Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

Fachbereich Informatik
Department of Informatik

Abschlussarbeit

im Bachelor-Studiengang Informatik

Vorhersage der Entwicklung von Aktienkursen am deutschen Markt

mit Hilfe von technischen Indikatoren und Techniken des maschinellen Lernens

von

Erik Autenrieth

Betreuer:	Dr. Marco Hülsmann
Zweitbetreuer:	Prof. Dr. Peter Becker

Eingereicht am: 22. August 2022

Erklärung

Erik Autenrieth
Mirecourtstr. 19
53225 Bonn

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbst angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Die Arbeit wurde bisher keiner Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

.....	.....
Ort, Datum	Unterschrift

Zusammenfassung

Die Aktienmarktprognose ist die Kunst, die Kursentwicklung von börsengehandelten Wertpapieren vorherzusagen. Da Aktienkurse multifaktoriellen Einflüssen wie der Konjunktur, Rezessionen, der politischen Lage und Umwelteinflüssen unterliegen, stellt eine zuverlässige Prognose eine Herausforderung dar. Diese Arbeit beschäftigt sich mit einer Vergleichsstudie von vier Techniken des maschinellen Lernens (ML) angewandt auf den deutschen Markt. Untersucht wurden zwei auf Entscheidungsbäumen basierte Ensemblemethoden, Random Forest (RF) und Extra Trees (ET), die mit häufig eingesetzten ML Modellen, Support Vector Machine (SVM) und einem künstlichen neuronalen Netz (KNN) verglichen wurden. Der deutsche Markt wird durch die vier größten Indizes DAX, MDAX, SDAX und TECDAX repräsentiert. Für die Indizes wurde ein Datensatz von 2013 - 2022 herangezogen und ausgewertet. 17 technische Indikatoren wurden als Eingabemerkmale basierend auf den Schlusskursen berechnet. Zudem dient der Rohöl- und Goldkurs sowie der deutsche IFO Geschäftsklimaindex als Eingabemerkmal. Für die Modellbewertung wurden die Leistungskennzahlen Genauigkeit sowie positiver und negativer Vorhersagewert verwendet. Die Forschungsergebnisse konnten zeigen, dass Ensemblemethoden unter den Vergleichsmodellen für eine mittelfristige Prognose am geeignetsten sind. Das ET Modell erreicht die höchste durchschnittliche Genauigkeit von über 87 % für eine 10, 15 und 20 Tages Prognose. Eine Untersuchung der aktuellen Studienlage indiziert, dass ein effektiver Performancevergleich der Vorhersagemodelle mit ML Modellen aus weiteren Publikationen nur möglich ist, falls Trainings- und Testdatensätze mit der gleichen Methodik erstellt worden sind.

Schlüsselwörter— maschinelles Lernen, Aktienkursvorhersage, deutscher Markt, technische Indikatoren, Merkmalextraktion

Abstract

Stock market forecasting is the art of predicting securities traded on the stock market such as stock prices. Since stock prices are subject to multifactorial influences such as economy, recessions, political situation and environmental influences reliable forecasting is challenging. This work consists of a comparative study of four machine learning techniques applied to the German market. Two tree-based ensemble methods are investigated namely Random Forest (RF) and Extra Trees (ET), which are compared with commonly used machine learning models such as Support Vector Machine (SVM) and an Artificial Neural Network (ANN). The German market is represented by the four largest indices DAX, MDAX, SDAX and TECDAX. A data set from 2013 - 2022 was used and evaluated for the indices. 17 technical indicators were calculated as input features based on closing prices. In addition, crude oil and gold prices as well as the German IFO index serve as input features. For the model evaluation the performance metrics of accuracy, precision and recall are used. The research results were able to show that ensemble methods are the most suitable among the comparative models for medium-term forecasting. The ET model achieved the highest average accuracy of over 87 % for 10, 15 and 20 day forecasting. An examination of the current state of studies shows that an effective performance comparison of the prediction models with machine learning algorithm from other publications is only possible if training and test data sets are created with the same methodology.

Keywords— machine learning, stock price prediction, German market, technical indicators, feature extraction

Inhaltsverzeichnis

Abbildungsverzeichnis vii

Tabellenverzeichnis viii

Abkürzungsverzeichnis ix

1. Einleitung 1
1.1. Einführung in die Aktienanalyse . . . 1
1.2. Problemstellung . . . 1
1.3. Zielsetzung . . . 2
1.4. Technologien . . . 3
1.5. Gliederung und Aufbau . . . 3

2. Wirtschaftliche Grundlagen 4
2.1. Marktmechanismen . . . 4
2.2. Deutsche Indizes und IFO Index . . . 4
2.3. Markteffizienzhypothese . . . 5
2.4. Fundamentalanalyse . . . 5
2.5. Technische Analyse . . . 6
2.6. Technische Indikatoren . . . 6

3. Stand der Forschung 8
3.1. Forschungsmethodik . . . 8
3.2. Literaturübersicht . . . 8

4. Modelle zur Aktienvorhersage 12
4.1. Maschinelles Lernen . . . 12
4.2. Support Vector Machine . . . 13
4.2.1. Lineare SVM . . . 13
4.2.2. Nicht-lineare SVM . . . 14
4.3. Entscheidungsbäume . . . 16
4.3.1. Qualitätsmaße zur Trennung . . . 17
4.3.2. Überanpassung und Beschneiden . . . 17
4.4. Random Forest . . . 18
4.4.1. Out-of-Bag Daten . . . 18
4.4.2. Bedeutung der Merkmale . . . 19
4.5. Extremely randomized Trees . . . 19
4.6. Künstliche neuronale Netze . . . 20
4.6.1. Aktivierungsfunktion . . . 21
4.6.2. Verlustfunktion . . . 21
4.6.3. Updatefunktion . . . 22

5. Datengrundlage 24
5.1. Extraktion der Indizes . . . 24
5.2. Extraktion der Merkmale . . . 25
5.2.1. Technische Indikatoren . . . 25
5.2.2. Fundamentale Daten . . . 26
5.3. Standardisierung der Merkmale . . . 27
5.4. Indextrends . . . 27
5.5. Performance Metriken . . . 28
5.5.1. Modellevaluation . . . 28
5.5.2. Genauigkeit in Prozent . . . 28
5.5.3. Konfusionsmatrix . . . 28

6. Vorhersagemethodik 30
6.1. Kursvorhersage . . . 30
6.2. Hyperparameteroptimierung . . . 32

7. Prognostizierte Ergebnisse 34
7.1. Performance der Merkmale . . . 34
7.2. Vorhersagen der Modelle . . . 36

7.3. Konfusionsmatrix der besten Modelle 37

8. Diskussion **40**

9. Ausblick **42**

10. Literaturverzeichnis **43**

A. Anhang **46**
A.1. Formale technische Indikatoren 46
A.2. DAX Statistik 47
A.3. Evaluation der Indikatoren 48
A.4. RF Parameter 49
A.4.1. Parameteroptimierung 49
A.4.2. Bedeutung der Merkmale 50
A.5. Ergebnisse 51
A.5.1. Vergleich der Genauigkeit 51
A.5.2. Mittelwert der Vorhersagegenauigkeiten 51
A.5.3. Vergleich mit allen Maßen 52
A.6. Skripte zur Aktienkursvorhersage 53

Abbildungsverzeichnis

1. DAX und SMA (10) mit Schnittpunkten 6
2. Verteilung der am häufig genutzten ML Algorithmen zwischen 2011 - 2021 9
3. Klassifikation von ML Modellen aus 138 untersuchten Artikeln 10
4. Prognosehäufigkeit nach Märkten aus 138 untersuchten Artikeln 11
5. SVM Klassifizierung mit trennender Hyperebene 14
6. Ein Entscheidungsbaum als hierarchische Datenstruktur 16
7. Beispiel eines Entscheidungsbaums 16
8. Modell eines künstlichen Neurons 20
9. Ein mehrlagiges neuronales Netz 21
10. Deutsche Indizes von 2013 - 2022 24
11. Gold und Rohöl Rohstoffkurse mit DAX Schlusskurs. 26
12. DAX und IFO Index. 27
13. Allgemeine Beschreibung einer Konfusionsmatrix 29
14. Vollständiger Vorhersageverlauf 30
15. Schema der Aktienkursvorhersage 31
16. Säulendiagramm über die Performance der Eingabemerkmale 35
17. Säulendiagramm der Vorhersagegenauigkeiten für den DAX 36
18. DAX Vorhersage 1 Tag mit SVM 37
19. DAX Vorhersage 5 Tage mit ET 38
20. DAX Vorhersage 10 Tage mit ET. 38
21. DAX Vorhersage 15 Tage mit ET 38
22. DAX Vorhersage 20 Tage mit ET 38

A.1. Unterteilung der DAX Trends mit steigendem und fallendem Trend 47
A.2. Verteilung der DAX Rendite 47
A.3. Entscheidungsbaum aus einem Random Forest zur DAX Vorhersage 49
A.4. Random Forest Fehlerrate bei der DAX Kursvorhersage 49
A.5. Die Wichtigkeit der Merkmale des RF bei der DAX 5 Tages Prognose 50
A.6. Die Wichtigkeit der Merkmale des RF bei der DAX 20 Tages Prognose 50

Tabellenverzeichnis

1. Anzahl der Eingabemerkmale der Indizes 24
2. Performance der Merkmale bei einer 5 Tages Prognose am DAX 34
3. Vergleich der Vorhersagemodelle am DAX 36
4. Performanceübersicht der Modelle am DAX 39
A.1. Technische Indikatoren mit angegebenen Referenzen 46
A.2. Performance der Merkmale bei einer 20 Tages Prognose am TECDAX 48
A.3. Performance der Merkmale bei einer 1 Tages Prognose am DAX 48
A.4. Vergleich der Vorhersagemodelle am MDAX 51
A.5. Vergleich der Vorhersagemodelle am SDAX 51
A.6. Vergleich der Vorhersagemodelle am TECDAX 51
A.7. Durchschnittliche Vorhersagegenauigkeit der Modelle 51
A.8. Performanceübersicht der Modelle am MDAX 52
A.9. Performanceübersicht der Modelle am SDAX 52
A.10. Performanceübersicht der Modelle am TECDAX 52

Abkürzungsverzeichnis

%R	Williams Prozent Range
Adam	Adaptive Moment Estimation
ADOSC	Accumulation/Distribution Oscillator
ANN	Artificial Neural Networks
API	Application Programming Interface
ARCH	Auto-Regressive Modelle mit bedingter Heteroskedastizität
ARIMA	Auto-Regressive Modelle mit integrierten gleitenden Durchschnitten
CART	Classification and Regression Trees
CCI	Commodity Channel Index
DAX	Deutscher Aktienindex
DJIA	Dow Jones Industrial Average
DNN	Deep Neural Networks
EMA	Exponential Moving Average
EMH	Markteffizienzhypothese
ET	Extra Trees
ETF	Exchange Traded Fund
FL	Fuzzy Logik
FN	Falsch Negativ
FP	Falsch Positiv
GA	Genetische Algorithmen
GARCH	Generalisierte Auto-Regressive Modelle mit bedingter Heteroskedastizität
HA	Hybride Ansätze
IFO	Information und Forschung
KGV	Kurs-Gewinn-Verhältnis
KNN	Künstliche neuronale Netze
LSTM	Long Short-Term Memory
MACD	Moving Average Convergence Divergence
MDAX	Mid-Cap DAX
ML	Maschinelles Lernen
NB	Naïver Bayes Klassifikator
NYSE	New York Stock Exchange
OBV	On Balance Volume
OOB	Out-of-Bag
RA	Regressions Algorithmen
RBF	Radiale Basisfunktion
RF	Random Forest
RN	Richtig Negativ
ROC	Rate of Change
RP	Richtig Positiv
RSI	Relative Strength Index
RWT	Random Walk Theorie
S&P 500	Standard & Poor's 500 Index
SAR	Stop and Reverse
SDAX	Small-Cap DAX

SGD	Stochastic Gradient Descent
SMA	Simple Moving Average
SVM	Support Vector Machine
SVR	Support Vector Regression
TECDAX	Technologie DAX
WMA	Weighted Moving Average
XGBoost	eXtreme Gradient Boosting

1. Einleitung

1.1. Einführung in die Aktienanalyse

Die Vorhersage von Aktienkursen ist ein zentrales Forschungsthema auf dem Gebiet der Investitionen und der Portfoliooptimierung. Privatanleger und institutionelle Anleger sind gleichermaßen daran interessiert, Gewinne aus Investitionen zu maximieren und gleichzeitig Verluste zu minimieren. Eine auf maschinellem Lernen (ML) basierte, präzise Prädiktion des Marktes kann einem Marktteilnehmer die Optimierung dieses Problems ermöglichen, indem höhere Renditen bei geringerem Risiko erwirtschaftet werden können (Weng u. a., 2017). Das Potential von Kapitalanlagen an den Börsen zeigt eine Statistik der Weltbank[1] zur Marktkapitalisierung aller börsennotierter Unternehmen. Dabei stieg die Summe der Marktkapitalisierungen von 32.42 Trillionen USD im Jahr 2008 auf 93.69 Trillionen USD im Jahr 2020. Die an der Börse gelisteten deutschen Unternehmen bilden im Jahr 2020 eine Marktkapitalisierung von 2.28 Trillionen USD, wodurch sich ein Anteil am Weltmarkt von etwa 2.4 % ergibt. Der deutsche Markt wird nahezu vollständig durch die vier großen Indizes, Deutscher Aktienindex (DAX), Mid-Cap DAX (MDAX), Small-Cap DAX (SDAX) und Technologie DAX (TECDAX) abgebildet. Eine Vorhersage dieser Indizes bezieht sich so direkt auf den gesamten deutschen Markt. Während Weltmärkte wie der amerikanische, chinesische und der indische Markt bereits umfangreich in Studien analysiert und prognostiziert wurden, besteht für den deutschen Markt aktuell noch keine hohe Publikationsdichte (Kumbure u. a., 2022).

Die aktuelle Studienlage kategorisiert die Vorhersagemodelle zur Aktienanalyse in drei Kategorien. Statistische Modelle wie Auto-Regressive Modelle mit integrierten gleitenden Durchschnitten (ARIMA) sind häufig genutzte Algorithmen. Mit dem Aufkommen der globalen Digitalisierung sind auch die Vorhersagemodelle in eine technologisch fortgeschrittene Ära eingetreten. Techniken des maschinellen Lernens wie Support Vector Machine (SVM) oder auf Entscheidungsbäumen basierte Algorithmen wie Random Forest (RF) werden immer beliebter (Rouf u. a., 2021). Die dritte Kategorie besteht aus einer ML Methode, dem Deep Learning, wobei künstliche neuronale Netze (KNN) zur Prädiktion verwendet werden. Auch das Bilden von hybriden Modellen aus unterschiedlichen Kategorien ist möglich (Prasad u. a., 2021).

1.2. Problemstellung

Die Herausforderungen bei der Aktienkursvorhersage sind vielfältig und anspruchsvoll. Finanzmärkte können als nicht-lineare Zeitreihen gesehen werden, welche eine hohe Komplexität beinhalten, nicht stationär und verrauscht sind. Zudem wird die Prognose durch zufällig auftretende Sprünge weiterhin erschwert (Patil u. a., 2021). Die Bildung von Aktienkursen hängt von einer Vielzahl an Faktoren ab. Neben Unternehmenskennzahlen wie dem Umsatz sind makroökonomische Daten wie Konjunkturzyklen, Zinsen sowie Rohstoffpreise zu berücksichtigen. Auch der wirtschaftliche Status und die prognostizierten Aussichten der Branche können Einfluss auf die Aktienkursbildung haben. Die Fundamentalanalyse bietet hier eine Investitionsmethode, bei der sowohl mikroökonomische Kennzahlen als auch makroökonomische Faktoren zur Prädiktion der Entwicklung von Aktienkursen einzelner Firmen berücksichtigt werden. Einen komplementären Ansatz stellt die technische Analyse von Börsenkursen dar. In diesem Zusammenhang werden Kurstrends aus dem vergangenen Kursverlauf abgeleitet und darauf basierend eine Prognose für den zukünftigen Verlauf erstellt (Hu u. a., 2015). Der gewählte Analyseansatz bestimmt so die Eingabemerkmale des zur Vorhersage gewählten Algorithmus. Beispiele für fundamentale Eingabeparameter sind die Gewinne eines Unternehmens zur mikroökonomischen Analyse oder die Rohstoffkurse für eine makroökonomische Analyse (Huang u. a., 2022). Wird ein charttechnischer Analyseansatz gewählt, bilden technische Indikatoren die Eingabemerkmale. Zudem besteht die Möglichkeit der Sentimentanalyse, in der Textdaten wie Artikel

[1] https://data.worldbank.org

oder Tweets zu einem börsennotierten Unternehmen in numerische Daten umgewandelt werden und so ebenfalls als Eingabemerkmale zur Verfügung stehen. Es ist noch eine Vielzahl weiterer Eingabemerkmale vorstellbar. Beispielsweise können wirtschaftliche Indizes wie der deutsche IFO Geschäftsklimaindex oder die Kurse verschiedener Weltindizes wie dem Dow Jones Industrial Average (DJIA) als Eingabeparameter verwendet werden. Neben der Wahl der Eingabeparameter ist der Vorhersagetyp des Algorithmus zu entscheiden. Während die Richtungsprognose eine binäre Ausgabe in Form von einer steigenden oder fallenden Richtung des Kurses liefert, erstellt die Niveauprognose eine Vorhersage der exakten Werte eines Kurses für ein gegebenes Zeitintervall in der Zukunft. Die Wahl des Vorhersagetyps entscheidet u.a. die Auswahl des Vorhersagealgorithmus. Im Bereich maschinellen Lernens, wäre ein Support Vector Machine (SVM) Modell für eine binäre Klassifizierung und ein auf Regression basiertes ML Modell wie Support Vector Regression (SVR) für eine Niveauprognose geeignet. Der Vorhersagezeitpunkt kann sich von einer Prognose der nächsten Stunden bis hin zu einer Prognose mehrerer Monate in der Zukunft erstrecken. So sind sowohl Kurzzeit- als auch Langzeitprognosen möglich. Die Wahl des Vorhersagezeitpunktes hängt vor allem von den Eingabemerkmalen ab, da diese sich meist über eine gewisse Zeitspanne interpretieren lassen (Rouf u. a., 2021).

Um eine erfolgreiche Aktienkursvorhersage zu erstellen, muss eine Vielzahl an Einflussgrößen berücksichtigt werden. Niveauprognosen erzielen zwar häufig hohe Genauigkeiten, leiden jedoch meist unter einer schlechteren Performance in der praktischen Anwendung (Thawornwong; Enke, 2004). Eine Trendprognose für steigende und fallende Kurse ist deshalb am vielversprechendsten. Das auf Entscheidungsbäumen basierende Extra Trees Modell hat in Aktienmarktuntersuchungen bessere Ergebnisse als das Random Forest Modell erzielt (Ampomah u. a., 2020). Ein Vergleich am deutschen Markt kann dieses Ergebnis bestätigen. Die Literatur bescheinigt einfachen künstlichen neuronalen Netzen (KNN) ähnliche Leistungen im Vergleich mit komplexeren Deep Learning Verfahren. In Gegenüberstellungen mit baumbasierten Modellen unterperformt ein KNN jedoch in der Regel (Ampomah u. a., 2020). SVM Modelle performen unterschiedlich gut, je nach gewähltem Vorhersagezeitraum (Rouf u. a., 2021). Ein Vergleich dieser Modellkombinationen auf dem deutschen Markt ist bisher nicht gegeben. Ein wichtiges Kriterium ist die Wahl des Vorhersagezeitraumes. Erfolgreiche Kurzzeitprognosen scheinen lukrativ, sind in der Praxis jedoch durch hohe Marktvolatilitäten schwer umzusetzen (Zhang u. a., 2018). Mittelfristige Prognosen von einigen wenigen Wochen sind vielversprechender (Rouf u. a., 2021). Technische Indikatoren und fundamentale Daten wurden bereits vielfältig in der Aktienmarktprognose verwendet. Ob eine ausgewählte Kombination dieser Indikatoren ebenfalls auf dem deutschen Markt zu einer Leistungssteigerung der Vorhersagemodelle führt, ist noch zu untersuchen. Bei der Zeitreihenanalyse eignen sich unterschiedliche Trainingsmethoden für ML Modelle. Die Trainingsmethode kann das Vorhersageergebnis maßgeblich beeinflussen. Ein Vergleich der unterschiedlichen Methoden ist für eine Evaluation der Vorhersageperformance notwendig. Für eine effektive Performancemessung ist die Wahl der Vergleichsmaße entscheidend. Eine Prüfung der Vergleichsmaße auf Nutzbarkeit für verschiedene Handelsstrategien stellt eine maximierte Renditeerwartung in Aussicht.

1.3. Zielsetzung

Es werden vier Modelle des maschinellen Lernens vorgestellt und zur Prädiktion der vier relevanten deutschen Aktienindizes DAX, MDAX, SDAX und TECDAX verwendet. Getestet werden Support Vector Machine (SVM), zwei auf Entscheidungsbäumen basierte Algorithmen, Random Forest (RF) und Extra Trees (ET) sowie ein künstliches neuronales Netz (KNN). Die Eingabemerkmale der Modelle bestehen aus bis zu 17 technischen Indikatoren und dem Schlusskurs der Aktie. Weitere Merkmale sind zwei Rohstoffkurse, der Gold- und Rohölschlusskurs und der deutsche IFO Geschäftsklimaindex. Die Prognose erfolgt trendbasiert für 1, 5, 10, 15 und 20 Handelstage. Es wird so eine Zeitspanne von ein bis vier Wochen prognostiziert. Die täglichen Schlusskurse der Indizes werden aus einer

Zeitspanne von neun Jahren (01. Juni 2013 - 01. Januar 2022) ermittelt. Der so ermittelte Datensatz bildet die Datengrundlage aus 2170 Handelstagen. Aus den Schlusskursen werden die technischen Indikatoren berechnet. Alle Eingabemerkmale werden standardisiert und in ein 70 % Trainingsdatensatz und 30 % Testdatensatz aufgeteilt. Ein Kursanstieg kann numerisch mit einer 1 klassifiziert werden und eine 0 kann für das Fallen des Kurses stehen. Der Eingabedatensatz erhält ein Etikett, indem jedes Datum zusätzlich mit 0 oder 1 versehen wird, je nachdem ob der Kurs in einer bestimmten Anzahl von Tagen höher oder niedriger steht. Die binären Vorhersagen der Modelle werden mittels der Genauigkeit in Prozent aus dem Testdatensatz evaluiert. Eine profitable Handelsstrategie kann bereits ab einer Genauigkeit von 60 % eines ML Modells etabliert werden (Novak; Velušček, 2016). Für die Einführung verschiedener Handelsstrategien wird der positive und negative Vorhersagewert zur weiteren Evaluation der Modelle hinzugezogen. Die Zielsetzung dieser Arbeit besteht somit aus der Beantwortung folgender Fragestellungen:

- Ist es grundsätzlich lohnend einen Aktienmarkt zu prognostizieren und welche Zeiträume sind hierfür am geeignetsten?
- Führt das Verwenden von technischen Indikatoren und weiteren fundamentalen Merkmalen wie Rohstoffkurse und dem IFO Index zu einer signifikanten Leistungsverbesserung der Modelle?
- Welches der vorgeschlagenen ML Modelle performt, angewandt auf den deutschen Indizes, am besten?
- Wie sind die Ergebnisse im Literaturvergleich einzuordnen?

1.4. Technologien

Für das Erstellen der Vorhersagemodelle sowie für die Datengewinnung wurde die Programmiersprache Python verwendet. Diese bietet umfangreiche Bibliotheken in den Bereichen Datenanalyse, maschinellen Lernen und Datenvisualisierung. Die Marktdaten der deutschen Indizes sowie der Rohstoffe Öl und Gold wurden über die Yahoo Finanzen[2] Programmierschnittstelle (API) extrahiert. Technische Indikatoren wurden mit Hilfe der freien Software-Bibliothek ‚Ta-Lib'[3] berechnet. Die Modelle SVM, RF und ET wurden mit der ‚scikit-learn' Bibliothek konstruiert und optimiert. Für die Entwicklung des KNN wurde die von Google entwickelte ‚TensorFlow' Bibliothek herangezogen.

1.5. Gliederung und Aufbau

Die wirtschaftliche Grundlage, um den deutschen Markt vorhersagen zu können, wird in Kapitel 2 gegeben. Die Frage, ob eine Aktienkursvorhersage überhaupt möglich ist, wird diskutiert, indem dafürsprechende Hypothesen verneinenden Theorien gegenübergestellt werden. Weiterhin werden sowohl technische Indikatoren als auch fundamentale Daten allgemein beschrieben. Ein Überblick der aktuellen Forschungslage im Bereich Aktienkursvorhersage mit maschinellem Lernen wird in Kapitel 3 gegeben. Es werden häufig verwendete Algorithmen vorgestellt und aufgeschlüsselt, welche Märkte bereits am häufigsten untersucht wurden. Die Vorhersagemodelle SVM, RF, ET und KNN werden formal in Kapitel 4 beschrieben. Die Extraktion der Indizedaten und der verwendeten Eingabemerkmale wird in Kapitel 5 erläutert. Des Weiteren werden die Evaluationsmetriken Genauigkeit, positiver und negativer Vorhersagewert formal beschrieben. Kapitel 6 stellt den methodischen Vorgang der Modellerstellung mit optimierten Hyperparametern vor. In Kapitel 7 werden zunächst die technischen Indikatoren mittels Genauigkeit evaluiert. Es folgt die Auswertung der prognostizierten Ergebnisse für kurz- und mittelfristige Prognosen des deutschen Marktes mittels Techniken des maschinellen Lernens. Eine Einschätzung der Prognoseergebnisse wird im Diskussionsteil, Kapitel 8, gegeben. Dabei werden die Vorhersagewerte in den Kontext der aktuellen Studienlage eingeordnet. Ein Ausblick auf mögliche zukünftige Forschungsarbeiten mit weiteren ML Modellen wird in Kapitel 9 gegeben.

[2] `https://de.finance.yahoo.com`

[3] `https://github.com/mrjbq7/ta-lib`

2. Wirtschaftliche Grundlagen

2.1. Marktmechanismen

Ein Finanzmarkt ermöglicht die Verbindung von privaten und institutionellen Anlegern mit sich finanzierenden Unternehmen sowie staatlichen Organisationen. Ein Kapitalaustausch erfolgt meist über Aktien oder Anleihen. Aktionäre können so an der wirtschaftlichen Entwicklung eines Unternehmens partizipieren und diesem gleichzeitig benötigte Finanzmittel zur Verfügung stellen. Die durch Kauf und Verkauf von Aktien entstehende Preisbildung spiegelt die Zukunftserwartung der an den Finanzmärkten teilnehmenden Organisationen und Personen wider. Finanzmärkte werden, neben Banken und Versicherungen, vor allem durch Börsen repräsentiert. Zusätzlich zu dem Handel von Wertpapieren wie Aktien ermöglichen Börsen den Handel mit Fondsanteilen, Devisen und Rohstoffen (Spremann; Gantenbein, 2014). Es besteht zudem die Möglichkeit auch Indizes über einen Exchange Traded Fund (ETF) zu erwerben. Dieser bildet den entsprechenden Index ab, indem anteilig die Einzelaktien des Index gekauft werden. Die weltweit größte Börse ist die amerikanische New York Stock Exchange (NYSE). Nach internationalen Börsenplätzen wie Tokyo, Hongkong und London stellt die Frankfurter Börse eine der weltgrößten Handelsplätze dar (Schuster, 2015).

Ergänzend zu dem Erwerb von Aktien bietet die Börse die Möglichkeit der Leerverkäufe bzw. ‚Shortselling'. Technisch wird das Setzen auf fallende Kurse mit Leerverkäufen realisiert, indem Aktien von Marktteilnehmern geliehen und anschließend verkauft werden. Anschließend werden diese nach dem Fall der Aktien zurückgekauft. Ein Gewinn entsteht mit der Differenz zwischen Verkaufskurs und dem Einkaufskurs bei Wiedereinstieg. Leerverkäufe ermöglichen neben weiteren Methoden wie Optionsscheinen eine Handelsstrategie zu etablieren, bei der Gewinne mit fallenden Kursen erzielt werden können (Schuster, 2015).

2.2. Deutsche Indizes und IFO Index

Die Frankfurter Wertpapierbörse, die die Computerbörse Xetra inkludiert, ist der größte Handelsplatz für die deutschen Aktienindizes. Der deutsche Raum wird vor allem durch den DAX repräsentiert, gefolgt von den Indizes MDAX, SDAX und TECDAX. Folgende Auflistung beschreibt die wichtigsten deutschen Indizes:

- DAX (Deutscher Aktienindex): Repräsentiert die 40 umsatzstärksten Unternehmen in Deutschland, darunter Unternehmen wie VW oder Bayer.
- MDAX (Mid-Cap DAX): Beinhaltet 50 mittelgroße Unternehmen, die nach Marktkapitalisierung auf die DAX Unternehmen folgen. Exemplarisch kann das Unternehmen Bechtle und die Commerzbank genannt werden.
- SDAX (Small-Cap DAX): Listet 70 kleine deutsche Unternehmen, wie beispielsweise Fielmann oder die Hornbach Holding AG.
- TECDAX (Technologie DAX): Der Index listet die 30 größten Technologiewerte, darunter die Deutsche Telekom und SAP. Unternehmen im TECDAX können auch in anderen deutschen Indizes gelistet werden (Schuster, 2015).

Die Kurse der Indizes werden, anteilig nach Marktkapitalisierung, aus den Einzelaktien gebildet (Schuster, 2015). Damit bilden die vier größten deutschen Indizes den deutschen Finanzmarkt nahezu vollständig ab.

Ein zentraler Indikator für die deutsche Wirtschaftslage ist der IFO Geschäftsklimaindex. Dieser wird monatlich aus Unternehmensumfragen zu der Beurteilung der aktuellen Geschäftslage sowie des erwarteten Geschäftsverlaufes gebildet. Die deutsche Konjunkturentwicklung kann so frühzeitig und zuverlässig numerisch ausgedrückt werden (Seiler;

Wohlrabe, 2013). In einer Studie von Hülsmann u. a. (2011) wurde neben weiteren Parametern der IFO Geschäftsklimaindex verwendet, um eine Absatzprognose des deutschen Automobilmarktes zu erstellen. Dabei wurde der IFO Index als relevantes Eingabemerkmal für unterschiedliche ML Algorithmen, darunter SVM und RF, beschrieben (Hülsmann u. a., 2011). Die Frage, ob der IFO Index auch Relevanz für ML Vorhersagen auf den deutschen Aktienmarkt besitzt, soll im weiteren Verlauf ausgearbeitet werden.

Vorhersagen von Finanzmärkten sollen Marktteilnehmern primär eine konsistente Renditemaximierung ermöglichen. Die Rendite ergibt sich so aus den Kapitalerträgen des angelegten Kapitals über einen bestimmten Zeitraum. Weiterhin ermöglichen präzise Vorhersagen eine Risikoverminderung, indem potenziell verlustreiche Aktienkäufe vermieden werden. Ob diese höheren und risikobereinigten Renditen überhaupt mit einer Marktanalyse erzielt werden können, ist weder bewiesen noch widerlegt. Es existieren jedoch mehrere theoretische Ansätze, die jeweils für eine Seite argumentieren (Kumbure u. a., 2022).

2.3. Markteffizienzhypothese

Die von Fama (1970) entwickelte Markteffizienzhypothese (EMH) antizipiert, dass Preise für Aktien bereits alle verfügbaren Informationen widerspiegeln. In den angenommenen, informationseffizienten Märkten würde ein aus Informationen generierter Vorteil keine Gewinnmaximierung ermöglichen. Die EMH ist verwandt mit der Random Walk Theorie (RWT), die davon ausgeht, dass sich Aktienkurse völlig zufällig bilden. Überproportionale Gewinne seien nicht möglich, da bereits geringe Informationsvorteile von einer Vielzahl an Marktteilnehmern ausgenutzt werden würden und sich so die Gewinnmöglichkeiten durch eine neue Preisbildung eliminieren würde (Lo, 2011). Die Markteffizienzhypothese unterteilt sich in drei unterschiedlich starken Formen:

- Die schwache Form: Sich in Zeitreihen befindende Informationen aus vergangenen Preisen können nicht für eine Vorhersage zukünftiger Kurse verwendet werden, da sich der Informationsgewinn bereits auf den aktuellen Kurs ausgewirkt hat. Eine überdurchschnittliche Rendite kann durch die technische Analyse von Zeitreihen nicht ermöglicht werden. Klassische Anlagemethoden wie die ‚Kaufen und halten' Strategie können somit nicht durch die technische Analyse optimiert werden (Fama, 1970).
- Die halbstarke Form: Alle öffentlich verfügbaren Informationen inklusive fundamentaler Daten zu Unternehmen ermöglichen einem Marktteilnehmer keine langfristig höheren Renditen.
- Die starke Form: Es existieren keine Informationen, die eine Maximierung der Renditen ermöglichen. Selbst Insiderinformationen können nicht zum Erzielen von höheren Gewinnen genutzt werden. Diese extreme Form wird jedoch von ihrem Erfinder als eher unwahrscheinlich erachtet (Fama, 1970).

Einige Forschungsergebnisse sprechen jedoch gegen die Markteffizienzhypothese, besonders gegen die schwache Form. In Untersuchungen konnten Über- und Unterreaktionen bei Finanzmärkten festgestellt werden. Auch kurzfristige Impulse oder langfristige Preisumkehrungen können als Marktanomalien gesehen werden, die zunächst nicht durch die Markteffizienzhypothese erklärt werden können (Kumbure u. a., 2022).

2.4. Fundamentalanalyse

Die Fundamentalanalyse steht im Kontrast zu der halbstarken Form der EMH. Basierend auf Unternehmenskennzahlen wie der Bilanz, der Marktkapitalisierung oder dem Kurs-Gewinn-Verhältnis (KGV) eines Unternehmens wird ein Aktienkurs prognostiziert, der diese Daten besser widerspiegelt. Es wird davon ausgegangen, dass sich der echte Kursverlauf dieser Bewertung über einen gewissen Zeitraum angleicht (Rouf u. a., 2021).

Weiterhin können für eine umfassendere Fundamentalanalyse auch makroökonomische Daten hinzugezogen werden. Der IFO Index sowie Rohstoffkurse wie der Gold- und Ölpreis stellen solche Datensätze dar. Falls der IFO Index sinkt und somit die deutsche Konjunktur schwächer wird, hat dies in der Theorie eine Auswirkung auf den Aktienkurs von DAX Unternehmen. Gleiches gilt für sinkende oder steigende Rohstoffpreise, von denen die Produktion der Unternehmen abhängig ist.

2.5. Technische Analyse

Die technische Analyse beschreibt eine Verfahrensmethode, bei der technische Indikatoren zur Vorhersage von Finanzzeitreihen verwendet werden. Basierend auf der Annahme, dass Kurstrends existieren, werden künftige Kursbewegungen mittels vergangener Daten prognostiziert. Ein Kurstrend kann dabei u.a. von psychologischen Faktoren abhängen. Wenn der Kurs einer Aktie bereits stark gestiegen ist, wächst die Gewinnerwartung der Kapitalanleger oft mit und ein positiver Trend entsteht (Kumbure u. a., 2022). Diese wiederkehrenden Muster können stochastisch gemessen und interpretiert werden. Die technische Analyse gründet auf der DOW Theorie, welche Finanzmärkten unterstellt, dass historische Trends existieren und diese in Zyklen wiederholt werden (Rouf u. a., 2021). Im Umgang mit ML Algorithmen werden bei der Aktienkursvorhersage häufig fundamentale und technische Analysemethoden kombiniert, um höhere Genauigkeitswerte zu erreichen (Patil u. a., 2021).

2.6. Technische Indikatoren

Als Werkzeug der technischen Analyse können technische Indikatoren erstellt werden, um Änderungen von Markttrends vorherzusagen. Durch das Existieren einer Masse an historischen Chartverläufen, steht der Entwicklung der Indikatoren eine große Datengrundlage zur Verfügung. Selbst traditionelle technische Indikatoren können angepasst von moderneren, mathematischen und statistischen Techniken verwendet werden (Colby; Meyers, 1988). Eine Basis der technischen Indikatoren bilden 140 unverarbeitete Indikatoren, die direkt dem Markt entnommen werden können. Diese beziehen sich zum größten Teil auf den Schlusskurs, Tageshöchstkurs, Tagestiefkurs und dem Volumen einer Aktie. Das Volumen beschreibt die Summe der pro Zeiteinheit gehandelten Aktien eines Unternehmens.

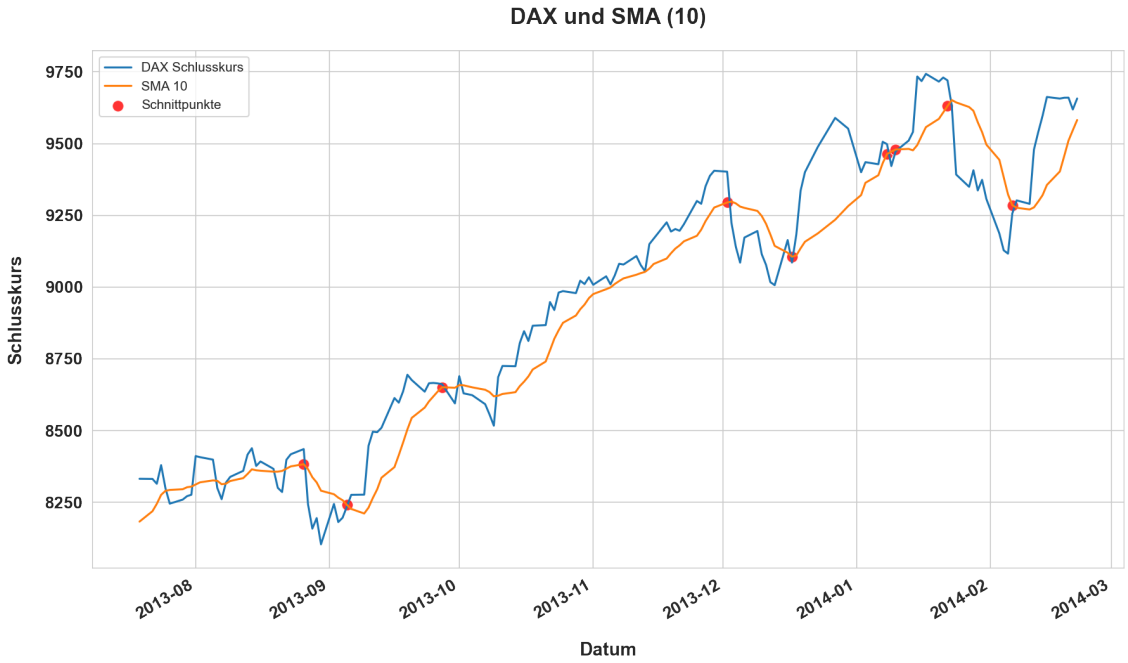

Abbildung 1: DAX und SMA (10) mit markierten Schnittpunkten.

Technische Indikatoren, die nicht zu der Basis zählen, werden meist aus deren Elementen wie dem Schlusskurs oder dem Volumen berechnet. Diese 1208 Indikatoren werden in vier Kategorien (Momentum, Trend, Volatilität und Volumen Indikatoren) unterteilt. Das Momentum gibt die Änderungsrate der Preisbewegung an. Vertreter der Momentumkategorie sind u.a. Relative Strength Index (RSI) und Williams Prozent Range (%R). Trendbasierte Indikatoren signalisieren bei bestimmten Kursbedingungen einen Trendwechsel (Kumbure u. a., 2022).

Exemplarisch wird der Simple Moving Average (SMA) Trendindikator aufgezeigt, der den Durchschnitt der Schlusskurse einer Aktie in einer bestimmten Zeitspanne berechnet. Ein Aufwärtstrend wird einem Aktienkurs dann bescheinigt, wenn der Schlusskurs über die SMA Linie steigt und sie somit von unten schneidet. Dieses Vorgehen ist in Abbildung 1 beschrieben, wobei die Zeitperiode n mit $n = 10$ gewählt wurde und somit den 10 Tages gleitenden Durchschnitt (SMA 10) angibt. Vertreter aus der Volatilität- und Volumenkategorie kommen im Gegensatz zu Momentum- und Trendindikatoren in der Praxis nicht häufig zum Einsatz. Der Accumulation/Distribution Oscillator (ADOSC) wird als Volumenindikator mitunter am häufigsten verwendet (Kumbure u. a., 2022).
Tabelle A.1 beschreibt die verwendeten technischen Indikatoren formal mit angegebenen Referenzen (siehe Anhang A.1).

3. Stand der Forschung

3.1. Forschungsmethodik

Für die Literaturrecherche zum Thema Aktienkursvorhersage mit maschinellem Lernen wurde der Zeitraum zwischen 2000 - 2022 gewählt. Berücksichtigt wurden alle Formen wissenschaftlicher Veröffentlichungen wie Artikel in Fachzeitschriften, Konferenzberichte sowie Fachbücher. Die Recherche wurde in den Forschungsbibliotheken ‚IEEE Xplore' und ‚ACM Digital Library' sowie der Datenbank ‚Web of Science' betrieben. Die erste Suche wurde mit der folgenden Anfrage erstellt:

(‚Prediction') AND (‚Stock Price') AND (‚Machine Learning')

Das Ergebnis der Suche in den Forschungsdatenbanken (Abfrage 22. Juni 2022):

- IEEE Xplore: 477 Einträge
- ACM Digital Library: 19,140 Einträge
- Web of Science: 3,752 Einträge

Zunächst wurden die jeweils ersten 15 nach Relevanz sortierten Veröffentlichungen untersucht. Um Informationen zu den selbst zu erstellenden Modellen zu erhalten, wurde der ersten Anfrage die Ergänzung (AND {Methode}) hinzugefügt. Die erweiterte Suche mit dem SVM Modell ergab exemplarisch bei der IEEE Xplore Methode 56 Einträge. Je Algorithmus wurden die zehn relevantesten Veröffentlichungen analysiert, die verwendeten Eingabemerkmale konnten außer Acht gelassen werden. Der Fokus lag vor allem darauf, wie sich die Modelle laut Literatur optimieren lassen.

Für den Performancevergleich der erstellten Modelle wurden vorwiegend Veröffentlichungen hinzugezogen, die ebenfalls eine binäre Klassifikation vornahmen und u.a. technische Indikatoren verwendeten. Um eine allgemeine und aktuelle Übersicht zur Prädiktion von Aktienkursen mit Hilfe von ML zu erhalten, wurde nach Literaturübersichten in den Jahren 2021 und 2022 gesucht. Die Suche ergab mit den Veröffentlichungen Kumbure u. a. (2022), Rouf u. a. (2021) und Patil u. a. (2021) drei aktuelle Literaturübersichten.

3.2. Literaturübersicht

Bevor Techniken des maschinellen Lernens entwickelt und in Breite auf die Prädiktion und Analyse von Aktienkursen angewandt wurden, beschränkte sich die Nutzung auf statistische Methoden. Neben der ARIMA Methode wurden weitere Verfahren wie das ARCH, GARCH Verfahren sowie Regressionsverfahren verwendet. Vergangene Untersuchungen haben gezeigt, dass die Performance statistischer Modelle unter der Nicht-Linearität und der Dynamik von Aktienkursen leidet. Ansätze des maschinellen Lernens sind für diese Problemstellung besser geeignet, da sie auch mit nicht-linearen Zeitreihen umgehen können (Padhi u. a., 2021). Weiterhin erreichen ML Techniken meistens höhere Genauigkeitswerte als statistische Modelle (Zhong; Enke, 2019).

Die Literaturübersicht von Rouf u. a. (2021) fasst die Studienlage zur Aktienmarktvorhersage mit Techniken des ML von 2011 – 2021 zusammen. Die Analyse ergab, dass folgende ML Modelle am häufigsten verwendet wurden:

- Künstliche neuronale Netze (KNN) engl. (ANN)
- Support Vector Machine (SVM)
- Naïver Bayes Klassifikator (NB)
- Genetische Algorithmen (GA)
- Fuzzy Logik (FL)

- Deep Neural Networks (DNN)
- Regressions Algorithmen (RA)
- Hybride Ansätze (HA)

Unter den Regressionsalgorithmen (RA) sind folgende Ansätze zusammengefasst: lineare Regression, multiple Regression, logistische Regression und Support Vector Regression (SVR). Zudem werden auch auf Entscheidungsbäumen basierte Regressionsverfahren zugezählt (Rouf u. a., 2021).

In Abbildung 2 wird die Nutzungsverteilung der vorgestellten ML Modelle in den letzten zehn Jahren beschrieben. Die Analyse ergab, dass SVM Modelle am häufigsten genutzt wurden. Deep Lerning Verfahren wie das KNN stellten eine weitere beliebte Methode dar. Untersuchungen indizierten, dass komplexere Architekturen wie DNN oft nicht signifikant besser als flachere Architekturen wie SVM oder KNN performten (Zhao, 2021). Unter den DNN Modellen übertraf das Long Short-Term Memory (LSTM) Netz in mehreren Studien andere Techniken der gleichen Art, da es in der Lage ist langfristige Abhängigkeiten zu lernen und sich somit als vorteilhaft bei der Zeitreihenvorhersage erweist (Shen; Shafiq, 2020).

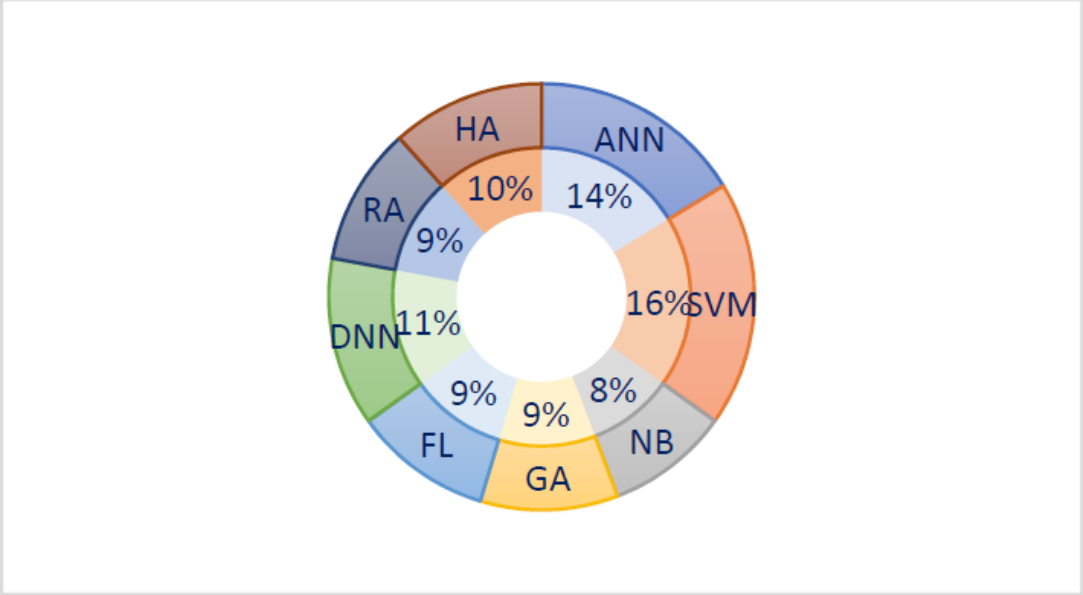

Abbildung 2: Verteilung der am häufig genutzten ML Algorithmen zwischen 2011 - 2021 (Rouf u. a., 2021).

SVM und KNN Modelle stellen somit ausführlich getestete Vergleichsalgorithmen dar, die eine vergleichsweise niedrigere Architekturkomplexität aufweisen und so einfacher zu implementieren sind. Die Genauigkeit der SVM bildet ein Mindestmaß an Performance, die ML Algorithmen leisten sollten. Weiterhin brachte die Analyse hervor, dass auf Entscheidungsbäumen basierte Ansätze oft deutlich besser performen (Rouf u. a., 2021). Da diese in der Untersuchung lediglich eine Teilmenge der Regressionsalgorithmen ausmachen, sind sie verhältnismäßig weniger untersucht. In einer weiteren Analyse wurde gezeigt, dass unter sechs baumbasierten Ensemble Modellen, darunter ET und RF, der Extra Trees Algorithmus, angewandt auf acht Aktienkurse des amerikanischen Marktes, am besten performte (Ampomah u. a., 2020).

Die Literaturübersicht von Kumbure u. a. (2022) hat 138 ausgewählte Primärstudien zwischen 2000 und 2019 untersucht und kategorisiert. Die in den Studien verwendeten ML Algorithmen zur Aktienkursvorhersage wurden in Abbildung 3 in drei Kategorien der Klassifikation eingeteilt.

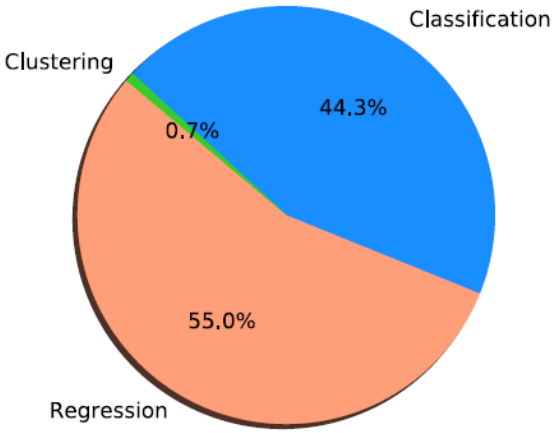

Abbildung 3: Klassifikation von ML Modellen aus 138 untersuchten Artikeln (Kumbure u. a., 2022).

Mit 55.0 % stellt die Regression die am häufigsten genutzte Methode dar. Dabei wird der exakte zukünftige Wert prognostiziert. Auf Klassifikation basierte Studien, die eine binäre Klassifikation mittels An- und Abstiegs vornehmen, haben einen Anteil von 44.3 % an den untersuchten Artikeln. Beide Methoden nutzen zum Training der ML Modelle das überwachte Lernen. Hierbei kennt der Algorithmus die wahre Vorhersage zu jedem Datum des Trainingsdatensatzes. Die Clustermethoden (Clustering) verwenden unüberwachtes Lernen, wobei der Algorithmus die Ergebnisse des Trainingsdatensatzes nicht kennt und eine Mustererkennung aus den Eingabemerkmalen vornimmt. Diese Methode ist eher der Datenvorverarbeitung dienlich und wurde deshalb lediglich in zwei Studien angewandt (Kumbure u. a., 2022).

Die Studie von Thawornwong; Enke (2004) konnte darlegen, dass in einem realen Handelsumfeld die binären Klassifikationen besser als Regressionsklassifizierungen performen. Zwar erzielen exakte Wertprognosen oft sehr hohe Genauigkeitswerte, jedoch werden diese Werte meist aufgrund von Überanpassung an den Trainingsdatensatz erreicht. Binäre Klassifikationen erzielen meist Genauigkeitswerte von 60 % - 80 %, da kleinste Steigungen oder Abstiege leicht falsch klassifiziert werden. In einer Handelssimulation auf einem realen Markt erweist sich die Klassifikationsmethode jedoch als zuverlässiger, da es wahrscheinlicher ist die passende Richtung des Handels gewählt zu haben (Zhong; Enke, 2019).

Gleichermaßen ist die Wahl des Vorhersagezeitraums für eine erfolgreiche Prognose entscheidend. Die Literaturübersicht von Kumbure u. a. (2022) legt offen, dass in den analysierten Studien am häufigsten die tägliche und der monatliche Vorhersagezeitraum gewählt wurde. Die Wahl eines Vorhersagezeitraumes von mehreren Minuten bis wenigen Stunden ist zwar möglich, aber laut aktueller Studienlage nicht üblich. Auch jährliche Prognosen werden nur in wenigen Artikeln behandelt (Kumbure u. a., 2022). Die Wahl des Vorhersagezeitraumes beeinflusst u.a. die Wahl der Eingabemerkmale für entsprechende Vorhersagealgorithmen. Technische Indikatoren sind eher für eine kurz- bis mittelfristige Prognose geeignet. Fundamentaldaten als Merkmale dienen meist einer mittel- bis langfristigen Prognose. Einige Ansätze des ML verbinden beide Arten von Datensätzen, um bessere Ergebnisse zu erreichen (Patil u. a., 2021). Es besteht zudem die Möglichkeit textuelle Daten zu erheben. Beispielsweise können hier Nachrichten aus verschiedenen Quellen

zu einem bestimmten Unternehmen für einen bestimmten Zeitraum gezählt werden und so numerisch als Eingabemerkmal genutzt werden. Eine Kombination von Marktindikatoren, fundamentaler Marktdaten sowie textueller Daten erreicht im Durchschnitt die höchste Genauigkeit, wenn sie als Eingabe für ML Techniken verwendet werden (Rouf u. a., 2021).

In der bisherigen Forschung wurden die Weltmärkte mit unterschiedlicher Quantität prognostiziert. Abbildung 4 listet die untersuchten Weltmärkte der 138 analysierten Studien in der Literaturübersicht von Kumbure u. a. (2022) auf. Mit 62 Vorhersagen wurde der amerikanische Markt (USA) am häufigsten prognostiziert. Ein oft genutzter Index war hier der Standard & Poor's 500 Index (S&P 500), der die 500 größten Firmen der USA abbildet. An zweiter Stelle steht Taiwan mit 28 Untersuchungen, gefolgt von China mit 25 Prognosen. Der europäische und speziell der deutsche Markt wurde hierbei nur marginal erforscht (Kumbure u. a., 2022).

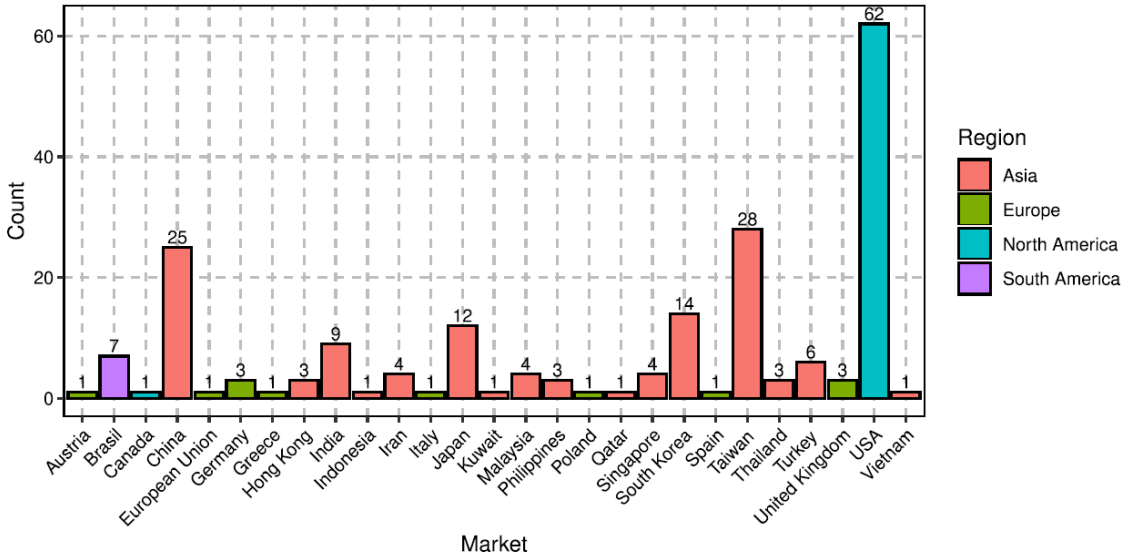

Abbildung 4: Prognosehäufigkeit nach Märkten aus 138 untersuchten Artikeln (Kumbure u. a., 2022).

In den Studien, die den DAX mit maschinellem Lernen untersuchten, wird dieser meistens für einen Performancevergleich mit anderen Weltindizes herangezogen. Dabei wurde oft eine Kurzzeitprognose von mehreren Stunden bis wenigen Tagen erstellt. Eine explizite Untersuchung des deutschen Marktes inklusive der Indizes MDAX, SDAX und TECDAX mit maschinellem Lernen ist nicht bekannt. Die Analyse der Studienlage hat deutlich gemacht, dass baumbasierte Methoden des maschinellen Lernens im Vergleich nicht häufig verwendet worden sind. Dabei untersucht diese Arbeit erstmals den deutschen Markt mit dem Ensemble Modell Extra Trees (ET). Auch eine auf den deutschen Markt angepasste Merkmalextraktion wie die Nutzung des IFO Index in Kombination mit maschinellem Lernen wurde bisher nicht vorgenommen. Des Weiteren ergibt die spezielle Wahl der technischen Indikatoren und fundamentalen Daten in Verbindung mit den untersuchten Vorhersagezeiträumen eine einzigartige Analyse.

4. Modelle zur Aktienvorhersage

4.1. Maschinelles Lernen

Das maschinelle Lernen (ML) ist eine Methode der Datenanalyse bei der erstellte Modelle auf die Eingabe von externen Informationen reagieren, indem das eigene Strukturprogramm so angepasst wird, dass die erwartete zukünftige Leistung verbessert wird. ML Methoden sollen in der Lage sein großen Eingabedatensätzen mittels Eingabe- und Ausgabefunktionen korrekte Ausgaben zuzuweisen. Im Bereich Data Mining werden ML Algorithmen genutzt, um versteckte Beziehungen und Trends in Datenmengen offenzulegen (Nilsson, 1990). Techniken des maschinellen Lernens lassen sich im Allgemeinen in drei Kategorien unterteilen:

- Überwachtes Lernen: Bei dieser Methode tragen die Eingabedaten bereits ein Etikett (engl. Label), das jedem Datenpunkt eine Klasse zuteilt. Die Eigenschaften der Datenobjekte x werden auch als Eingabemerkmale bezeichnet. Mit y wird die zugehörige Zielgröße beschrieben, auch als Ausgabe oder Etikett bezeichnet. Der Zusammenhang zwischen der Eingabe x und der Ausgabe y wird über eine Funktion $y = f(x)$ bestimmt. Die Eingabe x beschreibt in der Praxis häufig einen reellwertigen Vektor $x \in \mathbb{R}^d$, wobei $d \in \mathbb{N}$ die Anzahl der Merkmale bestimmt. Mit dem überwachten Lernen lassen sich sowohl Regressionsprobleme sowie Klassifikationsprobleme lösen. Bei einem Regressionsproblem gilt $y \in \mathbb{R}$, was bei der Aktienkursvorhersage einer Prognose des genauen zukünftigen Wertes entspräche. Bei einem Klassifikationsproblem ist die Zielgröße y die Anzahl der zu klassifizierenden Klassen und wird mit $y \in \{1, ..., K\}, K \in \mathbb{N}$ bestimmt. Bei einer binären, trendbasierten Aktienkursvorhersage kann $K = 2$ gewählt werden, wodurch eine Klasse für steigende und eine für fallende Kurse entsteht. Um ein Modell des überwachten Lernens auf die Extraktion von versteckten Mustern zu trainieren, werden Trainingsdaten als eine Teilmenge der Eingabedaten gebildet. Diese Trainingsdaten $(x_i, y_i), i = 1, ..., n$ werden in der Praxis oft aus $70 - 80\%$ der Eingabedaten gebildet und sind etikettiert. Anschließend wird das Modell mit den verbleibenden, nicht etikettierten Eingabedaten getestet (Richter, 2019). Die zur Prädiktion von Aktienkursen erstellten ML Modelle sind meist aus dem Bereich des überwachten Lernens, da bei vergangenen Zeitreihen zukünftige Werte bereits bekannt sind und die Trainingsdaten somit etikettiert werden können. Neben der Aktienkursvorhersage können verschiedenste Problemstellungen mit dem überwachten Lernen gelöst werden. Weitere Beispiele wären die Spamerkennung, bei der empfangene E-Mails in erwünscht und unerwünscht klassifiziert werden oder die Handschrifterkennung, bei der Zahlen oder Buchstaben aus Handschriften bestimmt werden können (Richter, 2019).
- Unüberwachtes Lernen: Hier haben die Trainingsdaten keine entsprechende Etikettierung und es gilt versteckte Muster ausschließlich aus den Eingabemerkmalen x_i zu erkennen. Algorithmen des unüberwachten Lernens arbeiten selbstständig mit den Eingabedaten und lernen unabhängig von vorgegebenen Zielvariablen. Ein Anwendungsbeispiel ist die Clusteranalyse, bei der Datenpunkte in unterschiedliche Cluster gruppiert werden. Beispielsweise können mit dem unüberwachten Lernen Bilder, die keine Etiketten tragen, gruppiert werden (Alpaydin, 2021). Diese Methode ist auch bei der Aktienkursvorhersage möglich, wird jedoch in der Praxis nicht oft umgesetzt, wie in Abbildung 3 aufgezeigt.
- Verstärktes Lernen: ML Modelle werden durch ein Belohnungssystem an eine bestimmte Umgebung möglichst optimal angepasst. Oft müssen bestimmte Ziele durch eine Entscheidungsfindung in einer komplexen Umgebung umgesetzt werden. Das verstärkte Lernen wird häufig in den Bereichen Robotik und autonomes Fahren verwendet (Patil u. a., 2021).

4.2. Support Vector Machine

Die von V. N. Vapnik (1995) vorgestellte Support Vector Machine (SVM) gehört zu den am häufigsten eingesetzten ML Modellen mit überwachten Lernen. Das SVM Modell wurde zunächst für die Lösung von binären Klassifizierungsproblemen konstruiert (Cortes; V. Vapnik, 1995).

Hierbei wird eine nicht-lineare Klassifizierung der Trainingsdaten ermöglicht, indem das SVM Modell nach einer Entscheidungsfläche sucht, die die Trainingsdatenpunkte in zwei Klassen trennt. Jeder Datenpunkt wird durch einen Vektor repräsentiert, die trennende Hyperebene ist jedoch nur von den ihr am naheliegendsten Vektoren, den ‚Support Vectors', abhängig. Durch die Hyperebene entstehen zwei Seiten, wobei jede Seite ausschließlich Punkte mit gleicher Klassenbezeichnung beinhaltet. Ziel ist es, eine optimal trennende Hyperebene zu finden, die den maximalen Abstand zu beiden Seiten hat. Eine nicht-lineare SVM kann die Eingabemerkmale auf einen höherdimensionalen Merkmalsraum abbilden und wendet dort die lineare SVM Methode auf den Merkmalsraum an. Dieser hochdimensionale Raum kann unendlich werden. Der Kern-Trick ermöglicht eine solche Transformation (Richter, 2019). Im Folgenden wird das SVM Modell zur Klassifikation beschrieben.

4.2.1. Lineare SVM

Zunächst wird ein Trainingsdatensatz aus n Datenpunkten gebildet $(x_i, y_i), i = 1, ..., n$. Durch die binäre Klassifikation werden die Klassen durch $y_i \in \{-1, 1\}$ mit zugehörigem Punkt x_i definiert. Der gesuchte Klassifikator in Form einer Hyperebene soll die Punkte x_i in einem höherdimensionalen Raum abbilden, um die beiden unterschiedlichen Klassen von Datenpunkten zu trennen. Zusätzlich soll die Hyperebene einen maximalen Rand aufweisen, der den Abstand zwischen Hyperebene und den nächstgelegenen Punkten der beiden Gruppen maximiert (Richter, 2019). Diese Hyperebene kann im linearen Fall folgendermaßen beschrieben werden:

$$\mathbf{x}_i\mathbf{w} + b = 0$$

Zwei parallele Hyperebenen, die einen maximalen Abstand haben sollen und die Datenpunkte trennen, können wie folgt beschrieben werden:

$$\mathbf{x}_i\mathbf{w} + b = +1$$

und

$$\mathbf{x}_i\mathbf{w} + b = -1$$

Die Distanz zwischen den zwei Hyperebenen ergibt sich durch $\frac{2}{||w||}$. Dies hat zur Folge, dass $\|w\|$ minimiert werden soll (Richter, 2019). Hiermit ergibt sich das folgende Minimierungsproblem mit Nebenbedingung:

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}||w||^2 \\ \text{wobei} \quad & y_i(\langle \mathbf{x}_i\mathbf{w} \rangle + b) - 1 \leq 0 \quad \forall i = 1, ..., n \end{aligned} \tag{1}$$

Dieses Optimierungsproblem kann in ein duales Maximierungsproblem überführt werden:

$$\max_{\alpha} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i \mathbf{x}_j \rangle \tag{2}$$

$$\text{wobei} \quad \sum_{i=1}^{n} \alpha_i y_i = 0, \quad \alpha \geq 0, \quad \forall i = 1, ..., n$$

In Abbildung 5 trennt die separierende Hyperebene eine rote und blaue Punktewolke. Die zwei parallelen Hyperebenen werden dabei als ‚Straßenrand' bezeichnet. Die sich auf dem Straßenrand befindenden Punkte werden auch als ‚Support Vectors' bezeichnet. Diese sind durch die Datenpunkte x_i beschrieben mit der Bedingung, dass die zugehörigen $\alpha_i \neq 0$ sein müssen.

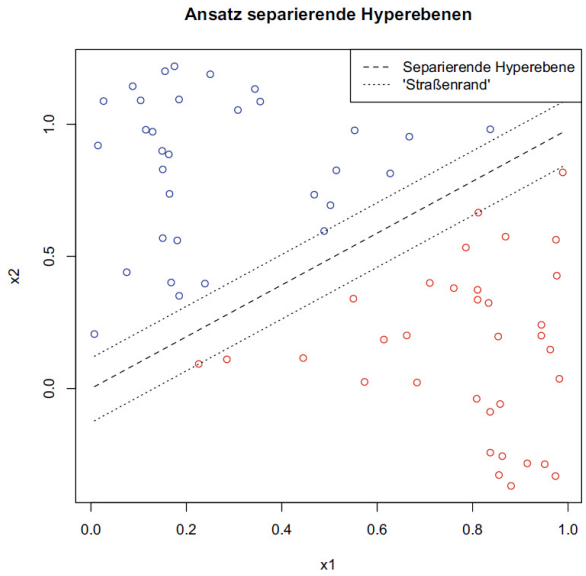

Abbildung 5: Support Vector Machine Klassifizierung mit trennender Hyperebene (Richter, 2019).

4.2.2. Nicht-lineare SVM

In der Praxis sind Trainingsdaten oft nicht linear separierbar und es kann keine passende Hyperebene gefunden werden. Dies hat zur Folge, dass die Einführung einer Schlupfvariable ξ_i, $i = 1, ..., n$ nötig ist, um Fehler in der Klassifizierung zuzulassen. Dieser Trainingsfehler, der ermöglicht, dass Punkte auch nahe der Hyperebene oder sogar auf der falsch klassifizierten Seite liegen können, soll minimal gehalten werden. Deshalb wird das Minimierungsproblem (1) um die Summe aller ξ_i erweitert. Dabei können große Werte für ξ_i verhindert werden. Dieser Strafterm lässt sich weiter kontrollieren, indem ein Kostenfaktor C eingeführt wird (Richter, 2019). Somit ergibt sich folgendes Optimierungsproblem:

$$\min_{w,b,\xi_i} \quad \frac{1}{2}||w||^2 + C \sum_{i}^{n} \xi_i \tag{3}$$

$$\text{wobei} \quad y_i(\langle \mathbf{x}_i \mathbf{w} \rangle + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad \forall i = 1, ..., n$$

Da dieses Vorgehen eine Berechnung mittels linearer Funktion durchführt, erweist sich diese Methode für nicht-lineare Klassifizierungsprobleme als nicht optimal. Hierzu besteht die Möglichkeit die Trainingsdaten mit einer Kernfunktion in einen höherdimensionalen

Raum zu projizieren, da sich diese dort mit hoher Wahrscheinlichkeit linear separieren lassen. Zur Klassifizierung der Datenpunkte werden hier ausschließlich die Support Vectors berücksichtigt, wodurch sich eine deutlich verminderte Rechenzeit ergibt (Richter, 2019). Allgemein lässt sich die Kernfunktion folgendermaßen beschreiben:

$$K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$$

Das duale Optimierungsproblem (2) wird lediglich so abgeändert, dass das Skalarprodukt $\langle \mathbf{x}_i \mathbf{x}_j \rangle$ durch die Kernfunktion $K(x_i, x_j)$ ersetzt wird.

Folgende Kernfunktionen werden häufig in der Praxis gewählt:

Linearer Kern:

$$K^{lin}(x_i, x_j) = \langle x_i, x_j \rangle \tag{4}$$

Polynomialer Kern:

$$K_p^{poly}(x_i, x_j) = (1 + \gamma \langle x_i, x_j \rangle)^p \tag{5}$$

Gauß Kern (RBF):

$$K_\gamma^{gauss}(x_i, x_j) = exp(-\gamma \cdot ||x_i - x_j||^2) \tag{6}$$

Sigmoidkern:

$$K_{\gamma,k_1}^{sigm}(x_i, x_j) = \tanh\left(\gamma \langle x_i, x_j \rangle + k_1\right) \quad \text{mit} \quad k_1 \in \mathbb{R}, \quad \gamma > 0 \tag{7}$$

Das duale Optimierungsproblem unter Berücksichtigung einer Kernfunktion und eines Kostenfaktors kann nun formal beschrieben werden:

$$\max_{\alpha} \qquad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i \mathbf{x}_j) \tag{8}$$

$$\text{wobei} \qquad \sum_{i=1}^{n} \alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C, \quad \forall i = 1, ..., n$$

Der Kostenfaktor C bestimmt die Anzahl der Support Vectors und die Breite des Abstandes der parallelen Hyperebenen. Für größere C Werte erhöht sich der Abstand der Hyperebenen und die Anzahl der Support Vectors. Somit werden mehr Trainingsfehler zugelassen. Der RBF Kern hängt insbesondere von dem Abstieg γ ab. Dieser Parameter definiert, wie viel Einfluss ein einziges Trainingsbeispiel auf die Klassifizierung hat. Je höhere Werte für γ gewählt werden, desto näher müssen andere Datenbeispiele liegen, um beeinflusst zu werden (Richter, 2019).

Die Wahl der passenden Kernfunktion hat oft eine signifikante Auswirkung auf das Klassifizierungsergebnis. Die Kernfunktion, der Kostenfaktor C sowie der Abstieg γ werden auch als Hyperparameter bezeichnet, da diese Parameter das SVM Modell maßgeblich beeinflussen und steuern. Es gilt, die auf dem Datensatz optimalen Werte für die Hyperparameter zu finden. Dies kann mit einer ‚GridSearch' bzw. Rastersuche umgesetzt werden. Hier werden alle möglichen Kombinationen der Hyperparameter im Vorfeld des Trainings auf einem Algorithmus durchlaufen. Besonders für das SVM Modell ist eine Optimierung der Hyperparameter notwendig, da die Performance stark von diesen abhängig ist (Agrawal, 2021).

4.3. Entscheidungsbäume

Entscheidungsbäume sind hierarchische Datenstrukturen, die überwachtes Lernen nutzen, um eine Klassifikation oder eine Regression auf einem Datensatz anzuwenden. Ein Entscheidungsbaum basiert auf dem CART (Classification and Regression Trees) Algorithmus, der mittels Entscheidungsregeln die Trainingsdaten klassifiziert. Der Entscheidungsbaum für die Klassifikation kann graphisch dargestellt werden, indem die Knoten t des Baumes eine Entscheidung basierend auf den Eingabemerkmalen symbolisieren. Ein Wurzelknoten W bestimmt den Anfang eines Baumes. Eine Klassifizierung wird an den Blättern b des Baumes vorgenommen (Frochte, 2018).

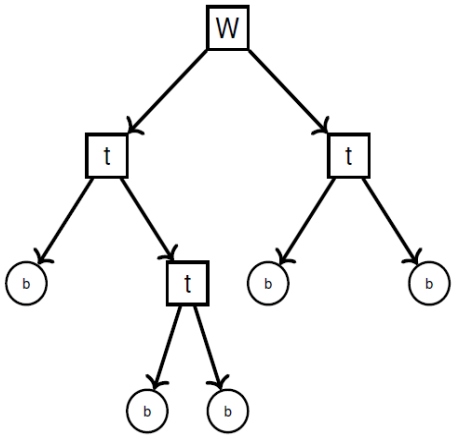

Abbildung 6: Ein Entscheidungsbaum als hierarchische Datenstruktur (Frochte, 2018).

An einem Knoten t wird die Datenmenge in zwei Teile getrennt. Eine Entscheidung wird getroffen, indem der Algorithmus testet, ob die Komponentenwerte der Datenpunkte zu einem bestimmten Merkmal unter oder über einem Schwellwert s liegen. Jeder Knoten t hat genau eine eingehende Kante und zwei ausgehende Kanten zu einem weiteren Knoten oder zu Blättern, von denen es keine ausgehenden Kanten gibt.

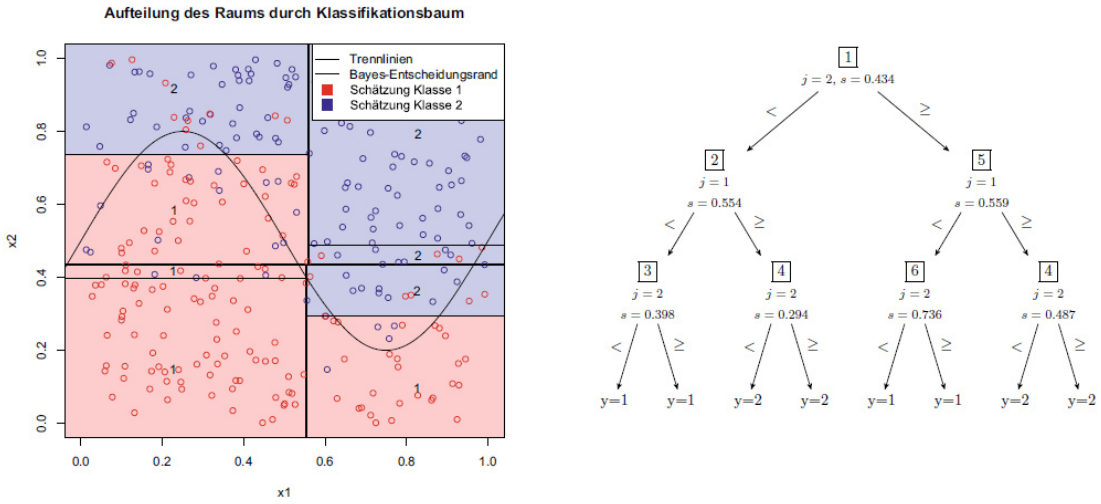

Abbildung 7: Beispiel eines Entscheidungsbaums (Richter, 2019).

In Abbildung 7 wird dieses Vorgehen beschrieben, indem auf der rechten Seite ein Entscheidungsbaum graphisch konstruiert wurde, der einen Datensatz in zwei Klassen $y \in \{1, 2\}$ trennt. Die Merkmale werden mit $x1, x2$ beschrieben. Einem Knoten des Baumes wird mit dem Trennungsindex $j \in \{1, 2\}$ ein Merkmal zugewiesen. Wenn ein Datenpunkt in diesem

Merkmal einen niedrigeren Wert als s hat, wird der linke Pfad weiter gegangen, ansonsten der rechte Pfad. Für jede Entscheidung wird im linken Bild eine Trennlinie gezogen, die die Datenmenge klassifiziert. Das Ende des Baumes ist erreicht, wenn die letzte Ebene nur noch aus Blättern besteht und somit alle Punkte klassifiziert sind (Richter, 2019).

4.3.1. Qualitätsmaße zur Trennung

Um die Güte der Aufteilung eines Knotens in einem Entscheidungsbaum zur Klassifikation zu messen, wird der durch Shannon; Weaver (1998) erstmals messbare Informationsgewinn genutzt. Diese zu jedem Knoten gehörenden Informationen können über ein Verunreinigungsmaß ausgedrückt werden. Eine Aufteilung gilt als rein, wenn alle zugehörigen Instanzen einer Verzweigung ausschließlich der richtigen Klasse zugeordnet wurden. Gesucht ist eine Trennung, die die gemessene Verunreinigung minimiert (Alpaydin, 2021). Bei Entscheidungsbäumen werden hauptsächlich zwei Funktionen zur Messung der Verunreinigung verwendet.

Die *Entropie*:

$$Entropie(T) = -\sum_{i=1}^{k} p_i \cdot log_2(p_i) \tag{9}$$

Wobei eine Menge aus dem Trainingsdatensatz mit T beschrieben wird und k die Anzahl der Klassen darstellt. Die Variable p_i beschreibt den Anteil der in Klasse i klassifizierten Datenpunkte am gesamten Datensatz. Die Werte der Entropie liegen zwischen 0 und 1. Der Wert 0 wird erreicht, wenn alle Punkte aus T einer Klasse zugeordnet wurden. Der höchste Wert wird erreicht, wenn die Punkte nur zur Hälfte richtig eingeordnet worden sind. Um festzustellen, welches Merkmal an einem Knoten die geringste Entropie ausmacht und somit den Entscheidungsbaum optimiert, kann der Informationsgewinn G berechnet werden. Dieser ermittelt den Entropieunterschied vor und nach einer Teilung durch ein bestimmtes Merkmal.

$$G(T, A) = Entropie(T) - \sum_{v \in V(A)} \frac{|T_v|}{|T|} \cdot Entropie(T_v) \tag{10}$$

Die Attribute oder Merkmale werden mit A bezeichnet und $V(A)$ stellt den Wert von A, also ein bestimmtes Merkmal dar. T_v sind die Trainingsdaten zu dem Merkmal $v \in V(A)$.

Die *Gini-Unreinheit*:

$$Gini(T) = 1 - \sum_{i=1}^{k} p_i^2 \tag{11}$$

Die Gini-Unreinheit gibt die Wahrscheinlichkeit an, dass Datenpunkte an einem Knoten falsch klassifiziert werden. Es gilt, diese zu minimieren. Geringe Werte stehen für eine effektive Separation der Klassen an einem Knoten (Richter, 2019). Gleichermaßen kann für die Gini-Unreinheit der Informationsgewinn berechnet werden. Ein Entscheidungsbaum setzt immer das Merkmal an einen Knoten, das für die größte Verunreinigungsabnahme sorgt.

4.3.2. Überanpassung und Beschneiden

Falls ein Entscheidungsbaum nach Erstellung lediglich reine Blätter aufweist und somit keine Fehlklassifizierungen vorhanden sind, liegt eine Überanpassung an den Trainingsdatensatz nahe. Um dies zu vermeiden, ist oft ein bestimmter Grad an Unreinheit notwendig (Alpaydin, 2021). Mit den folgenden Hyperparametern kann dies durch eine Anpassung der Größe des Baumes verhindert werden.

Die Tiefe des Baumes:

Ein Entscheidungsbaum kann durch das Beschneiden verkürzt werden. Es wird versucht Teilbäume, die zu einer Überanpassung führen, zu eliminieren. Zunächst wird der Entscheidungsbaum normal gebildet bis alle Blätter rein sind. Eine Stichprobe der Trainingsdaten wird zurückgehalten und auf dem verkürzten Baum getestet, der die überangepassten Teilbäume durch finale Blätter ersetzt. Falls die Ergebnisse gleich sind, wird der Teilbaum eliminiert und die Komplexität reduziert (Alpaydin, 2021).

Die Anzahl der Blätter:

Es ist möglich die Anzahl der finalen Blätter im Vorfeld der Erstellung des Baumes festzulegen. Diese Methode kann auch zu einer Beschneidung des Baumes verwendet werden. Dabei kann die Größe des Baumes erheblich reduziert werden (Alpaydin, 2021).

Entscheidungsbäume können mittels Hyperparameteranpassung optimiert werden, sind jedoch in ihrer Performance limitiert. Grund dafür ist, dass bei gleichbleibenden Daten nur ein Merkmal im Wurzelknoten verwendet wird, da immer das gleiche Merkmal für die höchste Verunreinigungsabnahme sorgt. Weiterhin kann die Wahl des Trennungskriteriums dazu führen, dass Merkmale nicht genutzt werden. Durch geringe Variationen in einem Datensatz werden stark unterschiedliche Entscheidungsbäume entworfen, einzelne Bäume sind somit oft nicht robust. Eine Überanpassung an den Datensatz ist bei einzelnen Entscheidungsbäumen grundsätzlich möglich (Cutler u. a., 2012). Um diese Probleme zu umgehen, wird in der Praxis oft ein Wald aus zufällig generierten Bäumen gebildet, ein Random Forest (RF).

4.4. Random Forest

Der von Breiman (2001) vorgestellte Random Forest ist eine Kombination aus Entscheidungsbäumen, um die Leistung einzelner Entscheidungsbäume zu verbessern und deren Robustheit zu erhöhen. Diese Gruppierung von Bäumen wird auch als Ensemble bezeichnet. Aus den Trainingsdaten werden zufällige Stichproben gezogen, die für die Erstellung eines Waldes an Entscheidungsbäumen verwendet werden. Diese als Bootstrapping vorgestellte Methode bildet möglichst unkorrelierte Bäume aus den Stichproben. Eine Stichprobe kann auch mehrmals vorkommen (Breiman, 2001). Für jeden Knoten eines Baumes steht eine zufällige Auswahl der Merkmale zur Verfügung. Welches Merkmal aus dieser Teilmenge verwendet wird, ergibt sich durch die Minimierung der Verunreinigung. Es wird also das Merkmal gesucht, das den Baum an einem Knoten optimal teilt. Die Größe der Teilmenge der für eine Teilung zur Verfügung stehenden Merkmale wird mit einer Funktion aus der Gesamtanzahl der Merkmale berechnet. Als Funktion für ein Klassifizierungsproblem kann die Wurzelfunktion $\sqrt{p}$ oder der binäre Logarithmus $log_2(p)$ verwendet werden, wobei p die Anzahl der Eingabemerkmale darstellt (Richter, 2019). In einem Wald von zufällig erstellten Entscheidungsbäumen wird eine Klassifizierung vorgenommen, indem aus den einzelnen Klassifizierungen mittels Mehrheitsentscheidung eine finale Klassifikation getroffen wird. Diese Aggregation der einzelnen Entscheidungen wird auch als ‚aggregating' bezeichnet und sorgt für eine Varianzreduktion der einzelnen Entscheidungsbäume. Der RF Algorithmus nutzt das Bagging-Verfahren (Bootstrap aggregating). Dabei wird das Verwenden von Stichproben zur Erstellung der Bäume und die Klassifizierung mittels Mehrheitsentscheidung verbunden. Dies führt zu einer Varianzreduktion der Vorhersagen und damit zu höheren Genauigkeitswerten. Die Bäume des RF werden nicht beschnitten, eine Überanpassung kann jedoch selbst bei hoher Baumanzahl ausgeschlossen werden (Cutler u. a., 2012).

4.4.1. Out-of-Bag Daten

Bei der Verwendung von Bootstrap-Stichproben besteht die Möglichkeit, dass einige Datensätze nicht zur Erstellung von Entscheidungsbäumen genutzt werden. Da gleiche

Bootstrap-Stichproben mehrfach vorkommen können wird der Umstand, dass Daten ignoriert werden, begünstigt. Unter der Nichtverwendung von Daten für das Training eines Modells kann die Vorhersagegenauigkeit eines Vorhersagemodells leiden. Diese Daten werden als Out-of-Bag (OOB) Daten bezeichnet und können verwendet werden um den Generalisierungsfehler (OOB Fehler) abzuschätzen. Dieser misst, wie genau neue Daten von einem ML Algorithmus vorhergesagt werden können. Für die nicht genutzten Bootstrap-Stichproben wird eine OOB Vorhersage für jeden Baum erstellt und der ermittelte Fehler wird zur Optimierung der Klassifikation verwendet (Breiman, 2001).

4.4.2. Bedeutung der Merkmale

Der Random Forest Algorithmus ermöglicht eine Messung der Merkmaleinflüsse auf die Bildung der Entscheidungsbäume in einem zufälligen Wald. Die Bedeutung der Merkmale kann über den Mittelwert der Verunreinigungsabnahme jedes Baumes berechnet werden. Das Ergebnis der Berechnung listet die Merkmale geordnet nach ihrer Wichtigkeit zur richtigen Klassifizierung und der damit verbundenen Unreinheitsabnahme auf. Dieses Verfahren eignet sich, um die gelöste Problemstellung genauer analysieren zu können. Durch eine gezielte Auswahl der Merkmale, die zu einer Abnahme der Verunreinigung führen, kann das Vorhersagemodell verbessert werden (Cutler u. a., 2012). Zudem besteht die Möglichkeit erste Einschätzungen zu den gewählten Merkmalsarten zu geben. Falls eine bestimmte Art von Merkmal häufiger zu einer größeren Verunreinigungsabnahme führt, kann dieser Ansatz weiterverfolgt und ausgebaut werden.

4.5. Extremely randomized Trees

Die Extremely randomized Trees Methode wurde von Geurts u. a. (2006) entwickelt und stellt ein weiteres auf Entscheidungsbäumen basiertes Ensemble Modell dar. Die Methode wird durch den Extra Trees (ET) Algorithmus umgesetzt, der dem Random Forest Algorithmus ähnlich ist. Es wird auch ein Wald aus unbeschnittenen, zufällig generierten Entscheidungsbäumen erstellt, wobei das Klassifizierungsergebnis per Abstimmung festgelegt wird. Ebenso stehen den Knoten nur eine Teilmenge der Merkmale für eine Trennung zur Verfügung. Der ET Algorithmus unterscheidet sich hauptsächlich in zwei Punkten von dem Random Forest. Es werden keine Bootstrap-Stichproben verwendet. Jeder Entscheidungsbaum wird aus den gesamten Trainingsdaten gebildet. Während der RF eine Teilung an den Knoten vornimmt, indem das optimale Merkmal eingesetzt wird, das die Verunreinigung minimiert, nimmt der ET Algorithmus eine Teilung durch ein zufällig gewähltes Merkmal vor. Die zufällig gewählten Teilungsmerkmale sorgen für eine extremere Form des Zufalls, was für die Namensgebung der Methode verantwortlich ist (Geurts u. a., 2006). Der ET Algorithmus bietet auch die Möglichkeit die Bedeutung der Merkmale zu berechnen. Die Aussagekraft ist jedoch gering, da die Merkmale zufällig den Knoten zugewiesen werden. Da standardmäßig kein Bootstrapping zum Einsatz kommt, werden auch keine OOB Daten berücksichtigt. Beide Methoden, Bootstrapping und das Nutzen von OOB Daten, können jedoch in der Praxis auch mit dem ET Algorithmus verwendet werden. Der Unterschied zu dem RF würde dann lediglich in der zufälligen Auswahl der trennenden Merkmale an den Knoten bestehen.

4.6. Künstliche neuronale Netze

Künstliche neuronale Netze (KNN) sind von der Arbeitsweise der biologischen Neuronen im menschlichen Gehirn inspiriert und bilden ein Teilgebiet der künstlichen Intelligenz (Frochte, 2018). Grundlage für ein neuronales Netz zur Klassifikation bietet das von Rosenblatt (1963) vorgestellte Perzeptron. Das einfache Perzeptron besteht aus einem einzelnen künstlichen Neuron.

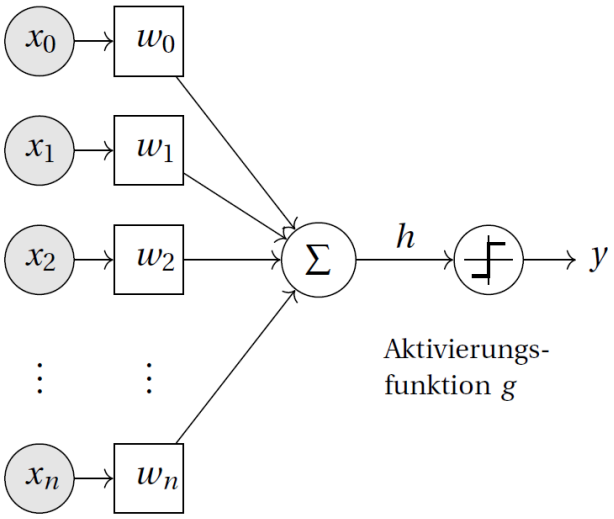

Abbildung 8: Modell eines künstlichen Neurons (Frochte, 2018).

Abbildung 8 beschreibt die Funktionsweise eines einzelnen künstlichen Neurons. Ein Merkmal wird als Vektor x mit Komponenten $x_0, x_1, ..., x_n$ dem Neuron als Eingabe übergeben. Anschließend werden die Vektorenkomponenten durch entsprechende Gewichte $w_i \in \mathbb{R}$ angepasst. Die Funktion $h(x)$ wird folgendermaßen beschrieben:

$$h(x) = \sum_{i=0}^{n} w_i x_i$$

Die Aktivierungsfunktion g verarbeitet die Berechnung von $h(x)$ weiter, indem der Eingabe ein bestimmter Wertebereich zugeteilt wird. Ein Beispiel stellt die Schwellwertfunktion als Aktivierungsfunktion dar, wobei ein Schwellwert θ eingeführt wird, der das Neuron aktiviert, falls dieser überschritten wird (Frochte, 2018). Mathematisch kann dies durch die Sprung- bzw. Stufenfunktion (‚Heaviside'-Funktion) umgesetzt werden:

$$y = g(h) = \begin{cases} 1 & \text{für } h > \theta \\ 0 & \text{für } h \leq \theta \end{cases}$$

Das Ausgabesignal y des einfachen Perzeptrons wird so durch die Aktivierungsfunktion bestimmt. Dieses Signal kann für weitere Neuronen als Eingabe dienen, wodurch ein künstliches neuronales Netz entsteht (Richter, 2019).

In Abbildung 9 wird ein solches künstliches neuronales Netz beschrieben. Dieses besteht aus mehreren Schichten. Die Eingabeschicht ‚Input layer' nimmt die Merkmalvektoren entgegen. Es können beliebig viele versteckte Schichten ‚Hidden neural layer' hinzugefügt werden. In Abbildung 9 sind exemplarisch zwei versteckte Schichten implementiert worden. Jedes Neuron ist mit allen folgenden Neuronen der nächsten Schicht verbunden. Die

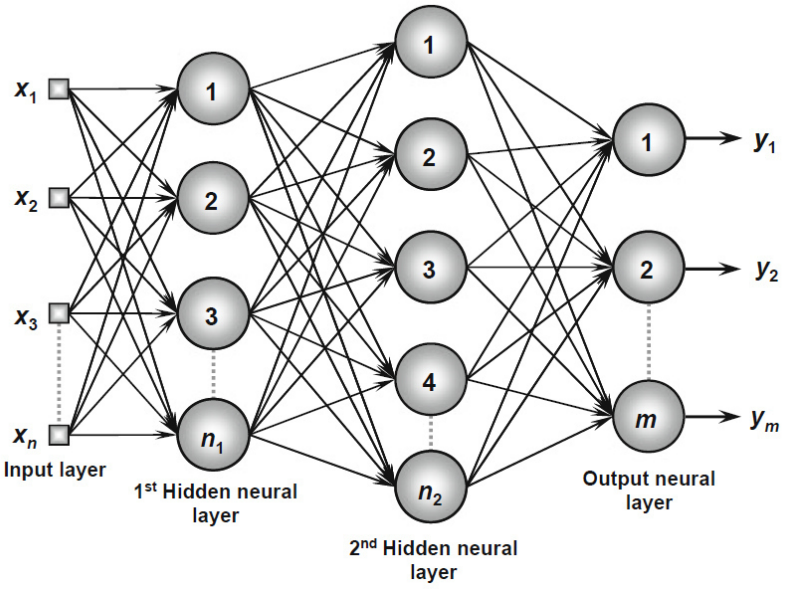

Abbildung 9: Ein mehrlagiges neuronales Netz (da Silva, 2017).

Ausgabeschicht ‚Output layer' bestimmt das finale Ergebnis. Für eine binäre Klassifizierung wird dafür ein einzelnes Neuron erstellt. Dieser Typ des neuronalen Netzes wird auch als ‚feedforward Netz' bezeichnet, da ausschließlich Verbindungen von Vorgängerschichten zu Nachfolgerschichten vorliegen und Informationen somit nur in eine Richtung weitergegeben werden (da Silva, 2017).

4.6.1. Aktivierungsfunktion

Neben der Heaviside-Funktion können weitere Aktivierungsfunktionen gewählt werden, um das Verhalten des Netzes zu steuern und die Ausgabe zu optimieren. Folgende Aktivierungsfunktionen werden häufig verwendet:

Sigmoid-Funktion:

$$f_{sig}(h) = \frac{1}{1 + e^{-h}} \tag{12}$$

Diese Funktion hat einen festen Ausgabebereich und gibt Werte zwischen 0 und 1 zurück. Zudem ist die Sigmoid-Funktion nicht-linear und stetig differenzierbar (Deng; Yu, 2014).

Rectified Linear Unit (RELU):

$$f_{RELU}(h) = \begin{cases} h & \text{für } h > 0 \\ 0 & \text{für } h \leq 0 \end{cases} \tag{13}$$

Tangens hyperbolicus (Tanh)

$$f_{tanh}(h) = \frac{e^{h} - e^{-h}}{e^{h} + e^{-h}} \tag{14}$$

Der Ausgabebereich der Tanh Funktion ist $[-1, 1]$. Die Funktion ist ebenfalls nicht-linear (Deng; Yu, 2014).

4.6.2. Verlustfunktion

Eine Verlustfunktion ermöglicht es nach dem Training eines neuronalen Netzes die Gewichte w anzupassen, um optimalere Prognosen zu liefern. Die Verlustfunktion berechnet

den Verlust aus den prognostizierten Werten y und den wahren Werten $\widehat{y}$. Ziel ist es, den Verlust während des Trainings zu minimieren. Für eine binäre Klassifikation wird häufig folgende Funktion verwendet:

Binärer Kreuzentropieverlust

$$L(y_i, \widehat{y}_i) = -(\widehat{y}_i \cdot log_2(y_i) + (1 - \widehat{y}_i) \cdot log_2(1 - y_i)) \tag{15}$$

Die Verlustfunktion gibt so den Verlust für ein Trainingsbeispiel wieder. Der gesamte Verlust kann mit einer Kostenfunktion berechnet werden, die den Durchschnitt der Verlustfunktionen berechnet (Deng; Yu, 2014).

4.6.3. Updatefunktion

Ein ‚rekurrentes Netz' stellt eine Erweiterung des ‚feedforward Netze' dar, indem einzelne Neuronen Informationen über Prognosefehler erhalten und daraufhin Anpassungen vornehmen. Ein rekurrentes Netz ermöglicht diese Funktion durch den ‚Backpropagation' Algorithmus. Dieser gibt den mittels Verlustfunktion berechneten Fehler an alle Schichten des Netzes zurück. Hierbei erhält jedes Neuron den Fehler und passt mit einer Updatefunktion die entsprechenden Gewichte an. Die zur Updatefunktion nötige Grundlage bietet der negative Gradient der Verlustfunktion. Dabei gibt der Gradient die Richtung an, in der die Steigung der Funktion am größten ist. Die Negation des Gradienten führt zu der Richtung des größten Abstieges (Richter, 2019). Da die Verlustfunktion minimiert werden soll, wird der Gradientenabstieg optimiert. Das Bestimmen des Minimums der Verlustfunktion führt so zu einer maximierten Vorhersagegenauigkeit des KNN Modells. Ein weiterer essenzieller Parameter ist die Lernrate η. Diese gibt an, wie schnell das neuronale Netz lernt. Der Lernfaktor steigt mit großen Werten für η (Ruder, 2016).

Das Stochastic Gradient Descent (SGD) Verfahren ist eine solche Updatefunktion. Der negative Gradient wird approximiert über den Fehler eines Neurons bestimmt. Durch einen zusätzlichen Parameter, die Lernrate, wird bestimmt, wie groß die Schritte sind, um ein lokales Minimum des Optimierungsproblems zu finden. Die Anpassung eines Parameters $\theta \in \mathbb{R}^d$ kann mit dem negativen Gradienten der Optimierungsfunktion $\nabla_\theta J(\theta)$ umgesetzt werden.

Stochastic Gradient Descent (SGD)

$$\theta = \theta - \eta \cdot \nabla_\theta J(\theta; y^{(i)}; \widehat{y}^{(i)}) \tag{16}$$

Das Parameterupdate wird für jedes Trainingsbeispiel mit dem Trainingsdatenpunkt y_i und dem Etikett $\widehat{y}_i$ durchgeführt (Ruder, 2016). Dabei wird in der Regel eine Teilmenge der Trainingsdaten für eine Trainingseinheit verwendet. Diese ‚Batch' wird vor dem Training festgelegt und sorgt bei großen Werten meist für eine hohe Berechnungszeit. Sobald der gesamte Datensatz einmal mit der Updatefunktion verbessert wurde, ist eine Epoche vergangen. Die Anzahl der Epochen kann beliebig groß gewählt werden. Je mehr Epochen für das Training des Modells einbezogen werden, desto höher ist die Laufzeit des Trainings (Richter, 2019).

Ein weiteres Gradientenabstiegsverfahren ist das Adaptive Moment Estimation (Adam) Verfahren. Das Adam Verfahren optimiert das SGD Verfahren insofern, dass die Lernrate nicht festliegt, sondern während des Abstiegsverfahrens angepasst wird (Ruder, 2016). Um die Suche nach dem Minimum der Optimierungsfunktion zu beschleunigen, wird das Momentum eingeführt. Das Momentum sorgt dafür, dass die Gradienten während des Abstieges in einem Vektor gespeichert werden.

Adaptive Moment Estimation (Adam)

Der Gradient der Optimierungsfunktion kann unter Angabe eines Zeitpunktes t beschrieben werden:

$$g_t = \nabla_{\theta_t} J(\theta_t)$$

Das Verfahren speichert den Durchschnitt vergangener Gradienten w_t. Des Weiteren wird der Durchschnitt der vergangenen quadratischen Gradienten v_t gespeichert. Die Parameter β_1 und β_2 werden standardmäßig mit 0.9 und 0.999 belegt und stellen Abklingraten dar (Ruder, 2016).

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$
$$v_t = \beta_2 m_{t-1} + (1 - \beta_2) g_t^2$$

Der Parameter m_t ist dabei eine Schätzung des ersten Momentums der Gradientenmittelwerte. Der Parameter v_t schätzt das zweite Momentum der Gradientenmittelwerte.

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$$
$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

Die Adam Updatefunktion ergibt sich mit:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t + \epsilon}} \hat{m}_t \tag{17}$$

Durch die Anpassungen erzielt das Adam Verfahren einen beschleunigten Abstieg, selbst, wenn der Gradient zu einem Zeitpunkt niedrig ist (Ruder, 2016).

5. Datengrundlage

5.1. Extraktion der Indizes

Die Kurse der deutschen Indizes wurden über den Zeitraum 01. Juni 2013 - 01. Januar 2022 durch die Yahoo Finanzen API extrahiert. Dies entspricht einer Datengrundlage von neun Jahren oder 2170 Handelstagen. Eine Handelswoche besteht aus fünf Tagen, da die Börsen an den Wochenenden geschlossen sind. Folgende Eigenschaften der Indizes werden mitgeliefert: Eröffnungskurs, Tageshöchstkurs, Tagestiefstkurs, Schlusskurs und das Volumen. Für die Berechnung der technischen Indikatoren sind alle Merkmale notwendig. Die Mehrzahl der Indikatoren wird aus den Schlusskursen gebildet. Es werden 17 technische Indikatoren berechnet, wovon drei Indikatoren auf dem Volumen basieren. Für die Indizes SDAX, MDAX und TECDAX wurden nur unzureichende Volumendaten mitgeliefert, wodurch drei Volumenindikatoren nicht berücksichtigt werden konnten. Weitere Eingabemerkmale stellen drei fundamentale Daten dar. Diese bestehen aus zwei Rohstoff-Schlusskursen, dem Rohöl- und Goldkurs und werden ebenfalls über die Yahoo Finanzen API für den gleichen Zeitraum extrahiert. Der IFO Index bildet den dritten makroökonomischen Datensatz.

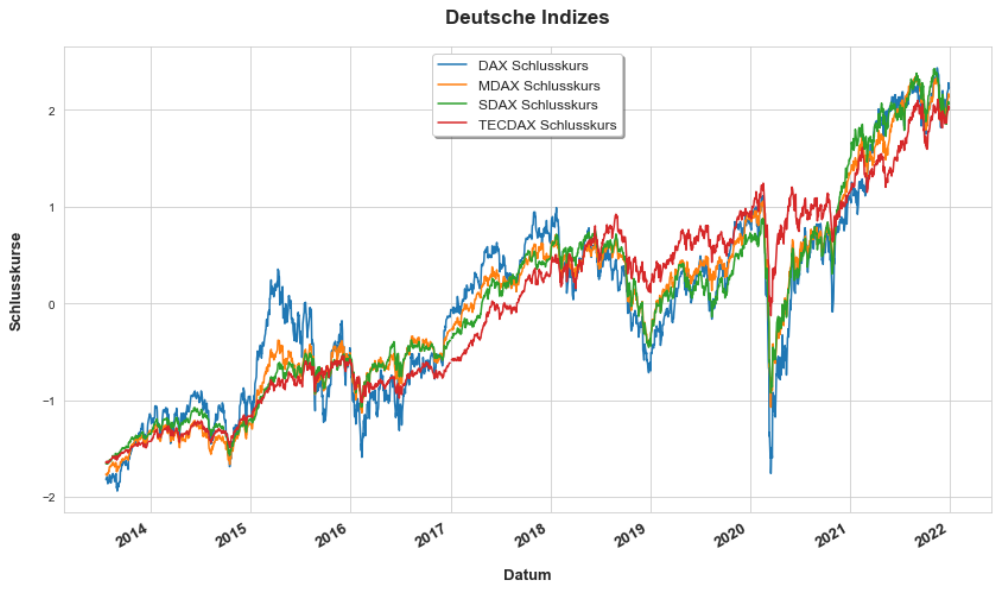

Abbildung 10: Deutsche Indizes von 2013 - 2022.

Die Kursverläufe des untersuchten Zeitraums der deutschen Indizes ist in Abbildung 10 dargestellt. Der Graph veranschaulicht, dass die Trendrichtungen der standardisierten Schlusskurse meist ähnlich sind. Im Vergleich steigt und fällt der DAX in einigen Zeiträumen jedoch stärker als die verwandten deutschen Indizes. Der TECDAX verläuft kontinuierlich mit geringeren Ausschlägen.

Eingabemerkmale	**DAX**	**MDAX SDAX TECDAX**
Schlusskurs	1	1
Technische Indikatoren	17	14
Fundamentale Daten	3	3
Summe	21	17

Tabelle 1: Anzahl der Eingabemerkmale der Indizes.

Tabelle 1 gibt die Anzahl der verwendeten Merkmale für den jeweiligen Indizes wieder. Für die DAX Vorhersage können alle extrahierten technischen Indikatoren inklusive der Volumenindikatoren zur Eingabe genutzt werden.

5.2. Extraktion der Merkmale

5.2.1. Technische Indikatoren

Als Grundlage der berechneten technischen Indikatoren dienen die Schlusskurse der Indizes. Die meisten Indikatoren sind von einer Zeitperiode n abhängig. Diese gibt an, aus bis zu wie vielen Tagen in der Vergangenheit die Indikatoren berechnet werden. Die folgende Auflistung beschreibt nach Colby; Meyers (1988) alle verwendeten technischen Indikatoren mit Angaben zur Zeitperiode.

- Rendite: Beschreibt die prozentuale Veränderung des Kurses im Vergleich zum Vortag. Dabei wird n auf 1 gesetzt.
- Volumen: Gibt die Anzahl der gehandelten Aktien pro Tag an. Eine Preisbewegung nach oben oder unten in Verbindung mit einem hohen Volumen wird in der technischen Analyse als stärker und relevanter angesehen als eine Steigung mit geringem Volumen (Schuster, 2015).
- SMA: Der Simple Moving Average berechnet das arithmetische Mittel der Zeitreihe. Typische Werte für n liegen zwischen 5 und 25 Tagen. Der SMA wird mit $n = 10$ berechnet.
- EMA: Der Exponential Moving Average gibt den exponentiell gewichteten Mittelwert an. Hierbei wird mehr Gewicht auf jüngere Beobachtungen gelegt, wodurch Kursveränderungen besser aufgegriffen werden. Es werden EMA Werte mit $n = 10$ und $n = 20$ gebildet.
- WMA: Der Weighted Moving Average wird ähnlich wie der EMA berechnet, nur mit linearer Gewichtung. Dabei wird $n = 10$ gesetzt.
- Momentum: Zeigt die Stärke einer Kursbewegung in einer bestimmten Zeitperiode auf. Das Momentum wird mit $n = 10$ bestimmt.
- RSI: Der Relative Strength Index ist eine Weiterentwicklung des Momentums und berechnet ein Verhältnis zwischen den jüngsten Kursbewegungen. Es werden Zahlen zwischen 0 und 100 erzeugt. Werte über 30 sprechen für ein steigendes Signal und Werte, die unter 70 fallen, signalisieren fallende Kurse. Die Zeitperiode wird mit $n = 14$ angegeben.
- ROC: Die Rate of Change berechnet eine Veränderungsrate einer Zeitperiode. Die Zeitperiode wird mit $n = 10$ gewählt.
- SAR: Der Stop and Reverse Indikator gibt an, dass ein Verkauf nötig ist, wenn der Kurs unter den SAR Wert fällt (Novak; Velušček, 2016).
- %R: Die Williams Prozent Range gibt Kauf- und Verkaufssignale an, indem der aktuelle Schlusskurs mit dem höchsten oder niedrigsten Kurs der jüngsten Vergangenheit verglichen wird. Verglichen wird je nach aktueller Trendrichtung. Die Vergleichsperiode wird mit $n = 14$ gewählt.
- MACD: Der Moving Average Convergence Divergence wird aus dem 12 und 26 Tage EMA gebildet. Die MACD Signallinie aus dem 9 Tage EMA. Das MACD Histogramm bildet die Differenz der beiden Funktionen. Falls der MACD die Signallinie überschreitet, wird ein steigender Kurs vorhergesagt, bei einer Unterschreitung, fallende Kurse.
- OBV: Das On Balance Volume berechnet sich durch drei Schritte. Falls der Schlusskurs höher als der vorherige Schlusskurs ist, wird das aktuelle Volumen zu dem vorherigen OBV Wert addiert. Falls der Kurs unverändert bleibt, wird 0 addiert und falls er niedriger ist, wird das aktuelle Volumen abgezogen.

- CCI: Der Commodity Channel Index ist ein Preisdynamik-Indikator und wird mit $n = 14$ berechnet.
- ADOSC: Der Accumulation/Distribution Oscillator ist eine Variation des OBV mit dem Unterschied, dass das aktuelle Volumen gewichtet in die Berechnung einfließt (Colby; Meyers, 1988).

Die extrahierten technischen Indikatoren werden formal in Tabelle A.1 beschrieben. (siehe Anhang A.1)

5.2.2. Fundamentale Daten

Die Schlusskurse für die Rohstoffe Rohöl und Gold werden mit fehlenden Werten übermittelt und müssen geringfügig angepasst werden. Über den angegebenen Zeitraum fehlen bei beiden Rohstoffen Werte, die bei der Extraktion nicht mitgeliefert werden. Die Anzahl der fehlenden Werte liegt jeweils unter 100 für den Datensatz von 2170 Handelstagen. Deshalb können fehlende Werte aufgefüllt werden, indem der Vorgängerwert kopiert und an die leere Stelle gesetzt wird.

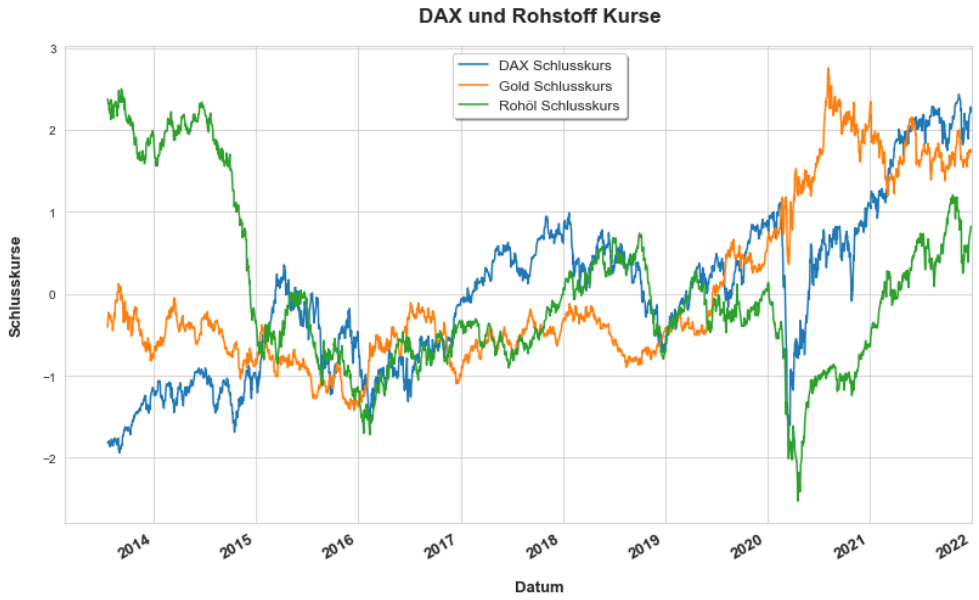

Abbildung 11: Gold und Rohöl Rohstoffkurse mit DAX Schlusskurs.

In Abbildung 11 werden die beiden Rohstoffe graphisch neben dem DAX dargestellt. Als Zeitraum dient die gesamte Analyseperiode. Um die unterschiedlichen Kurse nebeneinander darstellen zu können, liegen die Schlusskurse in standardisierter Form vor.

Der IFO Index kann für einen ausgewählten Zeitraum über das IFO Institut[4] als Exceltabelle bezogen werden. Die IFO Werte werden monatlich neu berechnet. Da die Indizekurse pro Tag gegeben sind, müssen die IFO Werte angepasst werden. Hierfür wird der monatliche Wert für jeden Tag des gleichen Monats eingetragen. Jeder Tag in einem bestimmten Monat hat so den gleichen IFO Wert. Abbildung 12 zeigt die standardisierten DAX- und IFO Werte als Zeitreihen. Da der IFO Index monatlich berechnet wird, eignet sich dieser eher für eine mittelfristige- bis Langzeitprognose.

[4] `https://www.ifo.de/ifo-zeitreihen`

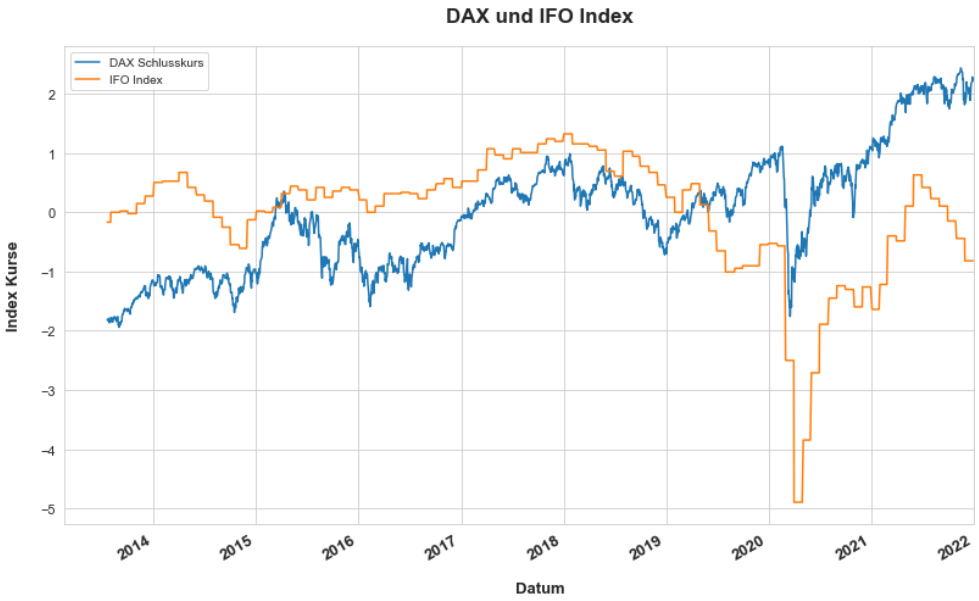

Abbildung 12: DAX und IFO Index.

5.3. Standardisierung der Merkmale

Für die meisten ML Methoden ist eine Umskalierung von Zeitreihendaten notwendig. Dies ist darin begründet, dass die Eingabemerkmale oft in unterschiedlichen Maßen oder Dimensionen vorliegen (Lazzeri, 2021). Während Indizewerte meist im Tausender-Wertebereich liegen, kann beispielsweise das Volumen Werte von mehreren Millionen aufweisen. Die Neuskalierungsmethode Standardisierung eignet sich für die Zeitreihenprognose besonders, da sie sich als robust gegenüber neuen Werten, die außerhalb des erwarteten Wertebereichs liegen, erweist. Die Standardisierung skaliert die Verteilung der Eingabewerte so um, dass der Mittelwert $\overline{x}$ gleich 0 ist und die Standardabweichung σ gleich 1 ist (Frochte, 2018). Formal kann die Standardisierung wie folgt beschrieben werden:

$$x_i' = \frac{x_i - \overline{x}}{\sigma}$$

Wobei x_i' die standardisierten Werte eines Eingabemerkmals x_i beschreibt. Alle Merkmale aus Tabelle 1 werden standardisiert als Eingebe der ML Vorhersagemodelle verwendet.

5.4. Indextrends

Nachdem die Schlusskurse der deutschen Indizes über neun Jahre extrahiert wurden, können die wahren Vorhersagen ermittelt und die Tage etikettiert werden. Jedem Tag der Zeitreihe wird ein Etikett zugeordnet. Dieses gibt an, ob der Schlusskurs in einer angegebenen Zeitperiode höher oder niedriger steht. Falls der Kurs nach der Periode höher steht, wird eine 1 etikettiert, falls niedriger eine 0. Es ist davon auszugehen, dass es keine identischen Vergleichskurse geben kann, da die Schlusskurse auf sechs Nachkommastellen angegeben werden und das Vorkommen von exakt gleichen Werten unwahrscheinlich ist.
Als zu prognostizierende Zeiträume werden 1, 5, 10, 15 und 20 Tage gewählt. Die Vorhersage eines Tages ist eine Kurzzeitprognose, während eine wöchentliche Vorhersage ab fünf Tagen eine mittelfristige Prognose darstellt. In Abbildung A.1 wird exemplarisch gezeigt, wie oft bei einer 5 Tages Prognose des DAX Indizes der Kurs nach fünf Tagen gestiegen oder gefallen ist. Prozentual sind DAX Werte innerhalb der neun Jahresperiode zu 57.7 % gestiegen und zu 42.3 % gefallen. Die Verteilung der Rendite über die neunjährige Zeitperiode ist in Abbildung A.2 beschrieben (siehe Anhang A.2).

5.5. Performance Metriken

5.5.1. Modellevaluation

Für einen Vergleich von ML Vorhersagemodellen mit überwachtem Lernen eignet sich bei einer binären Klassifizierung die Genauigkeit in Prozent. Da die Ergebnisse der Trainingsdaten im Vorfeld bekannt sind, können Fehlklassifizierungen kenntlich gemacht werden. Die Genauigkeit bietet ein Mittel, um schnell einschätzen zu können, wie hoch die allgemeine Trefferrate der Vorhersage liegt. Die Grundlage, um die Genauigkeit zu berechnen, bieten die folgenden Klassifizierungen der Datenpunkte. Für die Kategorisierung existieren hier zwei Klassen, die Klasse 0 und 1.

- Richtig Positiv (RP): Alle Punkte, die der wahren Klasse 1 angehören und als solche klassifiziert wurden.
- Richtig Negativ (RN): Die Punkte, die der wahren Klasse (0) zugehörig sind und auch dieser Klasse zugeordnet wurden.
- Falsch Positiv (FP): Fehlkassifizierte Punkte mit der wahren Klasse 0, die aber in Klasse 1 klassifiziert wurden.
- Falsch Negativ (FN): Hier wurden Punkte fehlerhaft der Klasse 0 zugeteilt. Die richtige Zuteilung wäre jedoch Klasse 1.

Aus den Klassifikationstypen können weitere Kennzahlen abgeleitet werden. Darunter die Genauigkeit des ML Modells (Kotu, 2015).

5.5.2. Genauigkeit in Prozent

Die Genauigkeit (engl. Accuracy) gibt die Fähigkeit eines Klassifikators an, die wahren Klassen der Datenpunkte richtig zu kategorisieren. Eine perfekte Genauigkeit mit 100 % würde keine Fehlklassifizierungen zur Folge haben. Die Anzahl der falsch klassifizierten Punkte würde sich mit $FN = FP = 0$ ergeben (Kotu, 2015). Die Genauigkeit lässt sich folgendermaßen berechnen:

$$Genauigkeit = \frac{RP + RN}{RP + RN + FP + FN}$$

Eine Multiplikation mit 100 ergibt die Prozentangabe der Genauigkeit. Die prozentuale Genauigkeit gibt eine Einschätzung über die allgemeine Vorhersageperformance eines ML Algorithmus an. Es kann gemessen werden, wie gut ein Modell in einem Handelsszenario performt, wobei jede Vorhersage des Modells zu einer Kaufentscheidung führt. Es wird bei einer Kaufentscheidung nicht unterschieden, ob fallende oder steigende Kurse vorhergesagt wurden.

5.5.3. Konfusionsmatrix

Die Konfusionsmatrix ist ein grundlegendes Werkzeug zur graphischen Darstellung der Klassifikationsbeziehungen. Die wahre Klasse beschreibt die vertikale Achse, während die vorhergesagte Klasse die horizontalen Spalten beschreibt. Die Anzahl der richtig klassifizierten Ergebnisse sind auf der Hauptdiagonalen von oben links nach unten rechts zu finden. Die Proportion der als falsch klassifizierten Punkte ist auf der Diagonalen von unten links nach oben rechts zu sehen. Die Angaben der einzelnen Klassifikationen können in Prozent als Anteil aller Klassifikationen angegeben werden. Dieser Anteil wird farblich gekennzeichnet, indem die Felder der Matrix umso dunkler werden, je mehr Klassifizierungen den Feldern zugeordnet wurden (Kotu, 2015). Eine Konfusionsmatrix zur binären Klassifizierung besteht aus zwei Klassen. Zur Aktienkursvorhersage wird die steigende Prognose mit Klasse 1 gekennzeichnet und die fallende Prognose mit Klasse 0.

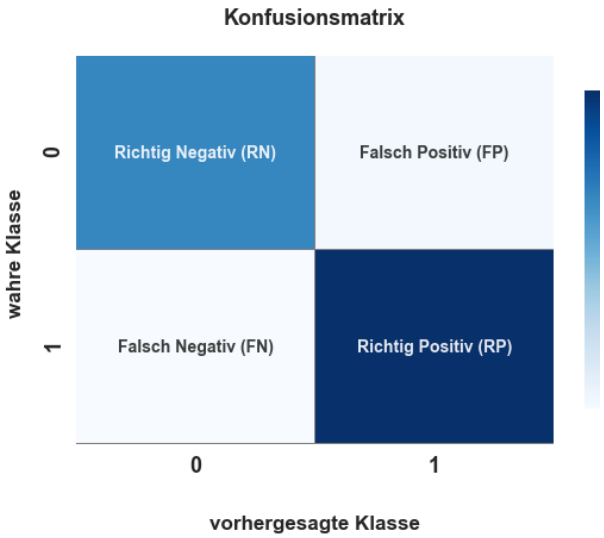

Abbildung 13: Allgemeine Beschreibung einer Konfusionsmatrix.

Abbildung 13 zeigt eine allgemeine Konfusionsmatrix mit zwei Klassen. Die Belegung der Achsen der Konfusionsmatrix ist nicht fest. Die Achsen- und die Felderbeschriftungen werden in der Praxis oft vertauscht, die Aussagekraft bleibt aber identisch (Kotu, 2015).

Aus der Konfusionsmatrix können weitere relevante statistische Maße abgeleitet werden. Für die Aktienkursvorhersage ist vor allem der positive und der negative Vorhersagewert nützlich. Diese entsprechen einem Verhältnis zwischen den als korrekt klassifizierten Daten einer Klasse und den Daten, die insgesamt einer Klasse zugeteilt wurden. Dabei beschreibt der positive Vorhersagewert (engl. Precicion) das Verhältnis zwischen den richtig-positiven Werten und allen Werten, die der positiven Klasse zugeordnet wurden.

$$Positiver\ Vorhersagewert = \frac{RP}{RP + FP}$$

In der Aktienkursanalyse entspricht dies dem Anteil an steigenden Kursen, die das Modell korrekt vorhergesagt hat aus allen Kursen, die das Modell als steigend klassifizierte. In einem Handelsszenario gibt der positive Vorhersagewert die Erfolgswahrscheinlichkeit an, falls ausschließlich auf steigende Vorhersagen gesetzt wird und fallende Vorhersagen nicht berücksichtigt werden.

Der negative Vorhersagewert (engl. Recall) beschreibt das Verhältnis zwischen richtig-negativ klassifizierten Daten und der Anzahl aller Datenpunkte der negativen Klasse (Kotu, 2015).

$$Negativer\ Vorhersagewert = \frac{RN}{RN + FN}$$

Die negative Vorhersage ermöglicht es Erwartungswerte für ein Handelsszenario anzugeben, bei dem auf fallende Kurse gesetzt wird. Fallende Kurse, die ein Modell korrekt vorhergesagt hat, werden in ein Verhältnis mit allen als fallend klassifizierten Kursen gesetzt. Steigende Kurse werden dabei außer Acht gelassen. Weiterhin ermöglicht eine Multiplikation mit 100 die prozentuale Angabe der Maße.

6. Vorhersagemethodik

6.1. Kursvorhersage

Methodisch ist die Aktienkursvorhersage so aufgebaut, dass zunächst einer der vier deutschen Indizes (DAX, MDAX, SDAX, TECDAX) ausgewählt wird. Aus den überlieferten Merkmalen des Indizes werden 17 Indikatoren berechnet. Weiter werden die drei fundamentalen Daten (IFO Index, Gold- und Rohölkurse) extrahiert und gespeichert.

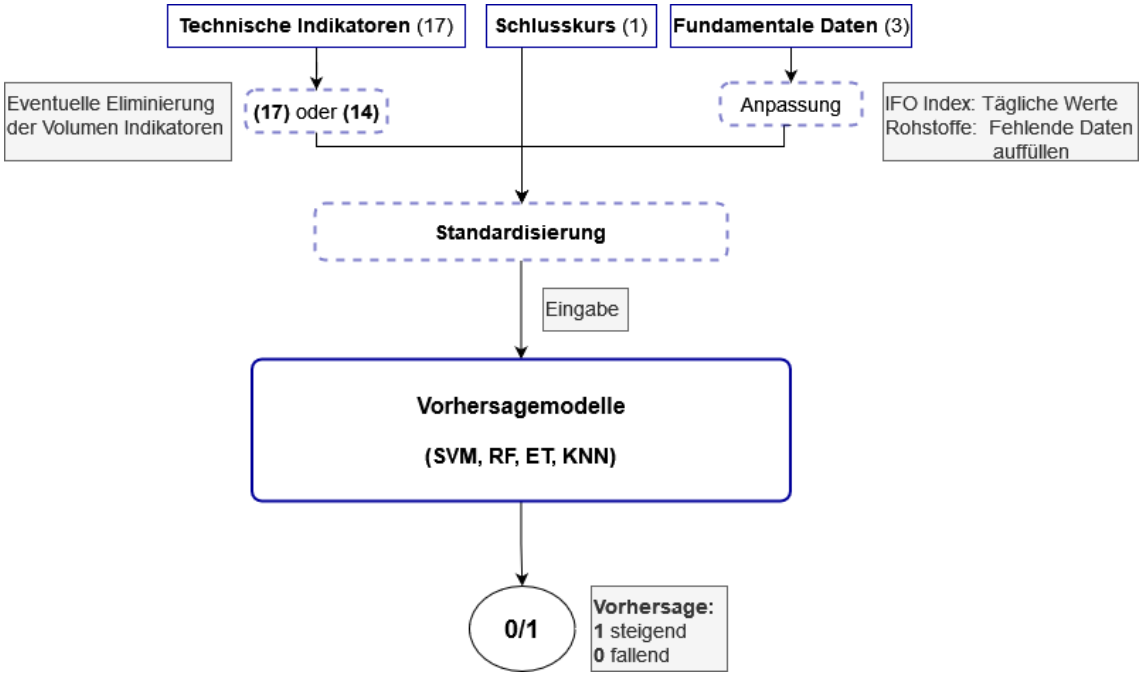

Abbildung 14: Vollständiger Vorhersageverlauf (in Anlehnung an (Patel u. a., 2015)).

In Abbildung 14 wird der vollständige Verlauf der Prognose beschrieben. Nach der Extraktion der Merkmale eines Indizes wird eine Anpassung der Eingabemerkmale vorgenommen. Die technischen Indikatoren werden auf eine Anzahl von 14 reduziert, falls das die Volumendaten des Indizes über 100 fehlende Werte aufweisen. Die fundamentalen Daten werden angepasst, indem für den IFO Index tägliche Werte als Kopien der monatlichen Werte eingesetzt und fehlende Daten der Rohstoffe mit den Vorgängerwerten aufgefüllt werden. Die Schlusskurse liegen unverarbeitet zur Eingabe vor. Im nächsten Schritt werden alle Eingabemerkmale standardisiert. Die standardisierten Merkmale werden anschließend den vorgestellten Modellen des maschinellen Lernens übergeben. Diese werden auf dem entsprechenden Datensatz des jeweiligen Indizes trainiert und angepasst. Eine genaue Beschreibung des Vorgehens wird im Folgenden erläutert. Die vier Vorhersagemodelle sind nun in der Lage für jedes Datum des Eingabedatensatzes eine steigende oder fallende Prognose zu geben. Ein Vergleich der Modelle findet anschließend mit der Genauigkeit statt.

Der Trainingsverlauf und die anschließende Hyperparameteroptimierung wird in Abbildung 15 genauer aufgezeigt. Die Vorhersagemodelle werden aus der standardisierten Datengrundlage eines Indizes trainiert. Der Datensatz der Eingabemerkmale spaltet sich so in ein Trainingsdatensatz bestehend aus 70 % der Daten und ein Testdatensatz mit 30 % der Daten. Die Datenpunkte werden dem jeweiligen Datensatz zufällig zugeordnet. Die Trainingsdaten sind etikettiert und das korrekte Ergebnis der Vorhersage ist bekannt. Diese Etiketten werden auch zur Eingabe der ML Algorithmen überliefert. Der Algorithmus kann nun einen Zusammenhang aus den Eingabemerkmalen und der korrekten Prognose der Daten lernen.

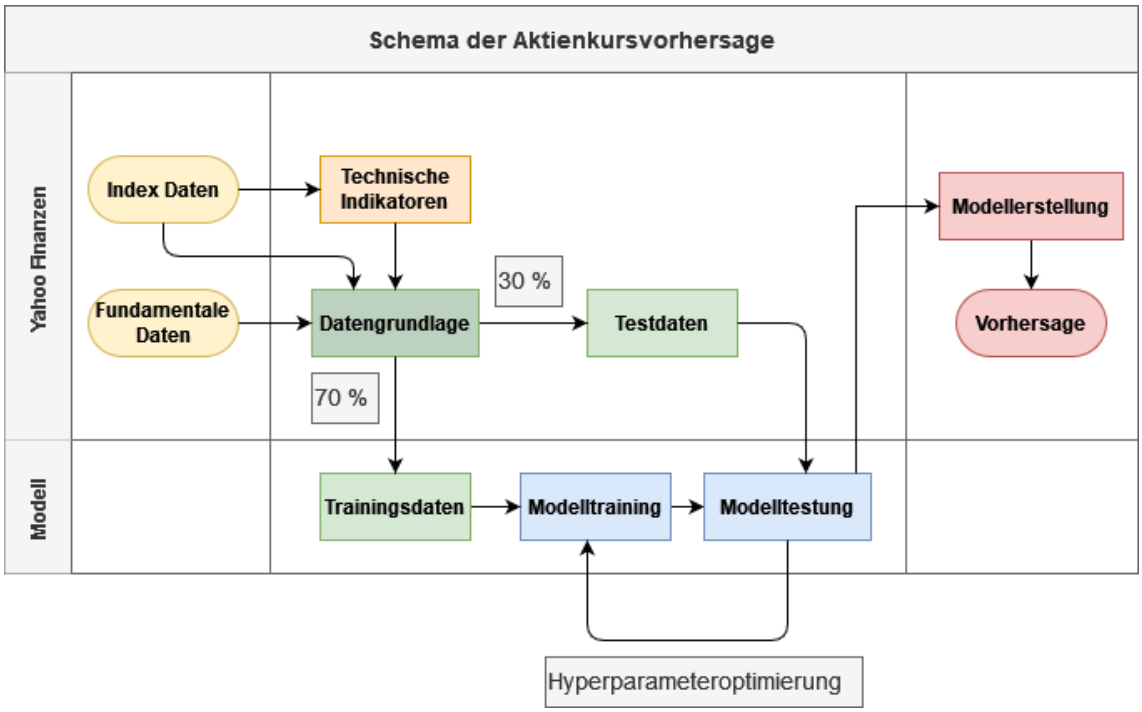

Abbildung 15: Schema der Aktienkursvorhersage.

Nach dem Training der Modelle erstellen diese eine Prognose für einen neuen, zuvor nicht gesehenen Datensatz, den Testdaten. Dieser Datensatz wird nicht etikettiert übergeben. Die Modelle klassifizieren anschließend die Testdaten basierend auf den gelernten Zusammenhängen der Trainingsdaten. Da die korrekten Klassifizierungen der Testdaten vorliegen, können diese mit den ermittelten Klassifizierungen der Modelle verglichen werden. Ein Vergleich findet mit der Genauigkeit statt. Um die Genauigkeit der Modelle zu maximieren, werden die Eingabemerkmale auf eine Verbesserung der Modellleistung getestet. Hierbei wird der Trainings- und Testprozess mit Teilmengen aus allen Eingabemerkmalen wiederholt und verglichen. Im ersten Schritt werden die Modelle mit nur einem Merkmal trainiert und ausgewertet. In dem jeweiligen nächsten Schritt wird ein weiteres Merkmal hinzugefügt und evaluiert. Nachdem die Anzahl der verwendeten Merkmale für einen Index bestimmt worden ist, werden die vier ML Modelle (SVM, RF, ET, KNN) angepasst und optimiert. Für jedes Modell wird das ‚GridSearch' Verfahren angewendet. Zunächst werden für jeden Hyperparameter eines Modells eine Liste von Werten erstellt (siehe Kapitel 6.2). Der GridSearch Algorithmus bildet im Folgenden jede mögliche Kombination der einzelnen Hyperparameterwerte. Mit den verschiedenen Kombinationen werden die Modelle erstellt, trainiert und getestet. Um die Genauigkeit einer Einstellung zu messen, werden vier Durchläufe mit dieser Einstellung durchgeführt. Die durchschnittliche Genauigkeit aus den vier Durchläufen wird der Parametereinstellung zugewiesen. Anschließend wird die Hyperparametereinstellung mit der höchsten Genauigkeit ausgewählt, um das ML Modell zu erstellen.

6.2. Hyperparameteroptimierung

Um die Performance der ML Modelle evaluieren zu können, müssen diese zunächst in einen bestimmten Zustand gesetzt werden, damit bei gleicher Eingabe auch konstant die gleichen Ergebnisse geliefert werden. Dies wird umgesetzt, indem die Modellvariable ‚RandomState' einen festen Wert erhält und sich das Modell dadurch in einem festen Zustand befindet (Lazzeri, 2021). Gleiches gilt für die Teilung der Datengrundlage in Test- und Trainingsdaten. Durch das Setzen des RandomState werden Datenpunkte zwar zunächst zufällig zugeordnet, bei identischer Datengrundlage bleibt die Verteilung jedoch bei Wiederholung gleich. Die Hyperparameteroptimierung der verwendeten ML Modelle basiert auf einer 5 Tages Prognose. Die optimale Einstellung der Modelle wird für die verbliebenen Vorhersagezeiträume 1, 10, 15 und 20 Tage beibehalten. Durch den Vergleich einer GridSearch der Modelle über alle Vorhersagezeiträume konnte ermittelt werden, dass sich die optimale Hyperparametrisierung für die Modelle nicht wesentlich ändert. Eine deutliche Performancesteigerung ist durch auf einen bestimmten Vorhersagezeitraum angepasste Hyperparameter bei allen Modellen nicht zu erwarten. Im Folgenden werden zu den vorgestellten Techniken des maschinellen Lernens die Wertebereiche der Hyperparameter für eine GridSearch vorgestellt und die ermittelte optimale Belegung gezeigt. In diesem Zusammenhang sind Referenzen auf die formale Beschreibung der Parameter ergänzt.

Support Vector Machine

- C : [1, 10, 100, 1000] (8)
- γ : [0.01, 0.1, 1, 10]
- Kerne: [Linearer Kern (4), Polynomialer Kern (5), RBF (6), Sigmoidkern (12)]

Optimale Belegung: $C = 100$, $\gamma = 0.1$, Kern=RBF

Eine Belegung für C mit 10 und 100 ergibt die höchste Performance. Als Kernfunktionen performen der RBF mit der höchsten Leistung sowie der Sigmoidkern am besten. Der Parameter γ sollte bei Verwendung des RBF unter 1 gewählt werden (Novak; Velušček, 2016). Falls der Sigmoidkern gewählt wird, sind γ Werte von 0.1 oder 1 zu wählen. Beim Heranziehen des RBF Kerns performt das SVM Modell mit einem γ Wert von 0.1 am besten. Das Nutzen eines linearen oder polynomialen Kerns führt zu keiner Verbesserung der Performance. Eine Anpassung der Hyperparameter auf weitere Vorhersagezeiträume ist nicht erforderlich. Es ist keine wesentliche Performancesteigerung zu erwarten (Novak; Velušček, 2016).

Random Forest

- Anzahl der Bäume: [100, 200, 500, 800, 1000, 2000]
- Kriterium: [Gini-Unreinheit (11), Entropie (9)]
- OOB Daten: [Ja, Nein]

Optimale Belegung: Anzahl Bäume = 1000, OOB Daten = Ja, Kriterium = Gini-Unreinheit

Ein exemplarischer Entscheidungsbaum eines zufälligen Waldes ist in Abbildung A.3 graphisch vorgestellt (siehe Anhang A.4). Um nur einen Ausschnitt zu zeigen, ist der Entscheidungsbaum beschnitten. Der eigentliche Baum ist um ein Vielfaches größer. Der RF zählt zu den robusten Methoden des maschinellen Lernens, was bedeutet, dass nach Belegung der Parameter mit Standardwerten keine wesentliche Performanceverbesserung durch eine weitere Optimierung zu erwarten ist. Die Wahl des Kriteriums ändert die Performance nur minimal. Das Heranziehen der OOB Daten kann abgestellt werden, führt aber zu einer verminderten Leistung. Die Anzahl der erstellten Entscheidungsbäume, auch als Parameter ‚n_estomators' bezeichnet, führt mit hohen Werten zu einer guten Performance mit

geringer Varianz. In Abbildung A.4 ist die Fehlerrate mit zunehmender Anzahl von Bäumen aufgezeigt (siehe Anhang A.4). Der Fehler wird durch $1 - Genauigkeit$ berechnet. Sobald die Anzahl der Bäume über 200 wächst, befindet sich die Fehlerrate in einem festen Intervall mit geringfügiger Schwankung.

Extra Trees

- Anzahl der Bäume: [100, 200, 500, 800, 1000, 2000]
- Kriterium: [Gini-Unreinheit (11), Entropie (9)]
- Bootstrap: [Ja, Nein]

Optimale Belegung: Anzahl Bäume = 1000, Kriterium = Gini-Unreinheit, Bootstrap = Nein

Für den ET Algorithmus gelten bei der Wahl des Kriteriums und der Anzahl der Bäume die gleichen Bedingungen wie bei dem RF Modell. Obwohl Bootstrap-Stichproben standardmäßig nicht einbezogen werden, besteht bei der Implementierung des Modells die Möglichkeit diese Eigenschaft zu nutzen. Dies führt jedoch zu keiner Verbesserung der Vorhersage. Auch die Wahl des Kriteriums beeinflusst die Vorhersagegenauigkeit nicht wesentlich. Die Parameter können zwar auf einen bestimmten Vorhersagezeitraum angepasst werden, es kann jedoch nur eine geringe, partielle Performanceverbesserung erreicht werden. Durch das Finden der optimalen Anzahl von Bäumen und dem besten Kriterium für einen Vorhersagezeitraum ist eine maximale Verbesserung von einem Prozentpunkt zu erwarten.

Künstliche neuronale Netze

- Aktivierungsfunktion: [Sigmoid (12), RELU (13), Tanh (14)]
- Anzahl versteckter Schichten: [0, 1, 5, 10]
- Anzahl Neuronen: $[1, ..., 200]$
- Epochen: $[100, ..., 8000]$
- Batchgröße: $[100, ..., 1000]$
- Updatefunktion: [SGD (16), Adam (17)]

Optimale Belegung: Aktivierungsfunktion = Sigmoid, Anzahl versteckter Schichten = 0, Epochen = 4500, Optimierungsfunktion = Adam

Ein künstliches neuronales Netz kann mit beliebig vielen versteckten Schichten aufgebaut werden. Die Untersuchung zur Parameteroptimierung hat hervorgebracht, dass der Einsatz von versteckten Schichten nicht zu höheren Genauigkeitswerten führt. Das Modell wird aus einer Eingabe- und einer Ausgabeschicht erstellt. Die Anzahl der Neuronen der Eingabeschicht wurde auf 100 festgelegt, die Ausgabeschicht besteht aus einem Neuron für eine binäre Klassifizierung. Die Sigmoid-Funktion liefert die besten Werte aus den Aktivierungsfunktionen. Die Trainingsepochen beschreiben die Anzahl an Durchläufen mit denen ein neuronales Netz trainiert wird. Nach jedem Durchlauf werden die Gewichte durch die Updatefunktion angepasst. Es wurden Epochen in einem Intervall von 100 bis 8000 getestet. Die Epochen werden pro Durchlauf um jeweils 100 Einheiten hochgezählt. Die Batchgröße wurde in 10er-Schritten zwischen 100 und 1000 getestet. Werte um 200 verbesserten die Leistung des KNN Modells. Als optimale Updatefunktion hat sich das Adam Verfahren erwiesen, welches zu deutlich höheren Genauigkeiten führte als mithilfe des SGD Verfahrens. Eine Anpassung auf weitere Vorhersagezeiträume ergab keine besseren Ergebnisse.

Ein Auszug aus der Implementierung der Vorhersagemodelle mit optimierten Hyperparametern kann im Anhang A.6 eingesehen werden.

7. Prognostizierte Ergebnisse

7.1. Performance der Merkmale

Um zu evaluieren, ob die einzelnen Eingabemerkmale zur Vorhersage der Indizes nützlich sind und zu einer Steigerung der Genauigkeit führen, wurden diese in einer 5 Tages Vorhersage für den DAX getestet. Tabelle 2 beschreibt die Performance der einzelnen Merkmale für die vier ML Modelle. Die Evaluation startet mit einem eingesetzten Merkmal, dem DAX Schlusskurs. Weiter wird die mit diesem Merkmal erreichte Genauigkeit in Prozent der Modelle gezeigt. In jeder Iteration bzw. Zeile der Tabelle wird ein Merkmal hinzugefügt und die prozentuale Genauigkeit erneut berechnet. Unter dem Spaltennamen ‚Merkmale' stehen die Eingabemerkmale, die in den Iterationen hinzugefügt werden. In der Spalte ‚Anzahl Merkmale' steht die Summe der genutzten Merkmale pro Auswertung. In der letzten Zeile wurden insgesamt 21 Merkmale als Eingabe der Modelle verwendet.

Anzahl Merkmale	SVM	RF	ET	KNN	Merkmale
1	56.1622	55.3822	55.5382	56.1622	Schlusskurs
2	56.3183	56.6303	56.6303	55.2262	Rendite
3	56.6303	56.4743	58.8144	54.4462	Volumen
4	56.3183	60.6864	59.5944	55.2262	SMA 10
5	55.6942	62.0905	61.7785	54.9142	EMA 10
6	55.5382	61.6225	64.8986	53.8222	EMA 20
7	55.8502	66.6147	65.8346	55.2262	WMA 10
8	56.0062	67.0827	68.6427	51.9501	Momentum 10
9	55.5382	68.1747	71.7629	53.1981	SAR
10	56.4743	66.9267	71.4509	58.1903	RSI
11	57.5663	67.2387	70.9828	57.4103	ROC
12	59.5944	66.7707	71.6069	57.0983	%R
13	60.0624	69.5788	73.0109	56.9423	MACD
14	60.3744	71.9189	75.1950	56.9423	OBV
15	61.3105	71.6069	76.7551	60.3744	MACD_SIGNAL
16	63.0265	73.0109	76.9111	60.2184	MACD_HIST
17	65.3666	74.2590	77.0671	60.8424	CCI
18	65.9906	73.0109	77.2231	65.6786	ADOSC
19	67.8627	75.8190	79.8752	64.4306	Gold Schlusskurs
20	67.7067	75.1950	79.2512	63.6505	Rohöl Schlusskurs
21	70.5148	76.2871	80.1872	67.7067	IFO Index

Tabelle 2: Performance der Merkmale bei einer 5 Tages Prognose am DAX.

Die Evaluation wird graphisch in einem Säulendiagramm in Abbildung 16 dargestellt. Auf der x-Achse wird der Eingabe jeweils ein Merkmal hinzugefügt, um die Genauigkeit in Prozent für alle vier Modelle zu berechnen. Die Ausgangsbedingungen wie die Einstellung der Modelle und der Vorhersagezeitraum sind für jede Iteration identisch, um ausschließlich die Performance der Merkmale zu evaluieren. Graphisch wird sichtbar, dass die Genauigkeit aller Modelle proportional zur Anzahl der genutzten Merkmale wächst.

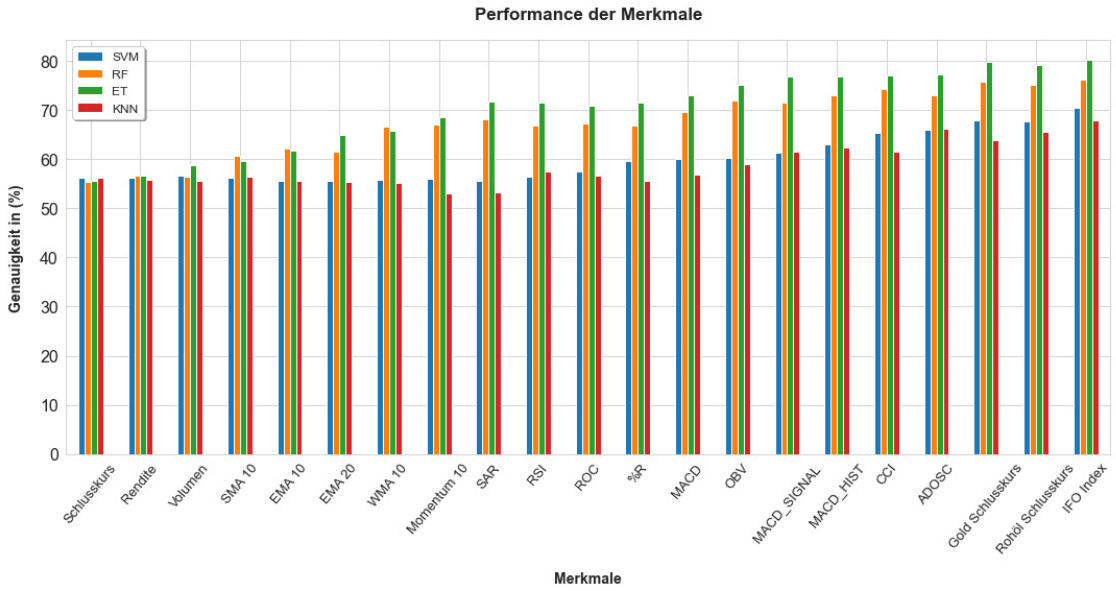

Abbildung 16: Säulendiagramm über die Performance der Eingabemerkmale.

Die Genauigkeit der Ensemble Methoden RF und ET fällt jeweils nur ein bis zweimal leicht ab und ist ansonsten ausschließlich steigend. Mit dem Einsatz von mehreren Merkmalen performen die Ensemble Methoden durchgängig besser als SVM und KNN. Der ET ist durchschnittlich um etwa zwei Prozentpunkte besser als der RF. Das SVM Modell kann zunächst nicht mit mehr als einem Merkmal optimiert werden. Die Genauigkeitswerte sinken mit zunehmender Anzahl von Merkmalen leicht ab, bis mit zehn Merkmalen eine Schwelle erreicht ist, nach der die Genauigkeit mit zunehmender Anzahl von Merkmalen steigt. Eine ähnliche Performance ist bei dem KNN Modell zu sehen. Hier schwanken die Genauigkeitswerte in den ersten Iterationen um den Anfangswert, bis das Modell ab dem Nutzen von zehn Merkmalen besser performt.
Die Evaluation der Merkmalperformance auf den verbliebenen Indizes für einen Vorhersagezeitraum von über 5 Tagen ergab die gleiche Tendenz. Tabelle A.2 zeigt die Performance der Eingabemerkmale einer 20 Tages Prognose für den TECDAX (siehe Anhang A.3).

In einer Evaluation der 1 Tages Prognose des DAX konnte ausschließlich das SVM Modell mit dem Nutzen weiterer Eingabemerkmale neben dem Schlusskurs leicht verbessert werden. Alle vier Modelle erzielten eine Genauigkeit von etwa 51 % mit der Verwendung des Schlusskurses als einziges Merkmal. Das SVM Modell konnte mit 20 weiteren Eingabemerkmalen um einen Prozentpunkt verbessert werden. Die Genauigkeit der restlichen Modelle wurde durch eine zusätzliche Merkmalnutzung reduziert. Tabelle A.3 zeigt die Merkmalperformance für die Kurzzeitprognose (siehe Anhang A.3). Eine erhebliche Verbesserung der Vorhersage für eine 1 Tages Prognose konnte durch die herangezogenen technischen Indikatoren nicht gezeigt werden. In diesem Fall ist es notwendig Merkmale explizit für eine Kurzzeitprognose zu selektieren und anzupassen.

Da die baumbasierten ML Algorithmen RF und ET für eine Vorhersage über 5 Tagen die höchste Performance liefern, ist es nützlich eine Einsicht über den Einfluss der Merkmale auf die Modelle zu erhalten. Das RF Modell bestimmt das optimale Merkmal eines Knotens, indem das Merkmal aus einer Teilmenge verwendet wird, dass für die größte Verunreinigungsabnahme der Teilung sorgt (siehe Abschnitt 4.4.2). Die Bedeutung der Merkmale des RF Modells einer 5 Tages DAX Prognose wird in Abbildung A.5 und für eine 20 Tages DAX Prognose in Abbildung A.6 gezeigt (siehe Anhang A.4.2). Die Abbildungen zeigen die mittlere Verunreinigungsabnahme für alle Eingabemerkmale des RF Modells. Die wichtigsten Merkmale für eine mittelfristige Prognose stellen die Rohöl- und Goldschlusskurse dar. Der IFO Index verbessert die Leistung bei einer 5 Tages Progno-

se am wenigsten, ist aber bei der 20 Tages Prognose das fünft wichtigste Merkmal. Der Volumenindikator OBV ist ein signifikantes Merkmal für beide Zeiträume. Die Rendite sorgt in beiden Vorhersagezeiträumen für die geringste Verunreinigungsabnahme. Bei einem Vorhersagezeitraum von 20 Tagen können sogar negative Werte vorkommen. Eine Auswertung der Verunreinigungsabnahme kann auch für das ET Modell erstellt werden. Hier ist es zwar möglich die Abnahme der Verunreinigung einer Teilung zu berechnen, dieser Wert ist jedoch nicht aussagekräftig, da die Teilung zufällig erfolgt. Die Evaluation der Bedeutung der Merkmale für das ET Modell ergab zwei nahezu identische Graphen. Dabei waren lediglich wenige benachbarte Merkmale auf der x-Achse vertauscht.

7.2. Vorhersagen der Modelle

Nachdem die gewinnbringende Nutzung der Eingabemerkmale für eine mittelfristige Prognose verdeutlicht werden konnte und die Modelle durch angepasste Hyperparameter optimiert worden sind, ist eine Prognose der deutschen Indizes möglich. Dabei wird die Auswertung anhand der Testdaten erstellt. Die Modellvorhersage für den DAX wird in Tabelle 3 präsentiert.

Vorhergesagte Tag	**1**	**5**	**10**	**15**	**20**
SVM	**52.2621**	70.5148	78.7832	80.4992	80.1872
RF	48.9860	76.2871	84.0874	87.6755	89.2356
ET	48.3619	**80.1872**	**87.2075**	**89.0796**	**90.9516**
KNN	51.1700	67.8627	77.6911	79.8752	82.5273

Tabelle 3: Vergleich der Vorhersagemodelle am DAX mittels der Genauigkeit in Prozent.

Das SVM Modell prognostiziert den Vorhersagezeitraum von einem Tag am besten. Die prozentuale Genauigkeit des KNN Modells ist den Werten der SVM über alle Zeiträume ähnlich. Die mittlere Abweichung der Prognose beträgt 0.62 Prozent. Beide Modelle liefern ab einer 5 Tages Prognose deutlich schlechtere Vorhersagewerte als die Ensemble Methoden. Das ET Modell weist ab einer 5 Tages Vorhersage die höchsten Genauigkeitswerte auf und performt im Mittel um 1.9 Prozentpunkte besser als das RF Modell.

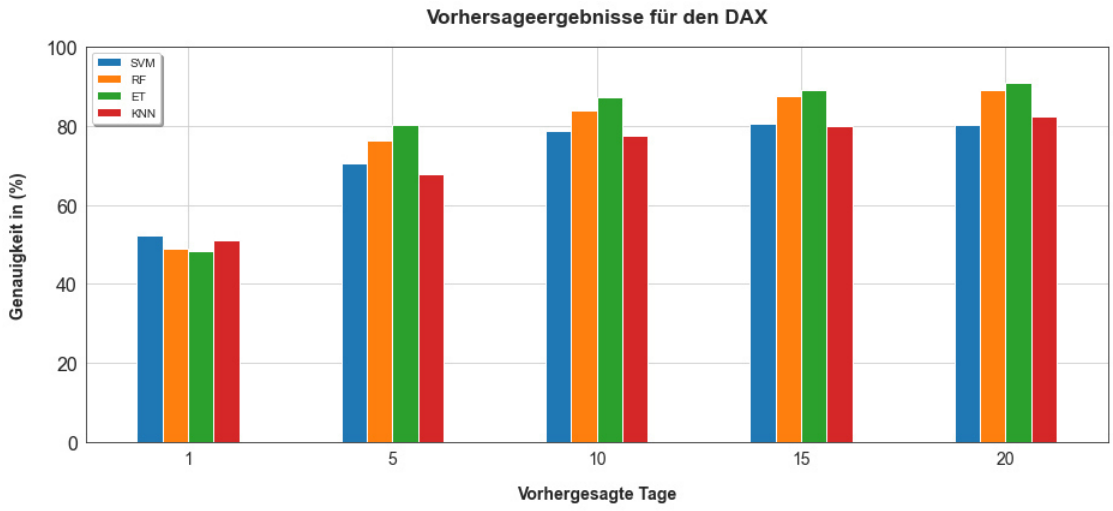

Abbildung 17: Säulendiagramm der Vorhersagegenauigkeiten für den DAX.

In Abbildung 17 werden die Vergleiche graphisch mit einem Säulendiagramm veranschaulicht. Eine mit der Größe des Vorhersagezeitraums steigende Tendenz der Vorhersagegenauigkeit wird für alle Modelle sichtbar. Weitere Vorhersagen sind für die verbliebenen

Indizes MDAX in Tabelle A.4, SDAX in Tabelle A.5 und TECDAX in Tabelle A.6 erstellt worden (siehe Anhang A.5.1). Bei den restlichen Indizes performt der ET ab einer 5 Tages Prognose ebenfalls am besten. Die Vorhersagewerte sind meist leicht höher als bei der DAX Vorhersage. Die höchste Genauigkeit wird mit dem ET Algorithmus für einen Vorhersagezeitraum von 15 Tagen am SDAX mit einer Genauigkeit von 92.9 % erreicht. Weiterhin steigen die Vorhersagewerte auch mit zunehmender Zeitperiode bei allen Modellen. Für eine 1 Tages Prognose erzielt das SVM Modell weiterhin jeweils die höchsten oder zweit höchsten Werte. Die Mittelwerte der Vorhersagegenauigkeiten über alle Indizes werden in Tabelle A.7 vorgestellt (siehe Anhang A.5.2).

7.3. Konfusionsmatrix der besten Modelle

Die folgenden Konfusionsmatrizen beschreiben eine Auswertung der jeweils besten Modelle zu einer Vorhersageperiode. Somit können die markierten, besten Genauigkeitswerte der DAX Vorhersage aus Tabelle 3 analysiert und evaluiert werden. In den einzelnen Feldern der Matrizen wird die Anzahl der jeweiligen Klassifizierungen und das prozentuale Vorkommen der Klassifizierungen abgebildet.

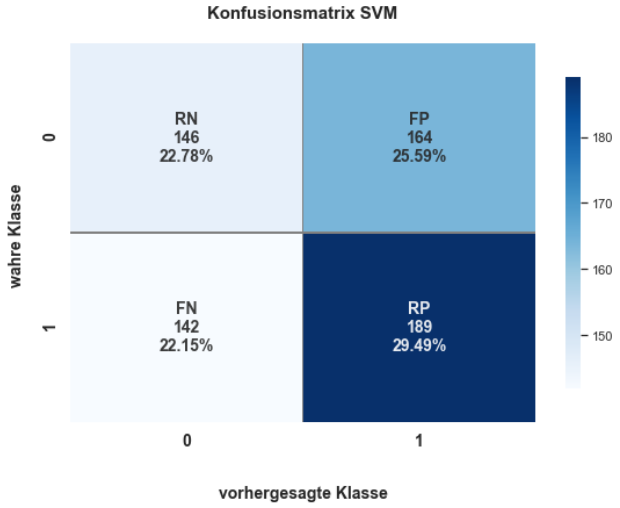

Abbildung 18: DAX Vorhersage 1 Tag mit SVM.

Die Konfusionsmatrix aus Abbildung 18 beschreibt die 1 Tages Prognose des SVM Modells auf den DAX. Die Genauigkeit der Vorhersage lag bei 52 %. Allgemein werden Trendvorhersagewerte über 50 % als lohnend angesehen (Novak; Velušček, 2016). Das SVM Modell übertrifft dieses Mindestmaß nur leicht, wobei die praktische Nutzbarkeit nicht nachgewiesen werden kann. Die Anzahl der falsch-negativ klassifizierten Daten ist fast genauso hoch wie die Anzahl der richtig-negativen Daten. Dies hat zur Folge, dass zukünftig fallende Kurse nicht gut erkannt werden. Der positive Vorhersagewert beträgt etwa 0.53, wobei das Setzen auf ausschließlich steigende Kurse eine Erfolgsrate von 53 % in Aussicht stellt.

Für die weiteren Vorhersagezeiträume erzielt das ET Modell die besten Leistungen. Während bei der 1 Tages Prognose falsch-positive Werte einen Anteil von etwa 25 % ausmachen, reduziert sich diese Rate auf unter 13 % bei den weiteren Prognosezeiträumen. Der Anteil falsch-negativer Werte wird mit steigendem Vorhersagezeitraum von 22 % auf unter 8 % reduziert. In Abbildung 19 wird die Evaluation der 5 Tages Prognose des ET Modells und in Abbildung 20 die Evaluation der 10 Tages Vorhersage gezeigt.

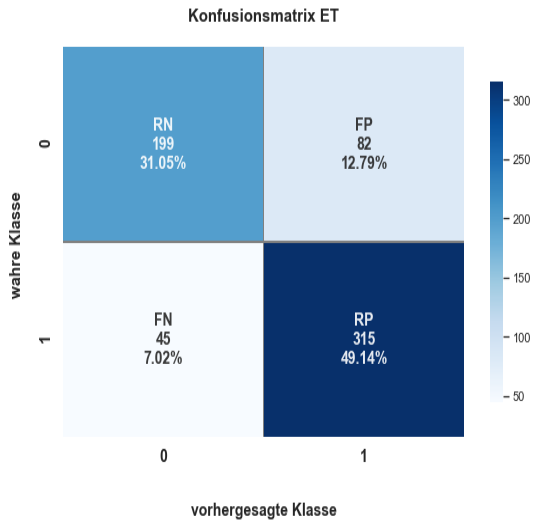

Abbildung 19: DAX Vorhersage 5 Tage mit ET.

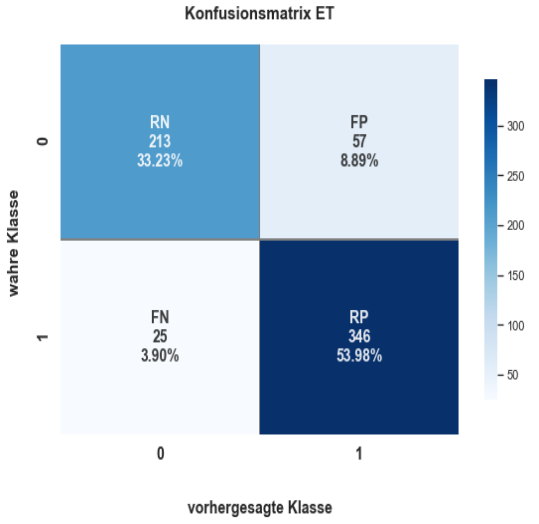

Abbildung 20: DAX Vorhersage 10 Tage mit ET.

Mit zunehmender Zeitperiode der Vorhersage werden falsch-positiv klassifizierte Werte des ET Modells weiter reduziert. Der Anteil falsch-negativer Werte steigt nach der 10 Tages Prognose um 1.5 % gegenüber der 15 Tages Vorhersage in Abbildung 21. Die geringste Anzahl an falsch-positiver Werte wurde in der 20 Tages Prognose in Abbildung 22 erreicht. Hier wurde auch die höchste Genauigkeit erzielt und die zweithöchste negative Vorhersage mit ca. 88 %. Mit über 92 % erzielte das ET Modell die höchste positive Vorhersage in einer 20 Tages Prognose für den DAX.

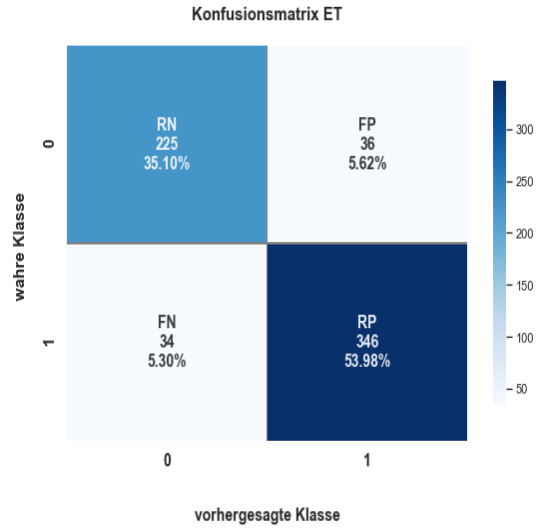

Abbildung 21: DAX Vorhersage 15 Tage mit ET.

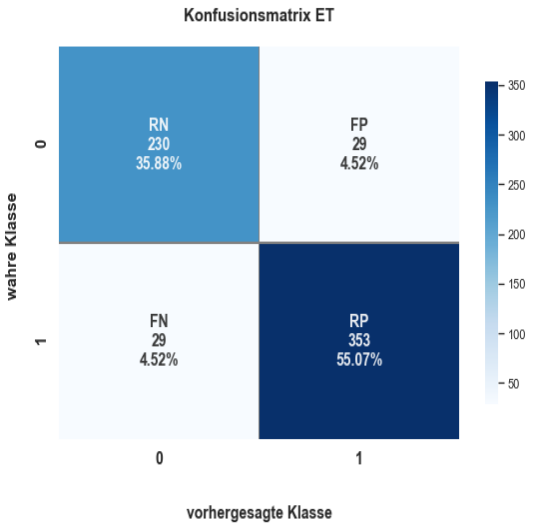

Abbildung 22: DAX Vorhersage 20 Tage mit ET.

Die Analyse der Konfusionsmatrizen ergibt, dass falsch-positive und falsch-negative Werte meist ungleich verteilt sind. Dabei erkennt das Modell entweder steigende oder fallende Kurse etwas besser. Dies gibt erste Hinweise darauf, dass eine Optimierung der Modellvorhersage mit einer Handelsstrategie etabliert werden kann. Bei Szenarien mit hohen falsch-positiven Werten, was bedeutet, dass steigende Kurse nicht so gut erkannt werden, kann auf fallende Kurse gesetzt werden. Für eine hohe Falsch-Negativ-Rate kann das Setzen auf steigende Kurse gewinnbringender sein. Um diese Tendenz zu bestätigen, kann die positive und negative Vorhersage aus den Konfusionsmatrizen berechnet werden (siehe Kapitel 5.5).

In Tabelle 4 werden die Performance-Maße Genauigkeit, positiver Vorhersagewert und negativer Vorhersagewert für eine DAX Prognose der besten Modelle vorgestellt.

Modell	Zeitraum	Genauigkeit	Pos. Vorhersage	Neg. Vorhersage
SVM	1	52.2621	**53.5410**	50.6944
ET	5	80.1872	79.3450	**81.5573**
ET	10	87.2075	85.8560	**89.4957**
ET	15	89.0796	**90.5759**	86.8725
ET	20	90.9516	**92.4083**	88.8030

Tabelle 4: Übersicht der prozentualen Performance für die besten Modelle über eine DAX Prognose.

Für jede Zeile der Tabelle wird das beste Modell zu einem Vorhersagezeitraum mit den erzielten Vorhersagewerten angegeben. Eine Handelsstrategie mittels Techniken des maschinellen Lernens kann in drei unterschiedlichen Formen etabliert werden. Falls eine allgemeine Strategie gewählt wird, bei der auf steigende und auf fallende Kurse gesetzt wird, sollte die Genauigkeit als Performance-Maß dienen. Diese gibt die allgemeine zu erwartende Performance des Vorhersagemodells an. Zudem kann eine Strategie gewählt werden, in der ausschließlich auf steigende Kurse gesetzt wird. Falls die Vorhersage des Modells fallend wäre, würde nicht gekauft werden. Dabei kann es vorkommen, dass kein Kauf getätigt wird, obwohl der Kurs gestiegen wäre. Diese Fehler entsprechen den falsch-negativ Werten und können ignoriert werden, da sie nur zu entgangenen Gewinnen führen und nicht zu Verlusten. Die Performance dieser Methode wird durch die Maße positive Vorhersage ausgedrückt. Das Vorgehen optimiert die allgemeine Handelsstrategie in einer 15 und 20 Tages Prognose mit dem ET Modell. Des Weiteren kann eine dritte Strategie entwickelt werden, bei der steigende Vorhersagen ignoriert werden und lediglich auf fallende Kurse gesetzt wird. Die falsch-positiv klassifizierten Daten müssen dann nicht mehr berücksichtigt werden. Die Vorhersage kann mit der negativen Vorhersage gemessen werden. Diese Strategie bietet einen Vorteil in der 5 und 10 Tages Prognose auf den DAX, da zu diesen Vorhersagezeiträumen die negative Vorhersage die Genauigkeit verbessert.

Eine Analyseübersicht der verbliebenen Indizes mit allen Performance-Maßen ist für den MDAX in Tabelle A.8, für den SDAX in Tabelle A.9 und für den TECDAX in Tabelle A.10 erstellt (siehe Anhang A.5.3). Dabei sind jeweils Werte der positiven oder der negativen Vorhersage meist höher als bei der DAX Vorhersage.

Die Auswertung macht deutlich, dass eine Modellvorhersage zu einem bestimmten Zeitraum immer entweder durch eine positive oder eine negative Vorhersage optimiert werden kann. Die höchste Performance wird mit einer positiven Vorhersage von etwa 94 % bei einer 15 Tages SDAX Prognose mit dem ET Modell gemessen. Die höchste negative Vorhersage findet sich mit 93 % in einer 20 Tages Prognose für den TECDAX. Bei einer Kurzzeitprognose erreicht das SVM Modell meist die höchsten Werte. Zwar wird in der 1 Tages Prognose für den SDAX der RF und für den TECDAX das KNN Modell als optimales Modell gelistet, so erzielt das SVM Modell jedoch in beiden Fällen ähnliche Werte. Die Differenz liegt dabei unter einem Prozent. Die höchste positive Vorhersage mit einem SVM Modell wird mit ca. 56 % bei dem MDAX erreicht.

8. Diskussion

Anhänger der Random Walk Theorie (RWT) gehen davon aus, dass jegliche Vorhersage der Preisbewegung einer Aktie eine Genauigkeit von etwa 50 % nicht überschreiten kann (Ampomah u. a., 2020). Bei der Untersuchung des deutschen Marktes über die Indizes DAX, MDAX, SDAX und TECDAX erzielt das SVM Modell die höchste durchschnittliche Genauigkeit von etwa 52 % in einer Kurzzeitprognose von einem Tag. Für mittelfristige Prognosezeiträume performt das ET Modell mit Genauigkeitswerten von durchschnittlich 81 % bis 91 % am besten. Dabei liegen die Genauigkeiten des ET Modells im Durchschnitt 1.5 % über denen des RF Modells. In einer mittelfristigen Prognose erzielt das SVM Modell mit maximal 81 % und das KNN Modell mit 84 % im Durchschnitt die niedrigsten Genauigkeiten.

Eine Untersuchung der negativen und positiven Vorhersagewerte der Modellprognosen hat gezeigt, dass einer der beiden Werte stets höher als die allgemeine Genauigkeit ist. Praktisch bedeutet dies, dass eine definierte Handelsstrategie zu einer höheren zu erwarteten Trefferrate führt. Statt bei jeder Vorhersage eine Kaufentscheidung in die entsprechende Richtung zu tätigen, kann beispielsweise eine Handelsstrategie etabliert werden, die nur bei prognostizierten, steigenden Kursen Kaufsignale indiziert. Dieses Vorgehen führt zu einer höheren Renditeerwartung. Nach Novak; Velušček (2016) kann eine erfolgreiche Handelsstrategie ab einer Vorhersagegenauigkeit von 60 % etabliert werden. Dabei wurden jedoch nur kurzfristige Handelszeiträume von mehreren Stunden bis zu wenigen Tagen getestet. Um eine qualifizierte Aussage über die tatsächlich zu erwartende Marktperformance treffen zu können, sollten die Vorhersagemodelle in einer realen Marktumgebung getestet werden.

Eine Analyse der Merkmalbedeutung des RF Modells brachte hervor, dass für mittelfristige Prognosen fundamentale Daten wie der Rohöl- und Goldkurs die Vorhersagegenauigkeit am effektivsten steigern. Ferner führte das Nutzen des IFO Index für die 20 Tages Prognose zu einer Leistungssteigerung. Die verwendeten technischen Indikatoren verbesserten ab einer 5 Tages Prognose die Leistung aller Modelle. Die Verwendung der Rendite führt zu der geringsten Verunreinigungsabnahme des RF Modells. Die Tendenz, dass die Einbeziehung von weiteren fundamentalen Daten zu einer Verbesserung führt, kann hier demonstriert werden. Hinzukommend ist durch den Einsatz von mehreren technischen Indikatoren eine Leistungssteigerung zu erwarten. Dabei können bis zu 60 Eingabemerkmale aus technischen Indikatoren und fundamentalen Daten ausgewählt werden (Zhong; Enke, 2019). Eine erhebliche Performancesteigerung für die mittelfristige Prognose durch das ET Modell ist in diesem Fall nicht zu erwarten. Für eine trendbasierte Aktienkursvorhersage sind Werte über 90 % bereits im oberen Feld der zu erwartenden Genauigkeiten (Rouf u. a., 2021). Eine Steigerung von $2-3$ Prozentpunkten wäre mit zusätzlichen Merkmalen für die mittelfristige Prognose schätzungsweise möglich. Durch die dynamischen Eigenschaften des Marktes sind Vorhersagegenauigkeiten um 100 % unwahrscheinlich und in einem realen Handelsumfeld ohnehin nicht zu erreichen. Je länger der Vorhersagezeitraum ist, desto mehr kann der Kursverlauf durch unerwartete Ereignisse beeinflusst werden.

Eine Kurzzeitprognose von einem Tag ist mit den herangezogenen Merkmalen nicht lohnenswert. In Novak; Velušček (2016) wurden Genauigkeitswerte von 60 % für ein SVM Modell erreicht. Die technischen Merkmale wurden hierbei aus den Tageshöchstkursen berechnet, um die Volatilität am Ende eines Handelstages auszugleichen (Novak; Velušček, 2016). Weiterhin wurden die Indikatoren mit speziell angepassten Zeitintervallen berechnet. Für die Kurzzeitprognose ist somit eine gesonderte Auswahl an Eingabemerkmalen und deren Bearbeitung notwendig. Auch der Zeitraum der untersuchten Datengrundlage sollte auf $2-3$ Jahre reduziert werden, damit nur kurzfristigere Trends berücksichtigt werden. Die Annahme dabei ist, dass weit zurückliegende historische Daten weniger Einfluss auf aktuelle Kurse haben (Lazzeri, 2021).

Ein maßgeblicher Faktor für die Modellperformance bei der Zeitreihenanalyse und Vorhersage ist die Aufteilung der Kursdaten in Trainings- und Testdaten. Für die Modellerstellung wurde der Trainingsdatensatz aus 70 % der Gesamtdaten gebildet und der Testdatensatz aus 30 %. Die Unterteilung erfolgt zufällig. Dieses Vorgehen ist bei der Aktienanalyse nicht unüblich und wird oft praktiziert, beispielsweise in Shubharthi Dey u. a. (2016) und Patel u. a. (2015), welche ähnlich hohe Prognosewerte veröffentlichten. In Shubharthi Dey u. a. (2016) wurde die Apple Aktie mit einer 60 und 90 Tages Prognose analysiert. Die Daten wurden ebenfalls zufällig in ein Trainingsset mit 80 % und ein Testset mit 20 % aufgeteilt. Dabei wurde der baumbasierte Ensemble Algorithmus XGBoost (eXtreme Gradient Boosting) zur trendbasierten Klassifikation getestet. Dieser erreichte Genauigkeitswerte von über 87 % und performte besser als ein RF, KNN und SVM Modell. Das XGBoost Modell verwendet die Boosting-Methode, bei der mehrere, schwach performende Entscheidungsbäume in einen einzigen besser performenden Entscheider zusammengeführt werden (Shubharthi Dey u. a., 2016). In einer Untersuchung von Ampomah u. a. (2020) wurden fünf Ensemble Methoden, darunter RF, ET und XGBoost auf dem amerikanischen und indischen Markt getestet. Der ET performte mit etwa 80 % bis 90 % am besten. Eine Untersuchung des deutschen Marktes mit dem XGBoost Modell wäre erstrebenswert, da dieser im Mittel ähnlich gut performte. Die Daten wurden kategorisiert, indem ein Trainingsdatensatz aus den ersten 70 % der Daten erstellt wurde und ein Testdatensatz aus den letzten 30 %. Mit einer 10-Schritte Kreuzvalidierung wurde das Modell auf den Trainingsdatensatz trainiert, indem die Zeitreihe in zehn Gruppen gegliedert wurde. Mit neun Gruppen wird das Modell trainiert, mit der letzten evaluiert. Dieser Schritt wird wiederum zehnmal wiederholt, wobei jeweils andere Trainingsgruppen verwendet werden (Ampomah u. a., 2020). Diese Unterteilung der Daten mit einer k-Schritte Kreuzvalidierung kann zu exakteren Angabe der Performanceleistung eines Modells in der Praxis führen. Dies ist darin begründet, dass der Testdatensatz ausschließlich den letzten Teil der Zeitreihe darstellt, und das Modell im Training keine Informationen über die Daten erhält. In einer Untersuchung des nepalesischen Marktes von Zhao (2021) mittels 17 ML Methoden wurde eine Unterteilung in Trainings- und Testdaten anhand des Zeitreihenverlaufes vorgenommen, aber keine zusätzliche Kreuzvalidierung verwendet. Die Trainingsdaten wurden mit den ersten 80 % der Zeitreihendaten gebildet und die Testdaten mit den letzten 20 %. Verwendete Modelle waren unter anderem SVM, RF, ET und XGBoost. Die Genauigkeit der 17 Modelle lag bei einer 1, 15 und 30 Tages Prognose meist um 50 % oder niedriger. Das XGBoost Modell performte mit maximal 60 % Genauigkeit am besten (Zhao, 2021). Der Vergleich der Untersuchungen zeigt, dass Vorhersagegenauigkeiten von ML Modellen bei der Zeitreihenanalyse stark von der Aufteilung der Trainings- und Testdaten abhängen.

Bei der Vorhersage des deutschen Marktes performt das KNN Modell am schlechtesten. Eine Vorhersage des koreanischen und amerikanischen Marktes von Nguyen; Yoon (2019) konnte zeigen, dass mit einem LSTM (Long Short-Term Memory) Netz deutlich höhere Genauigkeitswerte als mit einem einfachen KNN erzielt werden. Die Genauigkeit war dabei um über 8 % höher. Ein LSTM stellt ein erweitertes rekurrentes Netz dar. Um die Eingabesequenzen besser in einen Kontext setzen zu können, wird ein zusätzlicher interner Status, ein Gedächtnis, eingeführt. Dieser Status speichert Informationen über mehrere Zeitschritte. Die internen Berechnungen des neuronalen Netzes werden durch das Gedächtnis beeinflusst. Für neue, vorher nicht gesehene Eingaben müssen mehr Informationen aus dem Gedächtnis zur Aktualisierung der internen Funktionen verwendet werden. Falls ähnliche Eingaben aufeinander folgen, werden weniger vergangene Informationen abgerufen (Nguyen; Yoon, 2019).

9. Ausblick

Für eine weitere Aktienkursvorhersage am deutschen Markt kann ein Vergleich der Ensemble Modelle ET und XGBoost mit einem LSTM Netz erfolgen. Um die Eingabemerkmale spezifischer anzupassen, kann ein mittelfristiger Vorhersagezeitraum von 5 bis 30 Tagen gewählt werden. Zudem können technische Merkmale neben der Standardisierungsmethode mit weiteren Skalierungsmethoden transformiert werden. Die Polarisierung bietet einen Ansatz, bei dem ausgewählte Indikatoren wie der MACD Indikator in Werte +1 und -1 transformiert werden. Falls ein Indikator einen steigenden Zustand voraussieht, wird der aktuelle Wert in +1 umgewandelt, ansonsten in -1 (Shen; Shafiq, 2020). Die Arbeit von Patel u. a. (2015) konnte aufzeigen, dass die Polarisierung von bestimmten technischen Indikatoren zu einer Leistungssteigerung bei ML Modellen führt (Patel u. a., 2015). Neben der Nutzung von weiteren technischen Indikatoren und fundamentalen Daten ist das Nutzen von textuellen Daten gewinnbringend (Rouf u. a., 2021). Zudem kann eine k-Schritte Kreuzvalidierung für das Training der Modelle verwendet werden. Um eine weitere Untersuchung des deutschen Marktes vorzunehmen, ist die Prognose von Aktienkursen einzelner börsennotierter Unternehmen möglich. Es kann analysiert werden, ob eine Vorhersage von Einzelaktien im Durchschnitt höher als bei der Indexvorhersage ist. Die Vorhersagemodelle können mit den mittleren Genauigkeiten aus der Einzelaktienvorhersage evaluiert werden.

Die Performance von Vorhersagemodellen leidet oft darunter, dass Aktienkurse über eine bestimmte Zeitspanne nur leicht steigen oder fallen. Diese Daten sind schwer zu kategorisieren. In einer Arbeit von Zhang u. a. (2018) wurden vier Kategorien (aufwärts, abwärts, flach und unbekannt) eingeführt. In die flache und unbekannte Klasse sollen unlukrative und risikobehaftete Handelsmöglichkeiten aussortiert werden. Die vorgeschlagene Methode performte besser mit einer höheren Gewinnerwartung pro Handel als Veröffentlichungen mit herkömmlichen Ansatz (Zhang u. a., 2018). Zwar kann für ein Vorhersagemodell mit dem positiven Vorhersagewert die erwartete Performance bei dem Setzen auf steigende Kurse angegeben werden, die erwartete Rendite kann jedoch nicht genauer spezifiziert werden. Falls die Kurse bei richtiger Vorhersage oft nur leicht steigen, wäre eine niedrige Rendite zu erwarten. Eine Lösung des Problems wäre die Einführung der zusätzlichen Klassen. In diesem Fall würden nur noch steigende Kurse vorhergesagt werden mit denen eine hohe Renditeerwartung assoziiert wird.

Um eine genauere zu erwartende Performance auf ein reales Handelszenario zu geben, kann mit den erstellten Modellen ein ‚Backtesting' auf historischen Daten durchgeführt werden. Diese Methode führt dazu, dass für jede vorgeschlagene Kaufempfehlung eines Vorhersagemodells auf vergangenen Zeitreihendaten die Rendite gemessen wird. Die akkumulierten Renditen entsprechen dem zu erwartenden Gewinn, abzüglich der üblichen Handelskosten (Zhang u. a., 2018).

10. Literaturverzeichnis

AGRAWAL, Tanay, 2021. *Hyperparameter Optimization in Machine Learning: Make Your Machine Learning and Deep Learning Models More Efficient*. New York: Apress. Springer eBook Collection. ISBN 9781484265796. Abger. unter DOI: 10.1007/978-1-4842-6579-6.

ALPAYDIN, Ethem, 2021. *Machine learning*. Revised and updated edition. Cambridge, Massachusetts: The MIT Press. The MIT Press Essential Knowledge Ser. ISBN 9780262365369. Abger. unter DOI: https://doi.org/10.1515/9783110740196.

AMPOMAH, Ernest Kwame u. a., 2020. Evaluation of Tree-Based Ensemble Machine Learning Models in Predicting Stock Price Direction of Movement. *Information*. Jg. 11, Nr. 6. ISSN 2078-2489. Abger. unter DOI: 10.3390/info11060332.

BREIMAN, Leo, 2001. Random Forests. *Machine Learning*. Jg. 45, Nr. 1, S. 5–32. ISSN 0885-6125. Abger. unter DOI: 10.1023/A:1010933404324.

COLBY, R.W.; MEYERS, T.A., 1988. *The Encyclopedia of Technical Market Indicators*. Dow Jones-Irwin. ISBN 9781556230493. Auch verfügbar unter: https://books.google.de/books?id=8IIYAAAAIAAJ.

CORTES, Corinna; VAPNIK, Vladimir, 1995. Support-vector networks. *Machine Learning*. Jg. 20, Nr. 3, S. 273–297. ISSN 0885-6125. Abger. unter DOI: 10.1007/BF00994018.

CUTLER, Adele u. a., 2012. Random Forests. In: *Ensemble Machine Learning*. Springer, Boston, MA, S. 157–175. Abger. unter DOI: 10.1007/978-1-4419-9326-7_5.

DA SILVA, Ivan Nunes, 2017. *Artificial Neural Networks: A Practical Course*. Cham: Springer. Springer eBook Collection Engineering. ISBN 9783319431628. Abger. unter DOI: 10.1007/978-3-319-43162-8.

DENG, Li; YU, Dong, 2014. *Deep learning: Methods and applications*. Boston, Mass.: Now Publ. Foundations and trends in signal processing. ISBN 9781601988140.

FAMA, Eugene F., 1970. Efficient Capital Markets: A Review of Theory and Empirical Work. *The Journal of Finance*. Jg. 25, Nr. 2, S. 383. ISSN 00221082. Abger. unter DOI: 10.2307/2325486.

FROCHTE, Jörg, 2018. *Maschinelles Lernen: Grundlagen und Algorithmen in Python*. München: Hanser. ISBN 978-3-446-45705-8.

GEURTS, Pierre u. a., 2006. Extremely randomized trees. *Machine Learning*. Jg. 63, Nr. 1, S. 3–42. ISSN 0885-6125. Abger. unter DOI: 10.1007/s10994-006-6226-1.

HU, Yong u. a., 2015. Application of evolutionary computation for rule discovery in stock algorithmic trading: A literature review. *Applied Soft Computing*. Jg. 36, S. 534–551. ISSN 1568-4946. Abger. unter DOI: https://doi.org/10.1016/j.asoc.2015.07.008.

HUANG, Yuxuan u. a., 2022. *Machine Learning for Stock Prediction Based on Fundamental Analysis*. Abger. unter DOI: 10.1109/SSCI50451.2021.9660134.

HÜLSMANN, Marco u. a., 2011. General Sales Forecast Models for Automobile Markets Based on Time Series Analysis and Data Mining Techniques. In: PERNER, Petra (Hrsg.). *Advances in data mining*. Berlin und Heidelberg: Springer. Bd. 6870, S. 255–269. Lecture notes in computer science Lecture notes in artifificial intelligence. ISBN 978-3-642-23183-4. Abger. unter DOI: 10.1007/978-3-642-23184-1_20.

KOTU, Vijay, 2015. *Predictive analytics and data mining: Concepts and practice with RapidMiner*. Waltham, MA: Morgan Kaufmann. MyiLibrary. ISBN 9780128016503. Auch verfügbar unter: http://lib.myilibrary.com?id=666031.

KUMBURE, Mahinda Mailagaha u. a., 2022. Machine learning techniques and data for stock market forecasting: A literature review. *Expert Systems with Applications*. Jg. 197, S. 116659. ISSN 09574174. Abger. unter DOI: 10.1016/j.eswa.2022.116659.

LAZZERI, Francesca, 2021. *Machine learning for time series forecasting with Python*. Indianapolis, Indiana: Wiley. ISBN 9781119682394. Auch verfügbar unter: https://onlinelibrary.wiley.com/doi/book/10.1002/9781119682394.

LO, Andrew W., 2011. *A Non-Random Walk Down Wall Street*. Princeton: Princeton University Press. EBL-Schweitzer. ISBN 9781400829095.

NGUYEN, Thi-Thu; YOON, Seokhoon, 2019. A Novel Approach to Short-Term Stock Price Movement Prediction using Transfer Learning. *Applied Sciences*. Jg. 9, Nr. 22, S. 4745. ISSN 2076-3417. Abger. unter DOI: 10.3390/app9224745.

NILSSON, Nils J., 1990. *The mathematical foundations of learning machines*. San Mateo, Calif: Morgan Kaufmann. ISBN 1558601236.

NOVAK, Marija; VELUŠČEK, Dejan, 2016. Prediction of stock price movement based on daily high prices. *Quantitative Finance*. Jg. 16, Nr. 5, S. 793–826. ISSN 1469-7688. Abger. unter DOI: 10.1080/14697688.2015.1070960.

PADHI, Dushmanta Kumar u. a., 2021. A Fusion Framework for Forecasting Financial Market Direction Using Enhanced Ensemble Models and Technical Indicators. *Mathematics*. Jg. 9, Nr. 21. ISSN 2227-7390. Abger. unter DOI: 10.3390/math9212646.

PATEL, Jigar u. a., 2015. Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques. *Expert Systems with Applications*. Jg. 42, Nr. 1, S. 259–268. ISSN 09574174. Abger. unter DOI: 10.1016/j.eswa.2014.07.040.

PATIL, Pankaj Rambhau u. a., 2021. A Literature Review on Machine Learning Techniques and Strategies Applied to Stock Market Price Prediction. In: MATHUR, Rajeev u. a. (Hrsg.). *Emerging Trends in Data Driven Computing and Communications*. Singapore: Springer Singapore, S. 121–135. ISBN 978-981-16-3915-9.

PRASAD, Venkata Vara u. a., 2021. Prediction of Stock Prices Using Statistical and Machine Learning Models: A Comparative Analysis. *The Computer Journal*. Jg. 65, Nr. 5, S. 1338–1351. ISSN 0010-4620. Abger. unter DOI: 10.1093/comjnl/bxab008.

RICHTER, Stefan, 2019. *Statistisches und maschinelles Lernen: Gängige Verfahren im Überblick*. 1. Aufl. 2019. Springer Berlin Heidelberg. ISBN 9783662593547.

ROSENBLATT, Frank, 1963. PRINCIPLES OF NEURODYNAMICS. PERCEPTRONS AND THE THEORY OF BRAIN MECHANISMS. *American Journal of Psychology*. Jg. 76, S. 705.

ROUF, Nusrat u. a., 2021. Stock Market Prediction Using Machine Learning Techniques: A Decade Survey on Methodologies, Recent Developments, and Future Directions. *Electronics*. Jg. 10, Nr. 21. ISSN 2079-9292. Abger. unter DOI: 10.3390/electronics10212717.

RUDER, Sebastian, 2016. *An overview of gradient descent optimization algorithms.* arXiv. Auch verfügbar unter: http://arxiv.org/abs/1609.04747.

SCHUSTER, Thomas, 2015. *Finanzierung: Anleihen, Aktien, Optionen*. Berlin, Heidelberg: Springer Gabler. Springer eBook Collection. ISBN 9783662462393. Abger. unter DOI: 10.1007/978-3-662-46239-3.

SEILER, Christian; WOHLRABE, Klaus, 2013. Das ifo Geschäftsklima und die deutsche Konjunktur. *ifo Schnelldienst*. Jg. 66, Nr. 18, S. 17–21. ISSN 0018-974X. Auch verfügbar unter: http://hdl.handle.net/10419/165324.

SHANNON, Claude Elwood; WEAVER, Warren, 1998. *The mathematical theory of communication*. 21. print. Urbana: Univ. of Illinois Press. ISBN 9780252725487.

SHEN, Jingyi; SHAFIQ, M. Omair, 2020. Short-term stock market price trend prediction using a comprehensive deep learning system. *Journal of big data*. Jg. 7, Nr. 1, S. 66. ISSN 2196-1115. Abger. unter DOI: 10.1186/s40537-020-00333-6.

SHUBHARTHI DEY u. a., 2016. *Forecasting to Classification: Predicting the direction of stock market price using Xtreme Gradient Boosting*. Unpublished. Abger. unter DOI: 10.13140/RG.2.2.15294.48968.

SPREMANN, Klaus; GANTENBEIN, Pascal, 2014. *Finanzmärkte: Grundlagen, Instrumente, Zusammenhänge*. 3., überarb. Aufl. Konstanz: UVK-Verl.-Ges. UTB Betriebswirtschaftslehre Volkswirtschaftslehre. ISBN 3825286088.

THAWORNWONG, Suraphan; ENKE, David, 2004. The adaptive selection of financial and economic variables for use with artificial neural networks. *Neurocomputing*. Jg. 56, S. 205–232. ISSN 09252312. Abger. unter DOI: 10.1016/j.neucom.2003.05.001.

VAPNIK, Vladimir N., 1995. *The Nature of Statistical Learning Theory*. New York, NY: Springer. Springer eBook Collection Mathematics and Statistics. ISBN 978-1-4757-2442-4. Abger. unter DOI: 10.1007/978-1-4757-2440-0.

WENG, Bin u. a., 2017. Stock market one-day ahead movement prediction using disparate data sources. *Expert Systems with Applications*. Jg. 79, S. 153–163. ISSN 09574174. Abger. unter DOI: 10.1016/j.eswa.2017.02.041.

ZHANG, Jing u. a., 2018. A novel data-driven stock price trend prediction system. *Expert Systems with Applications*. Jg. 97, S. 60–69. ISSN 09574174. Abger. unter DOI: 10.1016/j.eswa.2017.12.026.

ZHAO, Shunan, 2021. Nepal Stock Market Movement Prediction with Machine Learning. In: *2021 the 5th International Conference on Information System and Data Mining*. New York, NY, USA: ACM, S. 1–7. ISBN 9781450389549. Abger. unter DOI: 10.1145/3471287.3471289.

ZHONG, Xiao; ENKE, David, 2019. Predicting the daily return direction of the stock market using hybrid machine learning algorithms. *Financial Innovation*. Jg. 5. Abger. unter DOI: 10.1186/s40854-019-0138-0.

A. Anhang

A.1. Formale technische Indikatoren

Merkmalname	Formel *	Referenz
Volumen	V	(Zhong; Enke, 2019)
Rendite	$\frac{S_t}{S_{t-n}} \times 100$	(Zhong; Enke, 2019)
SMA (10)	$\frac{S_t + S_{t-1} + ... + S_{t-9}}{n}$	(Patel u. a., 2015)
EMA (10)	$\mathrm{EMA}_{t-1} + \frac{2}{11} \times (S_t - \mathrm{EMA}_{t-1})$	(Novak; Velušček, 2016)
EMA (20)	$\mathrm{EMA}_{t-1} + \frac{2}{21} \times (S_t - \mathrm{EMA}_{t-1})$	(Novak; Velušček, 2016)
WMA (10)	$\frac{10 \times S_t + 9 \times S_{t-1} + ... + S_{t-10}}{10 + 9 + ... + 1}$	(Novak; Velušček, 2016)
Momentum (10)	$S_t - S_{t-9}$	(Novak; Velušček, 2016)
SAR	$\mathrm{SAR}_{t-1} + 0.02 \times (\mathrm{EP} - \mathrm{SAR}_{t-1})$	(Patel u. a., 2015)
RSI	$100 - \frac{100}{1 + \left(\sum_{i=0}^{n-1} \mathrm{UP}_{t-i}/n\right) / \left(\sum_{i=0}^{n-1} \mathrm{DW}_{t-i}/n\right)}$	(Patel u. a., 2015)
ROC	$\frac{S_t + S_{t-10}}{S_{t-10}} \times 100$	(Patel u. a., 2015)
%R	$\frac{H_n + S_t}{H_n + T_n} \times 100$	(Patel u. a., 2015)
OBV	$\mathrm{OBV}_{t-n} + \begin{cases} V & , falls \quad S_t > S_{t-n} \\ 0 & , falls \quad S_t = S_{t-n} \\ -V & , falls \quad S_t < S_{t-n} \end{cases}$	(Shen; Shafiq, 2020)
MACD	$\mathrm{MACD}(n)_{t-1} + \frac{2}{n+1} \times (\mathrm{DIFF}_t - \mathrm{MACD}(n)_{t-1})$	(Patel u. a., 2015)
MACD SIGNAL	$MACD \quad Signallinie$	(Shen; Shafiq, 2020)
MACD HIST	$MACD \quad Histogramm$	(Shen; Shafiq, 2020)
CCI	$\frac{M_t + SM_t}{0.015\mathrm{D}_t}$	(Patel u. a., 2015)
ADOSC	$\frac{H_t - S_{t-1}}{H_t - \mathrm{L}_t}$	(Shen; Shafiq, 2020)

* S_t ist der Schlusskurs, T_t der tiefste Kurs und H_t der höchste Kurs zu einem Zeitpunkt t. EP ist der Extrempunkt also der höchste Wert in einem aktuellen Aufwärtstrend oder der niedrigste Wert eines Abwärtstrends. UP_t beschreibt eine Preisänderung nach oben, während DW_t einen Änderung nach unten zu einem bestimmten Zeitpunkt t angibt.
Weiter gilt:
$\mathrm{DIFF}_t = \mathrm{EMA}(12)_t - \mathrm{EMA}(26)_t$, $\mathrm{M}_t = \frac{H_t + L_t + S_t}{3}$, $\mathrm{SM}_t = \frac{(\sum_{i=1}^{n} + M_{t-i+1})}{n}$ und $\mathrm{D}_t = \frac{(\sum_{i=1}^{n} |M_{t-i+1} - \mathrm{SM}_t|)}{n}$.

Tabelle A.1: Formale Beschreibung der verwendeten technischen Indikatoren mit angegebenen Referenzen.

A.2. DAX Statistik

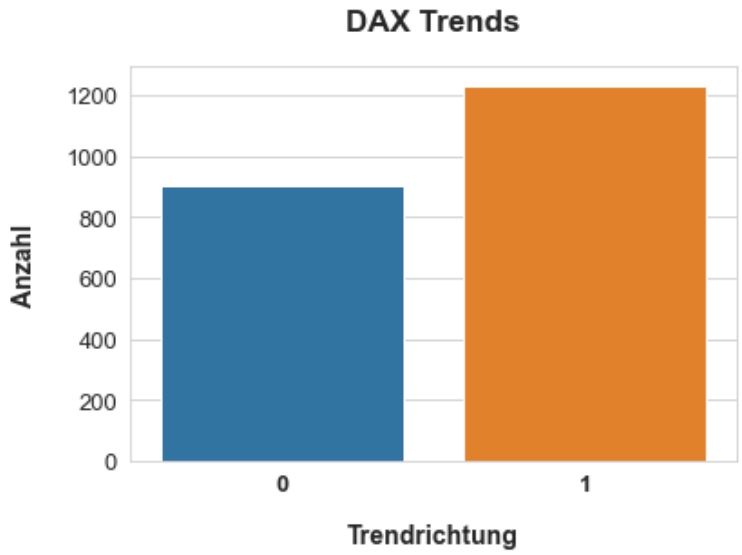

Abbildung A.1: Unterteilung der DAX Trends mit steigendem 5 Tage Trend (1) und fallendem Trend (0).

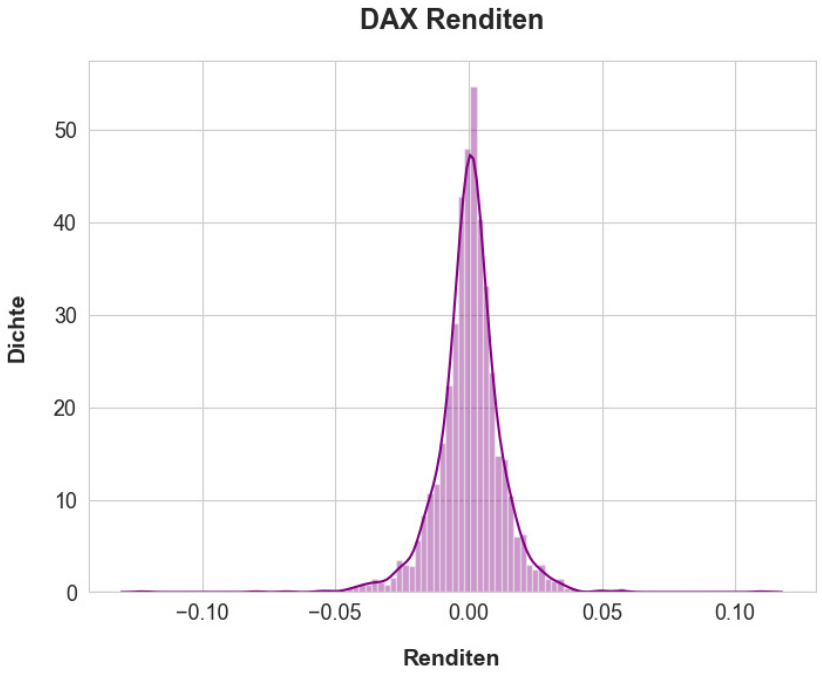

Abbildung A.2: Verteilung der DAX Rendite über neun Jahre.

A.3. Evaluation der Indikatoren

Anzahl Merkmale	SVM	RF	ET	KNN	Merkmale
1	66.5102	67.2926	67.2926	65.1017	Schlusskurs
2	66.3537	68.3881	68.3881	64.6322	Rendite
3	66.5102	74.1784	72.1440	66.0407	SMA 10
4	66.5102	74.6479	73.7089	66.5102	EMA 10
5	66.8232	76.9953	77.9343	65.8842	EMA 20
6	66.6667	77.3083	77.3083	64.6322	WMA 10
7	66.6667	78.4038	78.0908	66.5102	Momentum 10
8	66.6667	79.4992	80.1252	65.8842	SAR
9	66.3537	81.3772	82.7856	67.4491	RSI
10	67.6056	82.1596	83.4116	65.8842	ROC
11	67.1362	82.3161	83.8811	65.7277	%R
12	70.8920	84.5070	85.4460	65.4147	MACD
13	70.5790	85.2895	86.0720	68.5446	MACD_SIGNAL
14	71.2050	85.6025	87.6369	68.2316	MACD_HIST
15	72.1440	85.1330	87.6369	69.6401	CCI
16	74.9609	88.2629	90.4538	74.0219	Gold Schlusskurs
17	81.2207	89.3584	91.8623	83.0986	Rohöl Schlusskurs
18	84.6635	90.7668	92.3318	86.6980	IFO Index

Tabelle A.2: Performance der Merkmale bei einer 20 Tages Prognose am TECDAX.

Anzahl Merkmale	SVM	RF	ET	KNN	Merkmale
1	51.6381	51.4821	51.3261	51.6381	Schlusskurs
2	50.5460	50.7020	51.4821	50.0780	Rendite
3	51.3261	50.5460	51.3261	47.8939	Volumen
4	51.4821	48.9860	51.1700	49.4540	SMA 10
5	51.1700	49.9220	50.2340	49.1420	EMA 10
6	52.2621	50.0780	47.4259	48.2059	EMA 20
7	53.5101	49.7660	47.5819	49.2980	WMA 10
8	50.0780	50.8580	48.2059	51.1700	Momentum 10
9	51.3261	50.5460	48.2059	51.6381	SAR
10	51.9501	50.7020	47.8939	49.4540	RSI
11	51.3261	49.2980	49.6100	50.3900	ROC
12	50.7020	50.0780	47.1139	52.4181	%R
13	49.6100	49.7660	46.6459	50.8580	MACD
14	49.9220	49.4540	48.2059	50.2340	OBV
15	50.7020	50.5460	47.4259	49.6100	MACD_SIGNAL
16	50.8580	50.3900	47.8939	51.6381	MACD_HIST
17	51.0140	48.8300	48.2059	49.9220	CCI
18	53.9782	49.2980	48.2059	51.3261	ADOSC
19	53.1981	47.2699	48.5179	49.4540	Gold Schlusskurs
20	52.5741	48.3619	47.5819	51.1700	Rohöl Schlusskurs
21	52.2621	48.9860	48.3619	51.1700	IFO Index

Tabelle A.3: Performance der Merkmale bei einer 1 Tages Prognose am DAX.

A.4. RF Parameter

A.4.1. Parameteroptimierung

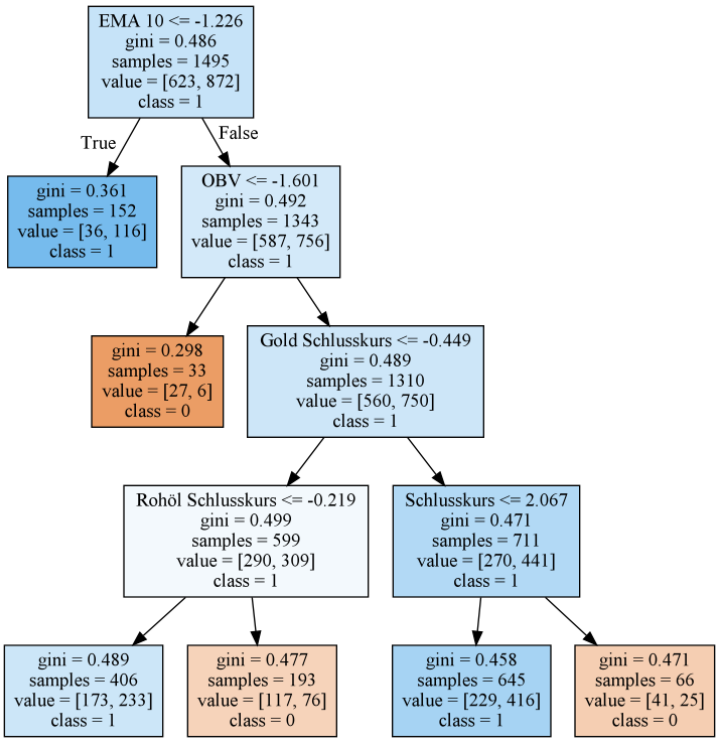

Abbildung A.3: Ein beschnittener Entscheidungsbaum aus einem Random Forest zur DAX Vorhersage.

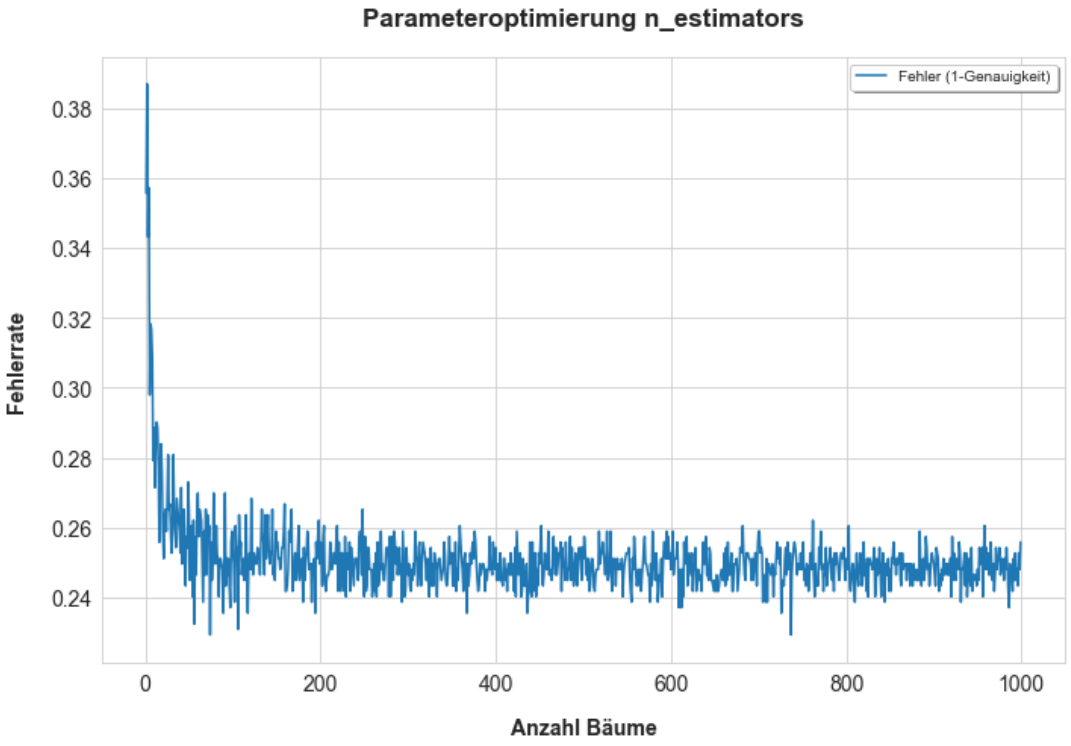

Abbildung A.4: Hyperparameteroptimierung von n_estimators eines Random Forest bei der DAX Kursvorhersage.

A.4.2. Bedeutung der Merkmale

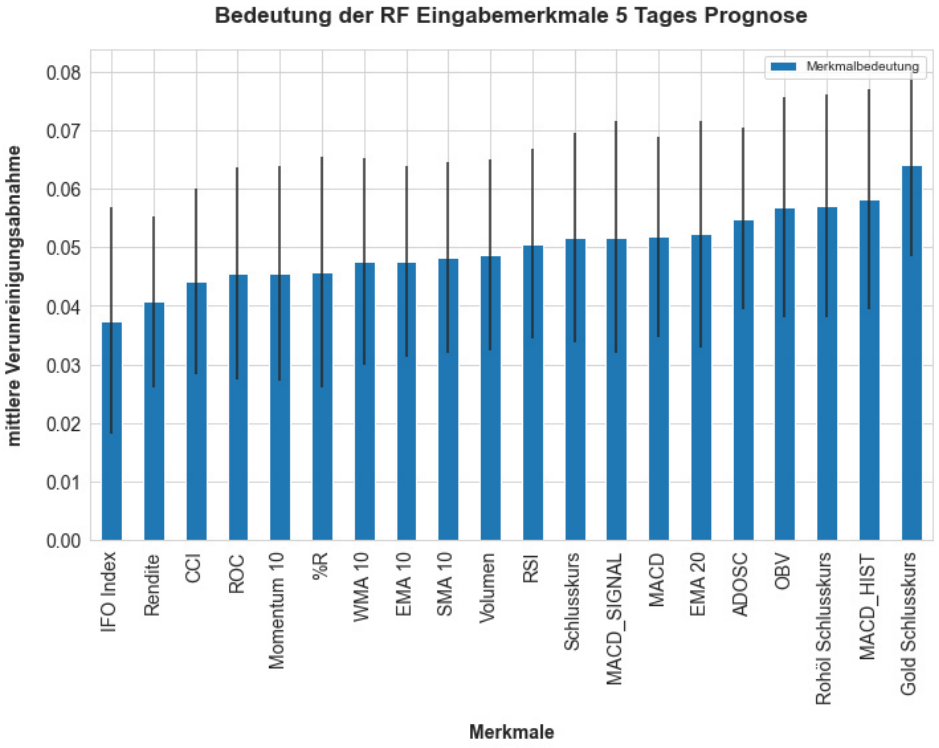

Abbildung A.5: Die Wichtigkeit der Merkmale des RF bei der DAX 5 Tages Prognose mit Standardabweichung.

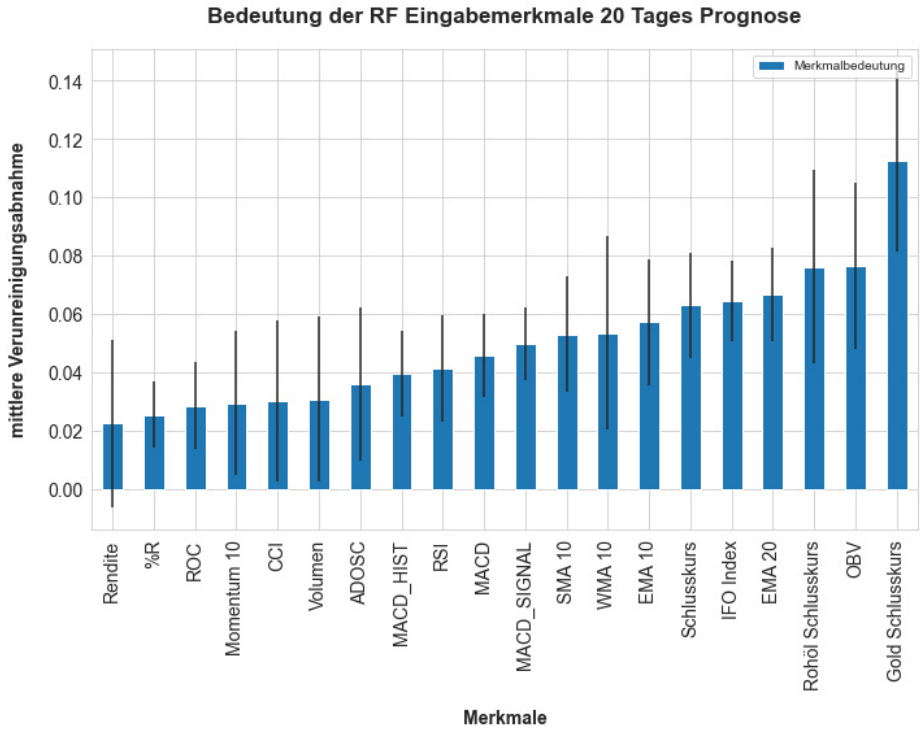

Abbildung A.6: Die Wichtigkeit der Merkmale des RF bei der DAX 20 Tages Prognose mit Standardabweichung.

A.5. Ergebnisse

A.5.1. Vergleich der Genauigkeit

Vorhergesagte Tag	1	5	10	15	20
SVM	**53.9062**	69.6875	76.4062	78.2812	81.8750
RF	51.0938	77.5000	85.9375	88.2812	89.3750
ET	52.5000	**81.2500**	**88.2812**	**89.5312**	**90.6250**
KNN	50.4687	71.2500	78.1250	82.3438	81.7188

Tabelle A.4: Vergleich der Vorhersagemodelle am MDAX mittels der Genauigkeit in Prozent.

Vorhergesagte Tag	1	5	10	15	20
SVM	52.7387	72.7700	78.8732	81.2207	84.1941
RF	**54.4601**	81.3772	87.0110	90.7668	89.8279
ET	52.2692	**81.8466**	**88.7324**	**92.9577**	**91.5493**
KNN	51.0172	70.5790	78.8732	82.6291	85.1330

Tabelle A.5: Vergleich der Vorhersagemodelle am SDAX mittels der Genauigkeit in Prozent.

Vorhergesagte Tag	1	5	10	15	20
SVM	50.7042	71.2050	74.0219	80.7512	84.6635
RF	50.3912	77.9343	83.8811	87.7934	90.7668
ET	49.1393	**81.6901**	**85.4460**	**89.9844**	**92.3318**
KNN	**50.8607**	71.8310	75.4304	81.0642	86.6980

Tabelle A.6: Vergleich der Vorhersagemodelle am TECDAX mittels der Genauigkeit in Prozent.

A.5.2. Mittelwert der Vorhersagegenauigkeiten

Vorhergesagte Tag	1	5	10	15	20
SVM	**52.4028**	71.0609	74.0219	80.1881	81.8315
RF	51.2328	78.2747	85.2292	88.6292	89.8013
ET	50.6926	**81.2435**	**87.4168**	**89.9844**	**91.3644**
KNN	50.8792	70.3807	77.5299	81.4781	84.0193

Tabelle A.7: Durchschnittliche Vorhersagegenauigkeit der Modelle für alle deutschen Indizes.

A.5.3. Vergleich mit allen Maßen

Modell	Zeitraum	Genauigkeit	Pos. Vorhersage	Neg. Vorhersage
SVM	1	53.9063	**56.1855**	50.3968
ET	5	81.2500	**82.7319**	78.9682
ET	10	88.2812	87.2817	**89.9581**
ET	15	89.5312	**90.5000**	87.9166
ET	20	90.6250	**93.9086**	85.3658

Tabelle A.8: Übersicht der prozentualen Performance für die besten Modelle über eine MDAX Prognose.

Modell	Zeitraum	Genauigkeit	Pos. Vorhersage	Neg. Vorhersage
RF	1	54.4601	**55.9808**	51.5837
ET	5	81.8466	**84.6347**	77.2727
ET	10	88.7323	88.1927	**89.7321**
ET	15	92.9577	**94.3349**	90.5579
ET	20	91.5492	**92.4528**	89.7674

Tabelle A.9: Übersicht der prozentualen Performance für die besten Modelle über eine SDAX Prognose.

Modell	Zeitraum	Genauigkeit	Pos. Vorhersage	Neg. Vorhersage
KNN	1	50.8607	**54.6916**	45.4887
ET	5	81.6901	**83.4123**	78.3410
ET	10	85.4460	**85.6818**	84.9246
ET	15	89.9843	**91.5094**	86.9767
ET	20	92.3318	91.9908	**93.0693**

Tabelle A.10: Übersicht der prozentualen Performance für die besten Modelle über eine TECDAX Prognose.

A.6. Skripte zur Aktienkursvorhersage

Auszug aus dem Skript der Vorhersagemodelle:

```
import numpy as np
from numpy.random import seed
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier,
    ExtraTreesClassifier
from keras.layers import Dense
from keras.models import Sequential
from sklearn.metrics import confusion_matrix

def svm(X_train:list, X_test:list, y_train:list, y_test:list):
    """Creates a support vector classifier and trains the model
       with the training data. The predictions are returned in
       the form of a confusion matrix.
    """
    model = SVC(C=100,
                gamma=0.1,
                kernel="rbf",
                random_state=42)

    model.fit(X_train, y_train)
    predictions = model.predict(X_test)
    return confusion_matrix(y_test, predictions)

def rf(X_train:list, X_test:list, y_train:list, y_test:list):
    """ Creates a random forest classifier. """
    model = RandomForestClassifier(n_estimators=1000,
                                   oob_score=True,
                                   criterion="gini",
                                   random_state=42)
    model.fit(X_train, y_train)
    predictions = model.predict(X_test)
    return confusion_matrix(y_test, predictions)

def extra_tree(X_train:list, X_test:list, y_train:list, y_test:list):
    """ Creates a extra tree classifier. """
    model = ExtraTreesClassifier(n_estimators=1000,
                                 criterion="gini",
                                 random_state=42)

    model.fit(X_train, y_train)
    predictions = model.predict(X_test)
    return confusion_matrix(y_test, predictions)

def knn(X_train:list, X_test:list, y_train:list, y_test:list):
    """ Creates a artificial neural network. """
    seed(2)
    model = Sequential()
    model.add(Dense(100, activation="sigmoid",
                    input_shape=(X_train.shape[1],)))
    model.add(Dense(1, activation="sigmoid"))
    model.compile(optimizer="adam", loss="binary_crossentropy",
                  metrics=["accuracy"])

    model.fit(x=X_train, y=y_train,
              epochs=4500, batch_size=200)
    predictions = model.predict(X_test)
    return confusion_matrix(y_test, np.round(predictions))
```

Auszug aus dem Skript der Merkmalextraktion:

```
import numpy as np
import talib as ta
from stocks.recources import all_recources

def extract_features(stockDf:pd.Dataframe, days_pred:int):
    """ Takes stock prices and calculates technical indicators
        and determines the true forecast values. Further
        fundamental data are attached.
    """
    time_period = 10              # timeperiod technical-indicators
    feature_parameter = "Close"   # inputparameter for
                                  # technical-indicators

    # convert volume from int to float
    stockDf.Volume = stockDf.Volume.astype('float64')

    # calculate return
    stockDf['Rendite'] = stockDf['Close'].pct_change()

    # calculate technical indicators
    inputs = stockDf[feature_parameter]
    stockDf[f"SMA {time_period}"] = ta.SMA(inputs,
                                           timeperiod=time_period)
    stockDf[f"EMA {time_period}"] = ta.EMA(inputs,
                                           timeperiod=time_period)
    stockDf[f"EMA {20}"]          = ta.EMA(inputs,
                                           timeperiod=20)
    stockDf[f"WMA {time_period}"] = ta.WMA(inputs,
                                           timeperiod=time_period)
    stockDf[f"Momentum {time_period}"] = ta.MOM(inputs,
                                          timeperiod=time_period)
    stockDf["SAR"] = ta.SAR(stockDf["High"], stockDf["Low"],
                            acceleration=0.02, maximum=0.2)
    stockDf["RSI"] = ta.RSI(stockDf["Close"], timeperiod=14)
    stockDf["ROC"] = ta.ROC(stockDf["Close"], timeperiod=10)
    stockDf["%R"]  = ta.WILLR(stockDf["High"], stockDf["Low"],
                              stockDf["Close"], timeperiod=14)
    stockDf["OBV"] = ta.OBV(stockDf["Close"], stockDf["Volume"])
    stockDf["MACD"], stockDf["MACD_SIGNAL"],
    stockDf["MACD_HIST"] = ta.MACD(stockDf["Close"], fastperiod=12,
                                   slowperiod=26, signalperiod=9)
    stockDf["CCI"]   = ta.CCI(stockDf["High"], stockDf["Low"],
                              stockDf["Close"], timeperiod = 14)
    stockDf["ADOSC"] = ta.ADOSC(stockDf["High"], stockDf["Low"],
                                stockDf["Close"],stockDf["Volume"],
                                fastperiod=3, slowperiod=10)

    # add ressources  and ifo index
    stockDf = all_recources(stockDf=stockDf)

    # closeprice prediction  (up -> 1 , down -> 0)
    stockDf['Prediction'] = np.where(stockDf[feature_parameter].shift
                    (-days_pred) > stockDf[feature_parameter], 1, 0)
    stockDf = stockDf.dropna()

    return stockDf
```

Auszug aus dem Skript der Datengenerierung:

```python
import pandas as pd
import yfinance as yf
from stocks.features import extract_features

class StockLoader:
    def __init__(self, wkns:list, names:list, start:str, stop:str):
        self.wkns = wkns
        self.names = names
        self.start = start
        self.stop = stop
        self.df_stocks = self.read_merge()

    def read_merge(self) -> pd.DataFrame:
        """ This function uses yfinance to get the stock information
            from the Yahoo Finance API.
        """
        df_list = []
        for wkn in self.wkns:
            aktie = yf.Ticker(wkn)
            df = aktie.history(start=self.start, end=self.stop)
            df_list.append(df)
        stocks = pd.concat(df_list, axis=1, keys=self.names)
        stocks.columns.names = ['Stock Ticker', 'Stock Info']
        return stocks

def save_data(name:str, days_pred:int) -> pd.DataFrame:
    """ The function loads four indices values and attaches the
        calculated features for a singel index in a dataframe.
    """

    indices = ['^GDAXI', '^MDAXI','^SDAXI','^TECDAX']
    indices_names = ['DAX', 'MDAX', 'SDAX', 'TECDAX']

    stockDf = StockLoader(wkns=indices,
                          names=indices_names,
                          start="2013-06-01",
                          stop="2022-01-01").df_stocks
    stockDf = pd.DataFrame(stockDf[name][['Open','High','Low',
                                          'Close', 'Volume']])
    # calculates features and attaches them
    stockDf = extract_features(stockDf,days_pred=days_pred)

    return stockDf
```