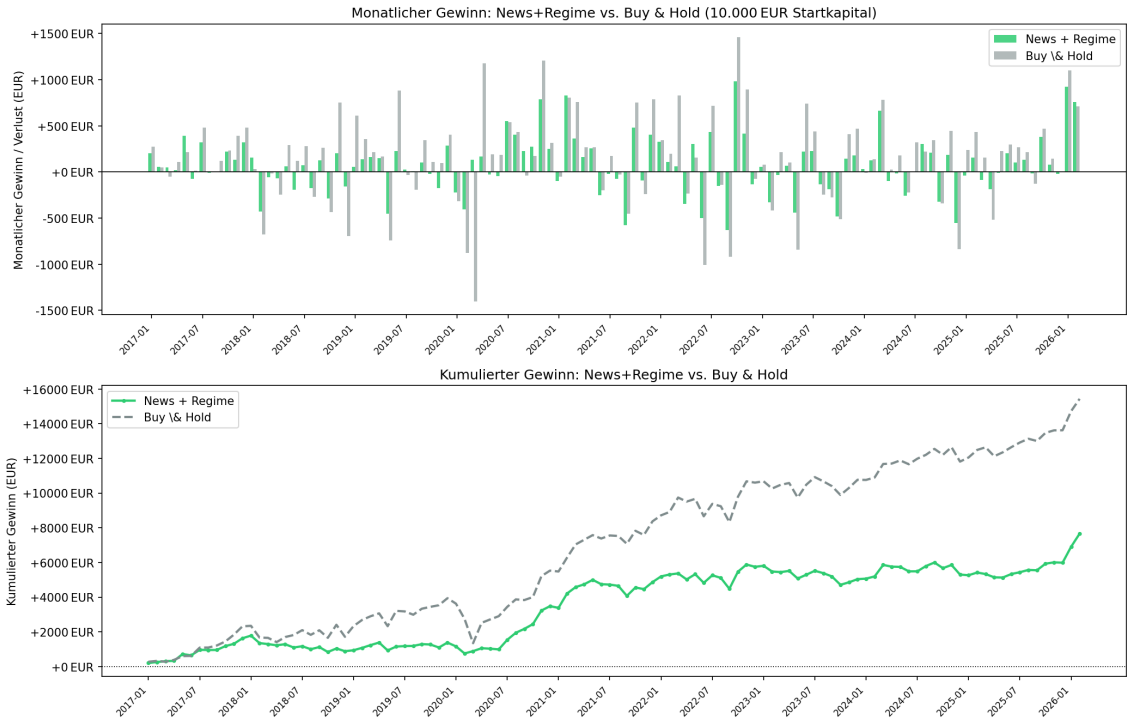

Abbildung 5.1: Monatlicher Gewinn und kumulierter Gewinn: News+Regime vs. Buy-and-Hold (10.000 EUR Startkapital, Januar 2017–Februar 2026)

News+Regime erzielt über den gesamten Zeitraum einen kumulierten Gewinn von 7.663 EUR in 66 von 110 Monaten (60 %). Buy-and-Hold kommt auf 15.437 EUR, ist aber in deutlich mehr Monaten stark im Minus. Der Vorteil der Strategie zeigt sich besonders in Crashphasen. Im März 2020, dem stärksten Einbruch im Simulationszeitraum, verliert Buy-and-Hold 1.401 EUR aus dem Startkapital, während News+Regime diesen Monat mit einem kleinen Plus von 177 EUR abschließt. Im Juni 2022, einem weiteren schwachen Monat, beträgt der Verlust bei Buy-and-Hold 1.005 EUR gegenüber 426 EUR bei der Strategie.

Der Unterschied in der Gleichmäßigkeit der Gewinne ist im unteren Teil der Abbildung gut erkennbar. Die kumulative Kurve von News+Regime steigt über den Zeitraum weitgehend stetig an, während Buy-and-Hold in mehreren Phasen stark zurückfällt. Ein Verlust von 1.400 EUR bedeutet in diesem Szenario, dass das Grundkapital im nächsten Monat zuerst wieder aufgefüllt werden muss, bevor überhaupt ein Gewinn entnehmbar ist.

5.3 Fazit

Die Simulation bestätigt den Befund aus der Analyse. News+Regime schlägt Buy-and-Hold bei der Gesamtrendite nicht, erzielt aber in 60 % aller Monate einen Gewinn und vermeidet starke Verlustmonate weitgehend. Das entspricht dem Ziel dieser Simulation, nämlich nicht den maximalen Gewinn herauszuholen, sondern möglichst jeden Monat im Plus zu bleiben.

6 Zusammenfassung

Im Rahmen dieser Arbeit wurde untersucht, inwieweit technische Indikatoren zur Ableitung belastbarer Handelssignale geeignet sind und wie sich kurs- und nachrichtenbasierte Informationen in einer Anwendung zur Entscheidungsunterstützung zusammenführen lassen. Zu diesem Zweck wurde mit dem *AlgoTrader* ein Prototyp entwickelt, der Kursdaten, technische Indikatoren, Fundamentaldaten und aktuelle Nachrichten aus externen Schnittstellen bündelt und in einer mit NiceGUI umgesetzten Benutzeroberfläche bereitstellt. Die Anwendung dient dabei nicht der automatischen Ausführung von Trades, sondern der strukturierten Aufbereitung relevanter Informationen für menschliche Nutzer.

Die empirische Analyse zeigt zunächst, dass einfache technische Handelssignale keine verlässliche Überrendite gegenüber einer Buy-and-Hold-Strategie liefern. Der naive Ansatz, der RSI, MACD sowie Golden Cross bzw. Death Cross in einem Mehrheitsvotum kombiniert, führte in den betrachteten Beispielen nicht zu überzeugenden Ergebnissen. Auch eine differenziertere Interpretation der Signalstärke verbesserte die Resultate nur geringfügig und blieb hinter Buy-and-Hold zurück. Damit bestätigt die Untersuchung, dass aus vergangenen Kursdaten abgeleitete Indikatoren für präzises Markt-Timing nur eingeschränkt geeignet sind.

Dieses Ergebnis wurde durch die anschließende historische Optimierung der Indikatorgewichtungen weiter untermauert. Zwar konnte eine aktienspezifische Grid Search auf den Trainingsdaten für 6 von 10 untersuchten Aktien bessere Ergebnisse als Buy-and-Hold erzielen, auf den Testdaten gelang dies jedoch nur noch in einem Fall. Die starke Verschlechterung zwischen Trainings- und Testphase zeigt, dass die historische Anpassung technischer Signale anfällig für Overfitting ist und sich gefundene Gewichtungen kaum auf neue Marktbedingungen übertragen lassen. Dies spricht gegen die Annahme, dass sich durch rein datengetriebene Kalibrierung ein robuster kurzfristiger Marktvorteil erzeugen lässt.

Auf Basis dieser Erkenntnisse wurde eine alternative Interpretation verfolgt: Technische Indikatoren wurden nicht länger als Timing-Instrument, sondern als Mittel zur Beschreibung von Marktregimen betrachtet. Der entwickelte Regime-Detektor klassifiziert den Markt anhand der Dimensionen Trend und Volatilität in vier Zustände und leitet daraus abgestufte Investitionsquoten ab. Die Backtests auf zehn Aktien zeigen, dass dieser Ansatz Buy-and-Hold in der Rendite meist nicht übertrifft, bei allen untersuchten Titeln jedoch einen besseren maximalen Drawdown erreicht.

Ergänzend wurde untersucht, ob News-Sentiment als zusätzliche Informationsquelle die Strategie verbessern kann. Da Nachrichten auf fundamentale Ereignisse reagieren, bevor sich diese vollständig im Kurs niedergeschlagen haben, eignet sich das Sentiment als früherer Indikator als rein preisbasierte Regime-Signale. Die Kombination aus News-Sentiment und Regime-Detektor erzielt bei allen getesteten Aktien einen deutlich besseren maximalen Drawdown als Buy-and-Hold und erreicht dabei höhere Gesamtrenditen als jede der beiden Einzelstrategien allein.

In einer abschließenden Portfoliosimulation über 110 Monate (Januar 2017 bis Februar 2026) wurde die News+Regime-Strategie in einem realistischeren Szenario mit 10.000 EUR Startkapital getestet. Die Strategie erzielte in 66 von 110 Monaten (60 %) einen positiven Gewinn und vermied dabei starke Verlustmonate: Im März 2020 verlor Buy-and-Hold

1.401 EUR aus dem Startkapital, während die Strategie diesen Monat mit einem kleinen Plus abschloss. Der kumulierte Gesamtgewinn lag mit 7.663 EUR unter dem von Buy-and-Hold (15.437 EUR), was dem Ziel der Simulation entspricht: nicht den maximalen Gewinn zu erzielen, sondern möglichst konstant im Plus zu bleiben.

Zusammenfassend zeigt die Arbeit, dass technische Indikatoren keine hinreichende Grundlage für verlässliche Kauf- und Verkaufssignale darstellen, sich aber sinnvoll zur Beschreibung von Marktregimen einsetzen lassen. In Kombination mit News-Sentiment entsteht ein Ansatz, der zwar Buy-and-Hold beim Gesamtertrag nicht schlägt, aber Verlustphasen systematisch reduziert und dabei über mehrere Jahre hinweg stabile Ergebnisse liefert. Mit dem AlgoTrader wurde ein Prototyp geschaffen, der diese Erkenntnisse praktisch aufgreift und technische Signale, Fundamentaldaten sowie Nachrichtenanalysen in einer gemeinsamen Oberfläche zusammenführt.

6.1 Ausblick

In diesem Abschnitt werden mögliche Erweiterungen und Verbesserungen des Projekts benannt, die in zukünftigen Arbeiten umgesetzt werden könnten.

Mobilversion des AlgoTraders

Aktuell ist der AlgoTrader nur als lokale Anwendung gedacht. Aufgrund der Nutzung von NiceGUI kann die Anwendung zwar als Webanwendung genutzt werden, jedoch ist die Benutzeroberfläche noch nicht für mobile Endgeräte optimiert. Ein weiteres Problem ist, dass die Funktionsweise von NiceGUI keine Progressive Web Apps (PWA)[1] ermöglicht, wodurch die Anwendung nicht als eigenständige App auf mobilen Geräten installiert werden kann. Zukünftige Arbeiten könnten sich daher mit der Optimierung der Benutzeroberfläche und dem Umbau der Anwendung beschäftigen, um eine mobile Version des AlgoTraders zu ermöglichen.

Community-Analysen

Die Analyse von Social-Media-Daten kann wertvolle Erkenntnisse über Markttrends liefern. So konnte gezeigt werden, dass das Handelsvolumen von GameStop während des Short Squeeze im Januar 2021 mit der Anzahl Posts darüber korrelierte [11]. Auch konnte gezeigt werden, dass Reddit dabei eine maßgebliche Rolle gespielt hat [11].

Um solche Trends zu erkennen, könnte der AlgoTrader um eine Funktion erweitert werden, die ausgewählte Communities, wie das Subreddit *r/wallstreetbets*, analysiert. Zwar warnen die Autoren von [11] davor, dass die Analyse von Reddit-Posts aufgrund des Vokabulars, der gegenteiligen Sentiments sowie des Volumens an Daten schwierig zu durchzuführen sein kann, jedoch könnten Fortschritte im Bereich des Text-Minings und der Einsatz von Large Language Models dabei helfen, diese Herausforderungen zu überwinden.

[1] `https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps`

Literatur

[1] *API Documentation | Alpha Vantage*. Letzter Zugriff am 11.03.2026. URL: https://www.alphavantage.co/documentation/.

[2] Dennis Kundisch Alexis Eisenhofer Tommy Jehmlich. *Algorithmic Trading*. Letzter Abruf: 14.03.2026. Apr. 2020. URL: https://www.gabler-banklexikon.de/definition/algorithmic-trading-70264/version-377279.

[3] *Alpha Vantage News API*. Letzter Zugriff am 03.03.2026. URL: https://www.alphavantage.co/documentation/#news-sentiment.

[4] Hilal Anwar Butt, James W. Kolari und Mohsin Sadaqat. „Market Volatility, Momentum, and Reversal: A Switching Strategy". In: *SSRN preprint* (2023). Available at SSRN: https://ssrn.com/abstract=4478316.

[5] Hans E. Büschgen. *Das kleine Börsen-Lexikon*. 23th ed. Handelsblatt-Bücher. Description based on publisher supplied metadata and other sources. Freiburg: Schäffer-Poeschel Verlag für Wirtschaft Steuern Recht GmbH, 2012. 11233 S. ISBN: 9783799265768.

[6] Terence Tai-Leung Chong, Wing-Kam Ng und Venus Khim-Sen Liew. „Revisiting the Performance of MACD and RSI Oscillators". In: *Journal of Risk and Financial Management* 7.1 (2014), S. 1–12. DOI: 10.3390/jrfm7010001.

[7] Deutsches Aktieninstitut. *Aktionärszahlen des Deutschen Aktieninstituts 2025*. Letzter Abruf: 14.03.2026. https://www.dai.de/fileadmin/user_upload/260113_Aktionaerszahlen_2025_Deutsches_Aktieninstitut.pdf, Jan. 2026.

[8] Javier Giner und Valeriy Zakamulin. „A regime-switching model of stock returns with momentum and mean reversion". In: *Economic Modelling* 122 (2023), S. 106237. DOI: 10.1016/j.econmod.2023.106237.

[9] MorningCrunch GmbH. *morningcrunch*. Letzter Zugriff am 08.03.2026. URL: https://morningcrunch.de/.

[10] *Golden Cross an der Börse*. Letzter Zugriff am 11.03.2026. URL: https://tradistats.com/golden-cross-an-der-boerse/.

[11] Kwansoo Kim, Sang-Yong Tom Lee und Robert J. Kauffman. „Social informedness and investor sentiment in the GameStop short squeeze". In: *Electronic Markets* 33.1 (Mai 2023). ISSN: 1422-8890. DOI: https://doi.org/10.1007/s12525-023-00632-9.

[12] Varun Narayan Kannan Pillai, Akshay Ajith und Sumesh K J. „Generating Alpha: A Hybrid AI-Driven Trading System Integrating Technical Analysis, Machine Learning and Financial Sentiment for Regime-Adaptive Equity Strategies". In: *arXiv preprint arXiv:2601.19504* (2026).

[13] Christian Reinboth. *Grundlagen der Statistik: Median, Perzentile und Modus*. Letzter Zugriff am 13.03.2026. Mai 2020. URL: https://wissenschafts-thurm.de/grundlagen-der-statistik-lagemasse-median-quartile-perzentile-und-modus/.

[14] Rene Rose, Hrsg. *Enzyklopädie der technischen Indikatoren*. 1. Aufl. Auf d. Schutzumschlag: Rene Rose (Hrsg.] Lt. Inhv. Abschnitte von versch. Autoren enth. ; Zus. zum HST auf d. Schutzumschlag. München: FinanzBuch-Verl., 2006. 767 S. ISBN: 9783898791045.

[15] *pandas - Python Data Analysis Library*. Letzter Zugriff am 11.03.2026. URL: `https://pandas.pydata.org/`.

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

Abkürzungsverzeichnis

EMA Exponential Moving Average
MACD Moving Average Convergence/Divergence
NATR Normalized Average True Range
RSI Relative Strength Index
SMA Simple Moving Average

Kolophon

Dieses Dokument wurde mit der LaTeX-Vorlage für Abschlussarbeiten an der htw saar im Bereich Informatik/Mechatronik-Sensortechnik erstellt (Version 2.25, August 2024). Die Vorlage wurde von Yves Hary und André Miede entwickelt (mit freundlicher Unterstützung von Thomas Kretschmer, Helmut G. Folz und Martina Lehser). Daten: (F)10.95 – (B)426.79135pt – (H)688.5567pt