Artificial Intelligence Review (2025) 58:167
https://doi.org/10.1007/s10462-025-11164-3

Machine learning operations landscape: platforms and tools

Lisana Berberi[1] · Valentin Kozlov[1] · Giang Nguyen[3,5] · Judith Sáinz-Pardo Díaz[2] · Amanda Calatrava[4] · Germán Moltó[4] · Viet Tran[3] · Álvaro López García[2]

Accepted: 19 February 2025
© The Author(s) 2025

Abstract
As the field of machine learning advances, managing and monitoring intelligent models in production, also known as machine learning operations (MLOps), has become essential. Organizations are increasingly adopting artificial intelligence as a strategic tool, thus increasing the need for reliable, and scalable MLOps platforms. Consequently, every aspect of the machine learning life cycle, from workflow orchestration to performance monitoring, presents both challenges and opportunities that require sophisticated, flexible, and scalable technological solutions. This research addresses this demand by providing a comprehensive assessment framework of MLOps platforms highlighting the key features necessary for a robust MLOps solution. The paper examines 16 MLOps tools widely used, which revolve around capabilities within AI infrastructure management, including but not limited to experiment tracking, model deployment, and model inference. Our three-step evaluation framework starts with a feature analysis of the MLOps platforms, then GitHub stars growth assessment for adoption and prominence, and finally, a weighted scoring method to single out the most influential platforms. From this process, we derive valuable insights into the essential components of effective MLOps systems and provide a decision-making flowchart that simplifies platform selection. This framework provides hands-on guidance for organizations looking to initiate or enhance their MLOps strategies, whether they require an end-end solutions or specialized tools.

Keywords Machine learning operations · MLOps platforms · Performance monitoring · Decision-making

1 Introduction

Artificial Intelligence (AI)/Machine Learning (ML) is progressively being included as a crucial solution in the design of new software systems across different industries. However, these ML-based systems bring new challenges of the software development process as compared to the ones we have been familiar with. For example, there are no model and/or data versions in the manual ML pipeline, no model lineage available, no tracking of different AI/

Extended author information available on the last page of the article

ML experiment-runs in an automatic way, no performance monitoring of models in production. Therefore, manual processes to create and control ML systems can cause additional costs, lack of reproducibility, and future problems in maintaining them (Ruf et al. 2021).

Furthermore, successful development requires collaboration between various specialists with different sets of skills and tools: software developers, data scientists, and ML engineers (Diaz-de Arcaya et al. 2023).

These challenges can be addressed through the following approaches: Automated ML (AutoML) is an attempt to solve the problem of expertise by providing fully automated off-the-shelf solutions for model choice and hyperparameter tuning. It aims to increase the user-friendliness of ML frameworks to make them more accessible to the nonexpert (Schmitt 2023). In addition, Machine Learning Operations (MLOps) complements this by integrating automated workflows into continuous training pipelines, enabling periodic retraining, monitoring, and feedback loops to maintain model quality and ensure scalability, reproducibility, and operational success (Kreuzberger et al. 2023).

In the domain of MLOps, continuous monitoring of model and data behavior is crucial for detecting potential performance degradations (Hu et al. 2020). Deployed ML models, trained on historical data, are susceptible to accuracy challenges when real-world data distributions change-a phenomenon known as data drift. Beyond data distribution shifts, concept drift can occur when the relationship between input variables and target variables evolves over time. To address these challenges, Continuous Training/Testing (CT), as part of the MLOps set of practices, aims to retrain and deploy the model automatically as needed (Google 2020).

The current AI/ML landscape is seeing a growing offer and adoption of MLOps platforms, which serve to automate and streamline tasks in building ML systems (AI-Infrastructure 2023). These platforms are essential for proper coordination of the AI/ML life cycle, there is, however, a need to go for a better understanding and grouping of their capabilities.

To advise companies and academia on the choice of appropriate MLOps platforms, the need to gain insight into the capabilities offered by modern popular open-source MLOps tools arises, which is a major foundation to making informed decisions. These include features such as orchestration, distributed training, code management, model development, model monitoring and others.

While there is a wide variety of MLOps tools available, understanding their capabilities-particularly in model performance monitoring (MPM), drift detection, and other critical features-remains a complex task.

Furthermore, the diverse approaches to drift detection and model performance management across platforms highlight the need for a clear framework that can guide organizations in evaluating and selecting the right tools for their specific needs. Despite the significance of model monitoring, many MLOps platforms offer limited or fragmented solutions, underscoring the necessity for deeper research and better decision-making support for MLOps adoption.

The motivation behind this research is driven by two main challenges posed by the evolving landscape of MLOps platforms and exploring the integration of drift detection capabilities within these platforms.

With a thorough examination of the MLOps platforms, we seek to present a better comprehension of what the platforms can do and to fill the knowledge gaps, if any.

By carrying out these investigations, our research is able to provide helpful insights and solutions, for the field of ML applications and services.

The work presented in this paper makes key contributions to the field of ML operations including:

- Introducing a three-step evaluation framework for MLOps platforms: we combine feature analysis, GitHub stars growth assessment, and a weighted scoring methodology to guide organizations in selecting the most suitable MLOps platform based on their needs.
- Identifying key features of MLOps platforms: we highlight essential features like experiment tracking, model development and orchestration.
- Creating a flowchart for platform selection: we develop a user-friendly flowchart to simplify decision-making, making it easier for organizations to choose the right MLOps platform based on their requirements.
- Investigating drift detection implementations across MLOps tools: we examine how different MLOps tools integrate drift detection, from built-in solutions like EvidentlyAI to customizable frameworks like Alibi-detect.

In this context, the remainder of this work is structured as follows. Section 1.1 presents an overview of ML life cycle management, highlighting the importance of efficient development in AI/ML applications and services. Section 1.2 provides a concise overview of the methodology used in this study and related work after reviewing the literature on the state-of-the-art development of MLOps in Sect. 1.3. Section 2 discusses the crucial role of MLOps in guiding the transition between data science and production systems; hence the article presents the numerous strategies and elaborate tools required in the automation, management, and monitoring of the whole life cycle of ML models. Section 2.1 investigates a notable list of open-source MLOps platforms, evaluating their support levels for key features such as orchestration, distributed training, code management, model development, and more. Section 2.2 describes in more detail each of the reviewed platforms/tools. Section 2.3 discusses the methods used to perform drift detection across the reviewed MLOps platforms. It highlights the diverse approaches to integrating monitoring libraries and methods, aimed at ensuring the ongoing reliability and performance of ML models. Section 2.4 concludes by highlighting the accelerating growth of MLOps platforms within the wider AI landscape. Despite the common goal of supporting the ML life cycle, these platforms vary in their objectives, creating space for diverse and complementary approaches. Finally, Sect. 3 provides a comprehensive overview of the key takeaways from this entire research work.

1.1 Context and scope

The increasing complexity of AI and ML applications in various industries requires a structured approach to development and deployment. The traditional ML life cycle is an iterative process that includes tasks like: *create/update model* (e.g. models can be created from scratch using the prepared dataset as input or use an existing model as a pre-trained model), *train/test model* (train model and log various metrics and parameters), *evaluate model* (evaluating model accuracy), *store model* (in a model registry), *deploy model* (for inference queries), and *monitor model performance* (track evaluation metrics derived from the model

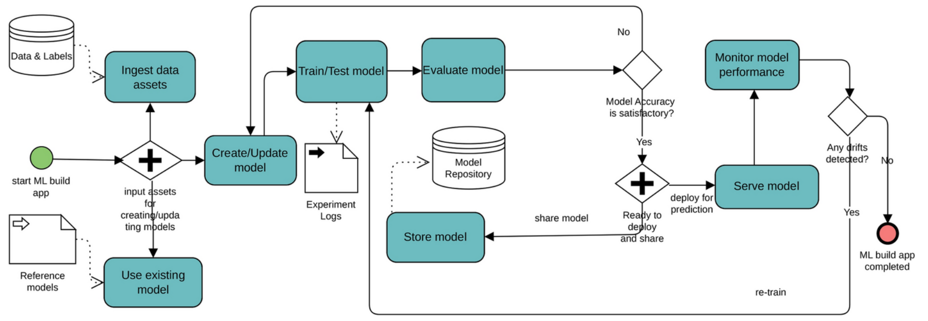


Fig. 1 Business Process Model and Notation (BPMN) diagram of simplified ML life cycle

Table 1 MLOps with CI/CD/CT

Type	Description
Continuous Integration (CI)	CI is no longer only about testing and validating code and components, but also testing and validating data, data schemas and models
Continuous Delivery (CD)	CD is no longer about a single software package or a service, but a system (ML training pipeline) that should automatically deploy another service (e.g. model prediction service)
Continuous Training/Testing (CT)	CT is a new property, unique to ML systems, that is concerned with automatically retraining and serving the models

inference running on new data), as shown in a simplified diagram in Fig. 1. We express this process model using the BPMN notation (OMG 2011) as the actual de facto standard of workflow languages. There are two decision points to check during this sequential flow as designed in the diagram: (1) if the accuracy of the model is not satisfactory, we need to update the model otherwise it is deployed into the production environment; (2) if the performance degrades, we trigger the retraining of the model. To properly manage these tasks as steps, MLOps comes into play. These steps can be completed manually or automatically.

In this paper, we refer to MLOps as an engineering practice that aims to automate and streamline the ML life cycle (Kreuzberger et al. 2023). To achieve this, a robust MLOps toolchain is required, offering functionalities like version control, testing frameworks, deployment automation, continuous integration/delivery (CI/CD), monitoring, and CT.

According to Google (2020), software systems, including ML ones, are similar in the continuous integration of source control, unit testing, integration testing, and continuous delivery of the software module or package (Shahin et al. 2017). However, there are a few notable differences for ML systems (Symeonidis et al. 2022) as presented in Table 1: CI/CD takes into account data and models, and there is a new CT property for continuous training.

MLOps is based on the existing DevOps discipline and its key features are: (1) Automation, (2) Collaboration, (3) Integration, and (4) Configuration Management (GreatLearning 2024; Shahin et al. 2023).

In addition to MLOps, in recent years there has also been a rise in DataOps, ModelOps, and AIOps. A brief description of these life cycle management practices is given in Table 2.

AIOps primarily focuses on improving the efficiency of AI for IT operations through the implementation of AI/ML (Diaz-de Arcaya et al. 2023). For instance, it can identify anomalies in system metrics, such as unusual CPU or memory usage, allowing potential issues

Table 2 Life cycle management practices

Type	Works	Description
DevOps	Luz et al. (2019), Qentelli (2024)	DevOps combines software development (Dev) and IT operations (Ops) to shorten the development life cycle and provide CI/CD with high software quality assurance
MLOps	Kreuzberger et al. (2023), Google (2020)	ML Operations applies DevOps principles to developing, deploying, and managing ML models
ModelOps	Hummer et al. (2019), Lefevre et al. (2022)	ModelOps is a subset of MLOps that enables organizations to operationalize ML models
DataOps	Munappy et al. (2020), Garriga et al. (2021)	DataOps is the practice of applying DevOps principles to data engineering and management
AIOps	Li et al. (2020), Notaro et al. (2021)	AI for IT operations use AI and ML techniques to automate IT issues such as detection, diagnosis, and resolution

to be spoted and resolved quickly before they disrupt operations. In contrast, MLOps is focused on optimizing the effectiveness of ML development and deployment procedures by integrating DevOps principles and methodologies (Remil et al. 2024). For example, MLOps workflows may include a task where metrics like precision and recall are used/provided to track the model performance and ensure continuous improvement. AIOps is the reverse of MLOps in one respect: AIOps is the application of ML to DevOps, rather than the application of DevOps to ML.

However, as there is no MLOps standardized approach to manage the ML life cycle, because each ML project has its own particularities, such as project goals and requirements, the MLOps maturity model (Microsoft 2022; Google 2020) has been recently introduced as a metric to indicate to what extent these steps are automated. This maturity model has been proposed by commercial vendors such as Google and Microsoft. The main difference between the MLOps maturity models of them lies in the granularity and the specific focus. Microsoft's model offers a broader view, with five levels, from zero (no MLOps) to four (Full MLOps Automated Retraining). It emphasizes the level of automation across the entire ML life cycle, from development to deployment and monitoring. Each stage highlights the most important functions and potential weaknesses of the ML life cycle steps (Microsoft 2022).

Google's model focuses on the level of automation in the training pipeline itself and comprises three stages, from zero (manual process) to two (CI/CD pipeline automation). It emphasizes the evolution from manual training to CT with automated pipelines (i.e., Data pipeline, ML pipeline) and CI/CD integration (Google 2020)

Therefore, industries involved in AI and ML projects can use this maturity model to clarify what level of automation they need to achieve first and then gradually improve it, rather than being overwhelmed by the demands of a fully mature environment.

In addition, certain open-source MLOps tools or a combination of them, which we have examined in this paper (see Sect. 2.2), can be used to achieve the different MLOps maturity levels, depending on their individual capabilities.

Commercial MLOps platforms such as Azure ML, AWS SageMaker, and GCP Vertex AI (López García et al. 2020) offer comprehensive ML capabilities, but can have higher ongoing costs due to the iterative nature of the ML life cycle. While powerful, these platforms can be expensive and do not directly solve the core ML problem through a single dashboard.

In contrast, open source MLOps products offer, such as: Cost-effectiveness (e.g., they are a good alternative to commercial platforms, suitable for projects with low or limited budgets), accessibility (e.g., they are freely available and often with permissive licensing models) and flexibility (e.g., they offer more control and customization options compared to some commercial platforms) we have focused our work exclusively on open-source products.

1.2 Methodology

Our methodology is illustrated in Fig. 2. It adopts a mixed approach, combining information from current scientific publications (see Sect. 1.1), identified MLOps tools (see Sect. 2.1), and reports to better understand technical components.

Our comprehensive analysis of the MLOps landscape (see Fig. 2) focuses on the ML life cycle management as the main component.

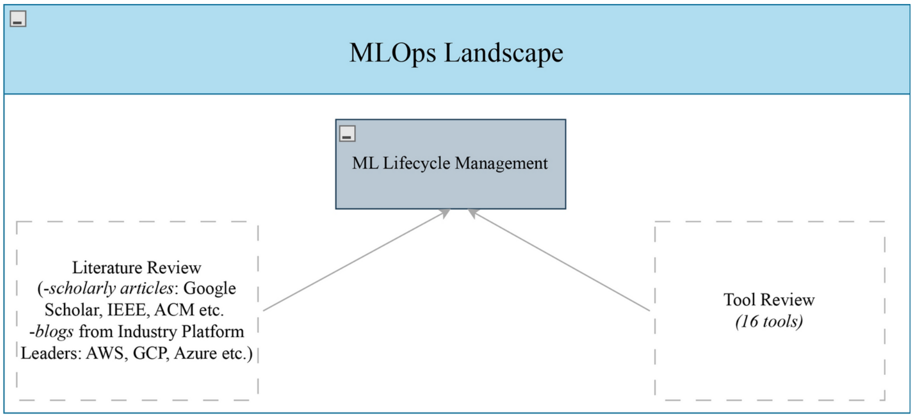

MLOps Landscape
ML Lifecycle Management
Literature Review (-scholarly articles: Google Scholar, IEEE, ACM etc. -blogs from Industry Platform Leaders: AWS, GCP, Azure etc.)
Tool Review (16 tools)

Fig. 2 Overview of the methodology

To gain valuable insight on emerging trends and best practices, we thoroughly reviewed more than 20 scholarly articles retrieved from Google Scholar, IEEE Xplore, ACM Library, Springer Link and Elsevier and more than 10 industry-leading blogs retrieved from Google Cloud Platform, Amazon Web Services (AWS), Microsoft Azure, etc. for ML life cycle management, along with a careful examination of 16 associated tools.

During our first exploratory analysis search in the recent year, we retrieved more than 230 articles on ML life cycle management. From this collection, around 45 papers were screened in detail. Out of those, more than 20 articles were selected based on specific inclusion criteria, such as containing terms like *MLOps AND Review*, *MLOps AND Survey*, or *ML life cycle* combined with concepts like *CI/CD*.

1.3 Related work

It is known that many research studies have contributed to the changing environment of MLOps through in-depth reviews and analysis. Insightful works, including (Subramanya et al. 2022; Mboweni et al. 2022; Kreuzberger et al. 2023; Diaz-de Arcaya et al. 2023), explore theoretical concepts in depth, while others (Zhou et al. 2020; Barrak et al. 2022) focus on practical applications. Additionally, investigations into MLOps tools presented by (Ruf et al. 2021; Hewage and Meedeniya 2022; Symeonidis et al. 2022) together provide a multifaceted perspective on the MLOps framework.

Research work of (Subramanya et al. 2022) is mostly about reviewing the state-of-the-art in MLOps, introducing DevOps principles related to MLOps, and applying an MLOps framework to a specific time-series forecasting application. However, our research, based on prior work such as (Hewage and Meedeniya 2022), goes further, covering a broader spectrum of MLOps tools by including open-source alternatives. Additionally, our investigation verifies a more diverse set of features than previously explored, recognizing the strengths and limits of MLOps tools.

Compared to (Symeonidis et al. 2022), which classifies tools by their primary use, such as data pre-processing (data versioning and data labeling), operationalization (model development, monitoring) or modeling ones (experiment tracking, hyperparameter tuning);

our work challenges this exclusive assignment, recognizing the potential limitations. We emphasize that tools can cross specific categories through library imports or the development of new plugins (in our work, we mark it as partially supported), thus expanding their functionality within the MLOps workflow.

The systematic review conducted by Mboweni et al. (2022) provides a scholarly exploration of ML-DevOps (MLOps) from 2015 to 2022. Our study instead explores not only the potential of MLOps, but we also examine the well-known open-source tools and frameworks so that the study is more practical. This approach contributes to a complete knowledge base aimed at researchers, practitioners, and industry experts.

The work in Kreuzberger et al. (2023) adopts a mixed method research approach, which covers literature reviews, tool evaluations, and expert interviews. Their work provides a holistic view of the principles, components, roles, architectures, and workflows necessary for MLOps. Our research complements this by offering a more detailed examination of various MLOps tools and their functionalities that can be useful for understanding how this can be applied in practice.

Recent works on MLOps underline its dynamic nature and challenges. Martinez and Kifle (2024) underline MLOps as a holistic framework for transitioning AI systems from development to deployment, one that is in line with data-centric AI principles and covers related domains such as DataOps and AIOps. They emphasize MLOps' increasing importance in dealing with end-to-end AI system design and operations. Complementing this, the research in Amrit and Narayanappa (2024) investigates the multifaceted challenges of MLOps implementation across different organizational contexts, identifying four key challenge dimensions and revealing both unique MLOps-specific and shared DevOps challenges through qualitative analysis and practitioner interviews.

Other relevant works include Zhou et al. (2020), analyzing time and resource consumption in the ML pipeline, Moreschini et al. (2022) proposing a graphical representation for DevOps in ML-based applications, Paleyes et al. (2022) exploring challenges in the deployment of ML, Barrak et al. (2022) investigating trends in the use of serverless computing for ML deployment, and Ruf et al. (2021) providing a recipe for tool selection, each contributing valuable insights to the MLOps domain.

This MLOps review paper differentiates itself from existing publications in the field by introducing a comprehensive assessment framework for MLOps platforms, which is organized in three clear steps. The framework includes feature analysis, GitHub stars growth, and a scoring mechanism using weights to help organizations find the platform that best fits their needs. It also highlights key functionalities not limited to experiment tracking and model development and provides a decision-making flowchart to help simplify the platform selection process.

In the next section, we describe in more detail the open-source MLOps platform capabilities.

2 MLOps platforms

MLOps as the bridge between data science and the production system is the backbone of ML systems. On the one hand, MLOps covers the practices and tools required to automate, manage, and monitor the entire life cycle of ML models, from development to deployment, and beyond. This includes tasks such as data preparation, model training, version control, CI/CD/CT, as well as data curation and model curation in real-time (Kreuzberger et al. 2023). On the other hand, AI infrastructure refers to the underlying technology stack that supports the MLOps process. This includes hardware, software, and services that enable efficient data storage, processing, model development, and model deployment. It involves cloud computing platforms, data pipelines, distributed computing frameworks, and monitoring tools that work together to ensure smooth operation of ML workloads.

MLOps and AI infrastructures work together to bring AI/ML models from experimentation to production. They enable companies to build, deploy, and maintain AI-powered applications with greater efficiency, scalability, and reliability. Workflow and workload management are essential components of MLOps, enabling companies to efficiently develop, deploy, and maintain ML models on scale.

2.1 MLOps platforms capabilities evaluation

We present a three-step evaluation framework for analyzing MLOps products, with the aim of identifying the critical features of a competent MLOps solution.

- **Feature analysis** Our analysis *begins* with an in-depth evaluation of 16 MLOps platforms across a compact list of categories that include various *capabilities* (also called *features* interchangeably in this work) in the management of AI infrastructure. These capabilities were selected based on the report of AI-Infrastructure (2023), which highlights the most notable AI/ML products, as well as insights gathered from the reviewed literature.
- **GitHub stars growth assessment** In the *second* step, we further examine the 16 platforms by analyzing their average stars growth on GitHub over the last 5 years. We assign weights to each repository by normalizing their growth values to a range of [1, 10] using linear scaling, with the lowest growth receiving a weight of 1 and the highest growth receiving a weight of 10. This metric highlights industry priorities and adoption trends, revealing which technical capabilities and tools developers find most valuable. By closely tracking these metrics, we can gain actionable insights into emerging priorities and pain points in MLOps for the platform design.
- **Weighted scoring and common feature identification** Finally, in the *third* step, we calculate a weighted score for each repository by combining their normalized average stars growth weights with the features they support. The top five repositories with the highest weighted scores are then analyzed to identify the common features among them. The results enable us to derive meaningful and scientifically grounded insights into the essential elements of a robust MLOps solution, bridging the gap between data-driven evaluation and practical industry relevance.

We begin by providing a detailed description of each step of the process, as outlined below:

2.1.1 Step 1: feature analysis

Table 3 compares MLOps platforms based on their level of support for different capabilities. It assigns full and partial scores to each platform to evaluate their overall effectiveness. The features analyzed are detailed below:

- *Orchestration (O)* coordinates and manages the individual workflows that resulted from the disassembling of the end-to-end ML pipeline.
- *Distributed Training (DT)* this feature allows to split up and share the workload of training a model among multiple processors, called worker nodes. These worker nodes work in parallel to speed up model training. Typically used to train DNNs models (Azure 2022).
- *Code Management (CM)* is the process of handling changes to the application source code with control versioning.
- *Model Development (MDV)* involves data acquisition from multiple trusted sources, data processing to build the model, choosing an algorithm to build the model and eventually building the model (Yaram 2021).
- *Model Testing/Validation (MTV)* is about accurately checking (e.g. unit tests) the expected behavior of the model and providing metrics/plots to summarize the performance on a validation/test dataset.
- *Model Inference (MI)* is the process of inferring results from input (i.e., live and unseen) data using a fully trained model.
- *Model Deployment (MDP)* an enterprise-grade MLOps system should allow organizations to deploy their models and generate consistent API access for application teams on the other end, regardless of deployment environments and choice of cloud services and providers (DataRobot 2023).
- *Experiment Tracking and Metadata Store (ETMS)* the former refers to keeping track of the details of each experiment, such as hyperparameters, training data, and evaluation metrics. The metadata store refers to keeping track of the context and dependencies of models and pipelines, such as data sources, code versions, and system configurations (DataCamp 2023). Note that these are two closely related concepts.
- *Data Versioning and Management (DVM)* is about tracking datasets by registering changes in a particular dataset. It also improves data compliance by reviewing data modifications in the audit process (lakeFS 2023).
- *Model Performance Monitoring (MPM)* is the process of observing ML models for possible changes, such as drift detection. The goal is to curate the model to achieve a satisfactory level of performance over time.

In addition to the above one, the following MLOps platform features are also frequently mentioned: *Feature Management* (FM) and *Model Versioning and Management* (MVM). FM is also called a *Feature Store* (FS), and MVM is also called *Model Store* (MS). These features can stay as an MLOps product feature or can be seen as a part of the ETMS feature. The number of desired capabilities is increasing. There is a tendency to break DVM into FM and MVM and a tendency to break ETMS into FS, MS and the rest of metadata. Such similar tendencies can be temporary, which may reflect in platform documentation that evolves

Table 3 Notable open-source MLOps platforms

Product	GitHub stars	O Orchestration	DT Distributed training	CM Code management	MDV Model development	MTV Model testing/validation	MI Model inference	MDP Model deployment	ETMS Experiment tracking and metadata store	DVM Data versioning and management	MPM Model performance monitoring	Full score	Partial score
MLflow	19 K			✓			✓✓	✓✓	✓✓			30%	10%
Prefect	17.7 K	✓✓		✓						✓✓		20%	10%
Kubeflow	14.5 K	✓✓	✓✓	✓	✓✓	✓	✓✓	✓✓	✓✓			60%	20%
Dagster	12 K	✓✓					✓	✓✓		✓✓		30%	10%
W&B (WB)	9.2 K	✓	✓✓	✓✓	✓✓	✓✓		✓	✓✓	✓✓	✓✓	70%	10%
MetaFlow	8.3 K	✓✓	✓✓	✓					✓✓	✓	✓	20%	30%
Mage	8 K	✓✓					✓✓	✓	✓	✓✓	✓	30%	30%
Pachyderm	6.2 K	✓✓	✓	✓✓	✓✓	✓✓	✓✓	✓	✓	✓✓		60%	30%
Flyte	5.8 K	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓		90%	0%
ClearML	5.7 K	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	100%	0%
Seldon core	4.4 K	✓✓				✓✓	✓✓	✓✓	✓		✓✓	50%	10%
ZenML	4.2 K	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	100%	0%
Polyaxon	3.6 K	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓	✓✓	90%	10%
TFX	2.1 K	✓✓	✓✓	✓✓	✓✓	✓	✓✓	✓✓	✓	✓✓		70%	20%
MLeap	1.5 K	✓✓					✓✓	✓✓			✓	30%	10%
MLRun	1.5 K	✓✓	✓✓	✓	✓✓	✓	✓✓	✓✓	✓✓	✓✓	✓✓	80%	20%

over time. The reason is the fast-changing and evolving state of MLOps products in the AI Infrastructure landscape.

In Table 3, we have marked the following *support levels*:

- empty = no support (means that the feature is not supported, not available, or not functional),
- 1x (✓ = partially supported, means that some but not all aspects of the feature are supported, or that the feature has limited functionality),
- 2x (✓✓ = supported, means that all aspects of the feature are supported, and it is expected to function as intended).

The most common features of the most notable open-source MLOps products presented in Table 3 are as follows.

- Most of the products are open-source under the Apache 2.0 license. The exception is Pachyderm and Seldon core, which have their specific open-source licenses, instead W&B (WandB) has a MIT license.
- All the above-presented MLOps products support Python, TensorFlow (and Keras as a part of TensorFlow 2) for DL, and Scikit-learn for ML. Most of them support the Jupyter Notebook (except for MLFlow and MLeap). PyTorch is also supported by almost all products (except Pachyderm and MLeap) (Oladele 2024).
- The minority languages supported by some of these products are R, Java, Scala, Go, C++, shell, and Jsonnet.

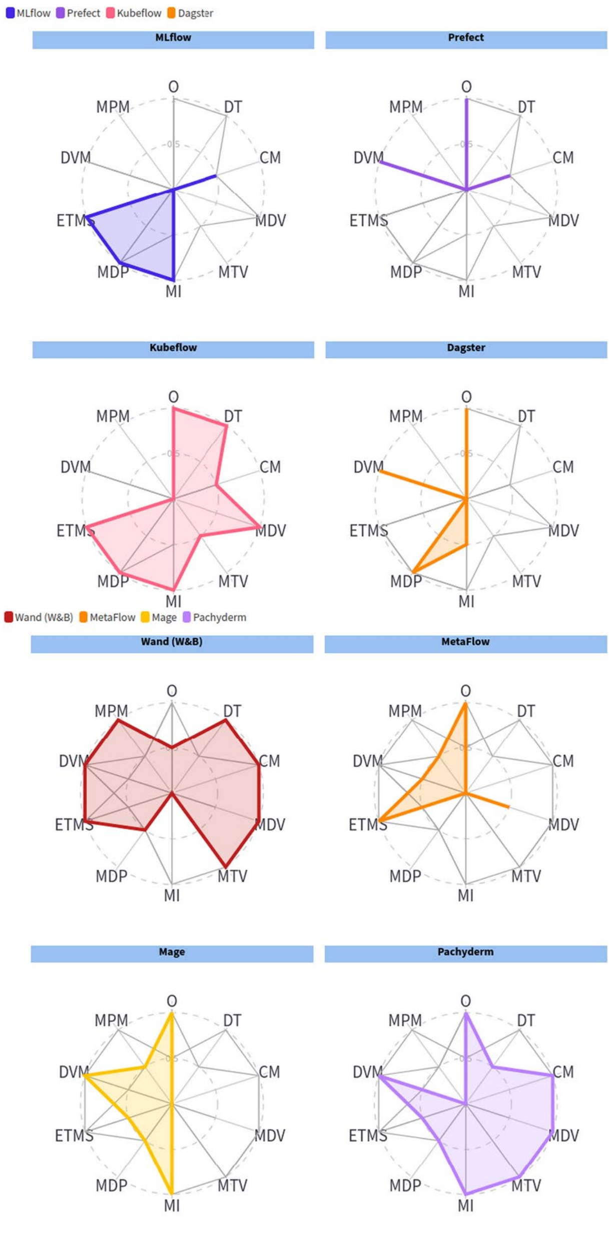
The results of the analysis of this step are visually represented in Figs. 3 and 4 that highlight the degree of support for different features in each of the 16 products using radar charts. Each figure offers a comprehensive overview of the capabilities and strengths of the respective MLOps platforms in terms of the features considered with more interactive details in Berberi (2024).

We assess the features denoted by the abbreviations on a scale within two concentric circles. The inner circle corresponds to a rating of one ✓ (0.5), indicating partial support, while the outer circle represents a rating of two ✓✓ (1.0), denoting full support. This visual representation allows a clear distinction between features with different levels of support in MLOps products.

In the following, we summarize the overall product score:

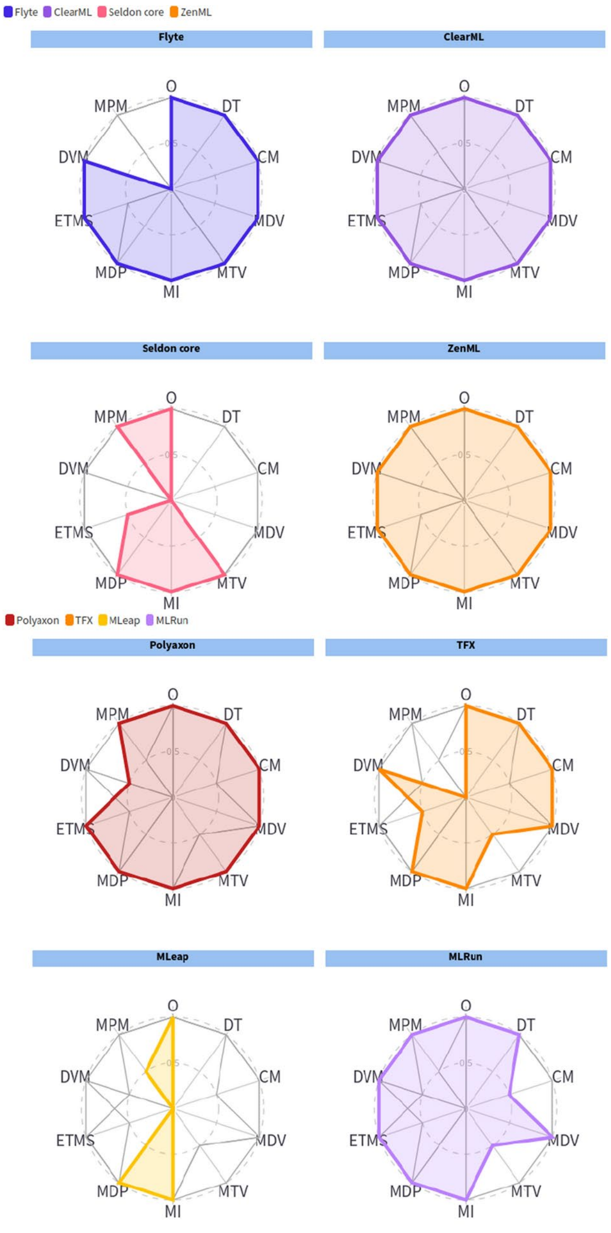
- ClearML, ZenML, Polyaxon, Flyte and MLRun support partly or fully all the features analyzed, scoring the highest degree of compliance.
- A good level of compliance (that is, > 70–80%) is achieved for ML products such as the following: W&B, TFX, Pachyderm and Kubeflow.
- However, Seldon core, and Mage record approximately a medium score (i.e., between 45 and 55%).
- Eventually, low-level compliance (that is, < 40%) is observed for MLflow, MLeap, Prefect, and Dagster products.

Fig. 3 Calculated degree of the category compliance for MLOps products (a)



Abbreviations of the Features: O [Orchestration] | DT [Distributed Training] | CM [Code Management] | MDV [Model Development] | MTV [Model Testing/Validation] | MI [Model Inference] | MDP [Model Deployment] | ETMS [Experiment Tracking and Metadata Store] | DVM [Data Versioning and Management] | MPM [Model Performance Monitoring]

Fig. 4 Calculated degree of the category compliance for MLOps products (b)



Abbreviations of the Features: O [Orchestration] | DT [Distributed Training] | CM [Code Management] | MDV [Model Development] | MTV [Model Testing/Validation] | MI [Model Inference] | MDP [Model Deployment] | ETMS [Experiment Tracking and Metadata Store] | DVM [Data Versioning and Management] | MPM [Model Performance Monitoring]

Not all the MLOps features mentioned above (O, DT, CM, MDV, MTV, MI, MDP, ETMS, DVM, MPM) in Table 3 are fully supported by all presented MLOps products. A graph with the current status of each feature support level is given in Fig. 5.

- The feature O is the most supported among all products;
- The features MPM, MTV and CM are the least supported ones;
- The remaining features are at a moderate level (>50%) supported by the selected products.

2.1.2 Step 2: GitHub stars growth assessment

In the following, we illustrate the plot of the average growth stars of each MLOps repository enabling an insightful analysis of their popularity and adoption within the community. The data has been generated using the Star-History (2022) tool which provides a history of stars accumulated over time for a specific GitHub repository (Fig. 6).

We use the linear scaling to calculate the weight of each repository after normalizing the average growth values of repositories to a range of [1, 10], where the repository with the lowest growth gets a weight of 1, and the highest growth gets a weight of 10. This method is similar to the normalization technique implemented in Scikit-learn's MinMaxScaler class (Scikit-learn 2024).

For a repository i with average growth x_i, the weight w_i can be calculated as follows (Eq. 1):

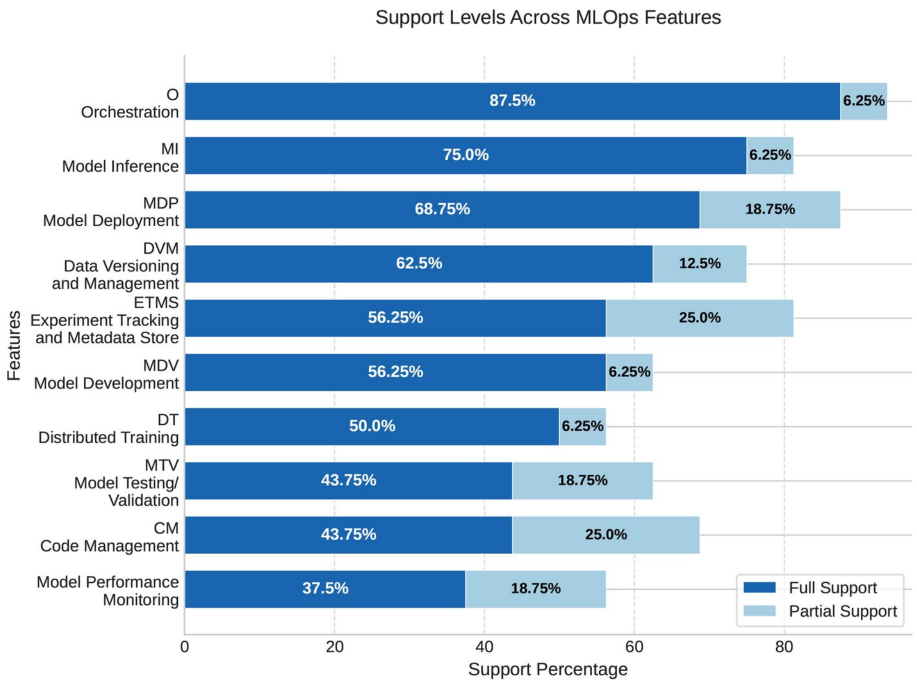


Fig. 5 Calculated percentage score of feature support levels

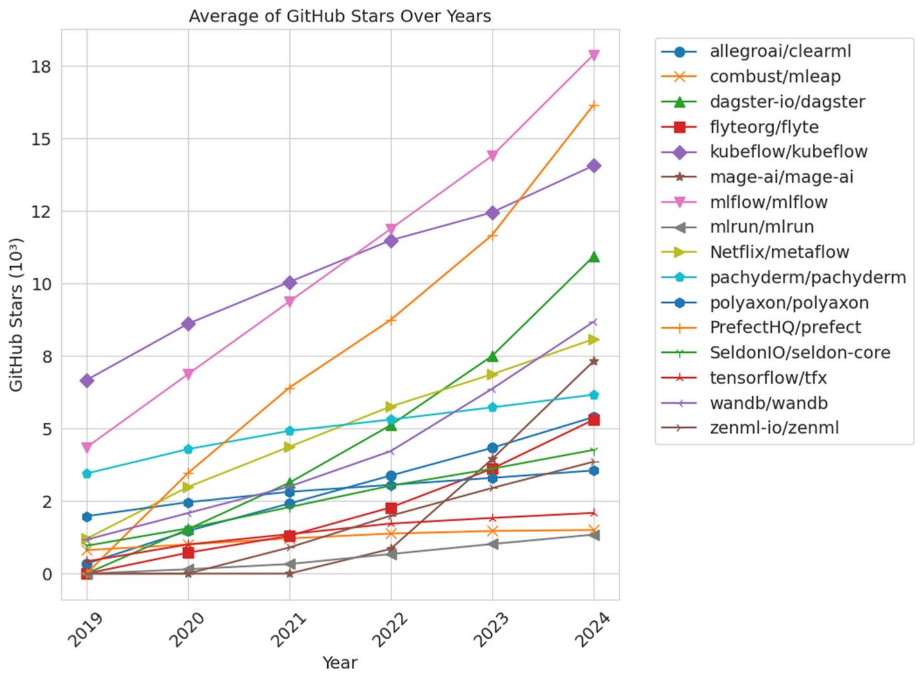


Fig. 6 Average stars growth per each MLOps repository over the last 5 years

$$w_i = 1 + \frac{(x_i - \min(x)) \cdot (10 - 1)}{\max(x) - \min(x)} \tag{1}$$

where:

- w_i is the weight assigned to repository *i*,
- x_i is the average growth of repository *i*,
- $\max(x)$ is the maximum average growth among all repositories,
- $\min(x)$ is the minimum average growth among all repositories.

We compute the weight values using the formula above, as demonstrated in the third step of our evaluation framework, with the results summarized in Table 4. The table highlights that the highest scores (based on w_i values) among the top five products are achieved by MLflow (10.00), followed closely by Kubeflow (8.89), Prefect (8.35), Dagster (6.13), and WandB (4.79), reflecting their significant growth and adoption in the field.

2.1.3 Step 3: weighted scoring and common feature identification

In this step, we use the weights from step two to calculate the weighted score for each repository based on their supported features.

To calculate the weighted score for each repository, we use the following formula (Eq. 2):

$$\text{Weighted-Score} = \sum (\text{Feature-Score} \times \text{Weight}) \tag{2}$$

Table 4 The weighted score for each product

Product	Weight (w_i)	Feature-score	Weighted-score
Kubeflow	8.89	7	**62.21**
WandB (W&B)	4.79	8	**38.30**
MLflow	10.00	3.5	**35.00**
Pachyderm	4.34	7.5	**32.55**
ClearML	2.94	10	**29.38**
Flyte	2.98	9	26.82
Polyaxon	2.65	9.5	25.22
ZenML	2.20	10	22.01
Dagster	6.13	3.5	21.44
Prefect	8.35	2.5	20.87
Mage	3.98	4.5	17.90
Metaflow	4.11	3.5	14.40
Seldon core	2.59	5.5	14.23
TFX	1.30	8	10.40
MLRun	1.00	9	9.00
MLeap	1.32	3.5	4.61

where:

- *Feature-Score* is the sum of the score for each supported feature (this can be 1 or ✓✓ for a feature that is fully supported, 0.5 or ✓ that is partially supported or 0 or empty for a feature that is not). These features correspond to the information we recorded in Table 3.
- *Weight* is the weight calculated for each repository based on the average growth numbers from step two.

After performing the calculations, the results are shown in the following table:

To determine the most common features among the **top five** products (i.e., Kubeflow, WandB (W&B), MLflow, Pachyderm and ClearML), we calculate the support values for each feature and identify those with the highest frequency.

Based on the findings from our three-step evaluation framework, a competent and robust MLOps solution must exhibit the following capabilities, which emerged as the most significant features supported by leading platforms:

2.1.3.1 Key capabilities of a competent MLOps solution (listed in descending order)

- *Experiment Tracking and Metadata Store (ETMS)* [support value $= 4.5$]

  ETMS ranks highest in support and is critical for monitoring experiments, tracking metadata, and managing iterative improvements in ML workflows. It reflects industry-wide adoption as a cornerstone for ensuring reproducibility and scalability in MLOps.

- *Model Development (MDV)* [support value $= 4.0$]

  Effective tools for developing ML models are essential for streamlining workflows and addressing the complexity of model creation, refinement, and iteration. This feature's prominence underscores its centrality in modern MLOps solutions.

- *Model Deployment (MDP)* [support value = 4.0]

  Seamless deployment capabilities allow for the transition of models from development to production, ensuring reliable and scalable operations.

- *Model Inference (MI)* [support value = 4.0]

  Efficient inference capabilities support real-time decision-making and operational scalability, making this feature integral to production pipelines.

- *Code Management (CM)* [support value = 4.0]

  Robust code management ensures version control, collaboration, and traceability of code changes, which are vital for effective MLOps.

- *Orchestration (O)* [support value = 3.5]

  Orchestrating complex workflows and managing tasks across distributed systems ensures efficiency and scalability. This feature is crucial for integrating various components of the MLOps pipeline.

- *Model Testing and Validation (MTV)* [support value = 3.5]

  Rigorous testing and validation ensure that models meet performance benchmarks and adhere to quality standards before deployment.

- *Distributed Training (DT)* [support value = 3.5]

  The ability to handle distributed training enables platforms to scale models efficiently, making this feature particularly important for resource-intensive applications.

- *Data Versioning and Management (DVM)* [support value = 3.0]

  Managing and versioning datasets is essential for maintaining data integrity, reproducibility, and compliance with regulations.

- *Model Performance Monitoring (MPM)* [support value = 2.0]

  While less common, performance monitoring provides critical insights into model behavior post-deployment, helping identify and address issues proactively.*Commercial note* Of these 16 MLOps products, Pachyderm, ClearML, Polyaxon, W&B are more or less commercial with an open source free or community version license. The limitations of their open-source free versions are in the number of users, the scale of resources, and the supported levels of services.

Other products The list of all existing MLOps platforms is flourishing, so we can talk about a rapid expansion of them. In addition to MLOps platforms, there is a long list of similar products oriented to ML life cycle management such as Microsoft *FLAML* (Wang et al. 2021), Neural Network Intelligence *NNI* (Microsoft 2021), Epistasis Lab *TPOT* (Fu et al. 2020) or Orchest (2021) (currently only beta version). The boundaries between MLOps platforms and ML life cycle management products are not strict, while, in general, MLOps products are more closely related to ensuring the ML application's effectiveness and scalability in production environments.

In the next parts, we present the content of the in-depth reviewed MLOps products.

2.2 MLOps platforms description

2.2.1 MLflow

MLflow (MLflow-Org(Databricks) 2023) is a platform to streamline ML development, including tracking experiments, packaging code into reproducible runs, and sharing and deploying models (Table 5). MLflow offers a set of lightweight APIs that can be used with any existing ML application or library (TensorFlow, PyTorch, XGBoost, etc.), wherever (e.g., in notebooks, standalone applications, or the cloud). The current components of MLflow are:

- MLflow Tracking: tracking experiments to record and compare parameters and results;
- MLflow Projects: packaging ML code in a reusable, reproducible form in order to share with other data scientists or transfer to production;
- MLflow Models: managing and deploying models from a variety of ML libraries to a variety of model serving and inference platforms;
- MLflow Model Registry: providing a central model store to collaboratively manage the full life cycle of an MLflow Model, including model versioning, stage transitions, and annotations.
- MLflow Evaluation: automated model evaluation tools integrated with experiment tracking, streamlining the process of recording model performance and providing intuitive visual comparisons across multiple models

Insights MLflow can be classified into the group of model metadata storage and management. It has the following main supported features (Table 3): Model Inference (MI) and Experiment Tracking and Metadata Store (ETMS). It is great as a basic ML life cycle platform to manage the entire ML life cycle that includes experimentation, reproducibility, deployment, and a central model registry. The tool is library-agnostic, which means it can be used with any ML library and in any programming language.

Table 5 MLflow

Motto	An open source platform for the ML life cycle
URL	https://mlflow.org/
Open-Source	https://github.com/mlflow/mlflow/ (19K GitHub stars)
Peculiarity	- MLflow Authentication is experimental and supports only basic HTTP auth

Table 6 Prefect

Motto	Prefect is a workflow orchestration tool that empowers developers to build, observe, and react to data pipelines
URL	https://www.prefect.io/opensource
Open-Source	https://github.com/PrefectHQ/prefect (17.7K GitHub stars)
Peculiarity	- Prefect Core provides only basic workflow orchestration features, whereas the Cloud version extends functionality with advanced features like Single Sign-on (SSO) for user management and Role-based Access Controls (RBAC)

Table 7 Kubeflow

Motto	The ML toolkit for Kubernetes. Kubeflow is the cloud-native platform for ML operations: pipelines, training and deployment
URL	https://kubeflow.org/
Open-Source	https://github.com/kubeflow/kubeflow/ (14.5K GitHub stars)
Peculiarity	- requires Kubernetes

2.2.2 Prefect

Prefect (2023) is the second-generation dataflow coordination and orchestration platform of the Prefect company. Prefect 2 has been designed from the ground up to handle the dynamic and scalable workloads that modern data stack demands. It is powered by Prefect Orion, a brand new asynchronous rule engine (Table 6).

Insights Prefect has two major concepts, Workflow and Task as the next data orchestration tool for Python. These concepts are quite similar to the nodes inside a Directed Acyclic Graph (DAG) and to Airflow DAGs. However, Prefect claims that it is much better than Airflow (Prefect-Docs 2022) thanks to the result of years of experience working on Airflow and related projects with a lightweight and user-friendly API backed by a powerful set of abstractions that fit most data-related use cases. Prefect treats workflows as standalone objects that can be run at any time, with any concurrency, for any reason. A schedule is a predefined set of start times with a flow parameter to define the workflow time dependency.

2.2.3 Kubeflow

The Kubeflow project (2023) is dedicated to making deployments of ML workflows on Kubernetes simple, portable and scalable. Its goal is not to recreate other services, but to provide a straightforward way to deploy best-of-breed open-source systems for ML tailored specifically for Kubernetes-based infrastructures (Table 7).

Insights Kubeflow has the following supported features (Table 3): Orchestration (O), Distributed Training (DT), Model Development (MDV), Model Inference, Model Deployment, and Experiment Tracking and Metadata Store (ETMS). Kubeflow can also be classified into the group of orchestration and workflow pipelines MLOps products. It is the ML toolkit for Kubernetes to maintain ML systems by packaging and managing Docker containers. The Kubeflow platform consists of different components that can be used either as standalone tools or to deploy the full Kubeflow suite for an end-to-end ML life cycle solution.

Table 8 Dagster

Motto	An orchestration platform for the development, production, and observation of data assets
URL	https://dagster.io/
Open-Source	https://github.com/dagster-io/dagster (12K GitHub stars)
Peculiarity	- *dagster dev* does not include authentication or web security

Table 9 W&B (WandB)

Motto	Building the best tools for ML practitioners
URL	https://wandb.ai/site
Open-Source	https://github.com/wandb/ (9.2K GitHub stars)
Peculiarity	- Production-grade features for W&B Server are available for enterprise-tier only (paid). Free teams are available for academics

2.2.4 Dagster

Dagster (DagsterLabs 2018) is an orchestrator that is designed to develop and maintain data assets, such as tables, datasets, ML models, and reports. It is built to be used at every stage of the data development life cycle: local development, unit tests, integration tests, staging environments, all the way up to production. Dagster can be used as the orchestration engine for ML pipelines (feature pipelines, training pipelines, and batch inference pipelines) (Table 8).

Insights Dagster and Prefect (both launched in 2018) are often compared with each other in the context of Airflow next generation (Navid 2022). Prefect adheres to a philosophy of negative engineering, built on the assumption that the user knows how to code, and makes it as simple as possible to take that code and build it into a distributed pipeline, backed by its scheduling and orchestration engine. Dagster takes a first-principles approach to data engineering. It is built with the full development life cycle in mind, from development to deployment, monitoring and observability. Dagster offers an expanding library of integrations with today's leading data tools, allowing seamless connection to existing tools and deployment within an infrastructure.

2.2.5 Weights & biases

Weights & Biases, W&B, or WandB (WeightsAndBiases 2023) is a comprehensive tool designed for tracking ML experiments and visualization of the model. It offers a variety of features to enhance the development and management of ML projects, including experiment logging, visualization of model metrics, and collaboration tools for teams (Table 9).

Insights W&B fully manages Experiment Tracking, capturing key parameters and metrics for a centralized view of ML development. It enhances collaboration through seamless sharing, integrates with popular frameworks, optimizes hyperparameters, prioritizes reproducibility, and efficiently automates and scales ML workflows. W&B offers different deployment options to accommodate various needs, including the possibility of setting up a server on your on-premise infrastructure. However, when it comes to the availability of features, it is essential to consider the terms of service and licensing agreements. To set up a W&B Server in a production environment, there are the following three ways:

- *Production Cloud* Set up a production deployment on a private cloud in just a few steps using Terraform scripts provided by W&B.

- *Dedicated Cloud* A managed, dedicated deployment on W&B's single-tenant infrastructure in your choice of cloud region.
- *On-Prem/Bare Metal* W&B supports setting up a production server on most bare metal servers in on-premise data centres by quickly starting by running the WandB server to easily start hosting W&B on a local infrastructure.

2.2.6 Metaflow

Metaflow (Netflix 2023) is a user-friendly Python/R library that helps scientists and engineers to build and manage real-life data science projects. Metaflow was originally developed at Netflix to boost the productivity of data scientists who work on a wide variety of projects, from classical statistics to state-of-the-art DL. It is a Python library that provides a framework to structure Python code as a DAGs describing a number of steps of work (e.g., data loading, model training, and evaluation) that will be executed, as well as dependencies between them. These steps can be run locally or distributed using AWS Batch (Table 10).

Insights Netflix released as open-source Metaflow in 2019. Metaflow helps design a workflow, run at scale, and deploy it to production. It versions and tracks experiments and data automatically with the ability to easily inspect results in notebooks. Today, Metaflow powers thousands of ML and data science applications in companies such as Netflix, CNN, SAP, 23andMe, realtor.com, REA, Coveo, and Latana. Commercial support and extended services for Metaflow are provided by Outerbounds, a company founded by the original Metaflow creators. Support running flows in notebooks and through Python scripts The release 2.12 introduces an new API that makes it simple to execute flows inside Notebooks, or as part of Python code.

2.2.7 Mage AI

Mage AI (2023) is a versatile platform that facilitates the seamless integration and synchronization of data from various third-party sources. Immediately see results from code output with an interactive notebook UI. Data is a first-class citizen: Each block of code in the pipeline produces data that can be versioned, partitioned, and cataloged for future use. ML pipelines can be built, and they consist of modular code blocks that are reproducible in any environment (Table 11).

Insights The main features supported by Mage are: *Orchestration*: Schedule and manage data pipelines with observability; *Notebook*: Interactive Python, SQL, & R editor for coding data pipelines; *Data integrations*: Synchronize data from third-party sources to your internal destinations; *Streaming pipelines*: Ingest and transform real-time data and data build tool (DBT).

Table 10 Metaflow

Motto	A framework for real-life data science
URL	https://metaflow.org/
Open-Source	https://github.com/Netflix/metaflow (8.3K GitHub stars)
Peculiarity	- Metaflow Sandbox allows you to experiment with modern ML and the infrastructure behind it, without having to install anything locally

Table 11 Mage AI

Motto	The modern replacement for Airflow. Build, run and manage data pipelines for integrating and transforming data
URL	https://mage.ai/
Open-Source	https://github.com/mage-ai/mage-ai (8K GitHub stars)
Peculiarity	- can be deployed on AWS, GCP, Azure, or Digital Ocean infrastructures - utilize the tool directly within mage.ai

Table 12 Pachyderm

Motto	Data-Centric Pipelines and Data Versioning
URL	https://www.pachyderm.com/
Open-Source	https://github.com/pachyderm/pachyderm (6.2K GitHub stars)
Peculiarity	- requires Kubernetes

2.2.8 Pachyderm

Pachyderm (2023) is cost-effective on the scale, enabling data engineering teams to automate complex pipelines with sophisticated data transformations across any type of data. It provides parallelized processing of multi-stage, language-agnostic pipelines with data versioning and data lineage tracking (CI/CD engine for data). Its main features are: (1) Data-driven pipelines automatically trigger based on detecting data changes. (2) Immutable data lineage with data versioning of any data type. (3) Autoscaling and parallel processing built on Kubernetes for resource orchestration. (4) Uses standard object stores for data storage with automatic deduplication. (5) Runs across all major cloud providers and on-premises installations (Table 12).

Insights Pachyderm combines the data lineage with end-to-end pipelines on Kubernetes. It is available in three versions, Community Edition (open source), Enterprise Edition, and Hub Edition (still a beta version). Pachyderm has moved some components of the Pachyderm Platform to a source-available limited license. Pachyderm serves, above all, for data processing and orchestration. It can be classified into data and pipeline versioning MLOps product group with data versioning principles:

- Repository—a Pachyderm repository is the highest level data object. Typically, each dataset in Pachyderm is its own repository.
- Commit—an immutable snapshot of a repository at a particular point in time.
- Branch—an alias to a specific commit, or a pointer, that automatically moves as new data is submitted.
- File—files and directories are actual data in your repository. Pachyderm supports any type, size, and number of files.
- Provenance—to track the dependencies and relationships among datasets.

2.2.9 Flyte

Flyte (2023) is an open-source Kubernetes-native workflow automation platform for complex mission-critical data and ML processes at scale. It has been tested on Lyft, Spotify, Freenome, and others. Flyte's main features are as follows: Kubernetes-native workflow automation platform Ergonomic SDKs in Python, Java, and Scala to develop Flyte workflows versioned, audit-

Table 13 Flyte

Motto	Kubernetes-native workflow automation platform for complex, mission-critical data and ML processes at scale
URL	https://flyte.org/
Open-Source	https://github.com/flyteorg/flyte (5.8K GitHub stars)
Peculiarity	- requires Kubernetes

Table 14 ClearML

Motto	ClearML - Auto-Magical CI/CD to streamline ML workflow. Experiment Manager, MLOps and Data-Management
URL	https://clear.ml/
Open-Source	https://github.com/allegroai/clearml/ (5.7K GitHub stars)
Peculiarity	The hosted version is free for a limited number of users, a self-hosted system is 100% free

able, and reproducible pipelines that are data-aware and strongly typed resource-aware and deployments at scale (Table 13).

Insights Flyte is community-driven and community-owned software. Flyte is more than a workflow engine; it uses workflow as a core concept, and task (a single unit of execution) as a top-level concept. Multiple tasks arranged in a data producer-consumer order create a workflow. Flyte is a platform for ML project maintenance released by Lyft with Large-scale project support; Improved reproducibility; Multi-language support such as Python, R, Java, Scala and Julia. Flyte has been tested out by Lyft internally before they released it to the public. Flyte can also be classified into the group of MLOps testing and maintenance products.

2.2.10 ClearML

ClearML (ClearML(AllegroAI) 2023) declares to turn an ML experiment into MLOps with only two lines of code, and it can easily develop, orchestrate, and automate ML workflows at scale. The hosted ClearML version comes with limited number of free users or different price plans. ClearML can be self-hosted, offering a 100% free solution with limits dictated by your own system's storage capacity (Table 14).

Insights ClearML is a ML/DL development and production suite containing 4 main modules:

- Experiment Manager—experiment tracking, environments and results;
- MLOps—Orchestration, Automation & Pipelines solution for ML/DL jobs (K8s/Cloud/bare-metal);
- Data-Management—Fully differentiable data management & version control solution on top of object-storage like Amazon Simple Storage Service (S3), Google Storage (GS), Azure storage or Network-Attached Storage (NAS);
- Model-Serving—deploy new model endpoints, includes optimized GPU serving support backed by NVIDIA-Triton, with out-of-the-box Model Monitoring.

ClearML enables many MLOps features (Table 3), for example, to track and upload metrics and models, reproduce experiments, create bots that send Slack messages based on experiment behavior, manage data (store, track, and version control), remotely execute experiments on any compute resource available with ClearML Agent, automatically scale cloud instances according to resource needs with ClearML's GPU Compute, AWS Autoscaler, and GCP Autoscaler GUI applications, run hyperparameter optimization and build pipelines from code.

2.2.11 Seldon core

Seldon Core (2023) declares itself as the standard open-source platform to rapidly deploy ML models on Kubernetes on a massive scale. It converts ML models (TensorFlow, PyTorch, H2O, etc.) or language wrappers (Python, Java, etc.) into production REST/gRPC (Google Remote Procedure Call) microservices. Seldon handles scaling to thousands of production ML models and provides advanced ML capabilities out of the box including Advanced Metrics, Request Logging, Explainers, Outlier Detectors, A/B Tests, Canaries and more (Table 15).

Insights Seldon Core is the open-source framework for easily and quickly deploying models and experiments at scale with the following summary:

- Simplify model deployment with various options like canary deployment;
- Monitor models in production with the alerting system;
- Use model explainers to understand why certain predictions were made.

Seldon Core is part of the commercial Seldon platform https://www.seldon.io/. Seldon also provides other MLOPs features, for example, drift detection by the Alibi Detect open-source library (Van Looveren et al. 2019).

2.2.12 ZenML

ZenML (2023) is an extensible open-source MLOps framework for creating portable, production-ready MLOps pipelines. It is built for Data Scientists, ML Engineers, and MLOps Developers to collaborate as they develop to production. ZenML offers a simple and flexible syntax, is cloud- and tool-agnostic, and has interfaces/abstractions catered toward ML workflows. The aim is to have all selected tools in one place to tailor a workflow that caters to specific needs (Table 16).

Insights In ZenML, a *stack* represents a set of configurations for the MLOps tools and infrastructure selected by users. For instance:

- Kubeflow for ML workflow orchestration;,
- Amazon S3 bucket as a storage to save ML artifacts;
- Weights & Biases for experiment tracking;
- Seldon or KServe for model deployment on Kubernetes.

Table 15 Seldon Core

Motto	An open-source platform to deploy ML models on Kubernetes at a massive scale
URL	https://www.seldon.io/solutions/open-source-projects/core
Open-Source	https://github.com/SeldonIO/seldon-core (4.4K GitHub stars)
Peculiarity	- requires Kubernetes - it is part of the Seldon commercial platform

Table 16 ZenML

Motto	Build portable, production-ready MLOps pipelines
URL	https://docs.zenml.io
Open-Source	https://github.com/zenml-io/zenml (4.2K GitHub stars)
Peculiarity	supports other MLOps tools for implementing MLOps *capabilities* (e.g. Kubeflow for orchestration, MLFlow for the model registry, etc)

Table 17 Polyaxon

Motto	A platform for reproducible and scalable ML and DL
URL	https://polyaxon.com/
Open-Source	https://github.com/polyaxon/polyaxon (3.6K GitHub stars)
Peculiarity	- requires Kubernetes

Apart from the infrastructure required to run ZenML itself, ZenML also boasts a ton of integrations into popular MLOps tools. The ZenML Stack concept ensures that these tools work nicely together, therefore bringing structure and standardization into the MLOps workflow. However, ZenML assumes that the stack infrastructure for these tools is already provisioned. Currently, ZenML hasn't developed a native large-scale workload distribution mechanism, but they've improved remote execution capabilities and expanded cloud integration options.

2.2.13 Polyaxon

Polyaxon (2023) is a platform for building, training, and monitoring large-scale DL applications with the aim of solving the reproducibility, automation, and scalability of ML applications. Polyaxon is deployed to any data center or cloud provider. It supports all major DL frameworks such as TensorFlow, MXNet, Caffe, Scikit-learn, and Torch. Polyaxon makes it faster, easier, and more efficient to develop DL applications by managing workloads with smart containers and node management (also with GPU) (Table 17).

Insights Polyaxon can also be classified into the group of run orchestration and workflow pipelines MLOps products. The tool can be deployed into any data center or cloud provider and can be hosted and managed by Polyaxon. When it comes to orchestration, Polyaxon can maximize the use of the cluster by scheduling jobs and experiments through its command-line interface (CLI), dashboard, software development kits (SDKs), or REST application programming interface (API). Polyaxon is, in general, a commercial product, but it has an open-source version. It comes in three versions: Community Edition (open source free tool), Hybrid Cloud and Enterprise Edition. It is a well-documented platform, with technical reference docs, getting started guides, learning resources, guides, tutorials, change-logs, etc.

2.2.14 TensorFlow Extended

TensorFlow Extended or TFX (Tensorflow-Extended(Google) 2023) is a Google production-scale ML platform based on TensorFlow. It provides a configuration framework to express ML pipelines consisting of TFX components. TFX pipelines can be orchestrated using Apache Airflow and Kubeflow pipelines. TFX components interact with a ML Metadata backend that keeps a record of component runs, input and output artifacts, and runtime con-

Table 18 TensorFlow Extended

Motto	TensorFlow Extended (TFX) is an end-to-end platform to deploy production ML pipelines
URL	https://www.tensorflow.org/tfx
Open-Source	https://github.com/tensorflow/tfx (2.1K GitHub stars)
Peculiarity	- requires Tensorflow
	- TFX pipelines can be orchestrated with Airflow and Kubeflow

Table 19 MLeap

Motto	Deploy ML Pipelines to Production
URL	https://combust.github.io/mleap-docs/
Open-Source	https://github.com/combust/mleap (1.5K GitHub stars)
Peculiarity	- no support for PyTorch

figuration. This metadata back-end enables advanced functionality like experiment tracking or warm-starting/resuming ML models from previous runs. TFX libraries include (1) TensorFlow Data Validation, (2) TensorFlow Transform, (3) TensorFlow Model Analysis, (4) TensorFlow Metadata, (5) ML Metadata. TFX requires Apache Beam, an open-source unified model for defining both batch and streaming data-parallel processing pipelines to implement data-parallel pipelines executed on, e.g., Apache Flink, Apache Spark, Google Cloud Dataflow, and others (Table 18).

Insights The current version of TFX 1.16.0 provides stable public APIs and artifacts along with nightly built packages.The popularity of TFX is still not at the top, but is growing thanks to the popularity of the TensorFlow family.

2.2.15 MLeap

MLeap (MLeap-Docs(CombustML) 2023) aims to deploy ML data pipelines and algorithms to avoid being a time-consuming or difficult task. MLeap allows data scientists and engineers to deploy ML pipelines from Spark and Scikit-learn to a portable format and execution engine. Its main goals are as follows. Allow building data pipelines and train algorithms with Spark and Scikit-Learn Extend Spark/Scikit/TensorFlow by providing ML pipelines serialization/deserialization to/from a common framework (Bundle.ML) Use MLeap Runtime to execute pipeline and algorithm without dependencies on Spark or Scikit (numpy, pandas, etc.) (Table 19).

Insights MLeap is a common serialization format and execution engine for ML pipelines. It supports Spark, Scikit-learn, and TensorFlow to train pipelines and export them to an MLeap bundle. For portability, MLeap is built on the JVM and uses only serialization formats that are widely adopted.

2.2.16 MLRun

MLRun (MLRun(Iguazio) 2023) is an open MLOps platform to quickly build and manage continuous ML applications throughout their life cycle. MLRun integrates into a development and CI/CD environment and automates the delivery of production data, ML pipelines, and online applications. It aims to reduce engineering efforts, time to production, and computational resources and break the silos between data, ML, software, and DevOps/MLOps teams and to enable collaboration and fast continuous improvements (Table 20).

Insights In MLRun the assets, metadata, and services (data, functions, jobs, artifacts, models, secrets, etc.) are organized into projects. Projects can be imported/exported as a whole, mapped to git repositories or IDE projects (in PyCharm, VSCode, etc.), which enables versioning, collaboration, and CI/CD. Project access can be restricted to a set of users and roles. MLRun supports the following MLOps tasks: (1) Project management and CI/CD automation, (2) Ingest and process data, (3) Develop and train models, (4) Deploy models and applications, (5) Monitor and alert. MLRun ecosystem supports the latest data stores, development tools, services, and platforms such as:

- Data stores: object (S3, GS, Azure Blob), files, NFS, Pandas/Spark DF, BigQuery, Snowflake, Redis, Iguazio V3IO object/key-value;
- Event sources: HTTP, cron, Kafka, Iguazio V3IO streams;
- Execution frameworks: Nuclio, Spark, Dask, Horovod/MPI, K8s Jobs;
- Dev environments: PyCharm, VSCode, Jupyter, Colab, AzureML, SageMaker, Codespaces;
- ML frameworks: scikit-learn, XGBoost, LGBM, TensorFlow/Keras, PyTorch;
- Platforms: Kubernetes, AWS EKS, Azure AKS, GKE, VMWar, Local (Kubernetes on Docker Desktop), Docker, Linux/KVM, NVIDIA DGX;
- CI/CD: Jenkins, Github Actions, Gitlab CI/CD, KFP.

However, from our point of view, the number of current GitHub stars for this software is still not high despite promising its MLOps options and features.

2.3 Exploring drift detection libraries across MLOps tools

In our research, we recognize a critical need to explore the integration of drift monitoring as a key component of MLOps workflows: automation of CT can only be established if there is a proper model monitoring and triggering of CT. Modern ML systems face increasingly complex challenges in maintaining long-term reliability and operational success.

In our comprehensive evaluation of MLOps tools, as presented in Table 3, we initially compare and analyze various platforms against different features, specifically the MPM which encompasses critical components such as drift detection and other technical performance metrics including prediction latency, throughput, model accuracy, error rates, inference time, and computational resource utilization (CPU/GPU) which lie outside our scope of study.

Building on the three-step analysis approach, we focus on the least supported feature-Model Performance Monitoring (MPM)-and its critical role in the MLOps pipeline. Our analysis uncovers distinct approaches to drift detection integration. For example, based

Table 20 MLRun

Motto	The Open Source MLOps Orchestration Framework
URL	https://www.mlrun.org/
Open-Source	https://github.com/mlrun/mlrun (1.5K GitHub stars)
Peculiarity	- recommended self-hosted installation requires Kubernetes with elastic scaling

on the MLOps tools we evaluated, we observe the following libraries being integrated or implemented:

- *MLRun* (MLRun(Iguazio) 2023) integrates Evidently (EvidentlyAI 2021) for comprehensive monitoring while offering additional flexibility through custom monitoring applications and Grafana (now part of New Relic's offerings) visualization (Relic 2014), beside the implementation of basic drift algorithms that serves as the starting point for developers to build and extend monitoring capabilities tailored to their specific needs.
- *ClearML* (ClearML(AllegroAI) 2023) uses scheduled jobs to analyse logged data for drift using external tools, triggering retraining or other workflows if drift is detected.
- *ZenML* (ZenML 2023) and *Seldon Core* (SeldonCore 2023) implement Alibi-detect (Van Looveren et al. 2019) as their primary drift detection library, providing standardized methods for detecting data and concept drift.
- *W &B* (WeightsAndBiases 2023) allow users to track data distributions and model performance over time to identify changes using custom scripts.
- The *Polyaxon* (Polyaxon 2023) ecosystem integrates a library called TraceML (TraceML 2024), which provides multiple utilities for logging, tracking, and event processing in ML experiments, including drift detection.
- *Metaflow* (Netflix 2023), *MLeap* (MLeap-Docs(CombustML) 2023) and *Mage* (Mage-AI 2023) do not offer built-in drift detection functionality, which is why they are only partially marked in our table as supporting this feature. Instead, both tools provide flexibility for integrating external drift detection solutions or implementing custom drift monitoring within their workflows.

By examining various MLOps platforms, we identified diverse approaches to drift detection, ranging from integrated libraries like EvidentlyAI and Alibi-detect to custom monitoring solutions, highlighting the evolving landscape of model performance management.

Therefore, investigating further current drift detection frameworks remains crucial for maintaining ML models efficiently in production. As the complexity of AI systems continues to grow, a comprehensive understanding and continuous exploration of drift detection methodologies will be essential for organizations seeking to ensure the sustained performance and reliability of their ML deployments.

2.4 MLOps landscape key finding

The MLOps landscape features an extensive array of products that support various ML life cycle tasks, with over 250 listed software products (RocketScience 2022). There are also approaches to classify MLOps products, e.g., into subgroups with narrowed purposes as: (a) Model metadata storage and management, (b) Data and pipeline versioning, (c) Hyperparameter tuning, (d) Run orchestration and workflow pipelines, (e) Model deployment and serving, (f) Production model monitoring.

The boundaries between the subgroups of the MLOps product are not strict. One MLOps product can belong to more groups or get lost/evolved with time going and feature requirements evolving.

When evaluating MLOps products, popularity-often measured by GitHub stars-is commonly considered. However, it is important to recognize that popularity alone can be misleading. Newer products, even those with advanced features, may have fewer stars simply because they have been on the market for a shorter time. Furthermore, as platforms continue to evolve, they expand their feature sets and functionalities, sometimes leading to shifts in their GitHub growth patterns.

In this study, we address the issue of relying solely on popularity, which is typically measured by GitHub stars, by also considering the growth trends of repositories over time. Specifically, growth trends reflect the platform's development activity, such as the rate of new feature implementations and the overall community engagement, rather than just the current number of stars. By analyzing these growth trends, we can gain insights into how platforms are evolving in terms of functionality and user adoption. This approach helps us capture a more comprehensive view of a platform's capabilities and potential, ensuring that newer and rapidly developing platforms are appropriately evaluated despite having fewer stars compared to more established tools.

Many use cases do not require a wide range of features. If required features are missing, it is often sufficient to add supplementary tools. In this review, we focus on a subset of MLOps solutions, specifically 16 open-source MLOps platforms. Despite the apparent common goal of supporting the ML life cycle, these platforms exhibit distinct objectives, allowing for diverse approaches.

Based on the findings from our three-step evaluation framework, a competent and robust MLOps solution must demonstrate the following capabilities, which emerged as the most significant features supported by leading platforms: *Experiment Tracking and Metadata Store (ETMS)*, *Model Development (MDV)*, *Orchestration (O)*, *Distributed Training (DT)*, *Code Management (CM)*, *Model Testing and Validation (MTV)*, *Model Deployment (MDP)*, *Data Versioning and Management (DVM)* and *Model Inference (MI)*.

Furthermore, based on our analysis results, we derive the following flowchart as shown in Fig. 7 to transform the complex tabular comparison of MLOps platforms into a more intuitive and user-friendly decision-making guide.

While the original table presented raw data on various MLOps tools across multiple dimensions, this flowchart simplifies the selection process by breaking down key decision points and highlighting the most suitable tools for different organisational needs.

It is important to note that the tools listed in each category represent a demonstration example and are not exhaustive. It translates the technical needs and compliance scores into a clear, navigable path that helps teams quickly identify the most appropriate MLOps platform based on their specific requirements, scale, and functional priorities. The visualisation serves as a bridge between the detailed technical analysis and practical tool selection, making the complex landscape of MLOps tools more accessible and comprehensible for data science and ML teams.

As indicated in this flowchart:

- **Comprehensive MLOps platforms provide full capabilities coverage** Tools like ClearML, ZenML, and Polyaxon are ideal for organizations requiring end-to-end MLOps capabilities, as they support nearly all essential features, making them suitable for enterprises with diverse ML life cycle demands.
- **Specialized tool recommendations depend on organizational focus** For startups or lighter setups, tools such as MLRun and Flyte are better suited due to their targeted feature coverage ($\approx 80\%$).

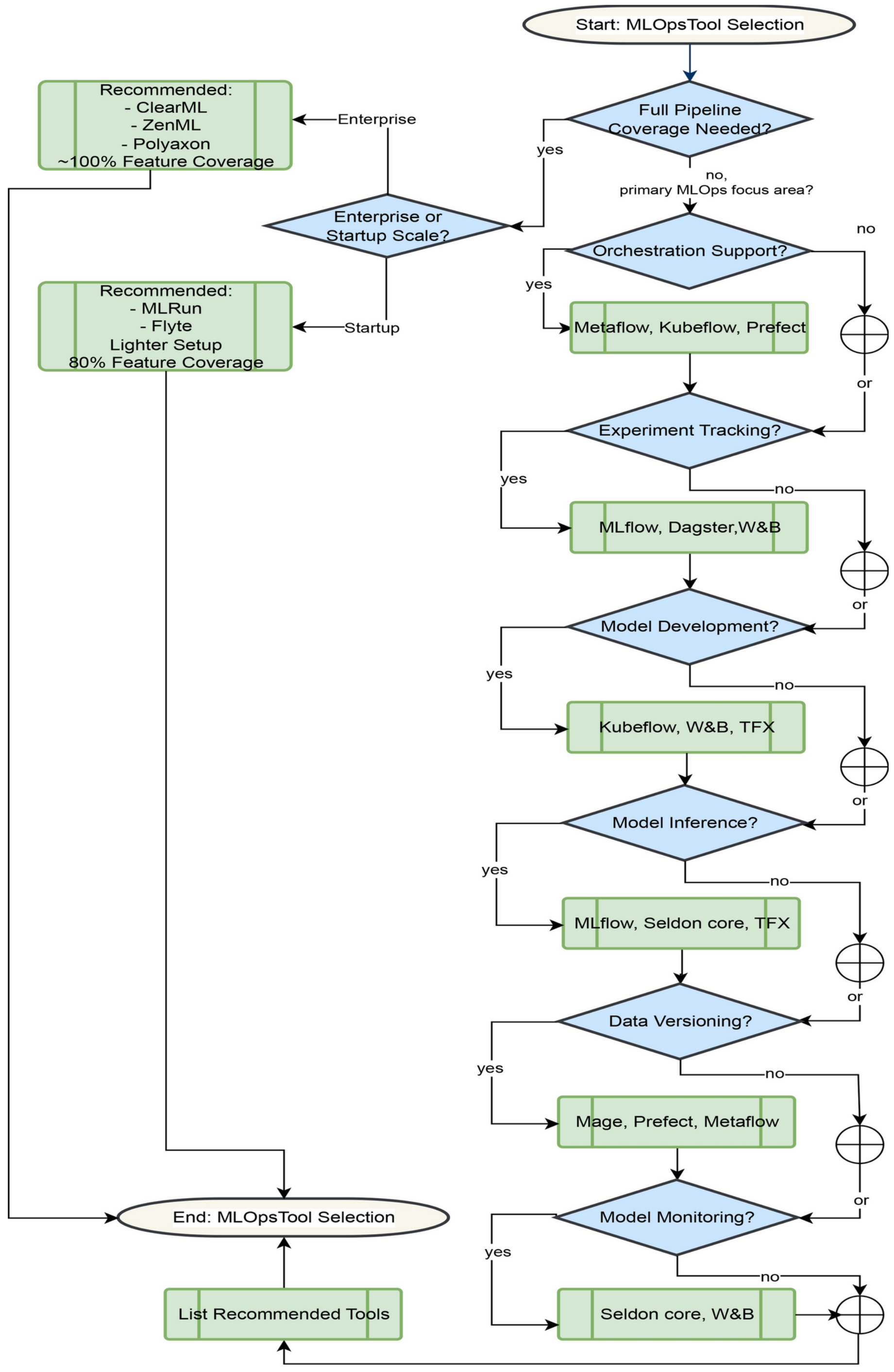


Fig. 7 MLOps tool selection flow diagram

- **Partial MLOps capability needs drive tool selection** When organizations require only specific MLOps features (e.g., orchestration or experiment tracking), tools like Prefect (for orchestration) or MLflow (for experiment tracking) provide tailored solutions without the complexity of full-stack platforms.

MLOps tools are extensively applied across diverse industries, with their implementations tailored to address specific operational needs (John et al. 2023). In healthcare, for instance, a company utilizes MLOps solutions to classify pathology images, enabling automated cancer detection and improving diagnostic efficiency. Similarly, in the automotive manufacturing sector, a speech-to-text system powered by MLOps is deployed to tag vehicle maintenance issues in workshops, streamlining operational workflows and reducing downtime. In banking, MLOps is leveraged to support sustainable initiatives by analyzing Environmental, Social, and Governance (ESG) factors, promoting transparency and enhancing client engagement.

The selective adoption of MLOps tools aligns closely with our AI4EOSC project's approach (AI4EOSC 2024), where specific use cases in environmental and agricultural domains require particular MLOps capabilities. In the automated thermography use case, their requirements emphasize the need for accurate experiment tracking and model monitoring features. The agrometeorology application requires distributing training, specifically federated learning approach to enhance model performance by utilizing distributed data images captured by weather radars, demonstrating slight but meaningful improvements in predictive accuracy across decentralized datasets (Sáinz-Pardo Díaz et al. 2024). While the integrated plant protection use case needs model deployment features that enable greater terrain coverage and spatial precision in disease detection models.

With MLOps tools carefully selected to fit the unique computational and monitoring demands of each use case, we can develop more efficient and adaptive ML workflows in environmental and agricultural research.

3 Conclusion and future work

While the MLOps system is going through a radical transformation, an increasing number of products are being launched across the ML life cycle in various dimensions.

The wide range of MLOps products presents us with the opportunity and dilemma of choosing the ideal software to be used in certain demands.

To address the dynamic landscape of MLOps we conducted a comprehensive survey of 16 MLOps platforms. This review provided valuable insights into the current state of ML platform capabilities, with a special focus on model performance monitoring as the least supported feature among them.

Although the study offers a comprehensive assessment of 16 free MLOps tools that gives an understanding of some specific capabilities and popularity of them, recognizing some *limitations* is also significant. Primarily, our study is restricted to the free open-source MLOps tools. This is so because it can be considered a limitation and also a strength: in order to support open-source tool development and use is what we want to achieve. Nevertheless, due to the fact that our findings cover only open source tools, enterprises that need proprietary solutions could have to do more research to get insight about the products beyond the suggestions in this review.

The research brought out the multi-faceted nature of MLOps platforms, illustrating their different approaches toward the key problems of ML life cycle management. Our three-step framework goes with feature analysis, GitHub stars growth analysis, and weighted scoring on these features.

As a result of our findings, our comprehensive analysis of the top five MLOps products (Kubeflow, WandB, MLflow, Pachyderm, and ClearML) revealed a set of essential capabilities that define a robust and competent MLOps solution.

The practical implications of our research are substantial. Our developed flowchart and evaluation framework provide organizations with a pragmatic guide to selecting MLOps platforms. By understanding the nuanced capabilities of different tools, companies can make more informed decisions about their ML infrastructure. The research contributes to bridging the knowledge gap in MLOps platform evaluation, offering a structured approach to understanding and selecting tools that can effectively manage the complexities of modern ML operations.

In summary, this research significantly contributes to demystifying the MLOps platform landscape. Our findings establish a basis for further investigation in this area and provide both academic and industry practitioners with a comprehensive framework for evaluation and selection of MLOps solutions.

Future work may involve a more tangible connection with end users and companies that have already adopted MLOps platforms. These goals may be achieved through receiving user feedback (via surveys), exploring the real-life implementation of the solution, and including user-based perspective, which can lead to more practical MLOps platform evaluations. Furthermore, investigating more in depth current drift detection frameworks remains crucial for maintaining ML models efficiently in production.

Another direction for future research is focusing on ethical considerations within MLOps platforms as ethical AI becomes vital. This includes evaluating how platforms are facilitating safe AI usage, data privacy, and transparency in model development and deployment.

Thus, the creation of standardized benchmarks and performance metrics for MLOps platforms will make the comparison fair more objectively. In the future, such work should be focused on identifying benchmarks that allow companies to evaluate and select the platforms based on clear criteria and standards in the industry.

Acknowledgements This work is funded by the European Union through the Horizon Europe AI4EOSC project under Grant Number 101058593.

Author contributions L.B.: formal analysis, methodology, investigation, project administration, validation, visualization, writing—original draft, writing—review and editing. V.K.: formal analysis, investigation, project administration, validation, writing—original draft, writing—review and editing. G.N.: conceptualization, formal analysis, investigation, methodology, writing—original draft, writing—review and editing. J.S.P.D.: formal analysis, investigation, methodology, validation, writing—original draft, writing—review and editing. A.C.: formal analysis, validation, writing—review and editing. V.T.: investigation, validation, writing—review and editing. G.M.: methodology, investigation, supervision, validation, writing—original draft, writing—review and editing. A.L.G.: conceptualization, investigation, supervision, validation, writing—review and editing. All authors read and approved the manuscript.

Funding Open Access funding enabled and organized by Projekt DEAL.
This work is funded by the European Union through the Horizon Europe AI4EOSC project under Grant Number 101058593.

Data availability No datasets were generated or analysed during the current study.

Declarations

Competing interests The authors declare no competing interests.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

AI4EOSC (2024) AI4EOSC use cases. https://ai4eosc.eu/use-cases/. Accessed 2 Dec 2024
AI-Infrastructure (2023) AI infrastructure landscape. https://ai-infrastructure.org/ai-infrastructure-landscape/. Accessed 30 Nov 2023
Amrit C, Narayanappa AK (2024) An analysis of the challenges in the adoption of MLOps. J Innov Knowl 10(1):100637. https://doi.org/10.1016/j.jik.2024.100637
Azure M (2022) Distributed training with Azure Machine Learning. https://learn.microsoft.com/en-us/azure/machine-learning/concept-distributed-training?view=azureml-api-2. Accessed 07 Dec 2022
Barrak A, Petrillo F, Jaafar F (2022) Serverless on machine learning: a systematic mapping study. IEEE Access 10:99337–99352. https://doi.org/10.1109/ACCESS.2022.3206366
Berberi L (2024) Interactive radar charts on analysis results of MLOps products. https://public.flourish.studio/visualisation/16000803/. Published 11 Dec 2024
ClearML(AllegroAI) (2023) Auto-magical CI/CD to streamline your ML workflow. experiment manager, MLOps and data-management. https://github.com/allegroai/clearml/. Accessed 10 Dec 2024
DagsterLabs (2018) An orchestration platform for the development, production, and observation of data assets. https://github.com/dagster-io/dagster. Accessed 10 Dec 2024
DataCamp (2023) 17 top MLOps tools you need to know. https://www.datacamp.com/blog/top-mlops-tools. Accessed 04 Dec 2023
DataRobot (2023) MLOps 101: the foundation for your AI strategy. https://www.datarobot.com/mlops-101/. Accessed 29 Dec 2023
Diaz-de Arcaya J, Torre-Bastida AI, Zárate G et al (2023) A joint study of the challenges, opportunities, and roadmap of MLOps and AIOps: a systematic survey. ACM Comput Surv 56(4):1–30. https://doi.org/10.1145/3625289
EvidentlyAI (2021) Evaluate and monitor ML models from validation to production. https://www.evidentlyai.com/. Accessed 29 Nov 2024
Flyte (2023) Kubernetes-native workflow automation platform for complex, mission-critical data and ml processes at scale. https://github.com/flyteorg/flyte. Accessed 10 Dec 2024
Fu W, Olson R, Nathan et al (2020) EpistasisLab/tpot: v0.11.5, v0.11.5. https://doi.org/10.5281/zenodo.3872281
Garriga M, Aarns K, Tsigkanos C et al (2021) DataOps for cyber-physical systems governance: the airport passenger flow case. ACM Trans Internet Technol 21(2):1–25. https://doi.org/10.1145/3432247
Google (2020) Cloud Architecture Center - MLOps: continuous delivery and automation pipelines in machine learning. https://cloud.google.com/architecture/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning. Accessed 12 Dec 2023
GreatLearning (2024) DevOps architecture features. https://www.mygreatlearning.com/devops/tutorials/devops-architecture-features. Accessed 16 Feb 2024

Hewage N, Meedeniya D (2022) Machine learning operations: a survey on MLOps tool support. https://doi.org/10.48550/ARXIV.2202.10169, https://arxiv.org/abs/2202.10169

Hu H, Kantardzic M, Sethi TS (2020) No free lunch theorem for concept drift detection in streaming data classification: a review. Wiley Interdiscip Rev Data Min Knowl Discov 10(2):e1327. https://doi.org/10.1002/widm.1327

Hummer W, Muthusamy V, Rausch T et al (2019) ModelOps: cloud-based lifecycle management for reliable and trusted AI. In: 2019 IEEE international conference on cloud engineering (IC2E). IEEE, pp 113–120. https://doi.org/10.1109/IC2E.2019.00025

John MM, Gillblad D, Olsson HH et al (2023) Advancing MLOps from ad hoc to Kaizen. In: 2023 49th Euromicro conference on software engineering and advanced applications (SEAA). pp 94–101. https://doi.org/10.1109/SEAA60479.2023.00023

Kreuzberger D, Kühl N, Hirschl S (2023) Machine learning operations (MLOps): overview, definition, and architecture. IEEE Access. https://doi.org/10.1109/ACCESS.2023.3262138

Kubeflow (2023) Machine learning toolkit for kubernetes. https://github.com/kubeflow/kubeflow. Accessed 10 Dec 2024

lakeFS (2023) Scalable data version control. https://lakefs.io/. Accessed 29 Dec 2023

Lefevre K, Arora C, Lee K et al (2022) ModelOps for enhanced decision-making and governance in emergency control rooms. Environ Syst Decis 42(3):402–416. https://doi.org/10.1007/s10669-022-09855-1

Li Y, Jiang ZM, Li H et al (2020) Predicting node failures in an ultra-large-scale cloud computing platform: an AIOps solution. ACM Trans Softw Eng Methodol 29(2):1–24. https://doi.org/10.1145/3385187

López García Á, De Lucas JM, Antonacci M et al (2020) A cloud-based framework for machine learning workloads and applications. IEEE Access 8:18681–18692. https://doi.org/10.1109/ACCESS.2020.2964386

Luz WP, Pinto G, Bonifácio R (2019) Adopting DevOps in the real world: a theory, a model, and a case study. J Syst Softw 157(110):384. https://doi.org/10.1016/j.jss.2019.07.083

Mage-AI (2023) Mage is an open-source, hybrid framework for transforming and integrating data. https://github.com/mage-ai/mage-ai. Accessed 10 Dec 2024

Martinez DR, Kifle BM (2024) MLOps: transitioning from development to deployment. In: Artificial intelligence: a systems approach from architecture principles to deployment. The MIT Press. https://doi.org/10.7551/mitpress/14806.003.0016

Mboweni T, Masombuka T, Dongmo C (2022) A systematic review of machine learning DevOps. In: 2022 international conference on electrical, computer and energy technologies (ICECET). pp 1–6. https://doi.org/10.1109/ICECET55527.2022.9872968

Microsoft (2021) Neural network intelligence. https://github.com/microsoft/nni. Accessed 29 July 2023

Microsoft (2022) Azure Architecture Center—machine learning operations maturity model. https://learn.microsoft.com/en-us/azure/architecture/ai-ml/guide/mlops-maturity-model. Accessed 12 Dec 2023

MLeap-Docs(CombustML) (2023) Deploy ML pipelines to production. https://github.com/combust/mleap. Accessed 10 Dec 2024

MLflow-Org(Databricks) (2023) MLflow: a machine learning lifecycle platform. https://github.com/mlflow/mlflow/. Accessed 10 Dec 2024

MLRun(Iguazio) (2023) Machine learning automation and tracking. https://github.com/mlrun/mlrun. Accessed 10 Dec 2024

Moreschini S, Lomio F, Hästbacka D et al (2022) MLOps for evolvable AI intensive software systems. In: 2022 IEEE international conference on software analysis, evolution and reengineering (SANER). pp 1293–1294. https://doi.org/10.1109/SANER53432.2022.00155

Munappy AR, Mattos DI, Bosch J et al (2020) From ad-hoc data analytics to DataOps. In: Proceedings of the international conference on software and system processes. pp 165–174. https://doi.org/10.1145/3379177.3388909

Navid P (2022) Airflow, prefect, and Dagster: an inside look. https://towardsdatascience.com/airflow-prefect-and-dagster-an-inside-look-6074781c9b77. Accessed 10 Dec 2024

Netflix (2023) Build and manage real-life data science projects with ease! https://github.com/Netflix/metaflow. Accessed 10 Dec 2024

Notaro P, Cardoso J, Gerndt M (2021) A survey of AIOps methods for failure management. ACM Trans Intell Syst Technol 12(6):1–45. https://doi.org/10.1145/3483424

Oladele S (2024) MLOps landscape in 2024: top tools and platforms. https://neptune.ai/blog/mlops-tools-platforms-landscape. Accessed 16 Feb 2024

OMG (2011) Business process model and notation. https://www.bpmn.org/. Accessed 08 Jan 2024

Orchest (2021) Build data pipelines, the easy way. https://github.com/orchest/orchest. Accessed 28 July 2023

Pachyderm (2023) Data-centric pipelines and data versioning. https://github.com/pachyderm/pachyderm. Accessed 10 Dec 2024

Paleyes A, Urma RG, Lawrence ND (2022) Challenges in deploying machine learning: a survey of case studies. ACM Comput Surv. https://doi.org/10.1145/3533378

Polyaxon (2023) MLOps tools for managing & orchestrating the machine learning lifecycle. https://github.com/polyaxon/polyaxon. Accessed 10 Dec 2024
Prefect (2023) Prefect is a workflow orchestration tool empowering developers to build, observe, and react to data pipelines. https://github.com/PrefectHQ/prefect. Accessed 10 Dec 2024
Prefect-Docs (2022) Why not airflow? https://docs-v1.prefect.io/core/about_prefect/why-not-airflow.html#overview. Accessed 10 Dec 2024
Qentelli (2024) Comprehensive list of DevOps tools 2024. https://www.qentelli.com/thought-leadership/insights/devops-tools. Accessed 16 Feb 2024
Relic N (2014) The open and composable observability and data visualization platform. visualize metrics, logs, and traces from multiple sources like prometheus, loki, elasticsearch, influxdb, postgres and many more. https://github.com/grafana/grafana. Accessed 22 Nov 2024
Remil Y, Bendimerad A, Mathonat R et al (2024) AIOps solutions for incident management: technical guidelines and a comprehensive literature review. http://arxiv.org/abs/2404.01363
RocketScience (2022) A review of +250 MLOps tools and solutions. https://rocketscience.one/mlops-software-review/. Accessed 01 June 2023
Ruf P, Madan M, Reich C et al (2021) Demystifying MLOps and presenting a recipe for the selection of open-source tools. Appl Sci. https://doi.org/10.3390/app11198861
Sáinz-Pardo Díaz J, Castrillo M, Bartok J et al (2024) Personalized federated learning for improving radar based precipitation nowcasting on heterogeneous areas. Earth Sci Inf 17(6):5561–5584. https://doi.org/10.1007/s12145-024-01438-9
Schmitt M (2023) Automated machine learning: AI-driven decision making in business analytics. Intell Syst Appl 18(200):188. https://doi.org/10.1016/j.iswa.2023.200188
Scikit-learn (2024) sklearn.preprocessing.minmaxscaler. https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html. Accessed 24 Nov 2024
SeldonCore (2023) Seldon core: blazing fast, industry-ready ML. https://github.com/SeldonIO/seldon-core. Accessed 10 Dec 2024
Shahin M, Babar MA, Zhu L (2017) Continuous integration, delivery and deployment: a systematic review on approaches, tools, challenges and practices. IEEE Access 5:3909–3943. https://doi.org/10.1109/ACCESS.2017.2685629
Shahin M, Rezaei Nasab A, Ali Babar M (2023) A qualitative study of architectural design issues in DevOps. J Softw Evolut Process 35(5):e2379. https://doi.org/10.1002/smr.2379
Star-History (2022) Star history: a tool to visualize the history of stars earned by repositories over time. https://github.com/star-history/star-history. Accessed 28 Nov 2024
Subramanya R, Sierla S, Vyatkin V (2022) From DevOps to MLOps: overview and application to electricity market forecasting. Appl Sci. https://doi.org/10.3390/app12199851
Symeonidis G, Nerantzis E, Kazakis A et al (2022) MLOps—definitions, tools and challenges. In: 2022 IEEE 12th annual computing and communication workshop and conference (CCWC). pp 0453–0460. https://doi.org/10.1109/CCWC54503.2022.9720902
Tensorflow-Extended (Google) (2023) Tensorflow extended (TFX)—end-to-end platform for deploying production ML pipelines. https://github.com/tensorflow/tfx. Accessed 10 Dec 2024
TraceML (2024) Polyaxon-traceml. https://github.com/polyaxon/traceml. Accessed 17 Jan 2024
Van Looveren A, Klaise J, Vacanti G et al (2019) Alibi detect: algorithms for outlier, adversarial and drift detection. https://github.com/SeldonIO/alibi-detect
Wang C, Wu Q, Weimer M et al (2021) FLAML: a Fast and Lightweight AutoML library. https://github.com/microsoft/FLAML. Accessed 28 July 2023
WeightsAndBiases (2023) 17 top MLOps tools you need to know. https://www.datacamp.com/blog/top-mlops-tools. Accessed 12 Dec 2023
Yaram S (2021) Machine learning model development and operations: principles and practice. https://www.kdnuggets.com/2021/10/machine-learning-model-development-operations-principles-practice.html. Accessed 07 Dec 2022
ZenML (2023) Build portable, production-ready MLOps pipelines. https://zenml.io. Accessed 10 Dec 2024
Zhou Y, Yu Y, Ding B (2020) Towards MLOps: a case study of ml pipeline platform. In: 2020 international conference on artificial intelligence and computer engineering (ICAICE). pp 494–500. https://doi.org/10.1109/ICAICE51518.2020.00102

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Authors and Affiliations

Lisana Berberi[1] · Valentin Kozlov[1] · Giang Nguyen[3,5] · Judith Sáinz-Pardo Díaz[2] · Amanda Calatrava[4] · Germán Moltó[4] · Viet Tran[3] · Álvaro López García[2]

✉ Lisana Berberi
lisana.berberi@kit.edu

Valentin Kozlov
valentin.kozlov@kit.edu

Giang Nguyen
giang.nguyen@stuba.sk

Judith Sáinz-Pardo Díaz
sainzpardo@ifca.unican.es

Amanda Calatrava
amcaar@i3m.upv.es

Germán Moltó
gmolto@dsic.upv.es

Viet Tran
viet.tran@savba.sk

Álvaro López García
aloga@ifca.unican.es

1 Scientific Computing Center (SCC), Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany

2 Instituto de Física de Cantabria (IFCA), CSIC-UC, Avda. los Castros s/n, 39005 Santander, Cantabria, Spain

3 Institute of Informatics, Slovak Academy of Sciences (IISAS), Dúbravská cesta 9, 845 07 Bratislava, Slovakia

4 Instituto de Instrumentación para Imagen Molecular (I3M), Centro Mixto CSIC - Universitat Politècnica de València (UPV), Camino de Vera S/N, 46022 Valencia, Spain

5 Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava (FIIT STU), Ilkovičova 2, 84216 Bratislava, Slovakia