**Framework and Best Practices**

**Framework:**

**Action:** Identifying popular MLOps frameworks that align with client's needs. Factors like:

1. MLOps streamlines meaching learning workflows, ensuring seamless integration from development to deployment.
2. By automating model training, testing and deployment, MLOps accelerates time-to-market for AI solutions.
3. Continuous monitoring and optimization in MLOps guarantee models remain efficient and effective in dynamic environments

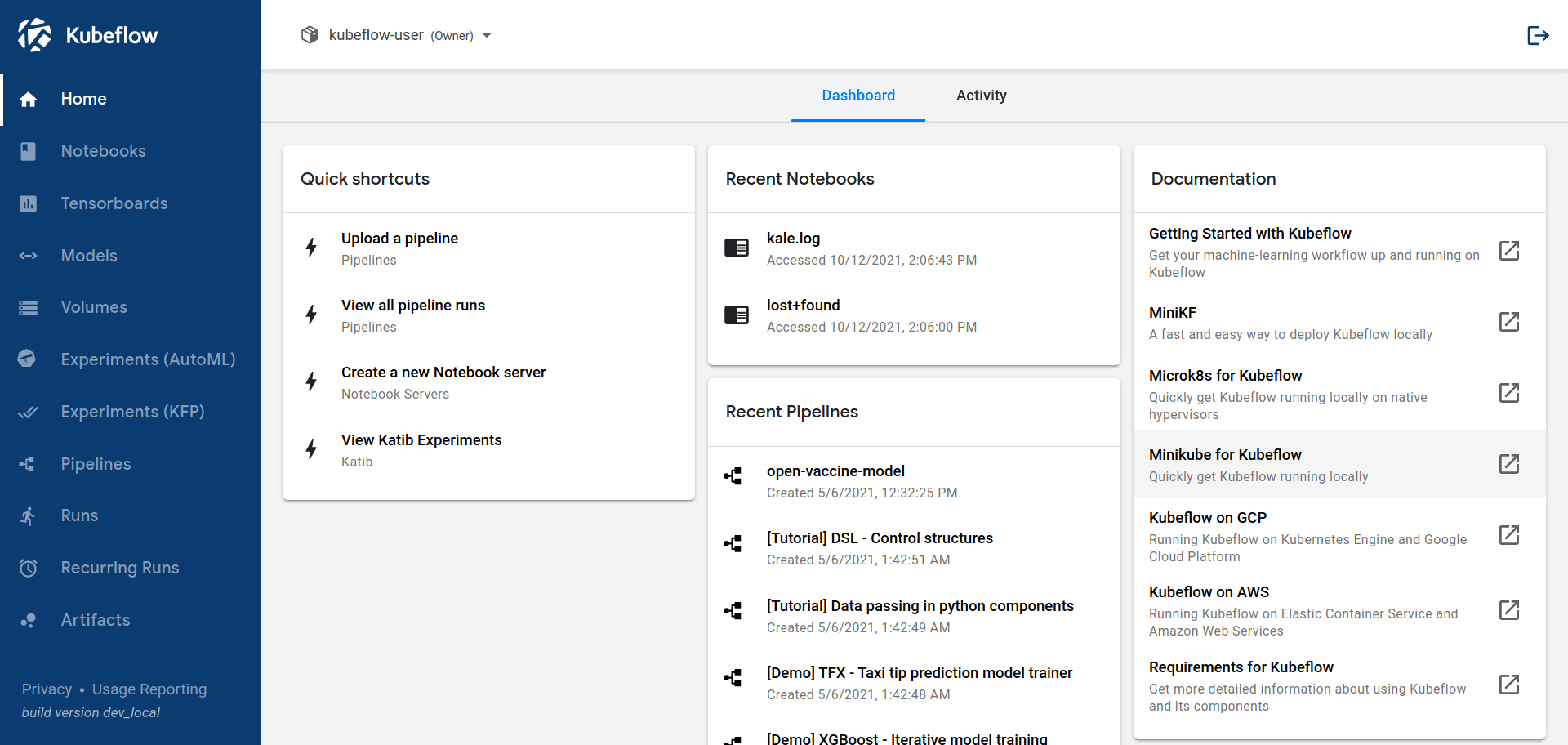
**Note:**

|  |
| --- |
| Ultimately, open-source tools can be tricky. Before you choose the tool for your project, you need to carefully study its pros and cons. Moreover, you need to make sure that the tools work well with the rest of your stack. This is why I prepared a list of popular and community-approved MLOps platforms, tools, and frameworks for different stages of the model development process. |

Let us start by exploring the open-source platforms first followed by frameworks and tools:

* Kubeflow

Almost immediately after Kubernetes established itself as the standard for working with a cluster of containers, Google created Kubeflow—an open-source project that simplifies working with ML in Kubernetes. It has all the advantages of this orchestration tool, from the ability to deploy on any infrastructure to managing loosely-coupled microservices, and on-demand scaling.



This project is for developers who want to deploy portable and scalable machine learning projects. Google didn’t want to recreate other services. They wanted to create a state-of-the-art open-source system that can be applied alongside various infrastructures—from supercomputers to laptops.

With Kuberflow, you can benefit from the following features:

**Jupyter notebooks**

Create and customize Jupyter notebooks, immediately see the results of running your code, and create interactive analytics reports.

**Custom TensorFlow job operator**

This functionality helps train your model and apply a TensorFlow or Seldon Core serving container to export the model to Kubernetes.

**Simplified containerization**

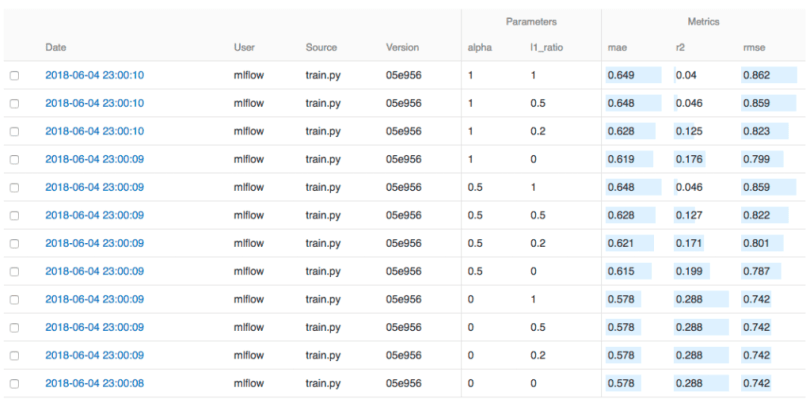
Kuberflow eliminates the complexity involved in containerizing the code. Data scientists can perform data preparation, model training, and deployment in less time.

All in all, Kuberflow is a full-fledged solution for the development and deployment of end-to-end machine learning workflows.

* **MLflow**

MLflow is an open-source platform for machine learning engineers to manage the machine learning lifecycle through experimentation, deployment, and testing. MLflow comes in handy when you want to track the performance of your machine learning models. It’s like a dashboard, one place where you can:

* monitor machine learning pipelines,
* store model metadata, and
* pick the best-performing model.



Right now, there are four components provided by MLflow:

**Tracking**

The MLflow Tracking component is an API and UI for logging parameters, code versions, metrics, and output files for running the code and visualizing the results. You can do log and query experiments using Python, REST, R API, and Java APIs. You can also record the results.

**Project**

MLflow Project is a tool for machine learning teams to package data science code in a reusable and reproducible way. It comes with an API and command-line tools to connect projects into workflows. It helps you run projects on any platform.

**Model**

MLflow Model makes it easy to package machine learning models to be used by various downstream tools, like Apache Spark. With this, deploying machine learning models in diverse serving environments is much more manageable.

Overall, users love MLflow because it’s easy to use locally without a dedicated server and has a fantastic UI where you can explore your experiments.

* **Metaflow**

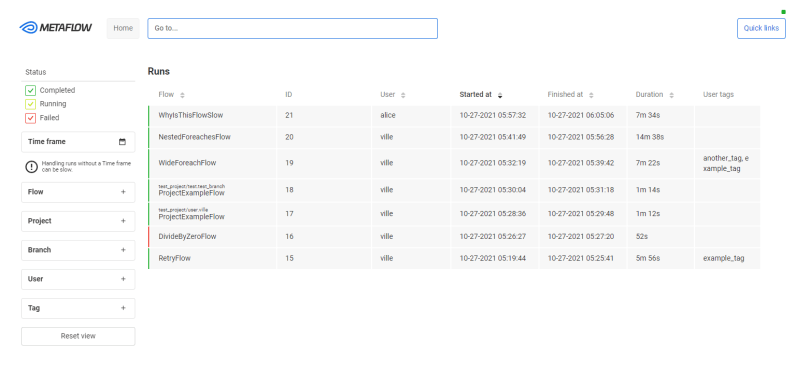
Netflix created Metaflow as an open-source MLOps platform for building and managing large-scale, enterprise-level data science projects. Data scientists can use this platform for end-to-end development and deployment of their machine-learning models.

**Great library support**

Metaflow supports all popular data science tools, like TensorFlow and scikit-learn, so you can keep using your favorite tool. Metaflow supports Python and R, making it even more flexible in terms of library and package choice.

**Powerful version control toolkit**

What is excellent about Metaflow is that it versions and keeps track of all your machine learning experiments automatically. You won’t lose anything important, and you can even inspect the results of all the experiments in notebooks.



As it was mentioned above, Metaflow was specifically created for large-scale machine learning development. The AWS cloud powers the solution, so there are built-in integrations to storage, compute, and machine learning services from AWS if you need to scale. You don’t have to rewrite or change the code to use any of it.

* **Flyte**

If you’re looking for a platform that will take care of experiment tracking and maintenance for your machine learning project, have a look at Flyte. It is an open-source orchestrator designed to simplify the creation of robust data and machine learning pipelines for production. Its architecture prioritizes scalability and reproducibility, harnessing the power of Kubernetes as its foundational framework.

Flyte offers a ton of features and use cases from a simple machine learning project to complex LLMs projects. To give you an I have distilled a some features and listed them below, but you check out their website and documentation.

**Large-scale project support**

Flyte has helped them to execute large-scale computing that’s crucial to their business. It’s not a secret that scaling and monitoring all pipeline changes can be pretty challenging, especially if the workflows have complex data dependencies. Flyte successfully deals with tasks of higher complexity, so developers can focus on business logic rather than machines.

**Improved reproducibility**

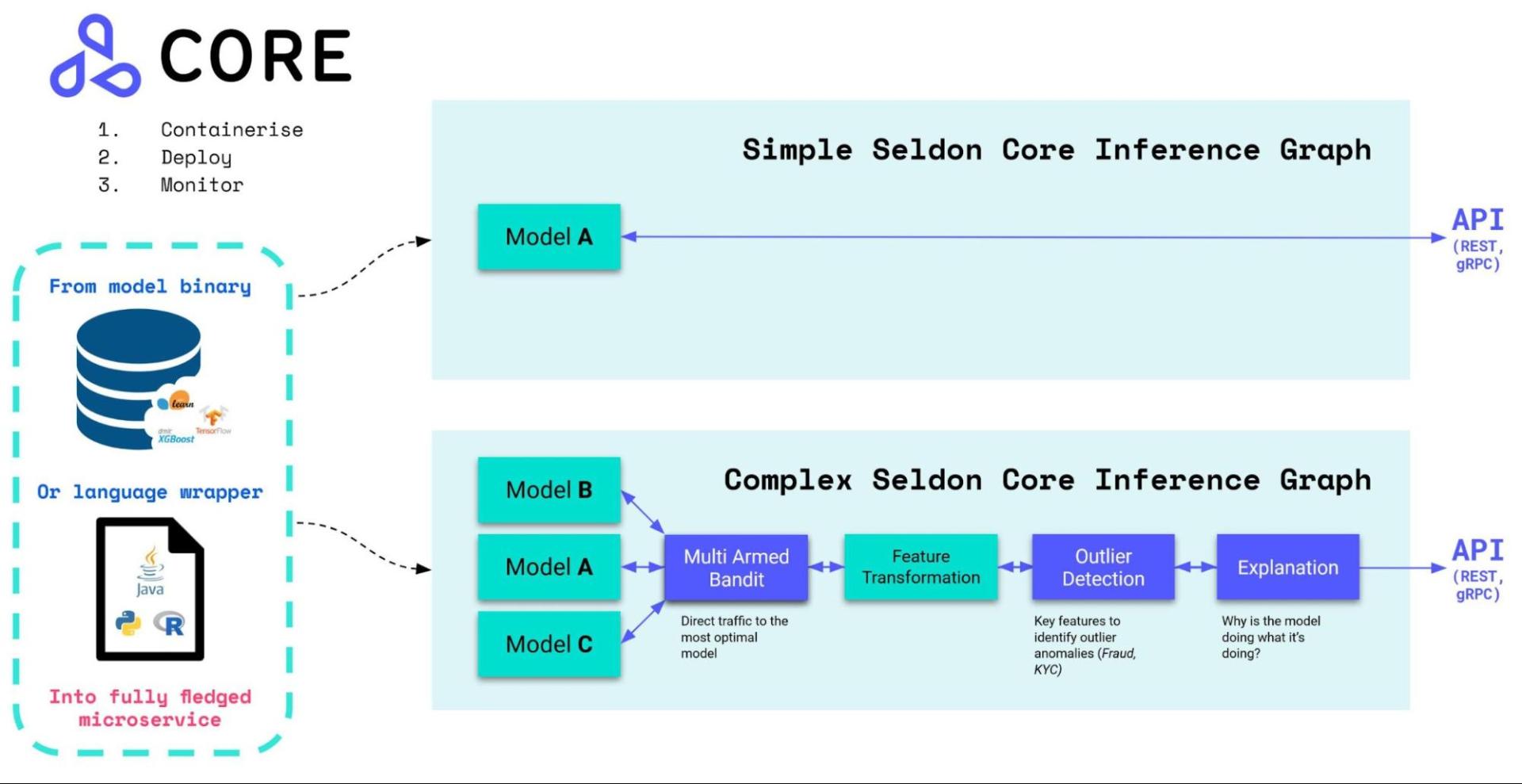
This tool can also help you be sure of the reproducibility of the machine learning models you build. Flyte tracks changes, does version control, and containerizes the model alongside its dependencies.

**Multi-language support**

Flyte was created to support complex ML projects in Python, Java, or Scala.

Flyte has been tested out by Lyft internally before they released it to the public. It has a proven record of managing more than 7,000 unique workflows totaling 100,000 executions every month.

* **Seldon Core**



Seldon Core is one of the platform for machine learning model deployment on Kubernetes. This platform helps developers build models in a robust Kubernetes environment, with features like custom resource definitions to manage model graphs. You can also merge your continuous integration and deployment tools with platform.

**Build scalable models**

Seldon core can convert your model built on TensorFlow, PyTorch, H2O, and other frameworks into a scalable microservice architecture based on REST/GRPC.

**Monitor model performance**

It will handle scaling for you, and give you advanced solutions for measuring model performance, detecting outliers, and conducting A/B testing out-of-the-box.

**Robust and reliable**

Seldon Core can boast the robustness and reliability of a system supported through continuous maintenance and security policy updates.

Optimized servers provided by Seldon Core allow you to build large-scale deep-learning systems without having to containerize them or worry about their security.

**Resources**: Tutorials, case studies, Documentation

**Best Practices Selection:**

**Action:** As an MLOps practitioner, you know firsthand the challenges of deploying machine learning models in real-world production environments. Staying informed about the latest MLOps best practices adopted by other production teams is a shortcut to doing things well the first time.

To mitigate this, newer versions of the model must be constantly shipped. This calls for continuous training and continuous monitoring in addition to the DevOps practices of CI/CD.

And then you want/need to monitor those models. As you have more models in prod you want naming schemas and a registry for model artifacts and packages.

And so on

So, in this article, we’re going to explore some of the best practices engineers need to consistently deliver the machine learning systems their organizations need.

**Examples:** Boot camp, Code Quality checks, Version Control (Git), Full stack Deep Learning, Setup experiment tracking and Monitor Predictive service performance. Etc….