

# Production frameworks for DataOps factory

<https://neptune.ai/blog/best-workflow-and-pipeline-orchestration-tools>

(Ed., see also: [AWS Step Functions](#), [Apache Airflow](#))

[MLOps Blog](#)

## Best Machine Learning Workflow and Pipeline Orchestration Tools



Krissanawat Kaewsanmua

7 min, 1st August, 2023

### Machine Learning Tools

Machine learning is rampaging through the IT world and driving a lot of high-end tech. It created a revolution of automation and flexibility for researchers and businesses.

When it comes to machine learning, workflows (or pipelines) are an essential component that drives the overall project.

In this article, we'll explore:

- what exactly workflows and pipelines are,
- and more than 10 tools that we can use to orchestrate workflows and pipelines.

### Interested in other MLOps tools?

When building their ML pipelines, teams usually look into a few other components of the MLOps stack. If that's the case for you, here are a few article you should check:

[The Best MLOps Tools and How to Evaluate Them](#)

[15 Best Tools for ML Experiment Tracking and Management](#)

[Best 8 Machine Learning Model Deployment Tools](#)

## What is a workflow in Machine Learning?

A workflow in ML is a sequence of tasks that runs subsequently in the machine learning process.

The workflows are the different phases of a machine learning project. These phases include:

- data collection,
- data pre-processing,
- building datasets,
- model training and refinement,
- evaluation,
- deployment to production.

# What are pipelines in Machine Learning?

Pipelines in machine learning are an infrastructural medium for the entire ML workflow. Pipelines help automate the overall [MLOps](#) workflow, from data gathering, EDA, data augmentation, to model building and deployment. After the deployment, it also supports reproduction, tracking, and monitoring.

ML pipelines help improve the performance and management of the entire model, resulting in quick and easy deployment.

## Dig deeper

- [How to Build an End-To-End ML Pipeline](#)

## The best Machine Learning orchestration tools

Machine learning orchestration tools are used to automate and manage workflows and pipeline infrastructure with a simple, collaborative interface. Along with the management and creation of custom workflows and their pipelines, these tools also help us track and monitor models for further analysis.

Orchestration tools make the ML process easier, more efficient and help data scientists and ML teams focus on what's necessary rather than waste resources trying to identify priority issues.

Orchestrating a proper workflow can be very useful for a company invested in machine learning. To do so, you must understand how to automate the entire process and how to extract valuable model output during production with monitoring and tracking.

So, to introduce some of the best tools for MLOps workflow/pipeline orchestration, we've compiled a list.

- [Kale](#) – Aims at simplifying the Data Science experience of deploying Kubeflow Pipelines workflows.
- [Flyte](#) – Easy to create concurrent, scalable, and maintainable workflows for machine learning.
- [MLRun](#) – Generic mechanism for data scientists to build, run, and monitor ML tasks and pipelines.
- [Prefect](#) – A workflow management system, designed for modern infrastructure.
- [ZenML](#) – An extensible open-source MLOps framework to create reproducible pipelines.
- [Argo](#) – Open source container-native workflow engine for orchestrating parallel jobs on Kubernetes.
- [Kedro](#) – Library that implements software engineering best-practice for data and ML pipelines.
- [Luigi](#) – Python module that helps you build complex pipelines of batch jobs.
- [Metaflow](#) – Human-friendly lib that helps scientists and engineers build and manage data science projects.
- [Coulter](#) – Unified interface for constructing and managing workflows on different workflow engines.
- [Valohai](#) – Simple and powerful tool to train, evaluate and deploy models.
- [Dagster.io](#) – Data orchestrator for machine learning, analytics, and ETL.
- [Netflix Genie](#) – Genie developed by Netflix is an open-source distributed workflow/task orchestration framework.

[...]

Here's how all those orchestration tools compare in terms of features.

	<a href="#">Kale</a>	<a href="#">Flyte</a>	<a href="#">MLRun</a>	<a href="#">ZenML</a>	<a href="#">Argo</a>	<a href="#">Kedro</a>	<a href="#">Luigi</a>	<a href="#">Metaflow</a>	<a href="#">Valohai</a>	<a href="#">Dagster</a>	<a href="#">Couler</a>	<a href="#">Genie</a>	<a href="#">Prefect</a>
Price	Free	Free	Free	Free	Free	Free	Free	Free	Get a custom quote	Free	Free	Free	Free
Open-source	X	X	X	X	X	X	X	X		X	X	X	X
Focus	Kubeflow pipeline & workflow	Create concurrent, scalable, and maintainable workflows	End-to-end ML pipelines	Creating production-ready ML pipelines	Kubernetes	Reproducible, maintainable	Build complex pipelines of batch jobs	Manage real-life data science projects	End-to-end ML pipelines	End-to-end ML pipelines	Manage other tools	Big Data orchestration	End-to-end data pipeline
Lightweight	X	X	X	X	X	X	X	X	X		X	X	X
Free plan limitation	Open source	Open source	Open source	Open source	Open source	Open source	Open source	Open source	No free plan	Open source	Open source	Open source	Open source
Hosted version available		X	No/ For storage (Yes)	X	X				X				X
Scales to millions of runs					X						X		
Dedicated user support		X	No/ For storage (Yes)t						X				
UI to visualize and manage workflows	X	X	X	X	X	X	X	X		X			X
Artifact support (S3, Artifactory, Alibaba Cloud OSS, HTTP, Git, GCS, raw)		X	X	X	X	X		X	X	X	X		X
Workflow templating to store commonly used Workflows in the cluster	X	X		X	X	X	X	X		X			X
Scheduled workflows		X	X										
Server interface with REST API		X	X	X	X			X	X	X			X
DAG or Steps based declaration of workflows		X	X	X	X	X			X	X	X		
Step level input & outputs (artifacts/parameters)		X	X	X	X								
Loops		X	X		X								X

And now, let's take a deeper look at each of those workflow/orchestration platforms.

## Kale

When working with a Jupyter Notebook, data scientists benefit from interactivity and visualizations. After finishing a task, refactoring the notebook to manage Kubeflow Pipelines can be difficult and time-consuming.

[Kale](#) solves this problem. It's a tool that simplifies the deployment process of Jupyter Notebooks into Kubeflow Pipelines workflows. It translates a Jupyter Notebook directly into a KFP pipeline. In doing so, it ensures that all the processing building blocks are well-organized and independent from each other. It also leverages the power of [experiment tracking](#) and workflow organization, provided out-of-the-box by Kubeflow.

The image shows the Kale deployment panel on the left and a JupyterLab notebook on the right. The notebook is titled 'titanic\_dataset\_ml.ipynb' and is running Python 3. The code in the notebook includes the following imports and data loading steps:

```

import numpy as np
import pandas as pd
import seaborn as sns
from matplotlib import pyplot as plt
from matplotlib import style

from sklearn import linear_model
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import Perceptron
from sklearn.linear_model import SGDClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.naive_bayes import GaussianNB

```

The notebook also shows a 'loaddata' step configuration with the following code:

```

path = "data/"
PREDICTION_LABEL = 'Survived'
test_df = pd.read_csv(path + "test.csv")
train_df = pd.read_csv(path + "train.csv")

```

Source: [Kale on GitHub](#)

Kale offers a platform to control and coordinate complex workflows on top of Kubernetes. Plus, you can create reusable components, and execute them along with workflows. With a simple UI for defining KFP pipelines directly from the JupyterLab interface, the tool is very efficient and effective for workflow pipeline orchestration.

Functionality:

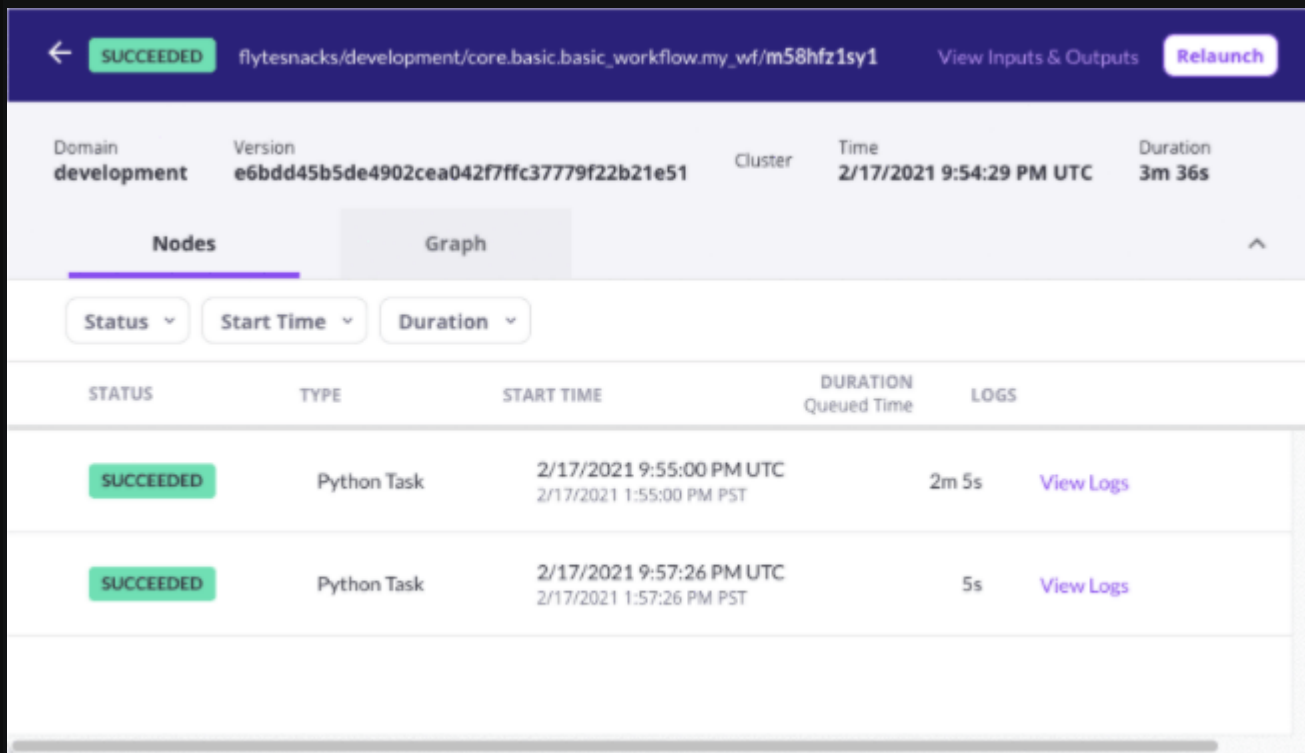
- Open-source
- Focus on: Kubeflow pipeline & workflow
- Lightweight

## Flyte

[Flyte](#) is another high-end tool to easily create ML workflows. It's a structured programming and distributed processing platform, with highly concurrent, scalable, and maintainable workflows for machine learning and data processing.

It already manages over 10,000 workflows. The stellar infrastructure lets you create an isolated repo, deploy, and scale without affecting the rest of the platform. It's built on top of Kubernetes, and offers portability, scalability, and reliability.

Flyte's interface is elastic, intuitive, and easy to use for multiple tenants. It offers parameters, data lineage, and caching for organizing your workflows.



The screenshot shows the Flyte interface for a workflow execution. At the top, a purple header bar contains a back arrow, a green 'SUCCEEDED' status badge, the workflow ID 'flytesnacks/development/core.basic.basic\_workflow.my\_wf/m58hfz1sy1', a 'View Inputs & Outputs' link, and a 'Relaunch' button. Below the header, a summary row displays: Domain: development, Version: e6bdd45b5de4902cea042f7ffc37779f22b21e51, Cluster: (empty), Time: 2/17/2021 9:54:29 PM UTC, and Duration: 3m 36s. Two tabs, 'Nodes' and 'Graph', are visible. Below the tabs are three filter buttons: 'Status', 'Start Time', and 'Duration'. The main content is a table with the following columns: STATUS, TYPE, START TIME, DURATION (Queued Time), and LOGS. Two rows of 'Python Task' nodes are shown, both with a 'SUCCEEDED' status. The first node started at 2/17/2021 9:55:00 PM UTC (2/17/2021 1:55:00 PM PST) and took 2m 5s. The second node started at 2/17/2021 9:57:26 PM UTC (2/17/2021 1:57:26 PM PST) and took 5s. Each row has a 'View Logs' link.

STATUS	TYPE	START TIME	DURATION Queued Time	LOGS
SUCCEEDED	Python Task	2/17/2021 9:55:00 PM UTC 2/17/2021 1:55:00 PM PST	2m 5s	<a href="#">View Logs</a>
SUCCEEDED	Python Task	2/17/2021 9:57:26 PM UTC 2/17/2021 1:57:26 PM PST	5s	<a href="#">View Logs</a>

Source: [Flyte](#)

The overall platform is dynamic and extensible, and offers a wide variety of plugins to assist workflow creation and deployment. Workflows can be reiterated, rolled back, experimented with, and shared to speed up the development process for the whole team.

Functionality:

- Open-source
- Focus on: Creating concurrent, scalable, and maintainable workflows
- Lightweight

## MLRun

[MLRun](#) is an open-source workflow/pipeline orchestration tool. It has an integrative approach to organizing machine-learning pipelines, from initial development, through model building, all the way to full pipeline deployment in production.

It summons an abstraction layer, integrated with a wide range of ML tools and plugins for working with features and models along with workflow deployment. MLRun has a feature and artifact store to control ingestion, processing, metadata, and storage of data across multiple repositories and technologies.

It has an elastic server-less service for converting simple code into scalable and organized microservices. It also facilitates automated experiments, model training and testing, and deployment of real-time pipeline workflows.

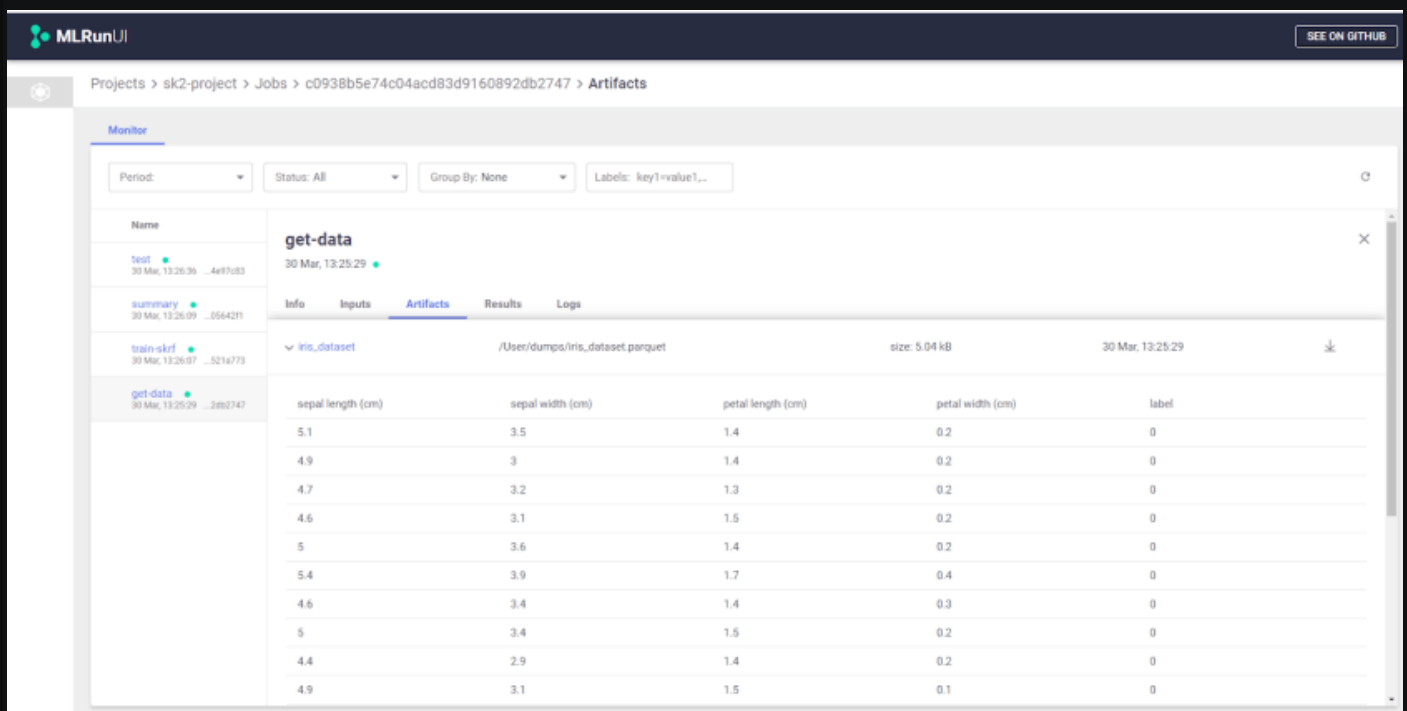
The overall UI has a centralized structure to manage ML workflows. Key features include rapid deployment, elastic scaling, feature management, and flexible usability.

Functionality:

- Open-source
- Focus on: end-to-end ML pipelines
- Lightweight

Functionality:

- Open-source
- Focus on: end-to-end ML pipelines
- Lightweight



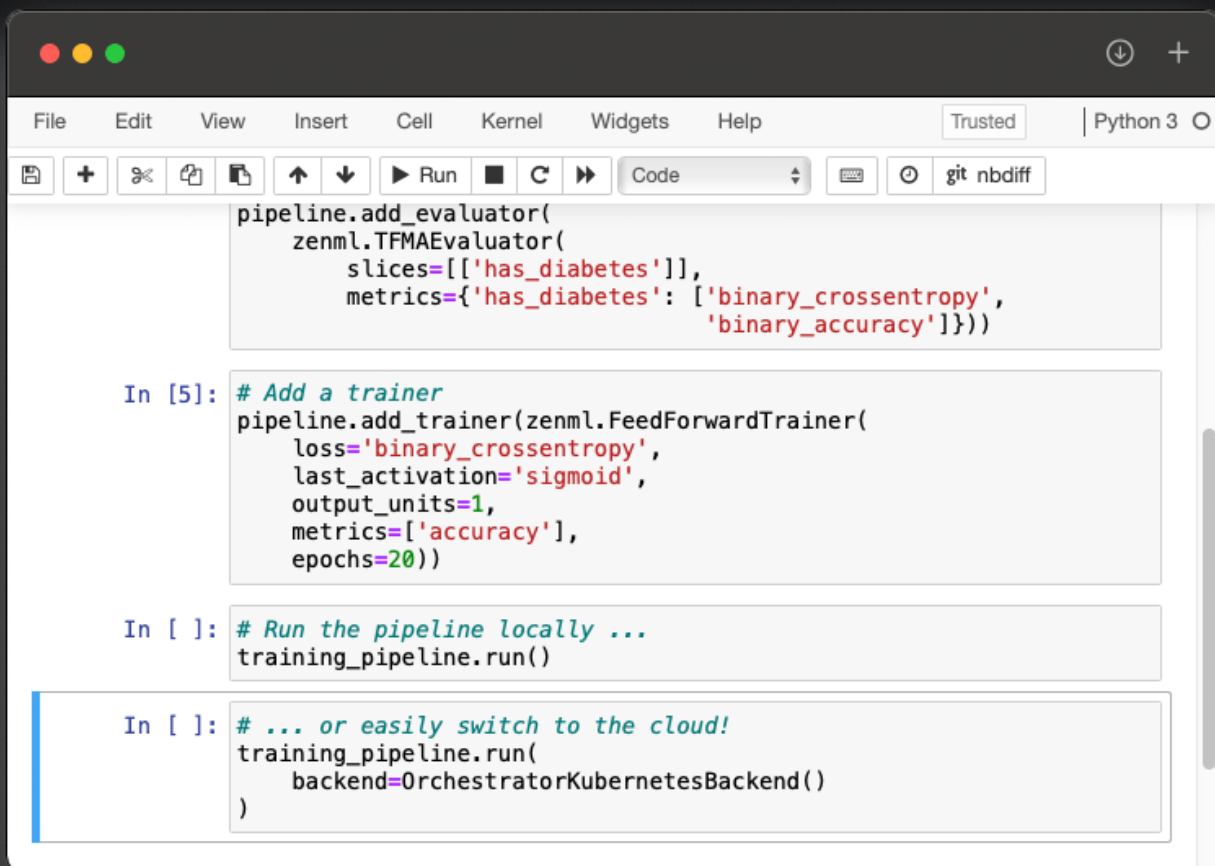
Source: [MLRun on GitHub](#)

## ZenML

[ZenML](#) is a popular open-source MLOps tool for creating reproducible workflows. It was built to solve the issue of translating observed patterns from Jupyter notebook research into a production-ready ML environment.

The tool focuses on production-based replication issues, such as versioning difficulties and models, reproducing experiments, organizing complex ML workflows, bridge training, deployment, and tracking metadata. It can work alongside other workflow orchestration tools to provide a simple path to getting your ML model into production.

At its core, ZenML breaks down ML development into steps representing individual tasks. The sequence of tasks operated together forms a workflow pipeline. You can leverage integrations, and switch seamlessly between local and cloud systems.



```
pipeline.add_evaluator(
    zenml.TFMAEvaluator(
        slices=[['has_diabetes']],
        metrics={'has_diabetes': ['binary_crossentropy',
                                'binary_accuracy']}))

In [5]: # Add a trainer
pipeline.add_trainer(zenml.FeedForwardTrainer(
    loss='binary_crossentropy',
    last_activation='sigmoid',
    output_units=1,
    metrics=['accuracy'],
    epochs=20))

In [ ]: # Run the pipeline locally ...
training_pipeline.run()

In [ ]: # ... or easily switch to the cloud!
training_pipeline.run(
    backend=OrchestratorKubernetesBackend()
)
```

Source: [ZenML on GitHub](#)

You can accurately version data, models, and configurations. It automatically detects the database schema, and lets you view statistics. It lets you evaluate the model (using built-in evaluators), compare training pipelines, and distribute preprocessing to the cloud.

Functionality:

- Open-source
- Focus on: creating production-ready ML pipelines
- Lightweight

## Might be useful

If you use ZenML, check the [neptune.ai + ZenML integration](#).

Neptune is an experiment tracker. So thanks to this integration, with less boilerplate code, you can log and visualize information from your ZenML pipeline steps (e.g., models, parameters, metrics).

- [Neptune-ZenML integration docs](#)

[Check full example](#) Full screen preview

# Prefect

I believe that [Prefect](#) is one of the best automated workflow management tools out there. Built for modern infrastructure, on top of an open-source Prefect Core workflow engine.

This workflow management system makes it easy to take data pipelines and add semantics, like retries, logging, dynamic mapping, caching, or failure notifications.

It facilitates two ready-to-use databases:

- Prefect Core's server,
- Prefect Cloud.

They have UI backends that automatically extend the Prefect Core engine with a rich GraphQL API, to make workflow orchestration simple. Prefect Core's server is an open-source, lightweight alternative to Prefect Cloud.

The screenshot displays the Prefect UI interface for a workflow run named 'chestnut-yak'. The top navigation bar includes a search icon, a notification bell, and a user profile icon 'J'. Below the navigation, the breadcrumb 'tutorial > hello-flow > FLOW RUN' is visible. The main content area features a 'Timeline' chart showing the workflow's progress with a green bar indicating the 'Running' state and a vertical dashed line for 'Success'. Below the timeline, there are two panels: 'chestnut-yak' overview and 'chestnut-yak Task Runs'. The overview panel shows the flow was created by 'jim\_prefect\_io', has a last state message of '[4 Dec 2020 11:26am]: All reference tasks succeeded.', and a duration of 5 seconds. The task runs panel lists several tasks, including 'say\_hello (Parent)', 'say\_hello (Mapped Child 2)', 'say\_hello (Mapped Child 1)', 'say\_hello (Mapped Child 0)', and 'people', all with a duration of less than 1 second and a 'Success' state.

Task	Start Time	End Time	Duration	State
say_hello (Parent)			...	Success
say_hello (Mapped Child 2)	4 Dec 2020 11:26...	4 Dec 2020...	< 1 second	Success
say_hello (Mapped Child 1)	4 Dec 2020 11:26...	4 Dec 2020...	< 1 second	Success
say_hello (Mapped Child 0)	4 Dec 2020 11:26...	4 Dec 2020...	< 1 second	Success
people	4 Dec 2020 11:26...	4 Dec 2020...	< 1 second	Success

Source: [Prefect](#)

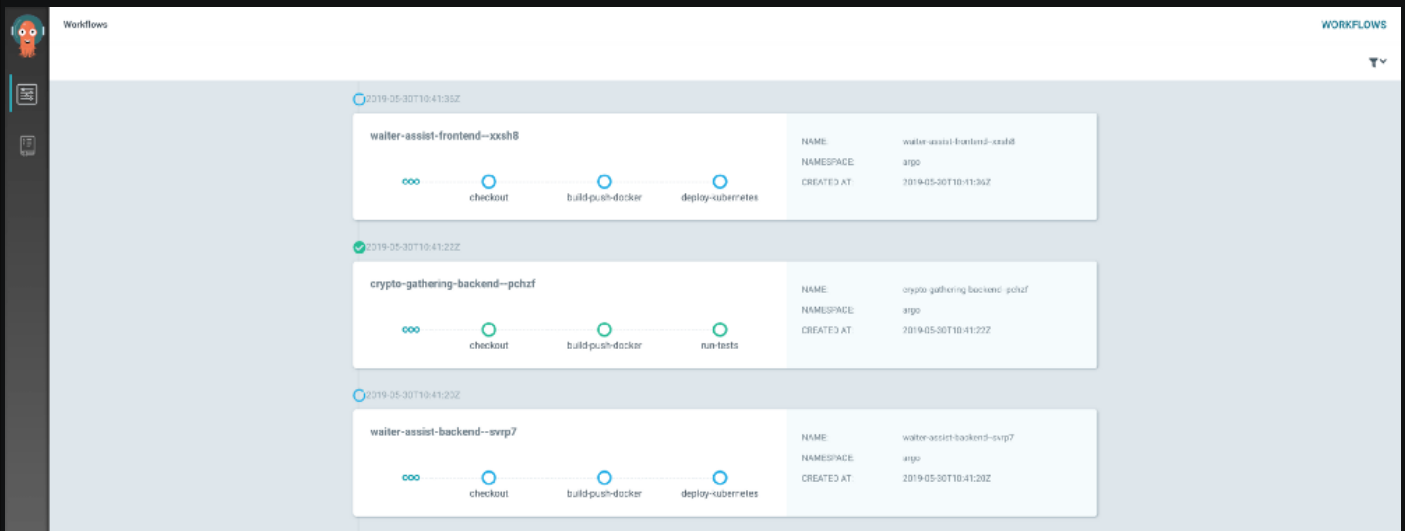
Prefect Cloud is a fully-hosted, deployment-ready backend for Prefect Core. It has enhanced features, like permissions and authorization, performance enhancements, agent monitoring secure runtime secrets and parameters, team management, or SLAs. Everything is automated, you just need to translate tasks into workflows and this tool will handle the rest.



# Argo

[Argo](#) is a powerful, container-native, open-source workflow engine. It's great for orchestrating parallel jobs on Kubernetes. It's been implemented as a Custom Resource Definition of Kubernetes. You can define pipeline workflows, where individual steps are taken as a container.

It lets you model multi-step workflows as a sequence of tasks. Also, Argo supports dependency tracking between tasks, thanks to a directed acyclic graph (DAG). It can easily handle intensive tasks for machine learning and data science, saving you a lot of time.



Source: [Argo on GitHub](#)

Argo has CI/CD configured directly on Kubernetes, you don't have to plug in any other software. It's cloud-agnostic, runs on any Kubernetes cluster, and enables easy orchestration of highly parallel jobs on Kubernetes.

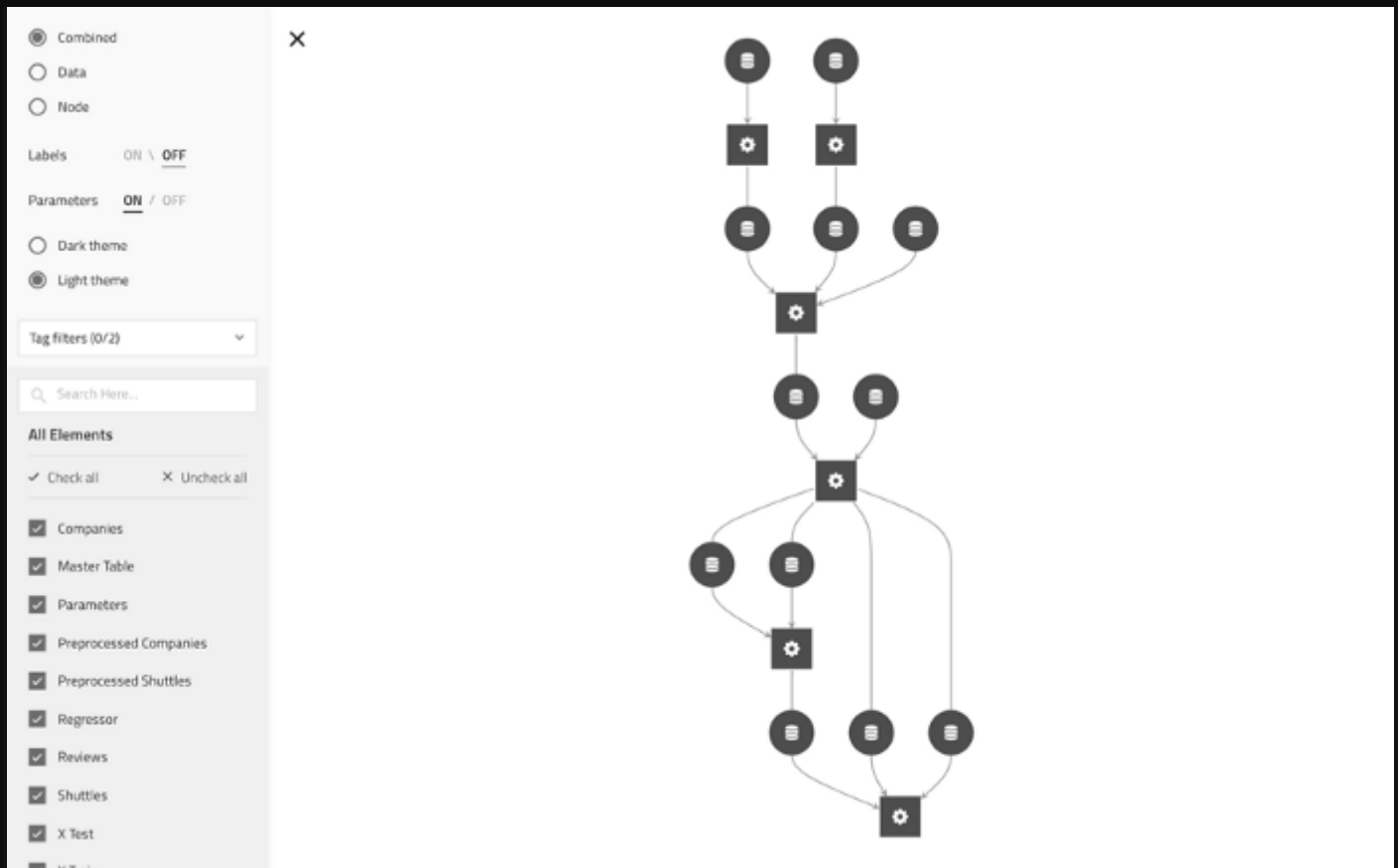
Functionality:

- Open-source
- Focus on: Kubernetes
- Lightweight

# Kedro

[Kedro](#) is a workflow orchestration tool based on Python. You can create reproducible, maintainable, and modular workflows to make your ML processes easier and more accurate. Kedro integrates software engineering into a machine learning environment, with concepts like modularity, separation of concerns, and versioning.

It offers a standard, modifiable project template based on [Cookiecutter Data Science](#). The data catalog handles a series of lightweight data connectors, used to save and load data across many different file formats and file systems.



Source: [Kedro on GitHub](#)

With pipeline abstraction, you can automate dependencies between Python code and workflow visualization. Kedro supports single- or distributed-machine deployment. The main focus is creating maintainable data science code to address the shortcomings of Jupyter notebooks, one-off scripts, and glue-code. This tool makes team collaboration easier at various levels, and provides efficiency in the coding environment with modular, reusable code.

Functionality:

- Open-source
- Focus on: reproduction, modularity, and maintainable
- Lightweight

## One more tip

**If you use Kedro, check the [neptune.ai + Kedro plugin](#).**

It lets you have all the benefits of a nicely organized Kedro pipeline with a powerful Neptune UI for filtering, comparing, displaying, and organizing ML metadata generated in pipelines and nodes.

**See in app** Full screen preview

•

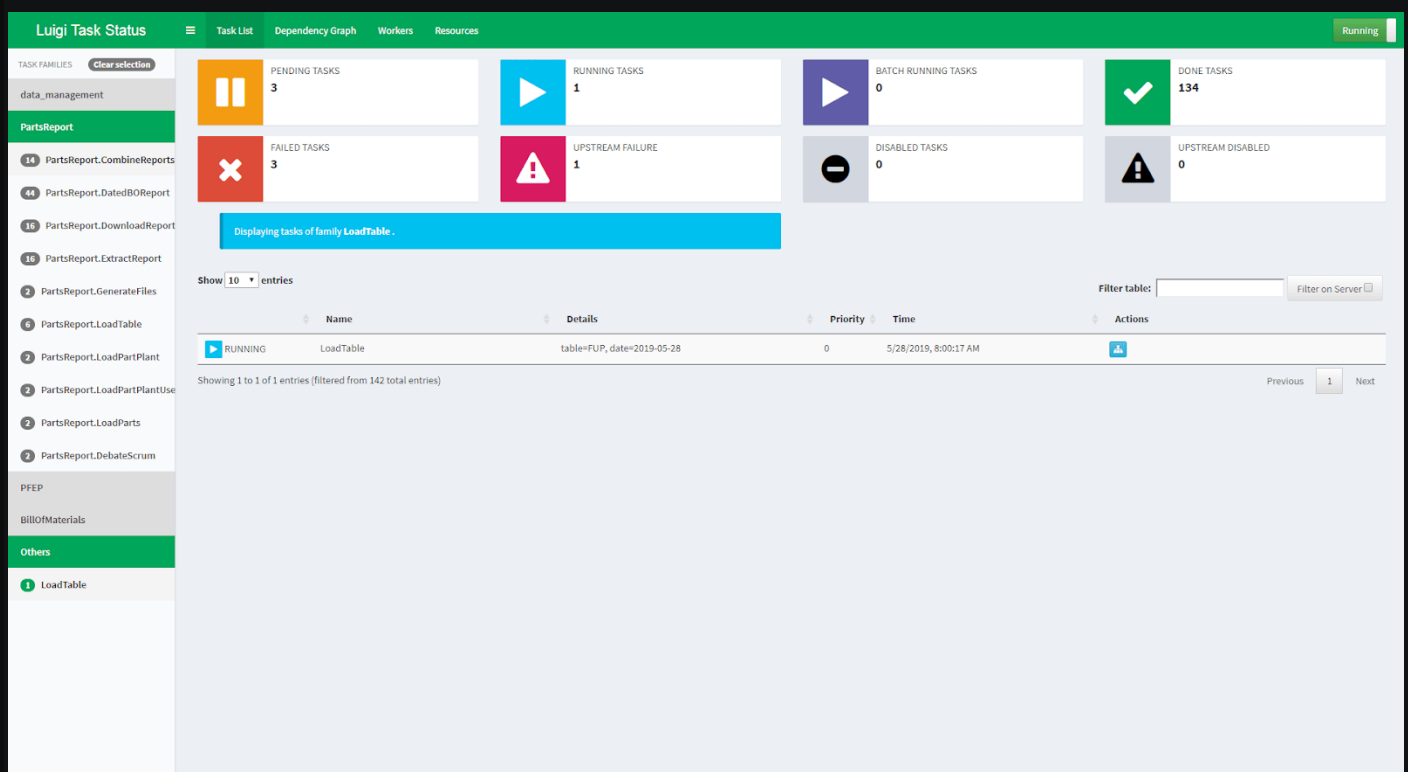
[Neptune-Kedro integration docs](#)

## [Case study on how ReSpo.Vision \(sport analytics company\) tracks their Kedro pipelines](#)

### Luigi

Luigi is an open-source Python package, optimized for workflow orchestration to perform batch tasks. With Luigi, it's easier to build complex pipelines. It offers different services to control dependency resolution and workflow management. It also supports visualization, failure handling, and command line integration.

It mainly addresses long-running complex batch processes. Luigi takes care of all workflow management tasks that may take a long time to finish, so that we can focus on actual tasks and their dependencies. It has a toolbox with common project templates.



Source: [Luigi on GitHub](#)

Luigi has state-of-the-art file system abstractions for HDFS and local files. This way, all file system operations are atomic. So, if you're looking for an all-Python tool that handles workflow management for batch job processing, then Luigi is for you.

Functionality:

- Open-source
- Focus on: building complex pipelines for batch jobs.
- Lightweight

### Metaflow

Metaflow is a powerful and modern workflow management tool built for demanding data science and machine learning projects. It simplifies and speeds up the implementation and management

of data science projects. We can generate models using any Python-related tool. This framework also has support for the R language.

You can design your workflow, run it at scale, and deploy it to production. Metaflow has automatic versioning and tracking for all experiments and data. There's built-in support to scale quickly and easily. It's integrated with AWS cloud, which provides support for storage, computation, and machine learning services.

Source: [Metaflow](#)

It has a unified API to the infrastructure, essential to execute data science projects from start to deployment. It focuses on usability and ergonomics. There's also code entropy management and a collaboration platform.

Functionality:

- Open-source
- Focus on: manage real-life data science projects
- Lightweight

## Couler

[Couler](#) is the only workflow orchestration tool for managing other workflow orchestration tools. Couler has a state-of-the-art unified interface for coding and managing workflows with different workflow engines and frameworks.

Different engines, like Argo Workflows, Tekton Pipelines or Apache Airflow, have varying, complex levels of abstractions. Couler's common interface makes it easier to manage these different levels of abstractions.



Source: [Couler on GitHub](#)

It has an imperative programming style for defining workflows, and support for automatic construction of a directed acyclic graph. Couler services are highly extensible, supporting other workflow engines. It facilitates distributed training for ML models, ensuring modularity and reusability. Couler supports automated workflows and resource organization for optimal performance.

Functionality:

- Open-source
- Focus on: management of other tools
- Lightweight

## Valohai

If you're looking for an MLOps tool to automate everything from data extraction and preparation, to model deployment in production, then [Valohai](#) might become your new favorite.

With Valohai, you can train, evaluate, and deploy models conveniently, without added manual work, and also repeat the process automatically. It supports an end-to-end machine learning workflow storing every model, experiment, and artifact automatically. After deployment, it also monitors deployed models in the Kubernetes cluster.

Valohai is a stable environment and UI interface, with just enough computing resources. You can manage your custom model instead of spending time on infrastructure and manual experiment tracking. It speeds up your work, and also supports automatic data, model, and experiment versioning.

The screenshot shows the Valohai web interface for a user named 'juha' in a project named 'nb9'. The interface includes a navigation bar with tabs for '# Executions', 'Tasks', 'Data', 'Deployment', and 'Settings'. A 'Fetch repository' button is visible in the top right. Below the navigation, there are buttons for 'Stop', 'Delete', and 'Compare'. A 'Create execution' button is also present. The main content is a table of notebook executions with the following columns: Execution, Environment, Created at, Duration, # Notes, Status, Actions, and accuracy. The table contains 7 rows of data, with the most recent execution (#7) being 'Started' and the others being 'Complete' or 'Error'.

Execution	Environment	Created at	Duration	# Notes	Status	Actions	accuracy
#7 Notebook execution	AWS eu-west-1 g2.2xlarge	a few seconds ago	-	0	Started	Copy	0.9
#6 Notebook execution	AWS eu-west-1 g2.2xlarge	2 minutes ago	2 minutes	0	Complete	Copy	0.9
#5 Notebook execution	AWS eu-west-1 g2.2xlarge	3 minutes ago	7 seconds	0	Error	Copy	
#4 Notebook execution	AWS eu-west-1 g2.2xlarge	3 minutes ago	2 minutes	0	Complete	Copy	0.9
#3 Notebook execution	AWS eu-west-1 g2.2xlarge	4 minutes ago	59 seconds	0	Complete	Copy	0.86
#2 Notebook execution	AWS eu-west-1 g2.2xlarge	4 minutes ago	59 seconds	0	Complete	Copy	0.85
#1 Notebook execution	AWS eu-west-1 g2.2xlarge	5 minutes ago	59 seconds	0	Complete	Copy	0.85

Source: [Valohai](#)

It includes a very stable MLOps environment that adapts to any framework and language. It can do hyperparameter sweeps, simplify team collaboration, experiment and model auditing, and it's secure with a firewall.

Functionality:

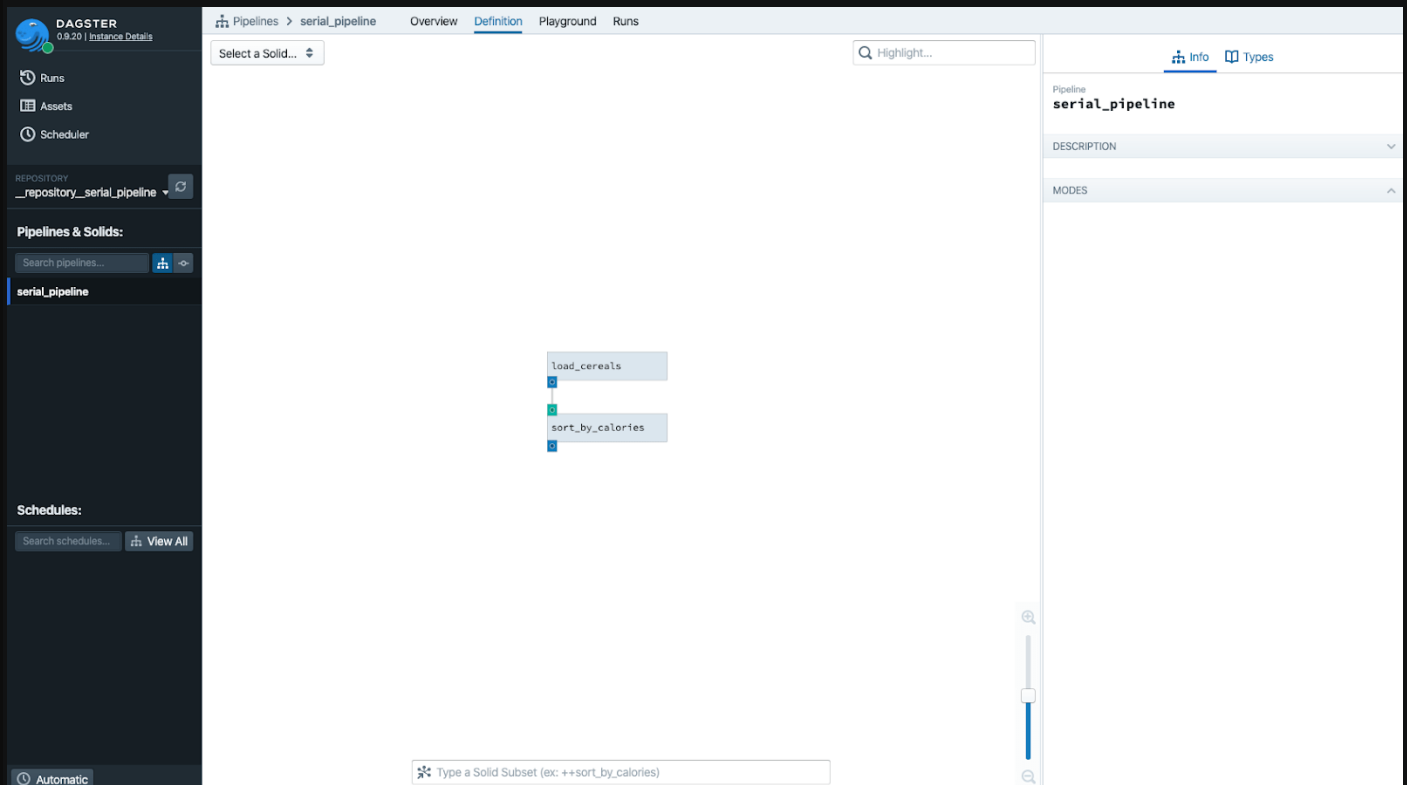
- Premium, no free plans
- Focus on: end-to-end ML pipelines.
- Lightweight

# Dagster

[Dagster](#) has a rich UI to perform workflow orchestration for machine learning, analytics, and ETL (Extract, Transform, Load).

You can build computation pipelines written in Spark, SQL, DBT, or any other framework. The platform lets you deploy the pipeline locally, or on Kubernetes. You can even create your own custom infrastructure for deployment.

Dagster shows you pipelines, tables, ML models, and other assets in a unified view. It provides an asset manager tool for tracking workflow results. It lets teams build custom self-service systems.



Source: [Dagster](#)

The web interface (Dagit) lets anyone inspect created task objects, and explore their properties. It eases the dependency nightmare. The codebases are isolated by repository models, preventing the problem of one workflow affecting another.

Functionality:

- Premium, no free plans
- Focus on: end-to-end ML pipelines.

# Netflix Genie

[Genie](#) is an open-source distributed workflow/task orchestration framework. It has APIs for executing different machine learning big data tasks, like Hadoop, Pig, or Hive. It offers centralized and scalable resource management for computing resources.

There are APIs for monitoring workflows on clusters, without installing any computational resources. It takes away the manual work of having to install computation resources yourself. It servers configurations APIs to register clusters and applications that run Genie.

GENIE Q Job Id: SPCS.FCT\_TICKET\_0054500815

[/](#) / [genie](#) / [applications](#) / [spark161](#) / [dependencies](#) / [spark-1.6.1](#)

Name	Size	Last Modified (UTC)
📁 <a href="#">R/</a>	--	12/09/2016, 1:07:43
📁 <a href="#">bin/</a>	--	12/09/2016, 1:07:43
📁 <a href="#">conf/</a>	--	12/21/2016, 4:13:13
📁 <a href="#">lib/</a>	--	12/09/2016, 1:07:58
📁 <a href="#">licenses/</a>	--	12/09/2016, 1:07:43
📁 <a href="#">python/</a>	--	12/09/2016, 1:07:43
📁 <a href="#">sbin/</a>	--	12/09/2016, 1:07:43
📄 <a href="#">RELEASE</a>	247 B	12/09/2016, 1:07:43

1 File(s), 7 Folder(s)

Source: [Netflix Genie on GitHub](#)

Genie's major advantage is scalability. It can house multiple machines depending on increasing and decreasing workload. The server APIs manage the metadata and commands of many distributed processing clusters.

Functionality:

- Open-source
- Focus on: big Data orchestration.
- Lightweight

## Conclusion

Now that you know some of the best MLOps workflow/pipeline orchestration tools, you can choose the right one for your ML project. Each tool has distinct advantages. Most of them are open-source, so you can test them without any financial commitment.

Automatic processes, scalability, unified design, global plugin integrations, and much more – the features that these tools provide make it easier to drive stellar results in machine learning projects. From the initial process of data extraction and experiment to deployment in production, these tools can make things easier and more accurate.

## What next?

If you're here, you're probably building or updating your MLOps stack. So here are a few resources you might look into next.

Best **tools** for other components of the ML pipeline:

- [The Best MLOps Tools and How to Evaluate Them](#)
- [15 Best Tools for ML Experiment Tracking and Management](#)
- [Best Tools for Model Tuning and Hyperparameter Optimization](#)

- [Best 8 Machine Learning Model Deployment Tools](#)
- [Best Tools to Do ML Model Monitoring](#)

Deeper **comparisons** between different workflow or pipeline orchestration tools:

- [Kedro vs ZenML vs Metaflow: Which Pipeline Orchestration Tool Should You Choose?](#)
- [Argo vs Airflow vs Prefect: How Are They Different?](#)

Real-world **examples** of how others built their MLOps:

- [Real-World MLOps Examples: End-To-End MLOps Pipeline for Visual Search at Brainly](#)
- [Real-World MLOps Examples: Model Development in Hypefactors](#)
- [Building ML Pipeline: 6 Problems & Solutions \[From a Data Scientist's Experience\]](#)
- [This Is Our MLOps Tool Stack: Continuum Industries](#)

## References

- <https://github.com/kubeflow-kale/kale>
- <https://flyte.org/>
- <https://github.com/mlrun/mlrun>
- <https://github.com/mlrun/mlrun>
- <https://docs.prefect.io/>
- <https://github.com/maiot-io/zenml>
- <https://github.com/argoproj/argo>
- <https://github.com/quantumblacklabs/kedro>
- <https://github.com/spotify/luigi>
- <https://metaflow.org/>
- <https://github.com/couler-proj/couler>
- <https://valohai.com/>
- <http://dagster.io>
- <https://netflix.github.io/genie/>