Vergleich von Open Source MLOps Tools zur Unterstützung von Machine Learning basierten Zeitreihenanalysen

Erik Autenrieth
Master Informatik
Hochschule Bonn-Rhein-Sieg
Sankt Augustin, Deutschland
erik.autenrieth@smail.inf.h-brs.de

Zusammenfassung*—Projekte des maschinellen Lernens (ML), insbesondere im Bereich der Zeitreihenanalyse, gewinnen heute zunehmend an Bedeutung. Die Bereitstellung solcher Projekte in einer Produktionsumgebung mit dem gleichen Automatisierungsgrad wie bei klassischen Softwareprojekten ist ein komplexes Unterfangen. Die Umsetzung in Produktionsumgebungen erfordert neben klassischen DevOps auch Machine Learning Operation (MLOps) Technologien und Werkzeuge. Ziel dieser Studie ist es, einen umfassenden Überblick über verfügbare MLOps Tools zu bieten und einen spezifischen Techstack für Zeitreihen ML Projekte zu entwickeln. Es werden aktuelle Trends und Werkzeuge im Bereich MLOps durch eine multivokale Literaturrecherche (MLR) untersucht und analysiert. Die Studie identifiziert passende MLOps Werkzeuge und Methoden für die Zeitreihenanalyse und präsentiert eine spezifische Implementierung einer MLOps Pipeline für die Aktienkursprognose des S&P 500. MLOps und DevOps Tools nehmen eine essenzielle Rolle bei der effektiven Konstruktion und Verwaltung von ML Pipelines ein. Bei der Auswahl geeigneter Werkzeuge ist stets eine spezifische Anpassung an die jeweiligen Projektanforderungen erforderlich. Die Bereitstellung einer detaillierten Darstellung der aktuellen MLOps Tool Landschaft erweist sich hierbei als wertvolle Ressource, die es Entwicklern ermöglicht, die Effizienz und Effektivität ihrer ML Projekte zu optimieren.

***Schlagwörter*—MLOps Tools, MLR, Machine Learning, Zeitreihenanalyse**

I. Einleitung

Die Generierung präziser Prognosen zukünftiger Ereignisse stellt eine fundamentale Herausforderung dar, welche eine Vielzahl an wissenschaftlichen und kommerziellen Disziplinen durchdringt. Vorhersagen sind von zentraler Bedeutung in den Bereichen Finanzen, Industrie, Wirtschaft und Ökonomie und finden zudem Anwendung in Disziplinen wie Medizin, Sozialwissenschaften und Politik [1]. Je nach strategischer Ausrichtung der Unternehmung können Kurzzeitprognosen von wenigen Tagen bis hin zu Langzeitprognosen von mehreren Jahren erstellt werden. Prognosen stützen sich dabei auf Daten oder Beobachtungen der betreffenden Variablen, die meist in Form von Zeitreihen vorliegen. Traditionelle Ansätze für die Zeitreihenvorhersage stützen sich hauptsächlich auf parametrische statistische Modelle, darunter lineare oder logistische Regression sowie autoregressive Modelle und Techniken mit exponential smoothing [2]. Algorithmen aus dem Bereich des maschinellen Lernens werden neuerdings immer beliebter, da sie es ermöglichen, Muster aus Zeitreihendaten zu extrahieren, um darauf basierend eine Vorhersage zu erstellen [3]. Neben Algorithmen aus dem Deep Learning wie künstliche neuronale Netze (KNN) oder rekurrente neuronale Netze (RNN) stellen ML Algorithmen wie die Support Vector Machine (SVM) oder der Naive Bayes Klassifikator eine beliebte Klasse an Analysemodellen dar [4]. Auf Entscheidungsbäumen basierte ML Algorithmen wie Random Forest, Gradient Boosting Machine oder der Extra Trees Algorithmus sind weitere häufig verwendete Analysemethoden [5].

Um Zeitreihendaten mit ML Methoden effektiv und skalierbar prognostizieren zu können, muss der gesamte Lebenszyklus des Prognosemodells überblickt und möglichst automatisiert werden. Eine End-to-End (E2E) Betreuung eines ML Modells von der Datenaufbereitung bis hin zum Bereitstellen des Modells erfordert die Auswahl an geeigneten Tools, welche die entsprechende Phase des Modelllebenszyklus betreuen. Aus der Verbindung von DevOps Praktiken, angewandt auf ML Projekte ergibt sich der Begriff MLOps, welcher vorwiegend die Bergriffe Continuous Integration (CI), Continuous Delivery (CD) und Continuous Training (CT) von ML Projekten vereint [6].

Für die effiziente Handhabung von Komplexität und Anforderungen in ML Projekten im End-to-End Kontext sind ML Pipelines unverzichtbar. Diese strukturieren die ML Prozesse von der Datenaufbereitung bis zur Modellbereitstellung in möglichst automatisierte Workflows [7].

Um zuverlässige und skalierbare ML Pipelines für den Geschäftsgebrauch und für die Industrialisierung von Projekten aufzubauen, gelten MLOps Tools und Praktiken heutzutage als Standard. Die Herausforderung besteht darin, MLOps Technologien mit einem hohen Reifegrad auszuwählen, da ansonsten nur traditionelle ML Methoden angewandt werden können [8]. Zudem ist es essenziell, dass MLOps Tools individuellen Anforderungen wie Kostenminimierung, Skalierbarkeit, Robustheit und Integrierbarkeit gerecht werden. Darüber hinaus sollte ein MLOps Workflow einen hohen Automatisierungsgrad bieten, um den Prozess von der Entwicklung bis zur Bereitstellung des Modells effektiv zu unterstützen [7].

Diese Arbeit wurde im Wintersemester 2023-24 im Masterseminar an der Hochschule Bonn-Rhein-Sieg im Fachbereich Informatik angefertigt.
Betreut wurde die Arbeit von Prof. Dr. Simone Bürsner.
https://doi.org/10.18418/opus-7847

Für den Aufbau einer effizienten Pipeline mit automatisierten Workflows sind DevOps Tools wie Github Actions oder Docker oft unverzichtbar [9]. Obwohl ML Projekte im Bereich der Zeitreihenanalyse von herkömmlichen DevOps Tools profitieren, erfordern sie für die Verwaltung des Lebenszyklus der ML Modelle eine spezifische Auswahl moderner MLOps Tools [7].

Daraus leitet sich für die Umsetzung solcher ML Projekte zur Zeitreihenanalyse folgende zentrale Forschungsfrage ab:

Welche MLOps Werkzeuge eignen sich unter gegebenen Anforderungen am besten für den Aufbau einer ML Workflow Pipeline zur Zeitreihenanalyse.

Diese Hauptfragestellung wird durch weitere unterstützende Forschungsfragen ergänzt:

- **RQ1:** Welche Anforderungen können an MLOps Tools gestellt werden?
- **RQ2:** Welche MLOps Tools existieren, um ML Pipelines aufzubauen?

Zur Beantwortung dieser Fragen leistet diese Arbeit die folgenden Beiträge. Der Background in Abschnitt II beschreibt den Aufbau bestehender ML Pipelines und ML Pipelines zur Zeitreihenvorhersage. Im Abschnitt III werden verwandte Arbeiten zu MLOps Praktiken und Tools analysiert. In Abschnitt IV erfolgt eine systematische Untersuchung aktueller Entwicklungen und Werkzeuge im Bereich MLOps durch eine multivokale Literaturübersicht. Diese Analyse integriert formale wissenschaftliche Beiträge, insbesondere durch Peer Review validierte Fachartikel, sowie informelle Literatur, die durch gezielte Google Suchen erschlossen wurde. Darauf aufbauend, beleuchtet Abschnitt V, spezifische MLOps Methoden und Herausforderungen und stellt zentrale Rollen und ihre Funktionen vor. Abschnitt VI fasst die aus der Literaturübersicht gewonnenen Erkenntnisse zusammen, einschließlich der Identifikation passender MLOps Werkzeuge und Methoden für die Zeitreihenanalyse. In Abschnitt VII wird eine spezifische Implementierung einer MLOps Pipeline für die Aktienkursprognose des S&P 500 vorgestellt, wobei verschiedene MLOps Werkzeuge zum Einsatz kommen. Abschließend bietet Abschnitt VIII eine Zusammenfassung der zentralen Erkenntnisse und diskutiert deren praktische Implikationen. Zudem wird ein Ausblick auf mögliche Erweiterungen der entwickelten Vorhersagepipeline gegeben.

II. Background

In den letzten zehn Jahren hat sich maschinelles Lernen in einer Vielzahl von Geschäfts- und Forschungsbereichen etabliert. Obwohl bedeutende Fortschritte in der Forschung und Entwicklung neuer Algorithmen und Trainingsmethoden für ML Modelle erzielt wurden, liegt die zentrale Herausforderung für Unternehmen und Experten derzeit vor allem in der effektiven Integration dieser Modelle in Produktionsumgebungen [10]. Im Folgenden werden Forschungsarbeiten vorgestellt, die sich auf die Entwicklung und den produktiven Einsatz von MLOps Pipelines konzentrieren, insbesondere solche, die Zeitreihenprognosen mittels ML implementieren.

A. ML Pipelines

In einer Studie von Pineda-Jaramillo et al. (2023) wurde ein automatisierter MLOps Workflow für den Güterzugbetrieb der Luxemburgischen nationalen Frachteisenbahngesellschaft (CFL Multimodal) entwickelt, der über 17 Monate Daten verarbeitete. Dieser Prozess nutzte maschinelles Lernen, vor allem ein LightGBM Modell, für Echtzeitvorhersagen von Güterzugverspätungen. Der Workflow umfasste die Datenerfassung, Modellentwicklung und -bereitstellung, und integrierte CI, CD und Continuous Training. Für Workflow und Orchestration wurde Apache Airflow genutzt. Weiterhin wurden Technologien wie Python, PostgreSQL und Docker eingesetzt. Um die Skalierbarkeit und Flexibilität der Pipeline zu erhöhen, wurden Cloud Plattformen wie Google Cloud, AWS, Microsoft Azure und IBM Cloud getestet. Eine Web App wurde mit Streamlit.io erstellt. Für das Erstellen einer ML Pipeline mit Open Scource Technologien wurde auf Kubeflow, MLflow und MLJAR verwiesen [11].

Filippou et al. (2023) entwickelten eine automatisierte maschinelle Lernen (AutoML) Pipeline für strukturbasiertes Lernen und Hyperparameteroptimierung. Die Pipeline wurde unter Verwendung von Technologien wie Git, Google Cloud Virtual Machine, Jenkins Server, Docker und dem Reverse Proxy Tool Ngrok entwickelt. Für die Evaluierung der Modelle wurde Neptune.ai verwendet. Insbesondere bei der Verarbeitung großer Datensätze mit komplexen ML Modellen konnte eine erhebliche Zeitersparnis im Vergleich zu manuellen Methoden festgestellt werden [8].

B. ML Pipelines zur Zeitreihenvorhersage

Pelkis et al. (2023) hat DeepTSF [12], ein MLOps Framework für Zeitreihenprognosen, in verschiedenen Bereichen wie Energie und Wetter konzipiert. Es hat sich bei der kurzfristigen Lastprognose im Energiesektor als wirksam erwiesen.

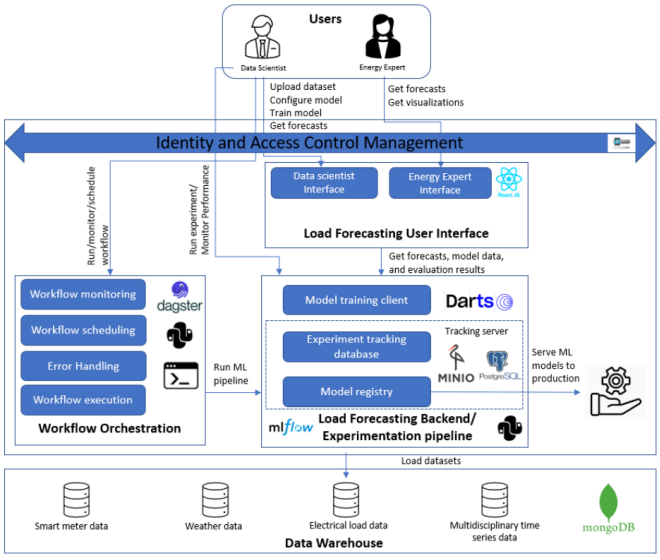

Abbildung 1. DeepTSF Architektur [2].

Das Framework ermöglicht einen codefreien Ansatz für ML und nutzt Python und JavaScript für die Backend- und Frontend Entwicklung sowie Docker für die Containerisierung. Es wurden MLOps Tools wie Dagster für den Workflow und MLflow für die Betreuung des ML Modells verwendet, wie in Abbildung 1 zu sehen [2].

Subramanya et al. (2022) entwickelten ein effizientes MLOps Framework zur Vorhersage der stündlichen Strommarktpreise für den nächsten Tag. In ihrer Arbeit nutzten sie PostgreSQL zur Speicherung von Daten, während Jenkins und Python für die CI/CD Prozesse eingesetzt wurden. Django diente zur Erstellung einer Web UI, Git zur Versionskontrolle, und Anaconda sowie Docker wurden für die Umgebungsverwaltung verwendet. Zusätzlich kamen MLflow und TFX (TensorFlow Extended) für das Modell Management und zur Automatisierung der Workflowprozesse zum Einsatz. Ihre Pipeline ermöglicht eine gewisse Generalisierung, da für viertel- oder halbstündliche Prognosen nur geringfügige Anpassungen notwendig sind [6].

Gürses-Tran und Monti, (2022) konzipierten das MLOps Framework ProLoaF [13], welches ein RNN Modell nutzt, um Stromlastprognosen vorherzusagen. Die Plattform performt besser bei der Nettolastvorhersage für Energiesysteme als AutoML Technologien wie Prophet (Facebook) und auto.arima. Das Framework verwendet dabei hauptsächlich Python Bibliotheken und Module wie Pandas, Numpy, Scikit-learn, Statsmodels, sowie Tensorboard und Pytorch. Für die Hyperparameteroptimierung wurde Optuna verwendet [14].

In der untersuchten Literatur, die sich mit der Implementierung von ML Pipelines für Zeitreihenprognosen befasst, wird eine Vielfalt von MLOps Tools integriert, wobei der Umfang ihres Einsatzes deutlich variiert. Es fällt auf, dass die Diskussion um alternative Werkzeuge für bestimmte Abschnitte der Wertschöpfungskette in den Studien nicht ausreichend berücksichtigt wird. Bislang existiert keine Publikation, welche eine breite Palette von MLOps Tools speziell für Zeitreihenanalysen darstellt. Um die Forschungslücke zu adressieren, erfolgt in Abschnitt III eine Analyse von Publikationen über MLOps Praktiken sowie eine Untersuchung relevanter Publikationen, welche Reviews zu MLOps Tools durchgeführt haben.

III. Verwandte Arbeiten

Die aktuelle Forschung und Entwicklung im Bereich des maschinellen Lernens und der damit verbundenen Praktiken und Werkzeuge für die Umsetzung in kontinuierlichen Entwicklungsprozessen schreitet rasch voran [9]. Viele Literaturübersichten im Bereich MLOps sind jedoch nicht in der Lage, die gesamte Bandbreite an Publikationen abzudecken, da parallele Forschungsaktivitäten oft unberücksichtigt bleiben. Ziel dieses Abschnitts ist es, den aktuellen Stand der Forschung im Bereich MLOps, insbesondere im Hinblick auf systematische Literaturübersichten (SLRs) und multivokale Literaturübersichten (MLRs), darzustellen und zu analysieren.

A. SLRs und MLRs zu MLOps Praktiken

In einer aktuellen Studie von Steidl et al. (2023) wurde eine umfassende Untersuchung zu Pipelines für die kontinuierliche Entwicklung von KI durchgeführt und dabei fünf Schlüsselbegriffe identifiziert sowie 25 Herausforderungen in den Bereichen Datenverarbeitung, Modelllernen, Softwareentwicklung und Systembetrieb aufgezeigt, wobei zukünftige Forschungen auf prototypische Implementierungen auf verschiedenen Plattformen abzielen. Dabei wurden 151 relevante formale und informale Quellen mittels MLR analysiert und Interviews geführt [15]. Aus diesen wurden folgende SLRs im Kontext von MLOps untersucht: (Karamitsos et al. (2020) [16], Figalist et al. (2020) [17], Nascimento et al. (2020) [18], Fredriksson et al. (2020) [19], Lo et al.(2021) [20], Lorenzoni et al. (2021)[21], Mboweni et al. (2022) [22], Testi et al. (2022) [23], Kolltveit und Li (2022) [10], Kreuzberger et al. (2022) [24]). Zudem wurden MLRs (John et al. (2021a) [25] John et al. (2021b) [26], Lwakatare et al. (2020) [27]) sowie eine Systematische Kartierungsstudie von Xie et al. (2021) [28] analysiert. Es wurde deutlich, dass sich viele SLRs und MLRs im Bereich MLOps auf spezifische Kontexte, wie beispielsweise Edge Cloud Architekturen [25] fokussieren.

Lima et al. (2022) untersuchten in einer SLR, MLOps Praktiken, Werkzeuge und Herausforderungen. Aus 1905 Artikeln wurden 30 für die Analyse ausgewählt. Die Studie ergab, dass es noch kein einheitliches Lebenszyklusmodell für ML in der Literatur gibt. Es wurden verschiedene Aktivitäten für Entwicklung, Training, Testen, Implementierung und Betrieb von ML Modellen identifiziert und festgestellt, dass diese in Phasen definiert und organisiert sind. Da es jedoch keinen wissenschaftlichen Konsens über die Ansätze zur Zuordnung von Aktivitäten zu Phasen gibt, besteht eine erhebliche Lücke in der Detaillierung der Aktivitäten zur Operationalisierung von ML Modellen. Weiterhin wurden verschiedene Rollen, Werkzeuge und Herausforderungen identifiziert, aber es fehlt an einem standardisierten Ansatz zur Bewertung und Implementierung von MLOps in Organisationen [29].

Rzig et al. (2022) führten eine großangelegte empirische Studie durch, wobei 4031 ML Projekte, darunter 1116 ML Tool Projekte und 2915 ML Projekte, mit 4076 nicht ML Projekten auf die Annahme von DevOps Praktiken und Tools verglichen wurden. Es konnte gezeigt werden, dass der Einbezug von DevOps Werkzeugen und Praktiken die Entwicklungsproduktivität und die Codequalität von ML Projekten erhöht. Vor allem in ML Projekten werden DevOps Tools und Praktiken wie Build-, CI-, Deployment-, Automation-, Monitoring-, Test- und Code Analyse Tools langsamer und in einem geringen Umfang angenommen, obwohl gerade diese Projekte mit der Annahme am meisten profitieren können [9].

B. SLRs und MLRs zu MLOps Tools

Moreschi et al. (2023) hatten eine MLR durchgeführt um 84 MLOps Tools aus 254 Primärstudien den verschiedenen DevOps Phasen zuzuordnen. Ihre Forschung baut auf den Erkenntnissen von Recupito et al. (2022) [31] auf. Die Studie

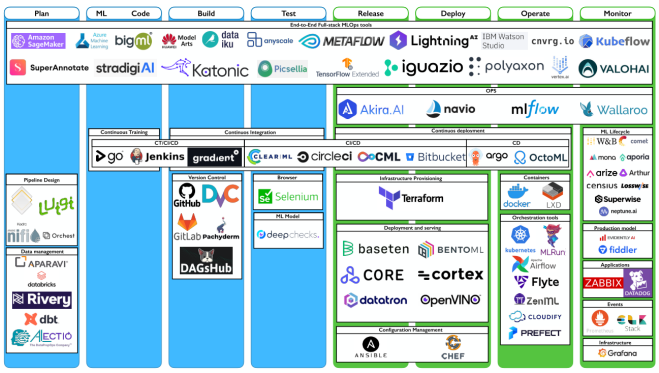

Abbildung 2. MLOps Tools nach Subkategorie [30].

ordnet DevOps Prozesse in Kategorie und Unterkategorie ein und teilt die untersuchten Tools den Kategorien grafisch zu, wie in Abbildung 2 dargestellt. Es wurde zudem festgestellt, dass die Mehrheit an E2E MLOps Plattformen eine Integration anderer Werkzeuge nicht ermöglichen, während bei anderen Tools meist keine Unverträglichkeit festgestellt wurde [30].

Heydari und Rezvani (2023) führten eine Umfrage unter 110 iranischen Softwareexperten zur Nutzung von MLOps Tools durch. Dabei stellte sich heraus, dass MLflow, Apache Airflow, Kubeflow, Apache Kafka und Tensorflow Extended (TFX) am häufigsten in Unternehmensumgebungen eingesetzt werden. Diese Präferenzen sind in Abbildung 3 dargestellt [32].

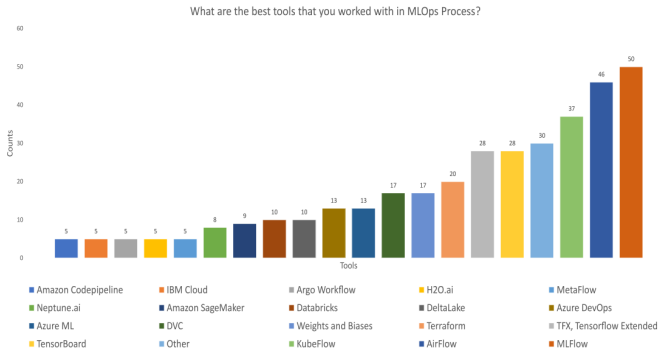

Abbildung 3. Am häufigsten genutzte MLOps Werkzeuge in Unternehmen [32].

Wazir et al. (2023) erstellten ein Review über MLOps, in dem 22 Publikationen analysiert wurden, um tiefere Einblicke in den ML Lebenszyklus zu gewinnen. Dabei wurden acht MLOps Tools verglichen. Die Studie ergab, dass für Unternehmen, die ML in ihrer Entwicklung einsetzen, Praktiken von MLOps für eine kontinuierliche Bereitstellung zunehmend unerlässlich werden. Dabei wurde festgestellt, dass die Anwendung von DevOps Technologien alleine, oftmals nicht angemessen ist, um die spezifischen Anforderungen von ML zu erfüllen. Die Studie hebt hervor, dass Tools wie MLflow oder Amazon SageMaker besonders geeignet sind, um ML in Produktivsysteme zu integrieren. Es wird auch angemerkt, dass insbesondere bei Cloud Diensten höhere Kosten anfallen können [33].

Symeonidis et al. (2022) beleuchteten den aktuellen Stand der Technik im Bereich des AutoML und stellten Verbindungen zum Bereich MLOps her. Zusätzlich bietet die Studie eine umfangreiche Zusammenstellung kategorisierter MLOps Tools. Dabei gibt es kein einheitliches Rezept zur Auswahl von Tools für ein bestimmtes Problem. Die Auswahl einer geringen Anzahl von Tools aufgrund von möglichen Integrierungsproblemen ist jedoch ratsam [34].

Testi et al. (2022) haben SLRs zu MLOps untersucht und Methoden und Operationen vorgeschlagen, um eine ML Pipeline zu erstellen. Eine ML Pipeline basiert auf zehn Schritten: Verständnis des Geschäftsproblems, Datenerhebung, ML Methodik, ML Training und -Testen, Continuous Integration, Continuous Deployment, Continuous Training, Continuous Monitoring, sowie Erklärbarkeit und Nachhaltigkeit. Zur Operationalisierung der Schritte wurden MLOps Tools vorgestellt und evaluiert [23].

Hewage und Meedeniya (2022) untersuchten den Einsatz und die Funktionalitäten kommerzieller MLOps Tools im Softwareentwicklungsprozess. Werkzeuge im Bereich MLOps müssen benutzerfreundlich und effizient sein, um in Softwareentwicklungsprozessen, welche ML Modelle, Continuous Integration und DevOps einbeziehen, die Konsistenz der Artefakte zu sichern [35].

Kreuzberger et al. (2022) führten eine SLR und Experteninterviews durch, um die Prinzipien, Komponenten, Rollen und die Architektur von MLOps zu untersuchen. Sie beschrieben MLOps Technologien und -Tools und ordneten diese der Nutzung der befragten Fachleute zu [24].

Ruf et al. (2021) demonstrierten die Vorteile von MLOps Workflows anhand eines Deep Learning basierten Objekterkennungsprojekts. Sie verglichen MLOps Tools für Prozesse wie Preprocessing, Training und Modell Management und erleichtern so die Auswahl spezifischer Lösungen und den Einstieg in MLOps Projekte. Die Studie stellt fest, dass kein einzelnes Tool einen vollständig automatisierten MLOps Workflow bereitstellen kann [7].

IV. Multivokale Literaturübersicht (MLR)

Die MLR erweitert die traditionelle systematische Literaturrecherche (SLR), indem sie neben formaler Literatur wie Fachzeitschriften und Konferenzbeiträgen auch graue Literatur wie Blogbeiträge, Videos und Whitepapers mit einbezieht. Diese Methode ist besonders wertvoll in der Softwareentwicklung, da sie einen umfassenden Einblick in den aktuellen Stand von Forschung und Praxis bietet [36]. Um eine ganzheitliche Analyse von MLOps Tools zu ermöglichen und die Forschungsfragen (RQs) adäquat zu beantworten, wurde eine MLR durchgeführt, die eine breite Marktabdeckung gewährleistet.

A. Systematische Literaturübersicht (SLR)

Das Ziel der SLR ist es, MLOps Werkzeuge und Techniken zu identifizieren und zu analysieren, um ihren Einsatz und Nutzen in der Entwicklung von ML Anwendungen für die

Zeitreihenanalyse zu erproben. Die SLR orientiert sich an den drei von Kitchenham et al. (2004) [37] definierten Schritten:

1. Entwicklung des Suchstrings, 2. Anwendung des Strings auf die Suchportale Web of Science und IEEE Xplore, 3. Sortierung und Extraktion der Studien basierend auf Ein- und Ausschlusskriterien. Für die Entwicklung der Suchanfrage werden die GQM Begriffe (Goal, Problem, Object, View) verwendet.

Goal: Systematische Kategorisierung
Problem: MLOps Tools, Praktiken und Herausforderungen
Objekt: ML Projekte zur Zeitreihenanalyse
View: MLOps Engineers und Data Scientist

Aus den GQM Begriffen und den RQs leiten sich die folgenden Suchanfragen ab, die für einen Suchzeitraum der letzten fünf Jahre (2018 - 2023) folgende Ergebnisse in Tabelle I liefern.

Tabelle I
ANZAHL DER VERÖFFENTLICHUNGEN NACH STICHWORTSUCHE (STAND: 27.10.2023).

Suchanfragen (SLR)	Total
(1) (“DevOps” ∨ “CICD”) ∧ “Tools” ∧ (“Machine Learning” ∨ “MLOps” ∨ “CD4ML”)	177
(2) “Time Series” ∧ “Forecasting” ∧ (“MLOps” ∨ “DevOps”)	6
(3) “Stock” ∧ “Prediction” ∧ “Direction” ∧ “Machine Learning”	159

Tabelle II
KRITERIEN FÜR DIE AUSWAHL DER STUDIEN.

Suchanfrage	Einschlusskriterien
(1)	• Diskussion über Integration von DevOps Praktiken und CI/CD in MLOps • Untersuchung von Tools im Kontext von ML, MLOps oder CD4ML • Studien, die eine Verbindung zwischen DevOps Praktiken und ML herstellen
(2)	• Studien über Zeitreihenprognosen in Verbindung mit MLOps oder DevOps • Arbeiten zu spezifischen MLOps Tools für Zeitreihenprognosen
(3)	• Studien zur Aktienmarktprognose unter Verwendung von ML Methoden • Forschungsarbeiten zur Vorhersage von Aktienkursrichtungen • Untersuchungen, die technische Indikatoren für die Merkmalsextraktion verwenden
Ausschlusskriterien	
	• Nichterfüllung eines der Einschlusskriterien • Nicht in Englisch oder Deutsch • Duplikat

Tabelle II enthält die Einschluss- und Ausschlusskriterien, die für die Auswahl relevanter Studien im Kontext der SLR angewendet wurden. Diese Kriterien stellen sicher, dass die ausgewählten Studien spezifisch auf die definierten Forschungsfragen (RQs) ausgerichtet sind und eine hohe Relevanz und Qualität für die Untersuchung von MLOps Werkzeugen, Praktiken und Herausforderungen in ML Zeitreihenanalyseprojekten aufweisen.

Tabelle III
ZUSAMMENFASSUNG UND KATEGORISIERUNG DER SLR.

Kategorie	Referenz
MLOps Analyse (RQ1)	(6)
Trends und Herausforderungen	(Tamburri, 2020) [38]
SLR, GLR	(John et al., 2021) [25]
Trends, Bemühungen und Vorteile	(Rzig et al., 2022) [9]
SLR	(Lima et al., 2022) [29]
Fähigkeiten, Praktiken	(Mucha et al., 2023) [39]
MLR, Interviews	(Steidl et al., 2023) [15]
MLOps Anwendungen (RQ1)	(7)
Zeitreihenanalyse	(Subramanya et al., 2022) [6], (Gürses-Tran; Monti, 2022) [14], (Pelekis et al., 2023) [2]
Cloud	(Lopez Garcia et al., 2020) [40]
CI/CD	(Karamitsos et al., 2020) [16], (Filippou et al., 2023) [8], (Tabassam, 2023) [41]
MLOps Tools (RQ2)	(10)
Anforderungen und Vergleich	(Ruf et al., 2021) [7]
SLR, Review, Interviews	(Kreuzberger et al., 2022) [24]
SLR, Review	(Kolltveit; Li, 2022) [10]
Survey Toolsupport	(Hewage; Meedeniya, 2022) [35]
Taxonomie und Methodik	(Testi et al., 2022) [23]
Übersicht und Verbindung zu AutoML	(Symeonidis et al., 2022) [34]
MLR über Pipelines und Tools	(Recupito et al., 2022) [31]
Review	(Wazir et al., 2023) [33]
Review und Umfrage	(Heydari; Rezvani, 2023) [32]
MLR	(Moreschi et al., 2023) [30]
Aktienkursvorhersage (RQ2)	(2)
Baumasiertes ML	(Ampomah et al., 2020) [5]
Richtungsprognose	(Nabipour et al., 2020) [42], (Jiang et al., 2021)[43]

Die erste Suche ergab 59 Einträge in Web of Science und 118 in IEEE Xplore. Die zweite Suche liefert 3 Einträge in Web of Science und 3 in IEEE Xplore. Die dritte Suchanfrage ergab 105 Einträge in Web of Science und 54 in IEEE Xplore. Aus den ersten 60 Publikationen jeder Datenbank wurden Publikationen anhand der Relevanz des Titels, des Abstracts und des Inhalts für die Fragestellung ausgewählt. Für die Auswahl wurden zudem die Einschluss- und Ausschlusskriterien aus Tabelle II verwendet. Dabei wurde eine Publikation [38], die von mehr als zwei untersuchten Papers zitiert wurde, der Auswahl für die MLOps Analyse hinzugefügt. Somit erhöhte sich die Anzahl der gefundenen Publikationen zu MLOps Trends und Herausforderungen von ursprünglich fünf auf sechs untersuchte Publikationen. Die Suchstrings (2) und (3) beziehen sich speziell auf die Anwendung von MLOps in ML Zeitreihenanalyse Projekten. Alle Suchstrings behandeln auch die Hauptforschungsfrage. Tabelle III zeigt die im Detail untersuchten Publikationen.

B. Qualitätsbewertung SLR

Um **RQ1: Welche Anforderungen können an MLOps Werkzeuge gestellt werden?** zu beantworten, wurden aktuelle SLR- und MLR Studien zu MLOps Praktiken und Herausforderungen analysiert und im Kapitel III-A zusammengefasst. Zusätzlich wurden Artikel untersucht, die ML Pipelines praktisch anwenden, um weitere Anforderungen an MLOps Werkzeuge zu identifizieren.

Zur Beantwortung von **RQ2: Welche MLOps Tools existieren, um ML Pipelines aufzubauen?** wurden 10 Publikationen analysiert, die einen Überblick über MLOps Tools geben. Dabei wurden alle erwähnten Tools und Plattformen in einer Tabelle notiert, um einen vollständigen Überblick über die aktuelle MLOps Landschaft in der Fachliteratur zu geben. Arbeiten wie [30] schließen Werkzeuge zur Vorverarbeitung und Etikettierung von Daten explizit aus, da diese die Automatisierung unterbrechen können. Diese Arbeit verfolgt den Ansatz, einen vollständigen Überblick über die aktuelle MLOps Landschaft zu geben und schließt alle gefundenen Tools ein.

Für die Unterstützung einer ML Pipeline durch MLOps Werkzeuge zur Aktienkursprognose wurden Publikationen untersucht, die Aktienkurse mit ML Modellen prognostiziert haben. Ziel war es, Modelle auszuwählen, die sich effizient in eine automatisierte Pipeline integrieren lassen und den Ressourcenverbrauch von CPU und GPU gering halten.

C. Graue Literaturrecherche (GLR)

Die graue Literatur wurde systematisch durchsucht, um die Ergebnisse der systematischen Literaturrecherche zu ergänzen. Die GLR umfasste eine sorgfältige Suche in Online-Datenbanken und -Plattformen, um relevante Beiträge zu identifizieren, die nicht in traditionellen wissenschaftlichen Publikationen erscheinen. Dabei wurden insbesondere Blogs, technische Berichte, Whitepapers und ähnliche Publikationen berücksichtigt, die nicht dem Peer Review Prozess unterliegen. Darüber hinaus wurde das Verfahren des “Snowballing” [44] angewandt, um weitere relevante Studien zu identifizieren, indem die Referenzlisten der ausgewählten Arbeiten analysiert wurden. Dies trug dazu bei, die Breite und Tiefe der Literaturabdeckung zu verbessern und sicherzustellen, dass wichtige Beiträge aus verschiedenen Quellen berücksichtigt wurden. Die verwendeten Suchstrings und Quellen sind in Tabelle IV zusammengefasst.

Tabelle IV
SUCHSTRING UND QUELLEN DER GLR.

Suchanfragen (GLR)	
Strings: (1) (MLOps Tools); (2) (Time Series MLOps Tools)	
Quellen	**Link**
Google	https://www.google.com
Medium	https://medium.com
GitHub	https://github.com

Zunächst wurde nach allen MLOps Werkzeugen gesucht (1), um zu prüfen, ob die Werkzeugliste aus der SLR weiter ergänzt werden kann. Um die MLOps Technologien explizit für Zeitreihenanalysen herauszufiltern, wurde eine weitere Suche durchgeführt (2).

D. Qualitätsbewertung GLR

Eine ausführliche Internetrecherche (Stand 12.12.2023)

Tabelle V
ERGEBNISSE GLR.

Quellen	**Link**
130 MLOps Tools [45]	github.com/awesome-MLOps
543 MLOps Tools [46]	github.com/awesome-production-ml
96 DataOps Tools [47]	github.com/awesome-dataops
178 DevOps Tools [48]	github.com/awesome-devops
374 Data Science Tools [49]	github.com/awesome-python-ds
417 Python Tools [50]	github.com/awesome-python

Die Suche nach dem Begriff ’Suchstring 1’ führte zu sechs Hauptquellen, die in Tabelle V aufgeführt sind. Diese Quellen bieten einen umfassenden Überblick über MLOps Tools. Zwei dieser Quellen stellen speziell MLOps Tools vor [45],[46], während eine weitere Quelle gängige DevOps Tools auflistet [48]. Darüber hinaus wurden in drei weiteren Quellen Tools und Bibliotheken beschrieben, die sich hauptsächlich auf Data Science mit Python konzentrieren [47],[50],[49]. Aus diesen Listen wurden insgesamt 1738 Werkzeuge extrahiert. Diese Werkzeuge wurden in zwei Hauptkategorien eingeteilt: Open Source und Plattform. Während Open Source Tools in der Regel kostenlos genutzt werden können, sind Plattformen oft kommerzielle Produkte und meist mit Kosten verbunden.

Die Einordnung von MLOps Tools in spezifische Kategorien variiert sowohl in der akademischen (weißen) als auch in der nicht akademischen (grauen) Literatur erheblich. Ein Beispiel hierfür ist das Tool Metaflow, das unterschiedlichen Kategorien wie Data Pipeline, Workflow, Orchestrierung und End-to-End ML zugeordnet wird. In dieser Arbeit wird die Kategorisierung gemäß den Quellen [45] - [49] übernommen, da diese Quellen eine umfassende Übersicht über nahezu alle verfügbaren Open Source Tools im Bereich MLOps bieten. Die weiße Literatur konnte keine zusätzlichen Open Source Tools liefern.

Für alle extrahierten und auf GitHub gelisteten Tools, wurde die Anzahl der aktuellen Sterne zum Stichtag 12.12.2023 erfasst.

Durch eine Internetrecherche mit dem Suchbegriff ’String 2’ konnten grundlegende Kategorien von MLOps Tools identifiziert werden, die für den Aufbau einer automatisierten Pipeline zur Zeitreihenanalyse unerlässlich sind. In Tabelle XI sind die MLOps Tools diesen grundlegenden Kategorien in absteigender Reihenfolge nach der Anzahl der GitHub Sterne zugeordnet. Tools mit weniger als 2000 Sternen wurden nicht berücksichtigt. Nützliche Tools für das maschinelle Lernen mit Python, insbesondere für die Zeitreihenanalyse, sind in

der Tabelle X aufgelistet. Da eine Cloud Umgebung für die Automatisierung unerlässlich sein kann, werden in Tabelle IX Cloud Plattformen sowie E2E ML Plattformen aufgelistet.

V. MLOps Praktiken

MLOps ist eine Praxis, die sich auf die effiziente Entwicklung und den Einsatz von ML Modelle in der Produktion konzentriert. Diese Methode kombiniert DevOps Prinzipien, also Continuous Integration (CI), Continuous Deployment (CD) und Continous Delivery (CD) von Software mit den Besonderheiten des maschinellen Lernens. Im Gegensatz zu herkömmlicher Software, die in der Regel nur aus Code besteht, beinhaltet ein ML Modell sowohl Code als auch Daten. Da sich Daten ständig ändern, müssen ML Modelle regelmäßig mit neuen Daten neu trainiert werden. Daher ergänzt MLOps die DevOps Ansätze um das Prinzip des Continuous Training (CT), um eine ständige Anpassung und Verbesserung der Modelle zu gewährleisten [34]. ML Modelle müssen zudem einem Continuous Monitoring (CM) unterliegen, um unterschiedliche Modelle auf Performance vergleichen zu können oder um Drift Detection zu ermöglichen [30]. Darüber hinaus ist es wichtig, Tests über den gesamten Workflow, beginnend mit der Datenvorverarbeitung und endend mit der Bereitstellung des Modells, zu implementieren, um eine durchgängige Qualitätssicherung zu gewährleisten [6].

A. MLOps Prozesse

Abbildung 4 gibt eine Übersicht über gängige MLOps Prozesse. Für jeden der Teilprozesse gibt es eine Vielzahl von Werkzeugen, wobei einige MLOps Werkzeuge mehrere Prozesse oder den gesamten Lebenszyklus abdecken.

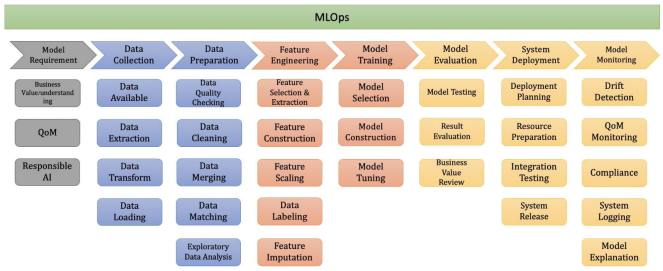

Abbildung 4. Machine Lerning Prozesse [51].

Model Requirement: Festlegung der Ziele und Anforderungen für ML Projekte einschließlich Business Value (BV), Quality of Model (QoM) und Responsible AI.
Data Collection: Sammlung der erforderlichen Daten, Überprüfung ihrer Verfügbarkeit, Formatkorrektur und gegebenenfalls das Labeling für supervised learning.
Data Preparation: Aufbereitung der Daten durch Qualitätsprüfung, Datenbereinigung, Datenzusammenführung, Data Matching und Exploratory Data Analysis (EDA).
Feature Engineering: Transformation der Rohdaten in nutzbare Features einschließlich Feature Selection, Extraction, Construction und Scaling.
Model Training: Auswahl, Konstruktion und Feinabstimmung von Modellen aus den Bereichen supervised, unsupervised und Reinforcement Learning (RF).
Model Evaluation: Bewertung der Modellleistung anhand unbekannter Daten einschließlich Model Testing, Result Evaluation und Business Review.
Model Deployment: Implementierung des Modells für den praktischen Einsatz inklusive Planung, Ressourcenvorbereitung, Integrationstest und Systemfreigabe.
Model Monitoring: Überwachung des Modells im Einsatz, um Leistungsabfälle frühzeitig zu erkennen und Gegenmaßnahmen einzuleiten, umfasst Drift Detection, QoM Monitoring, Compliance, System Logging und Model Explanation [51].

B. MLOps Pipelines

ML Pipelines lassen sich in unterschiedlichen Komplexitätsgraden, Integrationsniveaus und Automatisierungsstufen unter Berücksichtigung der beschriebenen MLOps Prozesse aufbauen. Im Folgenden werden drei Hauptstufen des Aufbaus von MLOps Pipelines dargestellt.

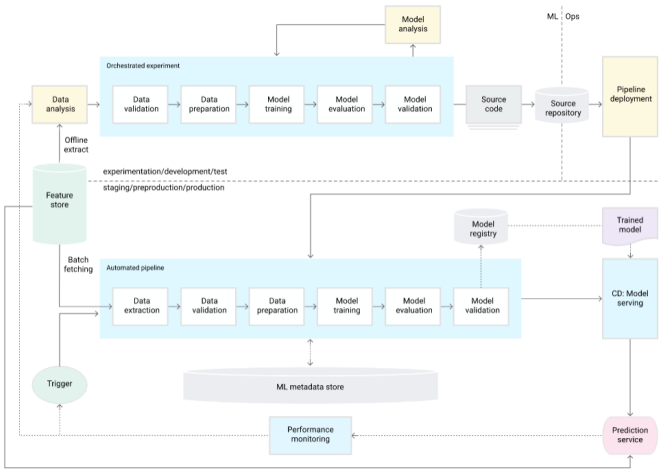

Abbildung 5. Google MLOps Pipeline Level 1 [34] [52].

- **MLOps Level 0: Manueller Prozess:** Hierbei erfolgt die ML Entwicklung hauptsächlich manuell ohne signifikante Automatisierung oder Integration in den Softwareentwicklungszyklus.
- **MLOps Level 1: Automatisierung der ML Pipeline:** Einführung automatisierter Pipelines, die den gesamten Prozess von der Datenaufbereitung bis zur Modellevaluierung ohne manuelle Eingriffe durchführen. ML Modelle werden automatisch mit neuen Daten trainiert, evaluiert und anschließend über eine API zur Verfügung gestellt. Zusätzlich wird ein zentraler Feature Store für die zentrale Verwaltung der Features eingerichtet. Ein ML Metadaten Store verwaltet Vergleichsdaten der Pipeline und der ML Modelle. Die Pipeline kann über verschiedene Trigger automatisch gestartet werden [52].
- **MLOps Level 2: CI/CD Pipelineautomatisierung:** Diese Stufe integriert ein umfassendes CI/CD System, das

automatisch neue ML Pipelines testet und integriert. Dadurch können kontinuierlich neue Implementierungen der ML Pipeline in die Zielumgebung überführt werden. Dies ermöglicht es, neue Vorhersagedienste, basierend auf neu trainierten Modellen, effizient und zuverlässig zu implementieren und bereitzustellen [34].

Abbildung 5 zeigt die Architektur einer MLOps Level 1 Pipeline. Die Pipeline kann auf verschiedene Trigger reagieren, darunter manuelle Ad-hoc Auslösungen, zeitbasierte Trigger, Ankunft neuer Daten, Leistungsverschlechterung des Modells und Änderungen in der Datenverteilung [52].

C. MLOps Rollen

In der dynamischen Welt des MLOps gibt es verschiedene Schlüsselrollen, die für die Entwicklung und das Management von MLOps Systemen wesentlich sind [34]. Die Akteure, die entsprechende MLOps Systeme entwickeln, sind in Abbildung 6 dargestellt.

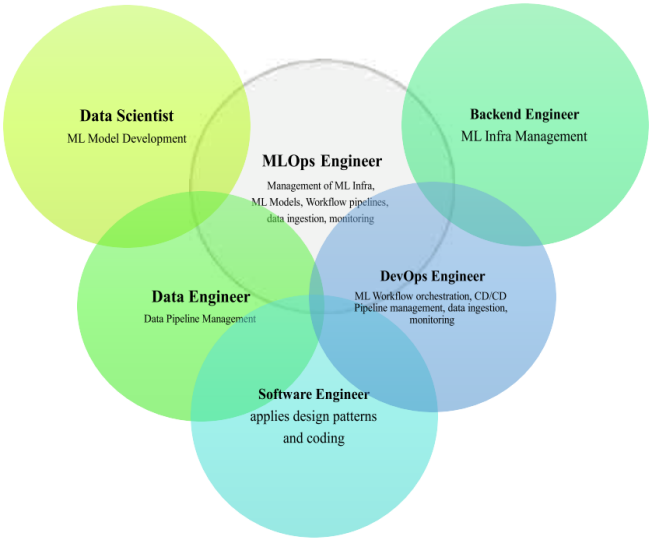

Abbildung 6. MLOps Rollen und Verantwortlichkeiten [32][24].

Zu den zentralen Rollen in einem MLOps Team gehören:
Data Scientist: übersetzt das Geschäftsproblem in ein ML Problem und kümmert sich um das Modell Engineering einschließlich der Auswahl des am besten performenden Algorithmus und der Hyperparameter [24].
MLOps Engineer: vereint Aspekte mehrerer Rollen und verfügt daher über domänenübergreifendes Wissen. Diese Rolle integriert Fähigkeiten von Data Scientists, Data Engineers, Software Engineers, DevOps Engineers und Backend Engineers. Diese domänenübergreifende Rolle baut die ML Infrastruktur auf, betreibt sie, verwaltet die automatisierten ML Workflowpipelines und die Bereitstellung von Modellen in der Produktion und überwacht sowohl das Modell als auch die ML Infrastruktur [24].
Business Stakeholder: definiert das geschäftliche Ziel welches durch den Einsatz von ML erreicht werden soll und übernimmt die unternehmensinterne Kommunikation, einschließlich der Darstellung des durch ein ML Produkt generierten Return on Investment (ROI) [24].

D. Herausforderungen

Bei der Operationalisierung von ML Modellen sind Entwickler mit verschiedenen Herausforderungen konfrontiert. Einerseits gibt es Probleme mit ML Frameworks, die oft in ihrer Konsistenz und Eignung für Produktions- oder Forschungsumgebungen variieren. Dies führt zu Schwierigkeiten bei der Bereitstellung und der Wahl zwischen suboptimalen Frameworks oder dem Eingehen technischer Schulden [10].

Tabelle VI
MLOps Herausforderungen in Unternehmen nach [32].

Kategorie	Herausforderungen
Allgemein	Ressourcenmangel, Definition der Geschäftsanforderungen, Mangel an Experten, Widerstand gegen Veränderungen
Daten	Datenqualität, Datenvorbereitung, Daten Drift, Daten Governance und Datenschutz, Datenverzerrung und Fairness, Datensilos und Fragmentierung
ML Modelle	Modellbereitstellung in der Produktion, Herausforderungen mit GPU Servern, Komplexität bei der Modellbereitstellung, Bewertung der Modellleistung, langsames Training und Testen, Modellversionierung, Erklärbarkeit und Interpretierbarkeit von Modellen, Ad-hoc-Modellentwicklung
Tools	Unzureichende Standardisierung von Frameworks und Werkzeugen

In einer Umfrage von Heydari und Rezvani (2023) [32] wurden verschiedene Herausforderungen im Bereich MLOps identifiziert und nach ihrer Bedeutung für die Unternehmen kategorisiert. Diese Herausforderungen sind in der TabelleVI in absteigender Reihenfolge der Wichtigkeit, die ihnen von den Befragten beigemessen wurde, aufgelistet.

Hewage und Meedeniya (2022) [35] weisen darauf hin, dass ML Modelle, die mit aktuellen Echtzeitdaten trainiert wurden, häufig neu trainiert werden müssen, um die Modellleistung konstant zu halten. Dazu ist es wichtig, dass ein kontinuierliches, automatisiertes Training ohne menschlichen Eingriff gewährleistet werden kann. Die Automatisierung dieses Entscheidungsprozesses mit MLOps Pipelines stellt eine Herausforderung dar [35].

VI. Ergebnisse der MLR

A. Anforderungen

In den letzten Jahren ist eine Vielzahl von Werkzeugen auf den Markt gekommen, die beim Aufbau und der Automatisierung einer ML Pipeline helfen [8]. Die Anforderungen an MLOps Werkzeuge ergeben sich im Wesentlichen aus der Fähigkeit, einzelne oder mehrere MLOps Prozesse (Abbildung 4) bearbeiten zu können. Dieses Kapitel gibt einen Überblick über Anforderungen an Tools, die insbesondere für die prototypische Entwicklung eines ML Projekts essenziell sind und behandelt Forschungsfrage **RQ1: Welche Anforderungen können an MLOps Tools gestellt werden?**

*1) **Allgemeine Anforderungen**:* Ein zentrales Kriterium, um die Kosten in der Entwicklungsphase einer ML Pipeline zu minimieren, ist die Verwendung von **Open Source** Tools. Diese Hauptanforderung stellt sicher, dass die Kosten während der Entwicklung möglichst niedrig gehalten werden. Die Annahme bei der Verwendung von Open Source Tools ist, dass grundsätzlich keine Kosten entstehen. Ausnahmen können zusätzliche Dienstleistungen des Anbieters darstellen, wie beispielsweise Cloud Services, welche üblicherweise kostenpflichtig sind.

Ein weiteres entscheidendes Kriterium ist die **Skalierbarkeit**, die insbesondere in der Phase der Prototypenentwicklung von Bedeutung ist. In einer ML Pipeline bezieht sich Skalierbarkeit auf die Auswahl von Werkzeugen, die flexibel mit Aspekten wie Rechenleistung, Speicherkapazität und Service Latenz umgehen können [7]. Dies ermöglicht die einfache Integration von ML Modellen mit höherem Ressourcenverbrauch in die Pipeline ohne, dass spezifische Werkzeuge ersetzt werden müssen.

Um Vendor-Lock-in Risiken zu minimieren und sich nicht von einem Anbieter abhängig zu machen, ist die **Integrierbarkeit** der Werkzeuge von zentraler Bedeutung. Bislang ist kein MLOps Tool bekannt, mit dem allein eine vollständige, automatisierte ML Pipeline aufgebaut werden kann [7]. Aus diesem Grund ist es unerlässlich, dass sich aufeinander aufbauende Werkzeuge nahtlos ineinander integrieren lassen.

Ein Werkzeug sollte einen möglichst hohen **Reifegrad** aufweisen, da Fehler in der Entwicklung erhebliche Auswirkungen auf Geschäftsziele haben können. Indikatoren für einen hohen Reifegrad sind eine umfassende Testung und eine breite Nutzerbasis [7]. Für Open Source Tools wird im Folgenden die Anzahl der Sterne auf GitHub als Maß für den Reifegrad herangezogen.

Auch die **Benutzerfreundlichkeit** eines Tools kann durch intuitive Handhabung oder Benutzeroberflächen zur Qualität einer MLOps Pipeline beitragen [7].

*2) **Datenverarbeitung**:* In einer MLOps Pipeline gibt es zahlreiche Datenverarbeitungsprozesse, die in Abbildung 4 in den Abschnitten "Data Collection" und "Data Preparation" dargestellt sind. In diesem Zusammenhang werden die grundlegenden Anforderungen an eine Datenverarbeitungspipeline erläutert, um die datenbezogenen Herausforderungen (siehe Tabelle VI) mit geeigneten Werkzeugen effizient zu adressieren.

Die **Datenquellenhandhabung** umfasst die Integration verschiedener Datenquellen in die Pipeline mittels Konnektoren oder Schnittstellen. Dies ist entscheidend für die effiziente Integration und Verarbeitung von Daten aus Quellen wie Datenbanken, Dateisystemen und Netzwerk Sockets [7].

Um effiziente **ETL Prozesse** (Extraktion, Transformation, Laden) zu gewährleisten, ist es unabdingbar, dass die eingesetzten Werkzeuge eine breite Palette an Datentypen verarbeiten können. Diese umfassen sowohl strukturierte als auch unstrukturierte Datenformate. Zu den strukturierten Daten zählen in der Regel Tabellen oder Datenbanken. Unstrukturierte Daten hingegen können aus Textdokumenten, numerischen Werten, Audiodateien, Bildern und Videos bestehen [53].

Beim **Datenmanagement** spielt die zentrale Verwaltung der Datensätze eine Schlüsselrolle, um Probleme oder Fehler schnell identifizieren zu können.

Um das Training von ML Modellen reproduzierbar zu archivieren, bietet **Datenversionierung** Git ähnliche Ansätze sowie zeitbasierte Methoden.

*3) **ML Modell**:* Zur Effizienzsteigerung von ML Modellen wird in der Regel während der Trainingsphase eine **Hyperparameteroptimierung** durchgeführt, um die Modellperformance zu verbessern. AutoML Ansätze bieten hier die Möglichkeit, die am besten performenden Modelle mit geeigneten Hyperparametern automatisiert zu ermitteln [16].

Nach der Erstellung eines optimierten Modells kann eine **Registrierung** erfolgen. Die Modellregistrierung dient als zentraler Speicherort für alle Modelle einschließlich ihrer jeweiligen Versionen und der zugehörigen Metadaten. Diese systematische Organisation ermöglicht eine effiziente und gezielte Suche sowie die Bereitstellung spezifischer Modellversionen. Dies verkürzt die Zeit bis zur Produktion [7].

Die **Bereitstellung** des Modells kann beispielsweise über eine REST API erfolgen. Als grundlegende Infrastrukturschicht wird eine skalierbare und verteilte Modellbereitstellungsinfrastruktur empfohlen. Der Zweck der Bereitstellung kann eine Online Inferenz für Echtzeitvorhersagen oder eine Batch Inferenz für Vorhersagen mit großen Datenmengen sein [24].

*4) **Betrieb und Überwachung**:* Die **Systemüberwachung** gewährleistet ein kontinuierliches Monitoring der bereitgestellten Infrastruktur. Dabei wird die Nutzung essenzieller Rechenressourcen wie CPUs, RAM und GPUs systematisch erfasst. Zusätzlich bietet sie einen detaillierten Überblick über die Verteilung dieser Ressourcen auf ein oder mehrere Systeme [24].

Bei der **Datenüberwachung** in maschinellen Lernsystemen ist es entscheidend, die Konsistenz und Qualität der Daten sicherzustellen. Dies umfasst die Überprüfung, ob das Datenschema während der Inferenzphase mit dem Training übereinstimmt, die Konsistenz und den erwarteten Bereich der Produktionsdaten zu überwachen, die Datenqualität mit den Trainingsdaten zu vergleichen und auf Datenveränderungen oder -drift sowie Ausreißer zu achten [24].

Die **Modellüberwachung** ermöglicht eine kontinuierliche Überwachung der Modellleistung. So kann anhand von Metriken wie Genauigkeit, AUC Wert, Precision, Recall, FPR, TPR und Konfusionsmatrix überprüft werden, ob das aktuelle Modell den Erwartungen entspricht. Zudem können weitere Metriken, wie die Feature Importance, Fairness oder Modell Drift aufgezeichnet werden [7]. Auch können hier Metadaten über ML Modelle, wie die aktuelle Version aufgezeichnet werden [24].

Tabelle VII fasst die Anforderungen zusammen, die zum Aufbau einer MLOps Pipeline Level 1 notwendig sind.

Tabelle VII
ANFORDERUNGEN AN MLOPS TOOLS

Kategorie	Anforderungen
Allgemeine Anforderungen	Open Source, Skalierbarkeit, Integrierbarkeit, Reifegrad, Benutzerfreundlichkeit
Datenverarbeitung	Datenquellenhandhabung, ETL Prozesse, Datenmanagement, Versionierung
ML Modell	Hyperparameteroptinierung, Registrierung, Bereitstellung
Betrieb und Überwachung	Systemüberwachung, Datenüberwachung, Modellüberwachung

Die Anforderungsliste kann noch erweitert werden, um alle MLOps Prozesse aus Abbildung 4 abzudecken [7].

B. MLOps Tools

Um die Forschungsfrage **RQ2: Welche MLOps Tools existieren für den Aufbau von ML Pipelines?** zu beantworten, wurden die in zehn relevanten Veröffentlichungen [7][24][10][35][23][34][31][33][32][30], aus Tabelle III (MLOps Tools (RQ2)), genannten MLOps Tools erfasst. Die Häufigkeit ihres Vorkommens wird in Abbildung 7 als Word Cloud dargestellt.

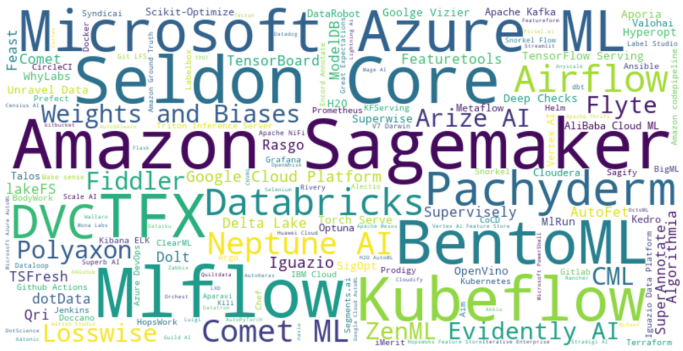

Abbildung 7. MLOps Tool Nennungen aus neun untersuchten Veröffentlichungen, Tabelle III (MLOps Tools (RQ2))

Hierbei zeigte sich, dass Plattformen wie **Amazon SageMaker**, **Microsoft Azure ML**, **Google Cloud Platform** und **Databricks** am prominentesten vertreten sind. Bei den Open Source Tools wurden **MLflow**, **Kubeflow**, **TensorFlow Extended (TFX)**, **Pachyderm**, **Data Version Control (DVC)** und **Airflow** am häufigsten genannt.

Die aus den zehn Veröffentlichungen extrahierten MLOps Tools wurden mit den in der GLR im Unterkapitel IV-C identifizierten Tools verglichen. Es zeigte sich, dass die in Tabelle IV aufgeführten Quellen die Landschaft der MLOps Open Source Tools zum aktuellen Zeitpunkt vollständig abdecken. Folglich konnte in Tabelle VIII eine umfassende Liste aller Open Source MLOps Tools zusammengestellt werden, die für den Aufbau einer automatisierten MLOps Pipeline erforderlich sind. Diese Zusammenstellung erfüllt alle in Unterkapitel VI-A definierten Anforderungen und bietet damit ein nahezu vollständiges Bild der Open Source MLOps Landschaft. In der Auflistung sind die Tools innerhalb jeder Kategorie nach der Anzahl ihrer GitHub Sterne in absteigender Reihenfolge geordnet. Dabei wurden nur jene Tools berücksichtigt, die mehr als 2000 Sterne auf GitHub aufweisen. Zudem wurde eine Obergrenze von maximal zehn Tools pro Kategorie festgelegt, um eine fokussierte und relevante Auswahl zu gewährleisten.

Eine Übersicht von den populärsten E2E ML Plattformen und Cloud Anbietern wurde in Tabelle IX erstellt. Eine E2E ML Plattform ist dabei eine Cloud Umgebung, die alle für ML Pipelines erforderlichen Schritte umfasst, von der Datenspeicherung bis zum automatisierten Deployment [31]. Es ist zu beachten, dass die Dienste dieser Anbieter in der Regel kostenpflichtig sind, wobei einige von ihnen eine begrenzte kostenlose Nutzungsoption anbieten, die allerdings mit eingeschränkten Ressourcen verbunden ist.

VII. MLOPS ZUR ZEITREIHENANALYSE

Um die zentrale Forschungsfrage zu adressieren, **Welche MLOps Werkzeuge eignen sich unter gegebenen Anforderungen am besten für den Aufbau einer ML Workflow Pipeline zur Zeitreihenanalyse?**, wurde eine automatisierte MLOps Pipeline für die Vorhersage der Kursrichtung des S&P 500 erstellt [54]. Hierbei wurde modelliert, ob der Kurs innerhalb der nächsten 15 Tage steigen (1) oder fallen (0) wird. Die Vorhersage der Kursrichtung hat sich als praktikabler herausgestellt als die Präzisierung des exakten zukünftigen Kurses [43]. Das Ziel war die Entwicklung eines prototypischen Ansatzes einer vollständigen MLOps Pipeline auf Niveau 1, wie in Abbildung 5 dargestellt, speziell für den Anwendungsfall der Aktienkursprognose. Im Rahmen der Implementierung wurden die in Unterkapitel VI-A definierten Anforderungen mit geeigneten MLOps Tools umgesetzt. Dabei wurde ein besonderes Augenmerk auf die Minimierung der initialen Lernkurve gelegt, um eine schnelle Einarbeitung zu ermöglichen, ohne dabei die Skalierbarkeit und Integrationsfähigkeit der Werkzeuge zu kompromittieren. Dieser Ansatz zielt darauf ab, einen Prototyp schnell und effizient erstellen zu können und zugleich eine solide Grundlage für zukünftige Erweiterungen im betrieblichen Kontext zu schaffen. Weiterhin wurde bei der Konzeption der Pipeline Wert auf eine modulare Architektur gelegt, um die Austauschbarkeit der Tools zu gewährleisten, ohne die kontinuierliche Betriebsbereitschaft der Pipeline zu beeinträchtigen. Die Notwendigkeit einer solchen Modularität ergibt sich aus der Erkenntnis, dass ein generalisiertes Pipelinedesign oft nicht realisierbar ist und somit die Anpassungsfähigkeit im Vordergrund steht [6].

A. Vorhersagepipeline

Bei der Implementierung der MLOps Pipeline kamen verschiedene MLOps Tools zum Einsatz, die sich auf die Hauptkategorien **Pipeline**, **Features**, **ML Modell** und **DevOps** verteilen. Die in der Pipeline verwendeten Open Source Tools sind in Abbildung XI in Grün hervorgehoben. Zusätzlich sind in der Abbildung alternative Tools aufgeführt, die als mögliche

Ersatzoptionen für die primär genutzten Werkzeuge dienen. Die Tools sind absteigend nach Anzahl der Sterne auf GitHub gelistet. Tabelle X listet weitere mögliche Optionen für den ML Prozess im Umgang mit Zeitreihen. Dabei werden ML und Deep Learning Open Source Tool Alternativen aufgezeigt sowie spezielle Tools für die Zeitreihenanalyse. Für den Aufbau der Pipeline wurden folgende Tools verwendet:

- **Ray**: Workflow/Datenpipeline, Computation Load Distribution, Modell Training Orchestration
- **Grafana, Prometheus**: Logging und Monitoring
- **InfluxDB**: Datenbank
- **TSFresh**: Feature Engineering für Zeitreihendaten
- **MLflow**: Modell Lifecycle, Serving, Monitoring und Versionierung
- **Sklearn**: ML Modell
- **Auto-Sklearn**: AutoML
- **Scikit Optimize**: Hyperparameter Tuning
- **Github**: Versionskontrolle
- **Github Actions**: CI/CD, Trigger
- **Streamlit.io**: APP Framework
- **AWS SageMaker**: Cloud Endpoint
- **Unittest**: Test Tool

In der Entwicklung der MLOps Pipeline, dargestellt in Abbildung 9, wurden die in Unterkapitel V-C definierten Rollen berücksichtigt und den entsprechenden Aufgabenbereichen innerhalb des Pipeline Erstellungsprozesses zugeordnet.

Die historischen Kursdaten des S&P 500 werden für den Zeitraum vom 01. August 2000 bis zum aktuellen Datum mittels der Yahoo Finance API bezogen [55]. Im Rahmen der Datenaufbereitung widmet sich der Data Scientist dem Feature Engineering mit dem Ziel, die Prognosegüte des ML Modells zu optimieren. Hierzu werden unter Verwendung der Python Bibliothek TA-Lib [56] 20 technische Indikatoren basierend auf dem Schlusskurs sowie dem Höchst- und Tiefstand des S&P 500 und dem Handelsvolumen berechnet. Ergänzend werden 29 zusätzliche Chartdaten einschließlich weiterer Aktienindizes und Rohstoffpreise extrahiert und in den Datensatz integriert. Im Zuge der Vorverarbeitung werden zwei separate Datensätze erstellt, ein Trainingsdatensatz für das ML Modell im CSV Format und ein weiterer Datensatz, der die Featuredaten des vorangegangenen Tages enthält, um Prognosen darüber abzugeben, ob der Aktienkurs steigen oder fallen wird. Dieser Datensatz wird automatisiert in einer Datenbank gespeichert. Für die Wahl des optimalen ML Modells nutzt der Data Scientist **Auto-Sklearn**, eine AutoML Bibliothek, welche zu einem bestimmten Datensatz das beste Sklearn Modell bestimmt. Als optimales **Sklearn** Modell wurde der Extra Tree Algorithmus bestimmt, welcher sich ähnlich wie das baumbasierte Random Forrest Modell verhält [5]. Im nächsten Schritt wurde das Extra Tree Modell mit dem aktuellsten Trainingsdatensatz trainiert und ein Hyperparameter Tuning durchgeführt. Für die Orchestrierung des Trainingsprozesses sowie die effiziente Verteilung der Ressourcen auf eine Virtual Machine (VM) kam das Tool **Ray** zum Einsatz. Ray ist ein Open Source Framework für das Skalieren von KI- und Python Anwendungen [57]. Die Implementierung von Ray ermöglicht eine optimierte Ausnutzung vorhandener Rechenressourcen und bietet die Flexibilität für eine potenzielle Skalierung zukünftiger Trainingsprozesse. Zusätzlich können durch die Integration der Tools **Prometheus** und **Grafana**, die verbrauchten Ressourcen und die Leistungsdaten des Systems in einem Dashboard übersichtlich dargestellt und überwacht werden. Ray wurde speziell ausgewählt, um den Trainingsprozess für größere Modelle wie Large Language Models (LLMs) oder Reinforcement Learning (RL) Modelle zu skalieren. Ressourcenintensive Modelle können so nahtlos in die bestehende Pipeline integriert werden. Es ist auch möglich Workflows mit Ray zu erstellen, was den Einsatz eines weiteren Tools für diese Aufgabe überflüssig macht. Wenn das trainierte und getunte Modell eine höhere Genauigkeit der Testdaten erreicht, wird es über das **MLflow** Tool registriert und in die Production Stage überführt. Dabei werden die Genauigkeitswerte und Versionen der Modelle gespeichert. Abbildung 8 zeigt das aktuell beste Sklearn Modell in der Production Stage.

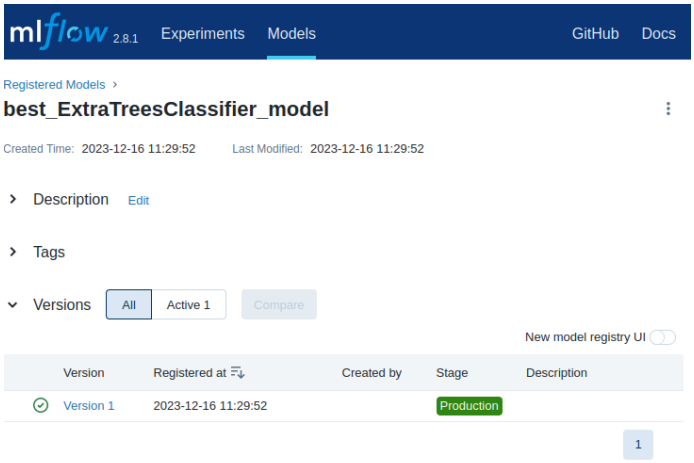

Abbildung 8. MLflow User Interface.

Nach dem Deployment ist das Modell über MLflow Serve ansprechbar und kann Prognosen zu aktuellen Daten liefern. Dieser Prozess muss zunächst manuell gesteuert werden. Um diesen Workflow zu automatisieren kann eine Cloud Umgebung wie **Amazon SageMaker** verwendet werden. Dabei wird das am besten bewertete, registrierte Modell in einem Container in der Cloud bereitgestellt und als Endpoint zugänglich gemacht. Durch die Integration mit dem DevOps Tool **GitHub Actions** kann der gesamte Prozess, von der Modelltrainierung über den Upload in die Cloud bis hin zur Bereitstellung des Modells, vollständig automatisiert werden. Der MLOps Engineer konzipiert hierbei die Automatisierungsschritte, in Absprache mit den Business Stakeholdern. Da das Bereitstellen von ML Modellen mit hohen Kosten verbunden sein kann, ist hier eine Abwägung nötig. Eine Cloud Umgebung bietet die Möglichkeit eine vollautomatisierte MLOps Pipeline zu realisieren. In einer solchen Umgebung können ML Modelle automatisch und regelmäßig, beispielsweise durch einen zeitlichen Trigger, mit neuen Daten trainiert werden. Sobald ein neu

trainiertes Modell eine bessere Leistung als seine Vorgänger aufweist, kann es automatisch das bestehende Modell ersetzen und als neuer Endpoint in der Produktion dienen. Falls auf eine Cloud verzichtet wird, funktioniert die Pipeline gleich, die Workflows müssen jedoch manuell angestoßen werden. Die Prognosen zu aktuellen Daten werden in der **InfluxDB** gespeichert und über **Streamlit.io** als Pfeilrichtungen auf einem S&P 500 Chart visualisiert, wie in Abbildung 10 dargestellt. Die Pipeline ist so aufgebaut, dass neue Features für die Trainingsdaten einfach getestet und integriert werden können, falls sie die Modellleistung verbessern. Dazu wurde das Tool **TSFresh** getestet, welches aus Zeitreihendaten weitere Features berechnet. Die erstellten Features brachten dabei keinen Mehrwert und wurden nicht aufgenommen. Um die MLOps Pipeline robuster zu gestalten, wurden **Unittests** für das Preprozessing erstellt. Die ML Pipeline ermöglicht somit die automatisierte tägliche Extraktion und Speicherung der aktuellen Daten für die Prognose in der Datenbank. Steht eine Cloud Umgebung zur Verfügung, wird die Prognose ebenfalls automatisiert erstellt und direkt in der Benutzeroberfläche angezeigt. Kann aus Kostengründen keine Cloud genutzt werden, muss der Prognoseprozess täglich manuell angestoßen werden. Hierfür ist ein einfacher Konsolenbefehl ausreichend, um den Workflow anzustoßen. Gleiches gilt für das kontinuierliche Trainieren des ML Modells.

VIII. Schlussfolgerung und weiterführende Arbeit

A. Schlussfolgerung

In dieser Arbeit wurde eine umfassende multivokale Literaturrecherche (MLR) zu MLOps Tools durchgeführt, und es wurde eine Auswahl an Open Source Tools für alle Phasen einer MLOps Pipeline vorgestellt (siehe Tabelle VIII). Zusätzlich wurde eine Liste von häufig genutzten proprietären ML Plattformen erstellt (siehe Tabelle IX). Eine ML Vorhersagepipeline unter Verwendung von MLOps Tools wurde erstellt, wobei auch Alternativen zu den eingesetzten Werkzeugen aufgezeigt wurden. MLOps Architekturen sind grundsätzlich nicht generalisierbar [6]. Dies trifft auch auf Pipelines für Zeitreihenvorhersagen zu. Die entwickelte Pipeline ermöglicht jedoch eine gewisse Generalisierbarkeit. Änderungen des Vorhersagezeitraums, Hinzufügen neuer Features und die Integration neuer Modelle sind möglich, ohne wesentliche Änderungen an der Pipeline Architektur vornehmen zu müssen. Für diese spezifischen Anwendungsfälle wurden Toolsammlungen in Tabelle XI und Tabelle X zusammengestellt. Bei einer Umstellung von Batch Verarbeitung auf Streaming Daten ist die Berücksichtigung zusätzlicher Tools, wie beispielsweise Apache Kafka, erforderlich [53].

Für die Erstellung einer MLOps Pipeline auf Level 1 sind spezialisierte Tools wie MLflow unverzichtbar. Darüber hinaus können gängige DevOps Tools wie GitHub Actions effizient in die Pipeline integriert werden. Diese Integration ist nicht nur möglich, sondern wird auch empfohlen [33]. Während der praktischen Umsetzung der ML Vorhersagepipeline hat sich herausgestellt, dass sowohl MLOps- als auch DevOps Tools und -Praktiken von großem Nutzen waren, um die gestellten Anforderungen zu erfüllen und auftretende Herausforderungen zu bewältigen. Das Konzipieren einer automatisierten MLOps Pipeline kann so bereits in der prototypischen Entwicklungsphase gewinnbringend sein, um eine spätere Skalierbarkeit und Integrierbarkeit zu gewährleisten.

Die Konzeption einer Level 2 MLOps Pipeline, charakterisiert durch die automatisierte Integration neuer Pipelines für zusätzliche ML Modelle in eine übergeordnete Struktur, präsentiert sich im Kontext der prototypischen Entwicklung als eine komplexe Herausforderung. Dieser Mehraufwand erscheint, insbesondere unter Berücksichtigung der Komplexität und Skalierungserfordernisse, erst im Rahmen von Business Anwendungen als gerechtfertigt und zielführend.

Für die erstellte ML Pipeline in Abbildung 9 hat sich das MLOps Tool **Ray** als besonders effektiv erwiesen, da es gängige Frameworks für ML integriert, die Ressourcenverteilung vornimmt und Workflows als Directed Acyclic Graph (DAG) erstellt werden können.

B. Weiterführende Arbeit

Um die Robustheit der erstellten MLOps Pipeline weiter zu erhöhen, ist die Erstellung zusätzlicher Unittests für alle Phasen des Lebenszyklus des ML Modells empfehlenswert. Diese Unittests können so konfiguriert werden, dass sie automatisch über GitHub Actions ausgeführt werden, sobald ein Push Request durchgeführt wird. Diese Praxis trägt nicht nur zur Fehlerminimierung bei, sondern fördert auch die kontinuierliche Qualitätssicherung des Codes und der Funktionalität der Pipeline [9]. Zur weiteren Verbesserung wäre es ratsam, die aktuellen Trainingsdaten mittels eines Feature Stores wie **FEAST** zu versionieren und in einer Datenbank wie **MongoDB** zu speichern. Ein zentrales Metadatenmanagement könnte durch die Implementierung von **DataHub** erreicht werden. Für eine verbesserte Übersicht über die Workflows könnten Tools wie **Apache Airflow** oder **Prefect** eingesetzt werden, da sie eine übersichtliche grafische Darstellung der erstellten Workflows bieten. Falls ein eigenes Kubernetes Cluster zur Verfügung steht, wären **Kubeflow** oder **Metaflow** geeignete Optionen für die Orchestrierung der Pipeline. Für eine vereinfachte Integration neuer ML Modelle in die Pipeline bieten sich Tools wie **ClearML** oder **CML** an.

LITERATUR

[1] D. C. Montgomery. *Introduction to Time Series Analysis and Forecasting*. Wiley Series in Probability and Statistics. 2015.

[2] S. Pelekis, E. Karakolis, T. Pountridis, G. Kormpakis, G. Lampropoulos, S. Mouzakits, and D. Askounis. Deeptsf: Codeless machine learning operations for time series forecasting. *arXiv e-prints*, 2023.

[3] B. Pavlyshenko. Machine-learning models for sales time series forecasting. *Data*, 4(1), 2019.

[4] F. Saâdaoui. A seasonal feedforward neural network to forecast electricity prices. *Neural Computing and Applications*, 28(4):835–847, 2017.

[5] E.K. Ampomah, Z. Qin, and G. Nyame. Evaluation of tree-based ensemble machine learning models in predicting stock price direction of movement. *Information*, 11:332, 2020. https://doi.org/10.3390/info11060332.

[6] R. Subramanya, S. Sierla, and V. Vyatkin. From devops to mlops: Overview and application to electricity market forecasting. *Applied Sciences*, 12(19), Sep 2022.

[7] P. Ruf, M. Madan, C. Reich, and D. Ould-Abdeslam. Demystifying mlops and presenting a recipe for the selection of open-source tools. *Applied Sciences*, 11(19), 2021.

[8] K. Filippou, G. A. Aifantis, G. A. Papakostas, and G. E. Tsekouras. Structure learning and hyperparameter optimization using an automated machine learning (automl) pipeline. *Information*, 14(4):232, 2023.

[9] D. E. Rzig, F. Hassan, and M. Kessentini. An empirical study on ml devops adoption trends, efforts, and benefits analysis. *Information and Software Technology*, 152:107037, 2022.

[10] A. B. Kolltveit and J. Li. Operationalizing machine learning models. In *Proceedings of the 1st Workshop on Software Engineering for Responsible AI*, pages 1–8, New York, NY, USA, 2022.

[11] Juan Pineda-Jaramillo and Francesco Viti. Mlops in freight rail operations. *Engineering Applications of Artificial Intelligence*, 123:106222, 2023.

[12] DeepTSF. Das von iccs entwickelte deeptsf zeitreihenprognose repository im rahmen des i-nergy h2020-projekts. https://github.com/I-NERGY/DeepTSF. Zugriff am 24.11.2023.

[13] ProLaf. Ein projekt zur probabilistischen lastprognose. https://github.com/sogno-platform/proloaf. Zugriff am 24.11.2023.

[14] G. Gürses-Tran and A. Monti. Advances in time series forecasting development for power systems' operation with mlops. *Forecasting*, 4(2):501–524, 2022.

[15] M. Steidl, M. Felderer, and R. Ramler. The pipeline for the continuous development of artificial intelligence models—current state of research and practice. *Journal of Systems and Software*, 199:111615, 2023.

[16] I. Karamitsos, S. Albarhami, and C. Apostolopoulos. Applying devops practices of continuous automation for machine learning. *Information*, 11(7):363, 2020.

[17] I. Figalist, C. Elsner, J. Bosch, and H. Olsson. An end-to-end framework for productive use of machine learning in software analytics and business intelligence solutions. 2020.

[18] E. Nascimento, A. Nguyen-Duc, I. Sundbø, and T. Conte. Software engineering for artificial intelligence and machine learning software: A systematic literature review. *arXiv:2011.03751 [cs.SE]*, 2020.

[19] T. Fredriksson, J. Bosch, and H. H. Olsson. Machine learning models for automatic labeling: A systematic literature review. In *Proc. of the 15th Int. Conf. on Software Technologies - ICSOFT*, pages 552–561. SciTePress, 2020.

[20] S. K. Lo, Q. Lu, C. Wang, H.-Y. Paik, and L. Zhu. A systematic literature review on federated machine learning: From a software engineering perspective. *ACM Comput. Surv.*, 54(5), 2021.

[21] G. Lorenzoni, P. Alencar, N. Nascimento, and D. Cowan. Machine learning model development from a software engineering perspective: A systematic literature review. *arXiv:2102.07574 [cs.SE]*, 2021.

[22] T. Mboweni, T. Masombuka, and C. Dongmo. A systematic review of machine learning devops. In *2022 International Conference on Electrical, Computer and Energy Technologies (ICECET)*, pages 1–6, 2022.

[23] M. Testi, M. Ballabio, E. Frontoni, G. Iannello, S. Moccia, P. Soda, and G. Vessio. Mlops: A taxonomy and a methodology. *IEEE Access*, 10:63606–63618, 2022.

[24] D. Kreuzberger, N. Kühl, and S. Hirschl. Machine learning operations (mlops): Overview, definition, and architecture. *arXiv e-prints*, 2022.

[25] M. M. John, H. H. Olsson, and J. Bosch. Towards mlops: A framework and maturity model. In *2021 47th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 1–8, 2021.

[26] M. John, H. Olsson, and J. Bosch. *Architecting AI Deployment: A Systematic Review of State-of-the-Art and State-of-Practice Literature*, pages 14–29. 2021.

[27] L. E. Lwakatare, I. Crnkovic, E. Rånge, and J. Bosch. From a data science driven process to a continuous delivery process for machine learning systems. In *Product-Focused Software Process Improvement*, pages 185–201. Springer International Publishing, 2020.

[28] Y. Xie, L. Cruz, P. Heck, and J.S. Rellermeyer. Systematic mapping study on the machine learning lifecycle. *arXiv preprint arXiv:2103.10248*, 2021.

[29] A. Lima, L. Monteiro, and A. Furtado. Mlops: Practices, maturity models, roles, tools, and challenges – a systematic literature review. In *Proc. of the 24th Int. Conf. on Enterprise Information Systems - Vol. 2: ICEIS*, pages 308–320, 2022.

[30] S. Moreschi, G. Recupito, V. Lenarduzzi, F. Palomba, D. Hastbacka, and D. Taibi. Toward end-to-end mlops tools map: A preliminary study based on a multivocal literature review. *arXiv preprint arXiv:2304.03254*, 2023.

[31] G. Recupito, F. Pecorelli, G. Catolino, S. Moreschini, D. Nucci, F. Palomba, and D. A. Tamburri. A multivocal literature review of mlops tools and features. In *2022 48th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 84–91, 2022.

[32] M. Heydari and Z. Rezvani. Challenges and experiences of iranian developers with mlops at enterprise. 2023.

[33] S. Wazir, G. Siddharth Kashyap, and P. Saxena. Mlops: A review. *arXiv e-prints*, 2023.

[34] Georgios Symeonidis, Evangelos Nerantzis, Apostolos Kazakis, and George A. Papakostas. Mlops - definitions, tools and challenges. *CoRR*, abs/2201.00162, 2022.

[35] N. Hewage and D. Meedeniya. Machine learning operations: A survey on mlops tool support. *arXiv e-prints*, 2022.

[36] V. Garousi, M. Felderer, and M. V. Mäntylä. Guidelines for including the grey literature and conducting multivocal literature reviews in software engineering. *CoRR*, abs/1707.02553, 2017.

[37] B. Kitchenham. Procedures for performing systematic reviews. *Keele University*, 33(2004):1–26, 2004.

[38] D. A. Tamburri. Sustainable mlops: Trends and challenges. In *2020 22nd International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, pages 17–23, 2020.

[39] T. Mucha, K. Abhari, and S. Ma. Riding a bicycle while building its wheels: The process of machine learning-based capability development and it-business alignment practices. *Internet Research*, 33, 2023.

[40] Á. et al. López García. A cloud-based framework for machine learning workloads and applications. *IEEE Access*, 8:18681–18692, 2020.

[41] A. I. U. Tabassam. Mlops: A step forward to enterprise machine learning. *arXiv e-prints*, 2023.

[42] M. Nabipour, P. Nayyeri, H. Jabani, S. Shahab, and A. Mosavi. Predicting stock market trends using machine learning and deep learning algorithms via continuous and binary data; a comparative analysis. *IEEE ACCESS*, 8:150199–150212, 2020.

[43] W. Jiang. Applications of deep learning in stock market prediction: Recent progress. *Expert Systems with Applications*, 184:115537, 2021. https://www.sciencedirect.com/science/article/pii/S0957417421009441.

[44] C. Wohlin. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *Proc. of the 18th International Conference on Evaluation and Assessment in Software Engineering (EASE '14)*, London, England, United Kingdom, 2014.

[45] Liste mi mlops tools. https://github.com/kelvins/awesome-mlops. Zugriff am: 12.12.2023.

[46] Liste mit production ml tools. https://github.com/EthicalML/awesome-production-machine-learning. Zugriff am: 12.12.2023.

[47] Liste mit dataops tools. https://github.com/kelvins/awesome-dataops. Zugriff am: 12.12.2023.

[48] Liste mit devops tools. https://github.com/wmariuss/awesome-devops. Zugriff am: 12.12.2023.

[49] Liste mit data science tools. https://github.com/krzjoa/awesome-python-data-science. Zugriff am: 12.12.2023.

[50] Liste mit python tools. https://github.com/vinta/awesome-python. Zugriff am: 12.12.2023.

[51] F. Zhengxin et al. Mlops spanning whole machine learning life cycle: A survey. *ArXiv*, abs/2304.07296, 2023.

[52] Google Cloud. Mlops: Continuous delivery und pipelines zur automatisierung im maschinellen lernen. https://cloud.google.com/architecture/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning?hl=de. Zugriff am: 06.12.2023.
[53] Mariam Barry and Montiel. Streammlops: Operationalizing online learning for big data streaming real-time applications. pages 3508–3521, 04 2023.
[54] Projekt Code. MLOps Stock Prediction Pipeline. https://github.com/erikautenrieth/stock-prediction, 2023.
[55] Yahoo Finance. Api für die extraktion von aktienkursen. https://developer.yahoo.com/api/. Zugriff am: 15.11.2023.
[56] TA-Lib. Framework zur berechnung von technischen indikatoren. https://ta-lib.org/. Zugriff am: 15.11.2023.
[57] Ray. Open source framework für das skalieren von ki- und python anwendungen. https://github.com/ray-project/ray. Zugriff am 02.11.2023.

ANHANG

MLOps Tools

Tabelle VIII
OPEN SOURCE MLOPS TOOLS MIT DEN MEISTEN GITHUBSTERNEN

Kategorie	Tools
Daten	
Daten Catalog	DataHub, CKAN
Daten Enrichment	Snorkel
Daten Exploration	Pandas Profiling, Jupytext
Daten Ingestion	Apache Kafka, Apache Pulsar, Fluentd, Apache Gobblin
Daten Labelling	Labelimg, Label Studio, CVAT, Doccano, cleanlab, Snorkel, makesense.ai, Argilla, Rubrix, modAL
Daten Management	Milvus, Dolt, Qdrant, Hub, Delta Lake, lakeFS
Daten Processing	Apache Spark, Apache Flink, OpenRefine, Dagster, Apache Beam, Faust, Apache Storm, Azkaban, Apache Nifi
Daten Quality	Cleanlab, Cerberus, Deequ
Daten Stream Processing	Apache Kafka, Apache Flink, Faust
Daten Validation	Cleanlab, Cerberus
Daten Visualization	Apache Superset, Dash, Facets, Lux
Data Warehouse	Apache Hive, Apache Kylin
Data Storage Optimisation	ClickHouse, InfluxDB, Milvus, TimescaleDB, Apache Druid, Apache Arrow, EdgeDB, Chroma, Weaviate, pgvector
Daten Serialization	ProtoBuf, Delta Lake, Kryo, Snappy, Apache Iceberg, Apache Hudi, Apache Avro, Pigz, Apache Parquet
Datenbank	Elasticsearch, Redis, etcd, CockroachDB, InfluxDB, RethinkDB, MongoDB, Milvus, Dragonfly, TimescaleDB
Feature Engineering	TSFresh, Featuretools
Feature Store	FEAST
Metadata Management	DataHub, Amundsen
ML Model	
AutoML	MindsDB, NNI, TPOT, Autokeras, Optuna, auto-sklearn, Featuretools, AutoGluon, FLAML, Model Search
Simplification Tools	Turi Create, PyCaret, Ludwig, Hydra, Koalas
Adversarial Robustness	CleverHans, Adversarial Robustness Toolbox (ART), Foolbox
Model Fairness und Privacy	AIF360
Model Interpretability	SHAP, LIME, InterpretML, Lucid, Captum, Alibi
Model Lifecycle	MLflow, Weights and Biases, Aim, Sacred
Model Serialisation	ONNX, MMdnn
Model Serving	Streamlit.io, Gradio, LocalAI, PredictionIO, Triton Inference Server, Cog, BentoML, Vespa, TorchServe, Opyrator
Model Serving und Monitoring	Jina, LocalAI, PredictionIO, Pandas Profiling, vLLM, Cortex, Triton Inference Server, BentoML, Tensorflow Serving
Model Testing und Validation	Deepchecks
Model Training Orchestration	Skaffold, Kubeflow, NVIDIA TensorRT, Accelerate, SkyPilot, CML, ZenML, Determined, Open Platform for AI, TFX
Model und Data Versioning	Dolt, MLflow, Data Version Control (DVC), ClearML, Aim, Sacred, lakeFS, Polyaxon, Catalyst, TerminusDB
Neural Search	Faiss, Qdrant, Annoy, CLIP-as-service, NMSLIB, DocArray
Hyperparameter Tuning	Hyperopt, KerasTuner, Scikit Optimize, Hyperas
Data Science Notebook	Jupyter Notebooks, Apache Zeppelin, Papermill, VoilÃ , Polynote, ML Workspace, RMarkdown, .NET Interactive
Drift Detection	Alibi Detect
Explaining Black Box Models and Datasets	NETRON, SHAP, MindsDB, LIME, FACETS, CleverHans, InterpretML, captum, DeepVis Toolbox, keras-vis
Benchmarking and Evaluation	Recommenders, Evals, Lucid, BIG-bench
Computer Vision	OpenCV, torchvision, Deep Lake, SuperGradients
NLP	Transformers, LangChain, LLaMA, FastChat, SpaCy, LlamaIndex, Ollama, MLC LLM, Flair, Sentence Transformers
Reinforcement Learning	ML-Agents, Dopamine, FinRL, Stable Baselines, TRL, Gymnasium, AI-Optimizer, CleanRL, Acme, PARL
Recommender Systems	Gorse, Surprise, LightFM, Implicit
Optimized Computation	Jax, XGBoost, Tensor2Tensor, PEFT, Modin, Numba, Nebullvm, Vowpal Wabbit, ggml, Vaex
Outlier and Anomaly Detection	PyOD, Deequ, Alibi Detect
Privacy Preserving ML	PySyft, FATE, FedML, Flower, Microsoft SEAL, Google's Differential Privacy
Visualisation	Apache ECharts, Superset, Streamlit.io, gradio, Redash, matplotlib, Bokeh, Plotly, seaborn, Altair
Visual Analysis and Debugging	Netron, Evidently, Yellowbrick, Whylogs
Pipeline	
CI/CD for Machine Learning	ClearML, CML
Data Pipeline	Apache Airflow, Luigi, Argo, Dagster, Azkaban, Prefect, Ludwig, Dagster, Kedro, DBT, Metaflow, Pachyderm
Machine Learning Platform	ML Workspace
Workflow Tools	Luigi, Argo, Ray Workflow, Prefect, Kedro, Kubeflow, Metaflow, Orchest, ZenML, Flyte, Ploomber
Computation Load Distribution	Colossal-AI, DeepSpeed, Ray, PyTorch Lightning, LightGBM, Horovod, Dask
Logging and Monitoring	Grafana, Prometheus, Loki, Whylogs

Tabelle IX
PLATFORMEN FÜR ML

Kategorie	Tools
Cloud	Amazon Web Services (AWS), Google Cloud Platform (GCP), Azure, Alibaba Cloud, Oracle Cloud, DigitalOcean, Scaleway, Vultr, VMware Cloud, IBM Cloud, Stackpath, Linode, Kinsta
E2E ML Plattform	Amazon SageMaker, Google Cloud Machine Learning, Azure ML, IBM Watson Studio, Anyscale, BigML, CNVRG, Dataiku, Huawei Cloud ModelArts, Iguazio, Katonic, Lightning Ai, Picsell.ia, Stradigi AI, superannotate MLStudio, Valohai, Vertex.ai, Watson Studio, Fiddler, H2O Driverless AI, Comet

Tabelle X
PYTHON TOOLS ZUR ZEITREIHENANALYSE ABSTEIGEND NACH GITHUBSTERNEN (12.12.2023)

Kategorie	Tools
Machine Learning	XGBoost, LightGBM, dlib, PyCaret, CatBoost, imbalanced-learn, mlpack, MLxtend, causalml, cuML, xLearn, Shogun, modAL
Deep Learning	TensorFlow, transformers, PyTorch, Caffe, JAX, pytorch-lightning, MXNet, Ludwig, TFLearn, Sonnet, TensorLayer, autograd, tensorpack, skorch, FLAX, ignite, Polyaxon, Catalyst, nnabla, Gluon, Tangent, Hyperas
Zeitreihenanalyse	Prophet, sktime, darts, maya, statsforecast, tslearn, PyFlux, neuralforecast

Tabelle XI
OPEN SOURCE MLOPS TOOLS ABSTEIGEND NACH GITHUBSTERNEN (12.12.2023)

Kategorie	Tools
Pipeline	
Workflow/Datenpipeline	Apache Airflow, Ray, Luigi, Argo, Prefect, Ludwig, Dagster, Kubeflow, Kedro, DBT, Metaflow, Pachyderm, Azkaban, Chronos, Flyte, Orchest, Ploomber, ZenML
Daten Processing	Apache Spark, Apache Flink, OpenRefine, Dagster, Apache Beam, Faust, Apache Storm, Azkaban, Apache Nifi
Computation Load Distribution	Colossal-AI, DeepSpeed, Ray, PyTorch Lightning, LightGBM, Horovod
Logging und Monitoring	Grafana, Prometheus, Loki, Whylogs
Metadaten Management	DataHub, Amundsen, Apache Atlas
Features	
Daten Management	Milvus, Dolt, Qdrant, Hub, Delta Lake, lakeFS
Datenbank	Elasticsearch, Redis, etcd, CockroachDB, InfluxDB, RethinkDB, MongoDB, Milvus, Dragonfly, TimescaleDB, Qdrant, RQLite, PostgreSQL, ArangoDB
Feature Engineering	TSFresh, Featuretools
Feature Store	FEAST
CI/CD ML	ClearML, CML
ML Model	
ML Plattform	ML Workspace
Model Lifecycle	MLflow, Weights and Biases, Aim, Sacred
Model Serving und Monitoring	MLflow, Jina, LocalAI, PredictionIO, Pandas Profiling, vLLM, Cortex, Triton Inference Server, BentoML, Tensorflow Serving, Nuclio, Evidently, Seldon Core, TorchServe, Deepchecks, m2cgen, KServe, DeepDetect, Giskard, WhyLogs
Model Testing und Validation	Deepchecks
Model Training Orchestration	Skaffold, Kubeflow, Ray Train, NVIDIA TensorRT, Accelerate, SkyPilot, CML, ZenML, Determined, Open Platform for AI, TFX
Model und Daten Versioning	Dolt, MLflow, Data Version Control (DVC), ClearML, Pachyderm, Aim, Sacred, lakeFS, Polyaxon, Catalyst, TerminusDB
Hyperparameter Tuning	Hyperopt, KerasTuner, Ray Optimize, Scikit Optimize, Hyperas
AutoML	MindsDB, Neural Network Intelligence, TPOT, Autokeras, Optuna, auto-sklearn, Featuretools, AutoGluon, FLAML, Model Search
DevOps	
Versions Control	Github, Gitlab
CI/CD	Github Actions, GitLab CI/CD, Jenkins
APP Framework	React, Angular, Django, Streamlit.io, Gradio
Tests	Unittest, Junit, Pytest

Vorhersagepipeline

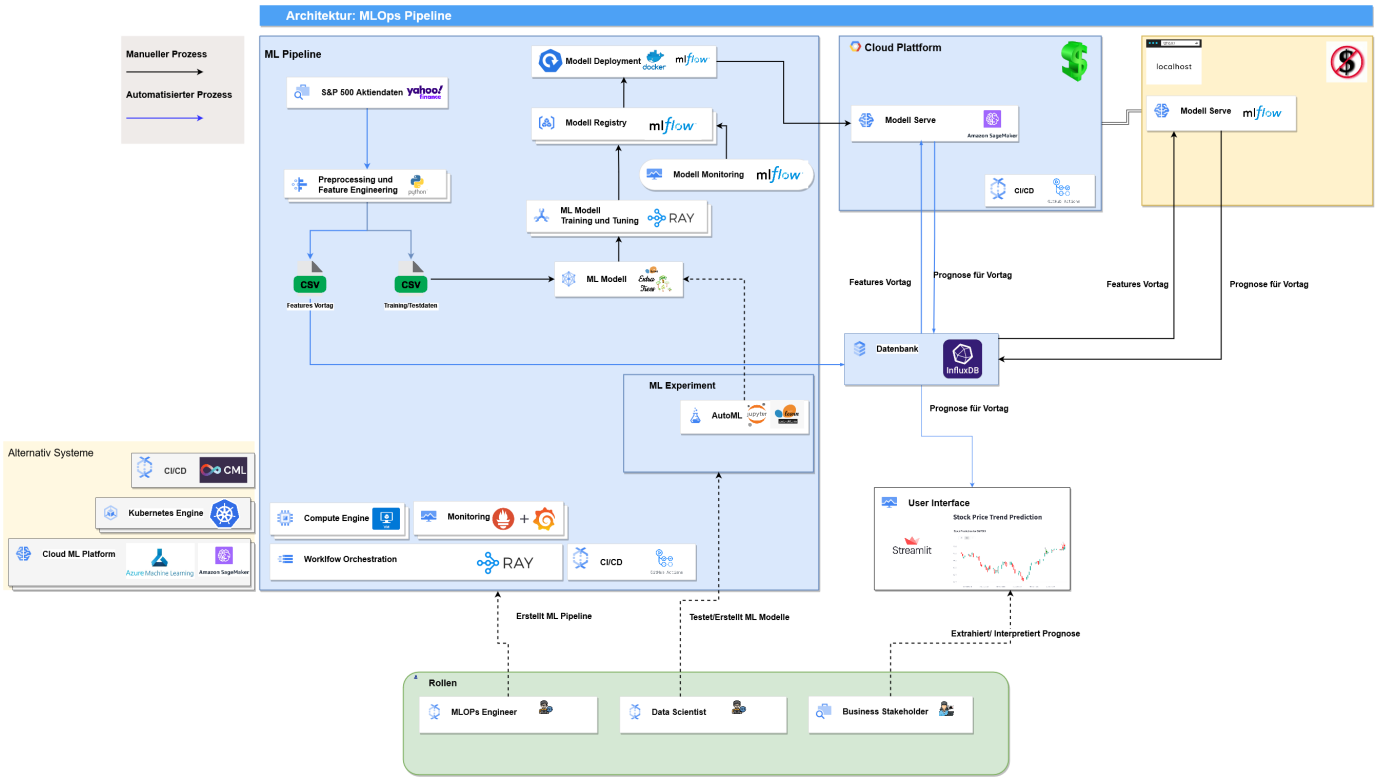

Abbildung 9. Architektur der Vorhersagepipeline [54].

User Interface

Model Information and Accuracy

	Date	Target	accuracy	model
16	2023-12-01	0	0.8477	ExtraTreesClassifier
17	2023-12-01	0	0.9151	ExtraTreesClassifier
18	2023-12-04	0	0.9048	ExtraTreesClassifier

Abbildung 10. User Interface der Vorhersagepipeline: https://stock-prediction-extra-tree.streamlit.app/.