

A guide to MLOps

Executive summary

With an adoption that doubled compared to 2017 based on the [McKinsey State of AI report](#), AI is going through a shift. Initiatives are going past the experimentation phase. Some noteworthy changes in the space include:

- An increasing number of capabilities being used by organisations as part of their AI initiatives.
- An increasing level of investment allocated to machine learning projects, which goes hand in hand with a higher adoption rate.
- More interest in collection, governance and ethics, aiming to ensure compliance for production deployments.
- Stable, secure, scalable tooling is a priority for enterprises. Having AI that enterprises can benefit from is critical.
- AI is more affordable and performant, with needs that are better addressed, tools that mitigate risk and an ecosystem that is better integrated.

How do you navigate these changes in the fast-paced world of AI? The answer lies in building a well-oiled MLOps practice. Much like DevOps changed the world of software development, MLOps will enable maturity for AI. This guide offers some guidance for data scientists and ML practitioners who are looking to take their AI models from the experimentation phase to the production stage with MLOps. We will start with an overview of what MLOps is, and cover its benefits and basic principles. We will then explore the typical MLOps lifecycle, available tooling and how Canonical can help you in your journey.

Contents

Executive Summary	1
An overview of MLOps	3
What is MLOps?	3
DevOps vs MLOps	4
1. Benefits of MLOps	4
1.1 Increased productivity	4
1.2 Reproducibility	5
1.3 Cost reduction	5
1.4 Monitorability	5
1.5 Sustainability	5
2. MLOps lifecycle	5
2.1 Fetching data	5
2.2 Engineering features	6
2.3 Training, testing and storing	6
2.4 Model deployment	6
MLOps tooling	6
MLOps principles	8
MLOps best practices	11
The MLOps landscape	14
Open source MLOps	15
3. The Canonical MLOps stack	17
3.1 Charmed Kubeflow	18
4. MLOps in your company	19

An overview of MLOps

MLOps is slowly evolving into an independent approach to the machine learning lifecycle that includes all steps – from data gathering to governance and monitoring. It will become a standard as artificial intelligence is moving towards becoming part of everyday business, rather than just an innovative activity.

MLOps benefits a wide range of functions within an enterprise that are impacted by machine learning initiatives. While data scientists can actually focus on modelling, the data engineering side has a tool to create pipelines and automate processes. At the same time, at a higher level, MLOps is a practice that ensures the stability that solution architects care about and the security that IT auditors demand today to get behind ML initiatives.

From a business angle, MLOps is a practice that brings reliability to machine learning projects. The goal is to have trustworthy projects that perform as expected in production. Depending on the use case, the outcome of an ML project can be different, but MLOps optimises associated functions or tasks, automates processes and plays an important role in time and cost savings. MLOps can be applied in any industry and any enterprise.

What is MLOps?

MLOps is the short term for machine learning operations and it represents a set of practices that aim to simplify workflow processes and automate machine learning and deep learning deployments. It accomplishes the deployment and maintenance of models reliably and efficiently for production, at a large scale.

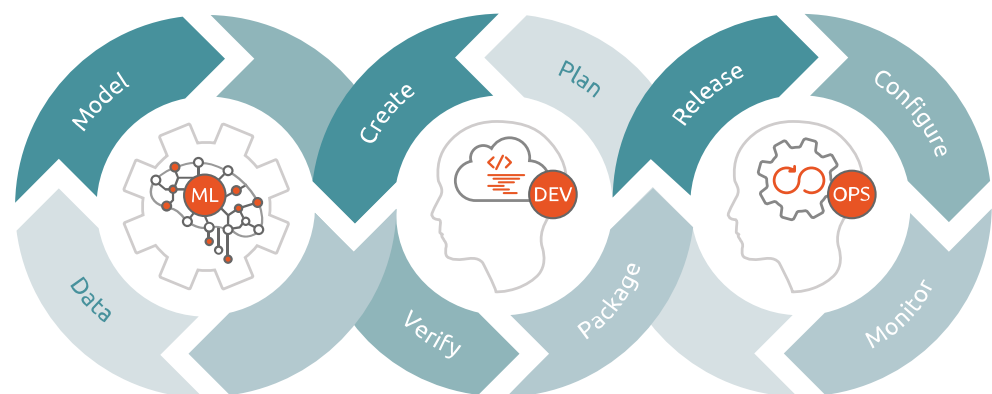


Image source: [Nvidia](#)

MLOps includes, besides the best practices, processes and underpinning technologies. They all provide a scalable, centralised and governed means to improve machine learning activities. It is a mix between machine learning development and operations.

From an ML perspective, it includes specific activities, such as model development or data gathering. It then goes further to development, where packaging, model creation and verification play an important role.

DevOps vs MLOps

Both DevOps and MLOps represent software development strategies that focus on collaboration between various parties that are involved in the process, such as developers, system administrators and data scientists. Whereas DevOps focuses mainly on applications and development, MLOps focuses on machine learning operations.

MLOps is, as already mentioned, a set of practices that borrow some of the widely adopted DevOps principles in software engineering. It uses them to take machine learning models to production. The expected outcome is improvements in code quality, faster patching, upgrades and updates, better customer satisfaction and finally, more efficient releases.

	DevOps	MLOps
Purpose	Focuses on integration between application development and IT operations	Focuses on integration between application development and IT operations
Primary user	Software engineer, DevOps engineer	Data scientist, data engineer, MLOps engineer
Motivation	CI/CD	CI/CD & CT/CM
Deliverables	Build executables which take hours	Train models which can take up to months
Ownership	Manage code	Manage code, data files, notebooks, models

Benefits of MLOps

MLOps accomplishes the deployment and maintenance of models reliably and efficiently for production, at a large scale. Knowing that it is a collaborative function, similar to DevOps, but focused on machine learning activities, it plays a crucial role in aligning business demands and regulatory requirements.

The benefits of MLOps include:

Increased productivity. MLOps accomplishes the deployment and maintenance of models reliably and efficiently for production, at a large scale. Knowing that it is a collaborative function, similar to DevOps, but focused on machine learning activities, it plays a crucial role in aligning business demands and regulatory requirements.

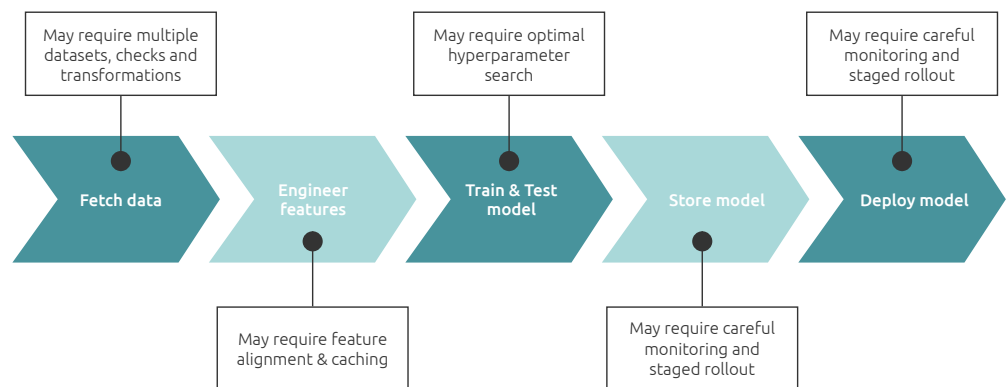
Reproducibility. Automating machine learning workflows leads to reproducibility, impacting the ML models and how they are trained, evaluated and deployed. Because of these benefits, both data versioning and model versioning are possible, ensuring the creation of snapshots of data as well as a feature store. MLOps enables further model optimisation using techniques such as hyperparameter tuning or in-depth experimentation of various model types.

Cost reduction. MLOps can significantly reduce costs, especially when considering scaling up AI initiatives and serving models to production. It impacts the entire machine learning lifecycle because of the minimisation of the manual efforts that come from task automation. It also enables easier error detection and improved model management.

Monitorability. Monitoring the behaviour of a machine learning model impacts not only the artificial intelligence project but also the area of business for which it has been designed. MLOps enables enterprises to monitor the model and gain insights about model performance in a systematic manner. It allows continuous model retraining, ensuring that it constantly offers the most accurate input. Furthermore, MLOps tools can send alerts in case of data drift or model drift, which is also flagging any vulnerability within the enterprise processes.

Sustainability. Because of the activity reduction and time-saving activities that MLOps brings, it enables enterprises to be more sustainable. Rather than letting employees get lost in repetitive tasks that use computing power, MLOps enables automation which, long-term, turns into an approach that is friendlier for the environment.

MLOps lifecycle



Fetching data. The machine learning lifecycle has a direct impact on the operations related to it. Data is the heart of any AI project, so without a big enough dataset, there is no machine learning modelling taking place. Fetching data includes various data sources that need to be further processed. It includes data coming from various devices, depending on the use case, having at the end of the day data that is numeric, text, video and much more.



.....

If you would like to learn more about hyperparameter tuning, [please watch our webinar](#)

.....

Engineering features. Once data is in place, there is a need to start processing and transforming it, such that it takes a shape that can be used for machine learning modelling. This includes activities that clean duplicates, aggregate and refine features to the ones that are relevant and finally make features visible across data teams.

Training, testing and storing. Training and testing models are what most people associate with machine learning. Training mostly uses machine learning libraries, such as PyTorch or Tensorflow, and has the ability to use more advanced techniques to optimise the machine learning models, such as hyperparameter tuning. This is a phase of experimentation that leads to the creation of multiple models which are then stored. Often training, as an experimentation step, leads to model versioning, as well as data versioning, which is further stored. Tracking model lineage, managing model artefacts and transitioning them through their lifecycle is part of the process that any AI initiative needs to take into account.

Model deployment. Once the review and analysis of the machine learning models is performed, deploying them is the next step. Model deployment addresses various aspects such as model refresh frequency or alert creation and management in case of failure.

Whereas the machine learning lifecycle is a repetitive process, offering the chance to perform all the activities within one tool represents one of the biggest challenges that machine learning engineers face. End-to-end tools, such as [Charmed Kubeflow](#), solve this problem, allowing specialists to do most of the steps within one tool.

MLOps tooling

The machine learning lifecycle can bring either clarity or become a burden for those who operate with models. Choosing the right tooling influences the delivery speed and ease of operations throughout the process. When it comes to MLOps activities, there are various aspects that need to be taken into account. There is no solution that fits all the scenarios in the case of machine learning, and it is mainly because of the wide variety of use cases that can be covered. However, when selecting an option, it's worth looking at a few key aspects:

- **Proprietary vendors:** whereas developing an MLOps solution in-house is possible, it can be time-consuming and not cost-effective. Building in-house helps enterprises design solutions that fit their own use case, which brings a lot of value, but also, it burns time and resources, which delays any return on investment from a machine learning initiative. On the other hand, a proprietary or closed-source product that is available as SaaS reduces the skill gap and increases the chances to find specialists in the area who have past experience with said tool.

- **Open source:** Opting for an open-source MLOps tool can save the company a lot of money and time that would have been spent on development. However, it requires staff that has enough experience to use such tooling and enable the outcomes of machine learning initiatives. In case you choose this option, you can consider working with companies like Canonical, which offers support for [open-source MLOps solutions](#).
- **Cloud provider:** It is common for companies to use cloud providers, as some of them offer specific services for AI/ML. While this seems convenient at first sight, there are a few things to consider, such as vendor lock-in, price per workload and TCO over time should the machine learning projects scale. Compliance regulations are also worth taking into account, because of the different types of data and where they can be hosted.
- **Managed environment:** Managing a new area such as machine learning operations requires new skills within the team. While it's still common for enterprises to maintain their own compute power needs, there are multiple options for MLOps tooling. Usually, for the exploratory phase, it's safe to simply try out multiple tools, without any commitment. Going further, the experimentation phase should be performed using a chosen tool. Once the production deployments take place, the enterprise should have support for the MLOps tool. This ensures the good functioning not only of the machine learning initiative but also of the business units that benefit from the AI projects.
- **User management:** Depending on the team that would be working on the machine learning initiatives, it's very likely that there is going to be a need for an MLOps tool that supports user management. This should enable on-hand data scientists and machine learning engineers to work better, but it also enables best practices such as the reproducibility of code.
- **Data Volumes:** Data is the core of any artificial intelligence initiative. Depending on the data types and volumes that the system needs to process, the MLOps tooling and the infrastructure underneath need to have different capabilities.
- **Speeding up pre-processing:** Before getting to the exciting part (model training), data scientists spend a lot of time preparing everything. Enabling them to perform the entire machine learning lifecycle in one tool not only saves you time, but it also improves the experience and enables collaboration between team members.

MLOps principles

The machine learning lifecycle includes multiple stages and addresses challenges at different levels of the stack. MLOps principles help practitioners navigate through the process and get positive outputs faster. Regardless of the utilised tooling, they should be followed closely. These include all three phases: design, experimentation and development, and operations.

1. **The design phase** involves business, data and design reaching a shared understanding of the ML-powered application. During this phase, the potential user is identified, as well as the problem it solves.
2. **Experimentation and development** aim to verify if machine learning is applicable to the chosen use case. It's realised by implementing a proof of concept (PoC) using a machine learning model. The main goal of this phase is to deliver a stable machine-learning model that can run in production.
3. **ML Operations** takes the developed machine learning model to production, using MLOps practices. It includes testing, versioning or monitoring.

All of these phases are interconnected and influence each other. MLOps principles can play a role at each level: data, ML model, and code. Modify any of the data, code or model and you might trigger a new build.

MLOps principles focus on:

- **Versioning**, which treats ML scripts, models and datasets as essential parts of DevOps processes. Versioning concerns data and model versioning, using system controls and alerts changes. The main reasons that data or ML model changes occur include:
 - ML models being retrained based on new data
 - Models being self-learning
 - Models degrading over time
 - Data residing across multiple systems
 - Data storage being immutable
- **Testing** needs to be performed at all levels of machine learning systems, having a different scope when ensuring performance and expected outcomes. It should include tests for features, with the goal to showcase how much predictions modify, compared to the existing version. In this case, data validation is required. Model development and infrastructure tests should also be taken into account.

- **Automation:** the level of automation determines the level of maturity of the ML initiative. The objective of any MLOps team is to automate the deployment of ML models. There are three levels of automation: manual process, ML pipeline automation and CI/CD automation. A manual process is performed when you start to implement the ML model and takes an iterative approach. ML pipeline automation executes model training in an automated manner. The last level, CI/CD automation, is introduced to perform machine learning deployments in production.
- **Reproducibility:** having reproducible and identical results in a machine learning workflow, given the same input, is a key MLOps principle. There are four phases: data collection, feature engineering, model training and model deployment. In order to ensure the reproducibility of the data collection phase, data should always be backed up. A snapshot of the data should be taken and data versioning should be in place. Feature engineering requires the code to use version control. In this case, reproducibility of the previous step is mandatory. When focusing on model training reproducibility is realised by having the same order of the features at all times. Documentation and automation of the feature transformation, hyperparameter selection and combination of ML models is needed. Model deployment has reproducibility ensured by using software versioning. It matches the production environment, as well as the use of containerisation. Furthermore, the same programming languages should be used for both training and deployment.
- **Deployment:** Model deployment should be done based on experiment tracking, which includes feature stores, containerisation of the ML stack and the ability to run on-prem, on the cloud or at the edge. A deployment service includes orchestration, logging, monitoring, and notifications, such that data, code and ML model artefacts are stable. MLOps includes four practices: continuous integration (CI), continuous development (CD), continuous training (CT) and continuous monitoring (CM). Whereas the CI/CD part is well documented from the DevOps practice, CT and CM are specific for machine learning models. CT is actually unique to ML systems and it covers the need to automatically retrain models for re-deployment. CM includes production data monitoring as well as model performance metrics, which are related to business metrics.
- **Monitoring:** Ensuring that ML models perform as expected, once deployed, is essential. Monitoring covers changes around dependencies, data, source systems and upgrades. It results in a notification and gives information about the specific modification. Monitoring of machine learning models in production should adopt an ML test score system, which measures the readiness of the system to be deployed to production and takes values between 0 and more than 5.

The table below explains how MLOps principles are aligned with the three phases of the complete ML lifecycle process. The table provides guidance when developing a new machine learning project and it helps experts have a clear overview of the initiative.

MLOps principles	Data	ML Model	Code
Automation	<ul style="list-style-type: none"> • Data transformation • Feature creation and manipulation 	<ul style="list-style-type: none"> • Data engineering pipeline • ML model training pipeline • Hyperparameter / Parameter selection 	<ul style="list-style-type: none"> • ML model deployment with CI/CD • Application build
Deployment	<ul style="list-style-type: none"> • Feature store is used in dev and prod environments 	<ul style="list-style-type: none"> • Containerisation of the ML stack • REST API • On-prem, cloud or edge 	<ul style="list-style-type: none"> • On-prem, cloud or edge
Monitoring	<ul style="list-style-type: none"> • Data distribution changes • Training vs serving features 	<ul style="list-style-type: none"> • ML model decay • Numerical stability • Computational performance of the ML model 	<ul style="list-style-type: none"> • Predictive quality of the application on serving data
Reproducibility	<ul style="list-style-type: none"> • Backup data • Data versioning • Extract metadata • Versioning of feature engineering 	<ul style="list-style-type: none"> • Hyperparameter tuning is identical between dev and prod • The order of features is the same • Ensemble learning: the combination of ML models is same • The model pseudo-code is documented 	<ul style="list-style-type: none"> • Versions of all dependencies in dev and prod are identical • Same technical stack for dev and production environments • Reproducing results by providing container images or virtual machines
Versioning	<ul style="list-style-type: none"> • Data preparation pipelines • Feature store • Datasets • Metadata 	<ul style="list-style-type: none"> • ML model training pipeline • ML model object • Hyperparameters • Experiment tracking 	<ul style="list-style-type: none"> • Application code • Configurations
Testing	<ul style="list-style-type: none"> • Data validation • Feature creation unit testing 	<ul style="list-style-type: none"> • Model specification is unit tested • ML model training pipeline is integration tested • ML model is validated before being operationalised • ML model staleness test (in production) • Testing ML model relevance and correctness • Testing non-functional requirements (security, fairness, interpretability) 	<ul style="list-style-type: none"> • Unit testing • Integration testing for end-to-end pipeline

MLOps best practices

“Developing and deploying ML systems is relatively fast and cheap, but maintaining them over time is difficult and expensive”

– D. Sculley et al.,
“Hidden Technical Debt in Machine Learning Systems,”
NIPS 2015

MLOps is a new practice and so is data science. While the industry is evolving fast and more and more courses become available on the market, most of them focus on machine learning practices. Developing a model itself is not such a difficult task anymore, but deploying it to production and then operating it comes with several challenges.

With only 19% of models making it to production in 2019 ([source](#)), companies could afford to leave MLOps best practices as a second thought. However, in 2022, around 35% of models moved past the experimentation ([source](#)) phase. This requires MLOps and constant collaboration between teams and departments, as well as best practices and continuous effort.

Each use case is different and tools have different requirements. However, following best practices ensures better collaboration, easier handovers and quicker movement through the ML lifecycle. These are our recommendations:

- Follow MLOps principles

They have been developed with a purpose and following them whenever machine learning models are developed or deployed to production increases your chances of success. From experimentation to monitoring, they cover all the steps that machine learning models go through. The MLOps principles take into account the three components that can influence the performance of the project as well, data, code and ML model, offering a clear overview.

- Build a DevOps culture

DevOps brought into the industry a different approach that encouraged collaboration between the development and operations areas of the business. In the case of AI initiatives, it would ensure a shared understanding of the business goals of the project and clear communication between the technical teams. Those responsible for the development, and those who drive the operations.

- Focus on security from day 1

Data is the most valuable asset that any enterprise can have. Ensuring the security of machine learning pipelines is essential to protect data, prevent unauthorised access and avoid any data or model poisoning. Knowing that ML models use a multitude of libraries that have a high number of packages which can become vulnerabilities in AI initiatives, security becomes an important part of the puzzle. Utilising MLOps platforms which ensure pipeline security decreases the risk of malicious attacks and chances to run machine learning projects securely.

- Stay compliant

Depending on the use case, industry, enterprise and region, there are various compliance regulations that need to be taken into account and followed in order to have a machine learning project that can actually be deployed to production. They mostly look after data privacy or financial regulations and whereas experimentation can be easily performed without taking this into account, production deployments and machine learning operations need to follow certain rules. In order to avoid blockage along the way, it is rather advisable to get informed beforehand and ensure that the AI initiatives follow the compliance guidelines in place.

- Have naming conventions

There is nothing new here, as it seems like a best practice for coding in general. Depending on the programming language used, there are several naming convention guides, such as [“Style guide for Python”](#), that can be read and followed. Using naming conventions is more than a nice to have, especially if multiple teams are working on different initiatives. Machine learning systems grow quickly and the same happens with the number of variables. Ensuring that various team members maintain and keep developing the project at a high quality is highly dependent on the basic rules that are set at the beginning.

- Consider disaster recovery

Even if undesired, there is always a chance of failures or outages happening. Like any other software system, machine learning should have a strategy in case it occurs, such that activities or processes within a company do not get paralysed or model performance improvement can continue to be done. When developing disaster recovery plans, it is worth taking into account the three main components of any machine learning project: code, data, and machine learning model. Thus, having data backups, trained models and methods to quickly retrain if needed is essential.

- Improve financial management

Reports like [AI Adoption by IBM](#) mentioned that the high costs that machine learning systems come with still represent a concern for enterprises. When talking about costs, there are various components such as compute resources, storage, new employees, and infrastructure expansion that need to be taken into account. It is critical to have a process in place that manages costs at all stages of the machine learning workflow, from exploration to production scaling. Opportunities to optimise them by automating processes, running on a hybrid cloud or using resource schedulers such as [Volcano](#) are options that can accelerate AI adoption within any enterprise.

- Try A/B testing

Machine learning models are very rarely a one-time activity. There is usually a lot of experimentation before reaching a winning version of the model, which meets the performance criteria and, very often, once deployed to production, models are retrained periodically. A/B testing comes in handy when comparing newer versions to an existing one, which delivers good results, because of the possibility to observe and address changes, as well as avoidance of any negative impact caused by a malfunctioning or an unexpected user experience.

- Check code quality

This is related to a few other best practices mentioned here, but having code quality checks is advised for any software system. Having code that does what is expected, is easy to read, maintain and extend and does not have a problem raises the chance to have, in the end, a model that can be deployed to production and actually improved over time. While unit testing seems obvious, utilising linters and formatters through machine learning systems will benefit those involved in its development of it. It eliminates bugs, delivers a code that is clean and speeds up the code review process. In the end, it leads to faster project delivery and better operations of it.

- Use MLOps platforms

Each machine learning project is different and so are its requirements. From the type of data used to the required processing type, data scientists and machine learning developers need to analyse everything before choosing a platform. Most of the time, the cost is one constraint, but also have a tool that can automate processes, run on various clouds and offer security is something that AI projects will benefit from. Depending on the scale of both the enterprise and projects, there is no one answer to all cases.

- Write documentation, ensure longevity

Documentation should cover all components of machine learning projects and it ensures the development, maintainability and operability of any project. In the case of ML systems, documentation should encompass data sources, processes that are used to make decisions related to the data, and data labelling. In addition to that, project documentation should include model selection criteria, model performance baselines, experiment designs and model pseudo-code. Lastly, the deployment process, how to run locally and how to operate the model should be part of the code documentation.

- Focus on structure

Project structure ensures fast project delivery, easy operations and long-term development of any machine learning system. In the case of data, it should include separate folders for raw data, data pipelines and tests. Also, trained models and notebooks feature engineering should be kept separately, in different folders. It looks like a tedious job, but it allows easy navigation through any machine learning project, allowing new developers to quickly onboard and old ones to be able to work without having to memorise a lot of actions.

The MLOps landscape

The MLOps landscape is steadily growing and changing in an accelerated manner, with new tools popping up daily and new needs being discovered. The productisation of ML models is the main priority, but the lifecycle is much longer than that. Companies have gone from being stuck with Jupyter Notebooks a few years back, to having too many options to choose from.

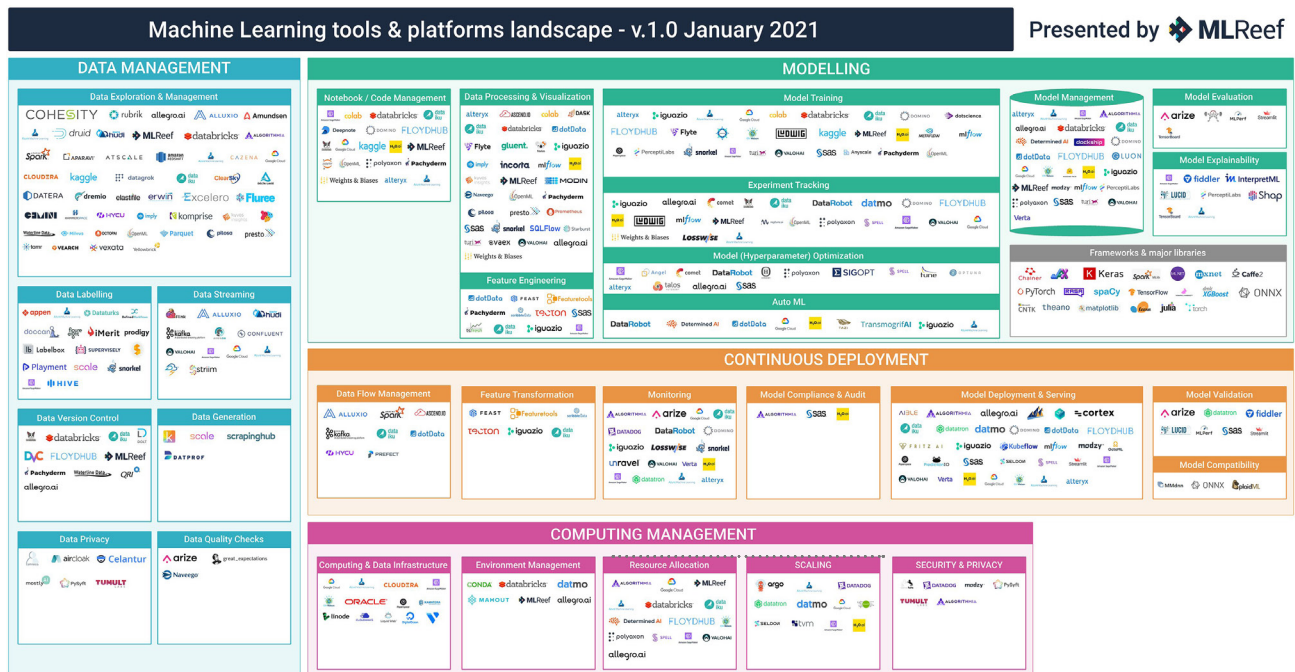


Image Source: [MLReef](#)

When looking more in-depth at this picture, it is easy to observe a lot of new names. The MLOps tooling landscape consists of a lot of startups. Data is quite a popular topic amongst new tooling providers. A clear market need for end-to-end MLOps tools can also be observed.

Navigating through such an extended offering can be intimidating. In order to make a choice, the end user of the tool, as well as the need that is going to be covered, have to be taken into account. The stack itself should cover needs related to data engineering, data version control, continuous integration and delivery pipelines, deployment automation, model performance analysis and model monitoring. While some tools solve specific challenges, others try to play a bigger role and cover more steps of the lifecycle.

To get a successful result, before selecting tools or even frameworks, the requirements of each component need to be gathered and analysed. The [AI Infrastructure Alliance](#) put together a useful blueprint which aims to create a shared vocabulary between machine learning vendors. It drives interoperability and eases the process of picking a stack, based on the user's needs. Furthermore, it encourages collaboration between vendors and highlights integration opportunities.

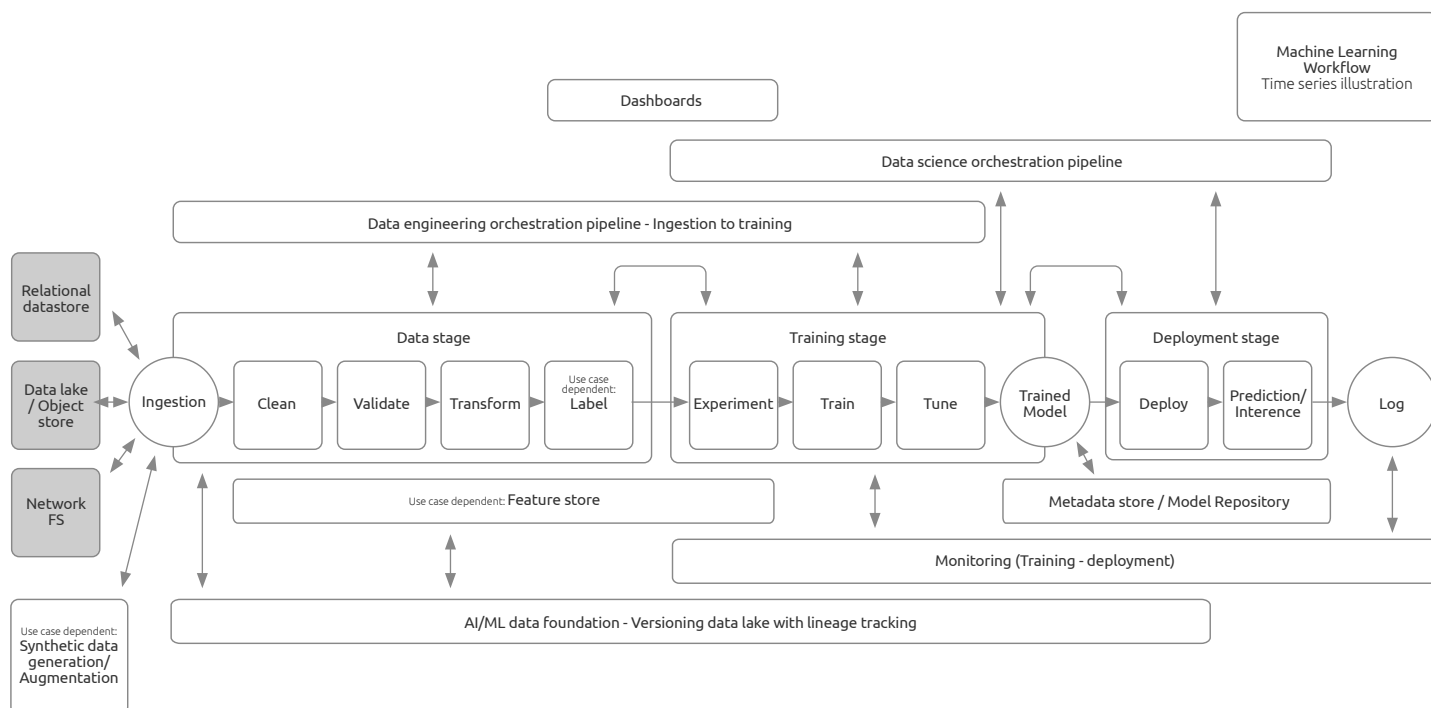


Image Source: [AI Infrastructure Alliance](#)















AIIA's blueprint brings clarity to the MLOps landscape. While all these represent steps in the ML lifecycle, whenever starting a new project, it is ideal to first map the main challenges that are likely to appear initially and focus on the requirements that need to be addressed.

Open source MLOps

Open source MLOps is an alternative that gives access to the source code, can be tried without paying and allows contributions in a more flexible manner. AI/ML benefited from interest from upstream communities right from the start. It led to an open-source MLOps landscape that produced solutions focused on various categories such as:

- **End-to-End platforms:** Kubeflow, MetaFlow or MLFlow are clear examples of this. Rather than covering a certain problem, they offer a solution that offers a suite of features. Amongst Kubeflow's components, there's Jupyter Notebooks, Tensorflow job operator and simplified containerisation. On the other hand, MLFlow's capabilities are around tracking, project and modelling.
- **Development and deployment:** MLRun, ZenML or SeldonCore are just some of the examples that fall into this category. Their main purpose is to automate tasks and perform deployments on multiple environments.

- **Data:** Being the core of any AI project, this category itself can be further split into:
 - **Validation:** Hadoop and Spark are the leaders when it comes to this kind of activity. Their main goal is to perform data quality checks, automating this process. Duplicates, inconsistencies or missing data are just some of the activities that are performed.
 - **Exploration:** Jupyter Notebook is the main example of this sub-category. It automates the data analysis phase, providing an easy way to visualise the data easily, track the changes and execute the code as you go.
 - **Versioning:** Pachyderm or DVC are just two of the examples that cover this functionality. Knowing that ML models require multiple experiments, a versioning control tool is required, in order to keep track of the evolution.
- **Testing:** Flyte is a good candidate to ensure these features. It is the last step in the machine learning lifecycle. Long term, It ensures the reproducibility of tasks and easily flags any problems that there might be in the process
- **Monitoring:** Tools such as Prometheus and Grafana are known on the market for covering these needs. They integrate with other tools and have as their main goal to ensure that models work as expected when they are deployed and that any change performed in the model does not have a negative impact.
- **The scheduling system:** Volcano is the recommended open-source tool for this kind of job. Optimised for intensive computing workloads, it is a Kubernetes native batch schedule and ensures container scheduling. In AI initiatives jobs are scheduled entirely, avoiding failure and allowing faster training

End-to-end MLOps platform	Data	Development and deployment	Testing	Monitoring
  	   	   		 

While open-source tools are ideal for several reasons, when it comes to enterprise-grade solutions, having experts offering support, ensuring security and providing guidance accelerates machine learning projects and allows data scientists to focus on modelling rather than debugging. Thus, many companies distribute open-source tooling, by providing patches, upgrades, and updates on top, in order to protect the data and model. They provide support, answering issues in a timely manner, and supporting both enterprises and open-source initiatives.

The Canonical MLOps stack

Canonical covers a wide spectrum of the components that are needed in the MLOps landscape. We built an open-source MLOps ecosystem that addresses data needs, training and deployment of machine learning models. The tooling is well integrated, creating a seamless experience for specialists. It benefits from the latest hardware updates, enabling access to improved computing power.

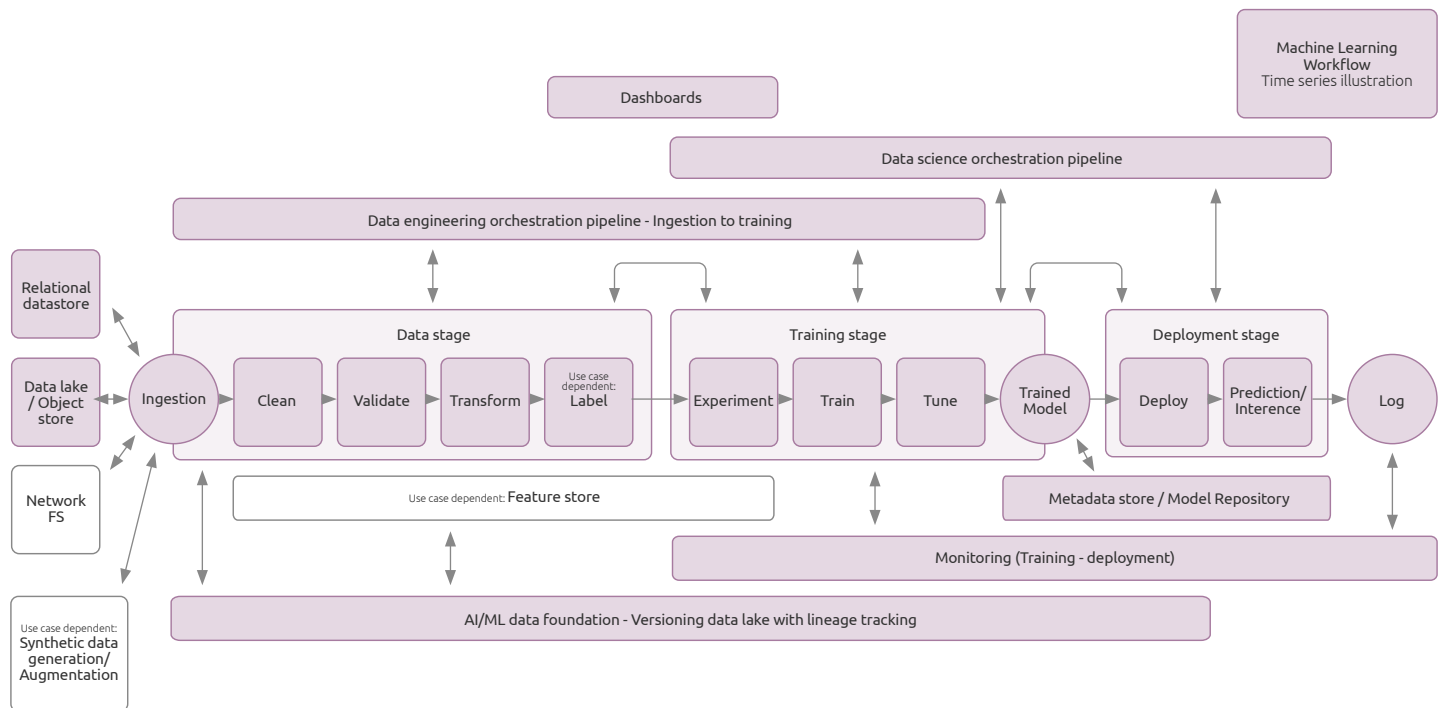


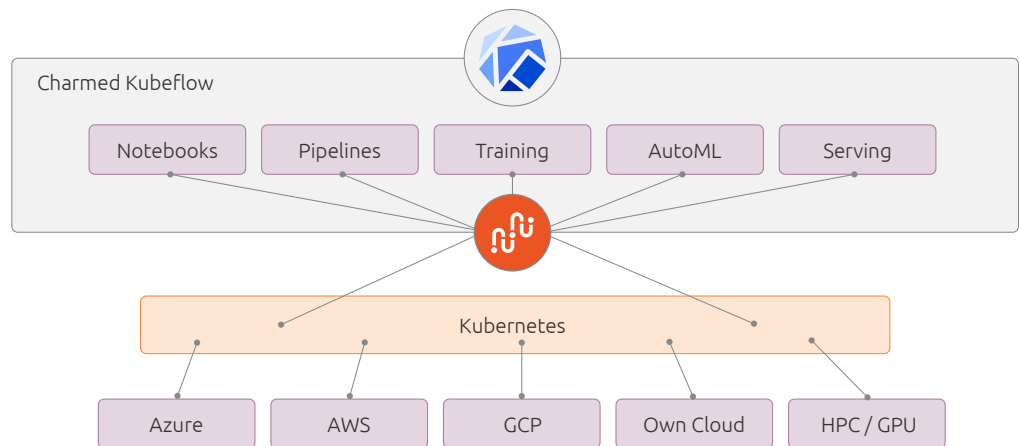
Image Source: [AI Infrastructure Alliance](#)

As the diagram above shows, the MLOps landscape is quite complete by choosing Canonical's solution. It covers most of the steps listed by the AIIA's blueprint, allowing consistency through the machine learning lifecycle. You can get a direct line to experts in the field, who can support your initiatives and guide you through the process. You will collaborate with an open-source vendor that offers security patching. Our solutions run on Ubuntu, but Canonical does not limit anyone to this operating system, offering access to Linux on Windows, through WSL, and on Mac, through Multipass.

From an AI/ML standpoint, Canonical does not cover only the application layer, highlighted in the blueprint presented above, but also the stack underneath, by offering container, cloud or bare-metal solutions. The Canonical MLOps stack fills the gaps through the entire process, covering data, development and experimentation, and operations.

Charmed Kubeflow

[Charmed Kubeflow](#) is a production-grade, end-to-end open-source MLOps platform that translates steps in the data science workflow into Kubernetes jobs. It is one of the official distributions of the [Kubeflow](#) upstream project. Using it, data scientists and machine learning engineers benefit from having ML deployments that are simple, portable and scalable. It has capabilities that cover a wide range, from experimentation using Notebooks to training using the Kubeflow Pipelines or tuning, using Katib.



Charmed Kubeflow allows faster project delivery, enables reproducibility and uses the hardware at its fullest potential. With the ability to run on any cloud, the MLOps platform is compatible with both public clouds, such as AWS or Azure, as well as private clouds. Furthermore, it is compatible with legacy HPC clusters, as well as high-end AI-dedicated hardware, such as NVIDIA's GPUs or DGX.

Charmed Kubeflow is a suite of tools that covers all the steps of the machine learning lifecycle, dedicated to machine learning activities that help its users to build ML models, analyse model performance, tune hyperparameters, manage computer power, deploy models to production and finally monitor them. The MLOps solution runs on hybrid and multi-cloud scenarios. It protects users to be locked in within one vendor and gives the flexibility to go with the cloud of choice. By having this enabled, compliance and data regulations are easier to be met.

MLOps in your company

The rising popularity of MLOps outlines a need for AI/ML projects to take a step further. As a practice that encourages collaboration between different areas of the business, MLOps increase the chance of success for machine learning initiatives. With a landscape in constant change, this whitepaper states the key points to consider when choosing the right tool for MLOps. Open source benefits from wide adoption, so having an MLOps stack that is fully open-source is completely possible.

Depending on your use case, there are different needs that Canonical covers through its services and products:

- **Charmed Kubeflow:** Available for free to be deployed. Canonical has an offering that includes support or managed services. We ensure deployment, and security patching for over 25,000 packages in the Ubuntu Universe and Main repositories, 10 years of support for Ubuntu LTS releases and bug fixes and operational support for open-source applications.
- **AI/ML advisory:** With four service lanes, Canonical helps enterprises that are at different stages with their AI initiatives. From exploring the opportunities through data exploration workshops to building PoCs. We help you transform MLOps from theory to a hands-on approach, using your own data and live sessions with domain experts

Learn more

[Read more](#) about Canonical's AI services

[Read more](#) about how to [get started with AI](#) and [Charmed Kubeflow](#)

Contact us to talk to an expert or direct all at (US Central) +1 737 204 0291 and or (UK) +44 203 656 5291

