Arrikto

From Notebook to Kubeflow Pipelines with HP Tuning

A Data Science Journey

A complete data science workflow for optimizing your models using Jupyter Notebooks, Kale, Katib, and Kubeflow Pipelines.

Stefano Fioravanzo
Ilias Katsakioris
Arrikto

From Notebook to Kubeflow Pipelines with HP Tuning

A Data Science Journey

Arrikto

Stefano Fioravanzo

Software Engineer

Ilias Katsakioris

Software Engineer

What You'll Learn In This Session

Arrikto

Run a pipeline-based hyperparameter tuning workflow starting from your Jupyter Notebook, with caching. Use Kale as a workflow tool to orchestrate Katib and Kubeflow Pipelines experiments.

Why is this important?

- ✓ **Simplify** your ML workflows using intuitive UIs
- ✓ ***Accelerate*** your ML lifecycle using Kale as an orchestration tool for Katib and Kubeflow Pipelines. Pipeline runs are now completing faster as the identical steps are cached
- ✓ **Collaborate** faster and more easily, and have complete visibility of your training

Don't forget, you can grab the slides right now at arrik.to/kubeconAMS as well as enter the draw to win a fabulous prize

*Get your questions answered **live** on Twitter and LinkedIn using the three hashtags* #kubecon #ml #arrikto

What is Kubeflow

Kubeflow

The Kubeflow project is dedicated to making deployments of machine learning (ML) workflows on Kubernetes: simple, portable and scalable.

Use cases

Arrikto

- Deployment and management of a complex ML system at scale
- Rapid experimentation
- Hyperparameter tuning
- Hybrid and multi-cloud workloads
- Continuous integration and deployment (CI/CD)

Arrikto

ML tools

- Chainer
- Jupyter
- MPI
- MXNet
- PyTorch
- scikit-learn
- TensorFlow
- XGBoost

Kubeflow applications and scaffolding

- Jupyter notebook web app and controller
- Kubeflow UI
- Training operators: MPI, MXNet, PyTorch, TFJob, XGBoost
- Hyperparameter tuning (Katib)
- Kale
- Metadata
- Pipelines
- KFServing

- PyTorch Serving
- TensorFlow Serving
- Seldon Core

- Istio
- Argo
- Prometheus
- Spartakus

Platforms / clouds

Kubernetes

- GCP
- AWS
- Azure
- IBM Cloud
- OpenShift
- On prem

Arrikto

ML tools

- Chainer
- Jupyter
- MPI
- MXNet
- PyTorch
- scikit-learn
- TensorFlow
- XGBoost

Kubeflow applications and scaffolding

- Jupyter notebook web app and controller
- Kubeflow UI
- Training operators: MPI, MXNet, PyTorch, TFJob, XGBoost
- Hyperparameter tuning (Katib)
- Kale
- Metadata
- Pipelines
- KFServing

- PyTorch Serving
- TensorFlow Serving
- Seldon Core

- Istio
- Argo
- Prometheus
- Spartakus

Platforms / clouds

Kubernetes

- GCP
- AWS
- Azure
- IBM Cloud
- OpenShift
- On prem

Arrikto

ML tools

- Chainer
- Jupyter
- MPI
- MXNet
- PyTorch
- scikit-learn
- TensorFlow
- XGBoost

Kubeflow applications and scaffolding

- Jupyter notebook web app and controller
- Kubeflow UI
- Training operators: MPI, MXNet, PyTorch, TFJob, XGBoost
- Hyperparameter tuning (Katib)
- Kale
- Metadata
- Pipelines
- KFServing

- PyTorch Serving
- TensorFlow Serving
- Seldon Core
- Istio
- Argo
- Prometheus
- Spartakus

Platforms / clouds

Kubernetes

- GCP
- AWS
- Azure
- IBM Cloud
- OpenShift
- On prem

Arrikto

ML tools

- Chainer
- Jupyter
- MPI
- MXNet
- PyTorch
- scikit-learn
- TensorFlow
- XGBoost

Kubeflow applications and scaffolding

- Jupyter notebook web app and controller
- Kubeflow UI
- Training operators: MPI, MXNet, PyTorch, TFJob, XGBoost
- Hyperparameter tuning (Katib)
- Kale
- Metadata
- Pipelines
- KFServing

- PyTorch Serving
- TensorFlow Serving
- Seldon Core

- Istio
- Argo
- Prometheus
- Spartakus

Platforms / clouds

Kubernetes

- GCP
- AWS
- Azure
- IBM Cloud
- OpenShift
- On prem

ML workflow

Arrikto

Identify problem and collect and analyse data	Choose an ML algorithm and code your model	Experiment with data and model training	Tune the model hyperparameters	Serve the model for online/batch prediction
Jupyter Notebook	PyTorch	Jupyter Notebook	Katib	KFServing
	scikit-learn	Kale		NVIDIA TensorRT
	TensorFlow	Pipelines		PyTorch
	XGBoost			TFServing
				Seldon Core

Kubeflow

Interacting with Kubeflow

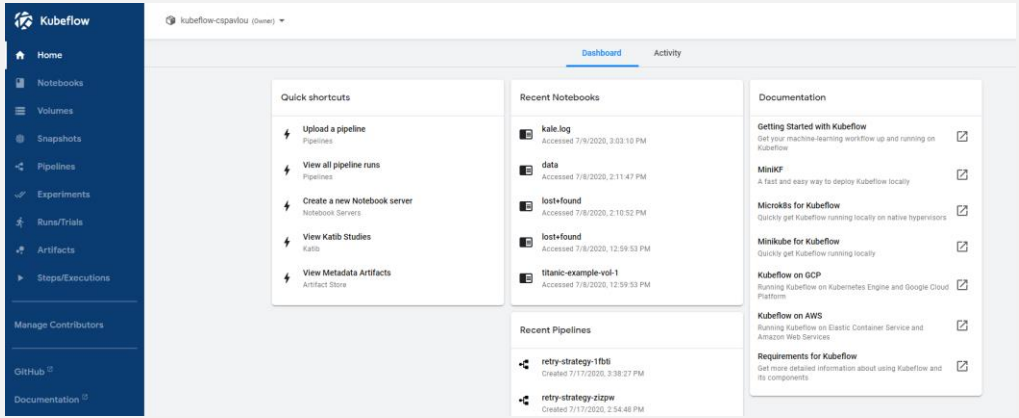
Arrikto

User interface (UI) →

kfctl CLI

kubectl CLI

APIs and SDKs



Kubeflow

kubeflow-cspavlou (Owner)

- Home
- Notebooks
- Volumes
- Snapshots
- Pipelines
- Experiments
- Runs/Trials
- Artifacts
- Steps/Executions

Manage Contributors

GitHub

Documentation

Dashboard | Activity

Quick shortcuts

- Upload a pipeline — Pipelines
- View all pipeline runs — Pipelines
- Create a new Notebook server — Notebook Servers
- View Katib Studies — Katib
- View Metadata Artifacts — Artifact Store

Recent Notebooks

- kale.log — Accessed 7/9/2020, 3:03:10 PM
- data — Accessed 7/8/2020, 2:11:47 PM
- lost+found — Accessed 7/8/2020, 2:10:52 PM
- lost+found — Accessed 7/8/2020, 12:59:53 PM
- titanic-example-vol-1 — Accessed 7/8/2020, 12:59:53 PM

Recent Pipelines

- retry-strategy-1fbti — Created 7/17/2020, 3:38:27 PM
- retry-strategy-zizpw — Created 7/17/2020, 2:54:48 PM

Documentation

- **Getting Started with Kubeflow** — Get your machine-learning workflow up and running on Kubeflow
- **MiniKF** — A fast and easy way to deploy Kubeflow locally
- **Microk8s for Kubeflow** — Quickly get Kubeflow running locally on native hypervisors
- **Minikube for Kubeflow** — Quickly get Kubeflow running locally
- **Kubeflow on GCP** — Running Kubeflow on Kubernetes Engine and Google Cloud Platform
- **Kubeflow on AWS** — Running Kubeflow on Elastic Container Service and Amazon Web Services
- **Requirements for Kubeflow** — Get more detailed information about using Kubeflow and its components

Interacting with Kubeflow

Arrikto

User interface (UI)

kfctl CLI → `kfctl apply -V -f ${CONFIG_URI}`

kubectl CLI → `kubectl -n kubeflow get all`

APIs and SDKs

Interacting with Kubeflow

Arrikto

User interface (UI)

kfctl CLI

kubectl CLI

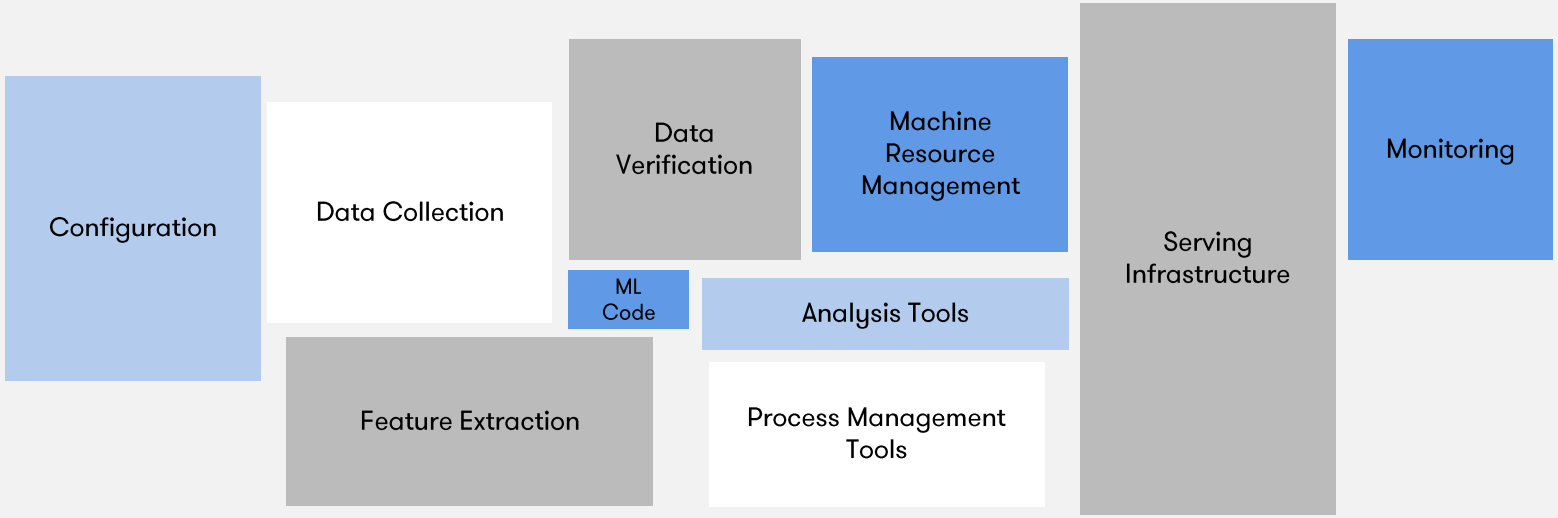
APIs and SDKs →

Examples:

- Pipelines SDK
- Katib API
- Metadata SDK

ML Applications are Distributed Systems

Arrikto



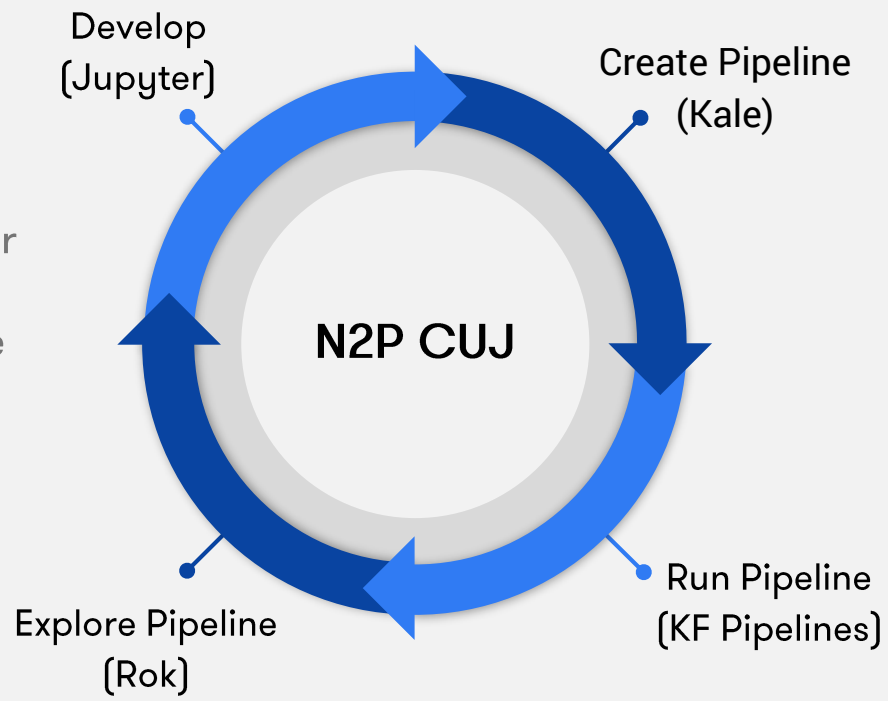
Credit: Hidden Technical Debt of Machine Learning Systems, D. Sculley, et al.

CI/CD for ML

Arrikto

How can data scientists continually improve and validate models?

- Develop models and pipelines in Jupyter
- Convert notebook to pipeline using Kale
- Run pipeline using Kubeflow Pipelines
- Explore and debug pipeline using Rok



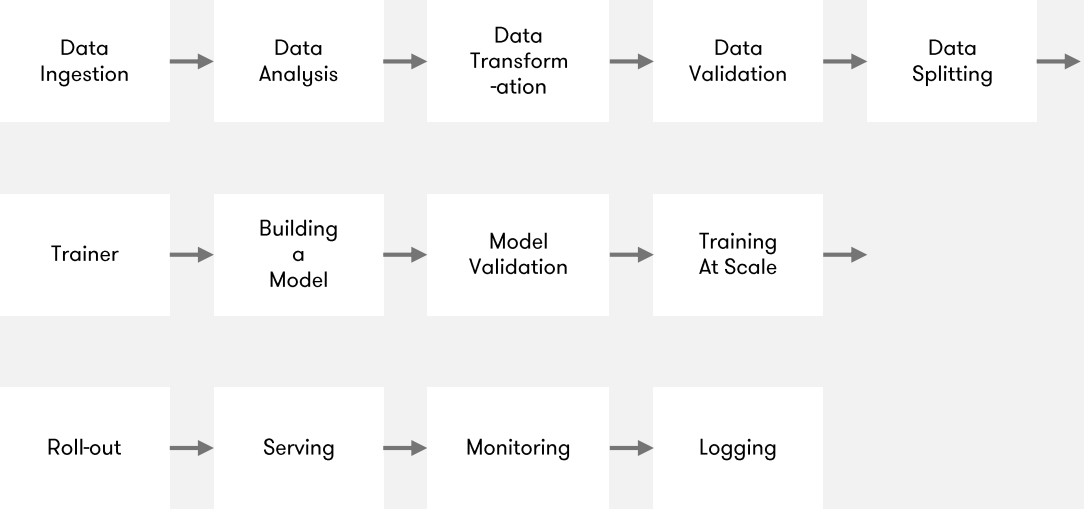
Data Science with Kubeflow

Arrikto

Kubeflow Pipelines exists because Data Science and ML are inherently **pipeline processes**

This workshop will focus on two essential aspects:

- **Low barrier to entry:** deploy a Jupyter Notebook to Kubeflow Pipelines in the Cloud using a fully GUI-based approach
- **Reproducibility:** automatic data versioning to enable reproducibility and better collaboration between data scientists



Data Science with Kubeflow

Arrikto

Kubeflow Pipelines exists because Data Science and ML are inherently **pipeline processes**

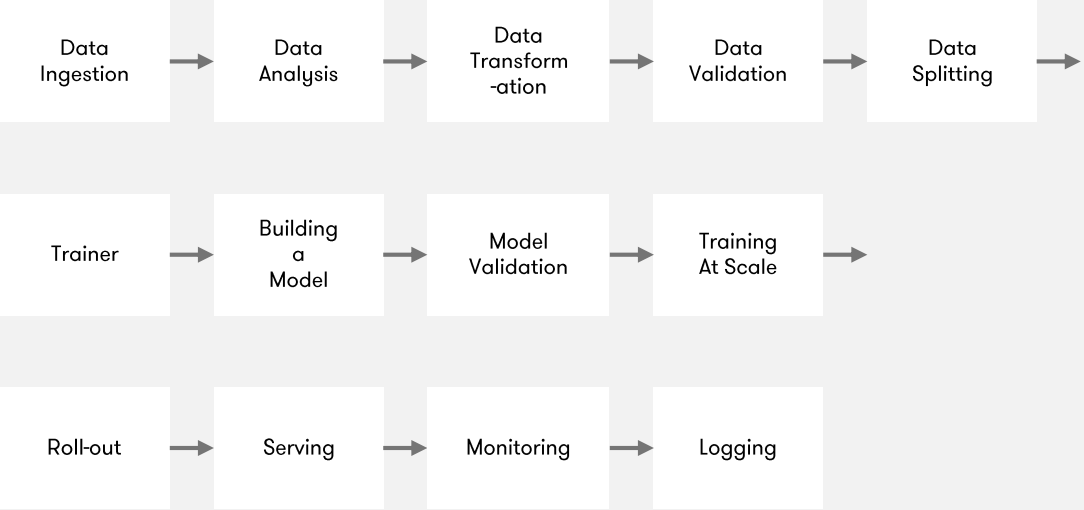
This workshop will focus on two essential aspects:

- **Low barrier to entry:** deploy a Jupyter Notebook to Kubeflow Pipelines in the Cloud using a fully GUI-based approach

Kale

- **Reproducibility:** automatic data versioning to enable reproducibility and better collaboration between data scientists

Arrikto



Benefits of running a Notebook as a Pipeline

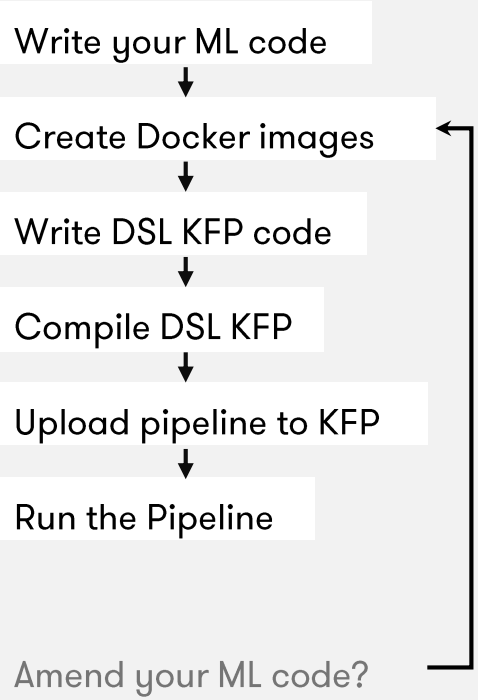
Arrikto

- The steps of the workflow are clearly defined
- Parallelization & isolation
  - Hyperparameter tuning
- Data versioning
- Different infrastructure requirements
  - Different hardware (GPU/CPU)

Workflow

Arrikto

Before



Workflow

Arrikto

Before

Write your ML code

↓

Create Docker images

↓

Write DSL KFP code

↓

Compile DSL KFP

↓

Upload pipeline to KFP

↓

Run the Pipeline

Amend your ML code?

After

Write your ML code

↓

Tag your Notebook cells

↓

Run the Pipeline at the click of a button

Amend your ML code? → Just edit your Notebook!

Workflow

Arrikto

Before

Write your ML code
↓
Create Docker images
↓
Write DSL KFP code
↓
Compile DSL KFP
↓
Upload pipeline to KFP
↓
Run the Pipeline

Amend your ML code?

After

Write your ML code
↓
Tag your Notebook cells
↓
Run the Pipeline at the click of a button

A Data Scientist can now reduce the time taken to write ML code and run a pipeline by 70%.

That means you can now run 3x as many experiments as you did before.

What that really means is that you can deliver work faster to the business and drive more revenue

Amend your ML code? → Just edit your Notebook!

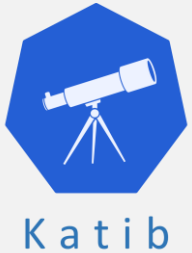
Hyperparameter optimization

Arrikto

The two ways of life

- Change the parameters manually
- Use Katib

What is Katib



Katib is a Kubernetes-based system for Hyperparameter Tuning and Neural Architecture Search. It supports a number of ML frameworks, including TensorFlow, Apache MXNet, PyTorch, XGBoost, and others.

Hyperparameter optimization

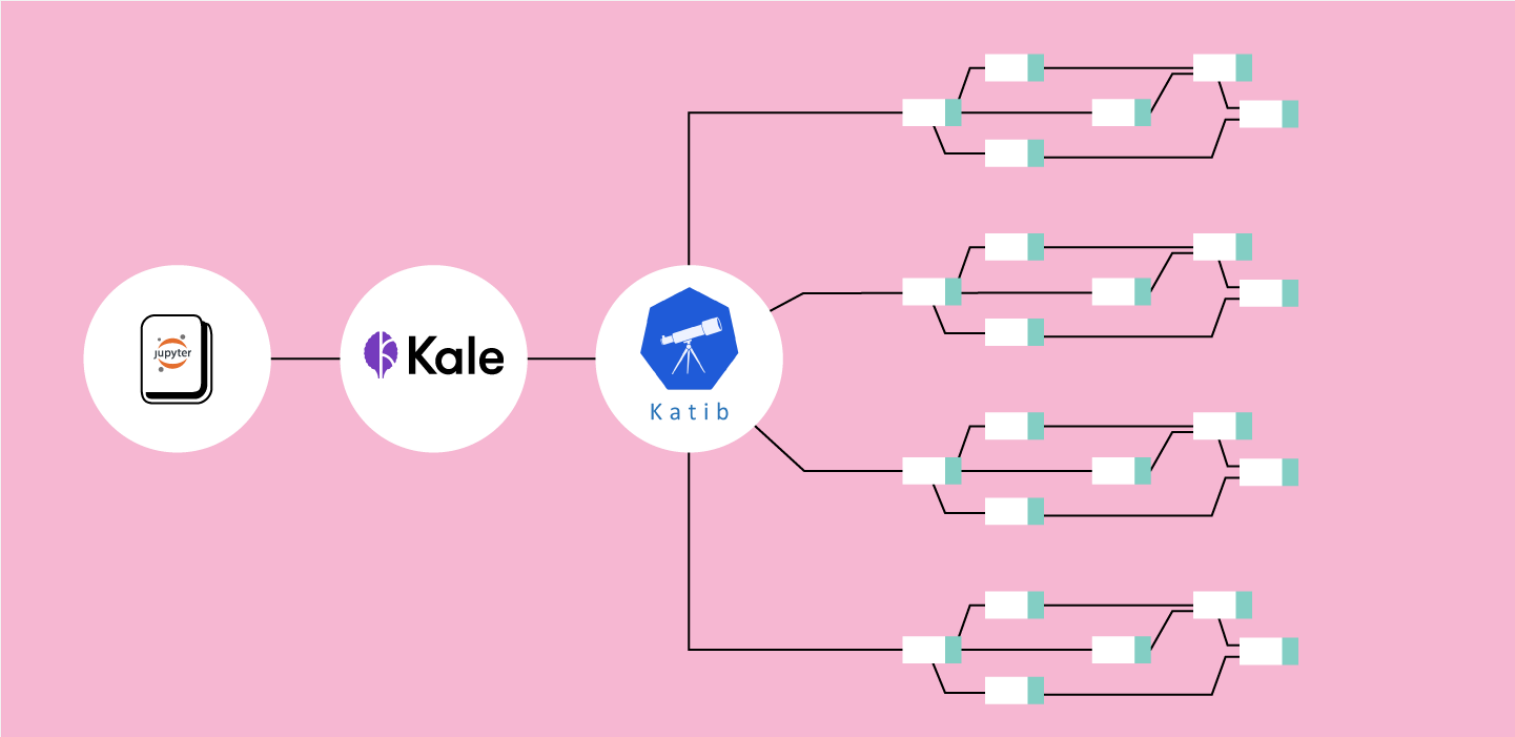
Arrikto

Combining the N2P CUJ with Katib

- Configure parameters, search algorithm, and objectives using a GUI
- Start HP tuning with the click of a button
- Reproducibility of every pipeline and every step
- Run Katib Trials as Pipelines
- Complete visibility of every different Katib Trial
- Caching for faster computation

A data science journey

Arrikto



Agenda

Arrikto

Go to arrik.to/demowfhp to find the Codelab with the step-by-step instructions for this tutorial

1

Install MiniKF

2

Explore the ML code of the dog breed identification example

3

Convert notebook to a Kubeflow pipeline

4

Explore the accuracy of the various models

5

Optimize a model with hyperparameter tuning

6

Clean up

Agenda

Arrikto

1

Install MiniKF

2

Explore the ML code of the dog breed identification example

3

Convert notebook to a Kubeflow pipeline

4

Explore the accuracy of the various models

5

Optimize a model with hyperparameter tuning

6

Clean up

What is MiniKF

- Kubeflow on GCP, your laptop, or on-prem infrastructure in just a few minutes
- All-in-one, single-node, Kubeflow distribution
- Very easy to spin up on your own environment on-prem or in the cloud
- MiniKF = MiniKube + Kubeflow + Arrikto's Rok Data Management Platform

Arrikto Rok

Arrikto

KDD 2017 Applied Data Science Paper

KDD'17, August 13–17, 2017, Halifax, NS, Canada

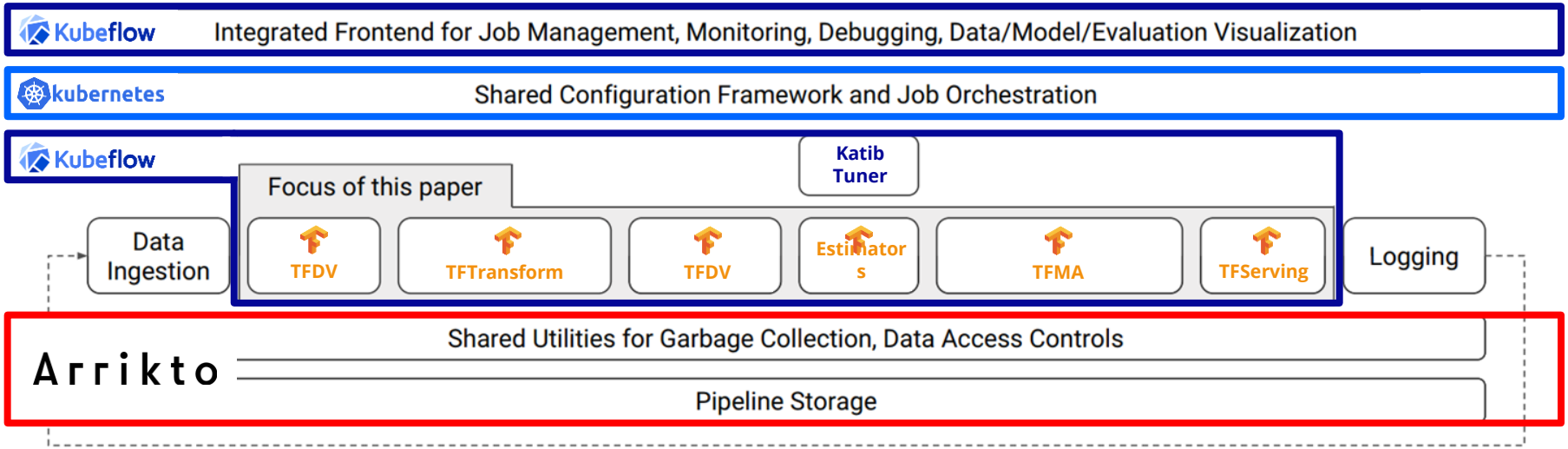


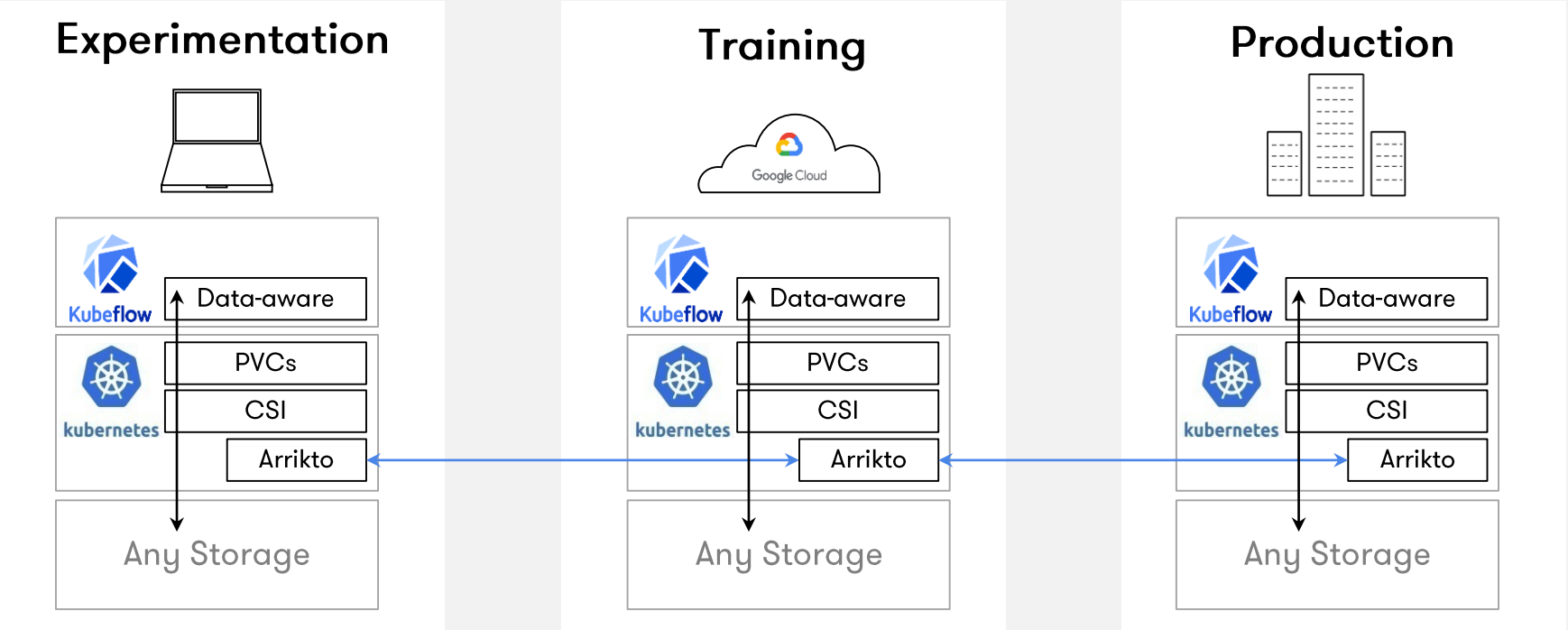
Figure 1: High-level component overview of a machine learning platform.

Arrikto Rok

Arrikto

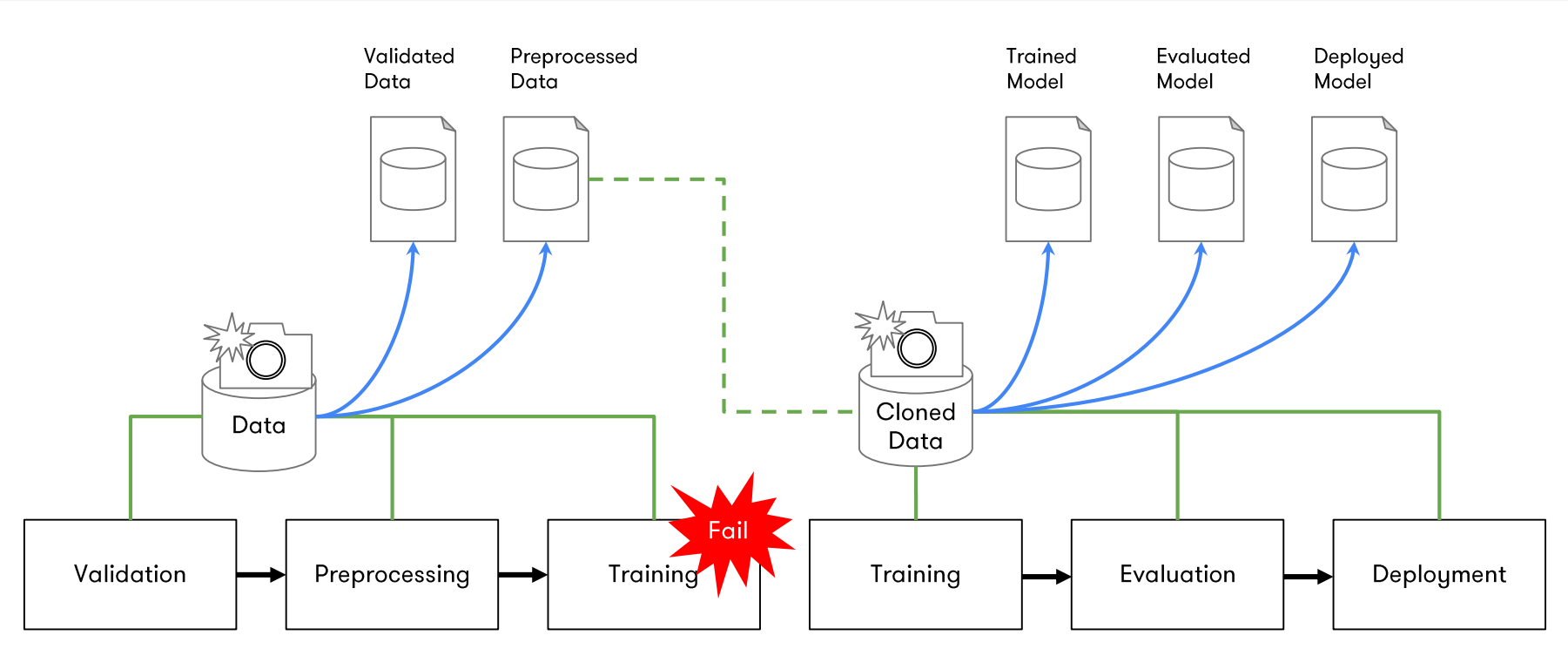
Data Versioning, Packaging, and Sharing

Across teams and cloud boundaries for complete Reproducibility, Provenance, and Portability



Arrikto Rok

Arrikto



Agenda

Arrikto

1

Install MiniKF

2

Explore the ML code of the dog breed identification example

3

Convert notebook to a Kubeflow pipeline

4

Explore the accuracy of the various models

5

Optimize a model with hyperparameter tuning

6

Clean up

Agenda

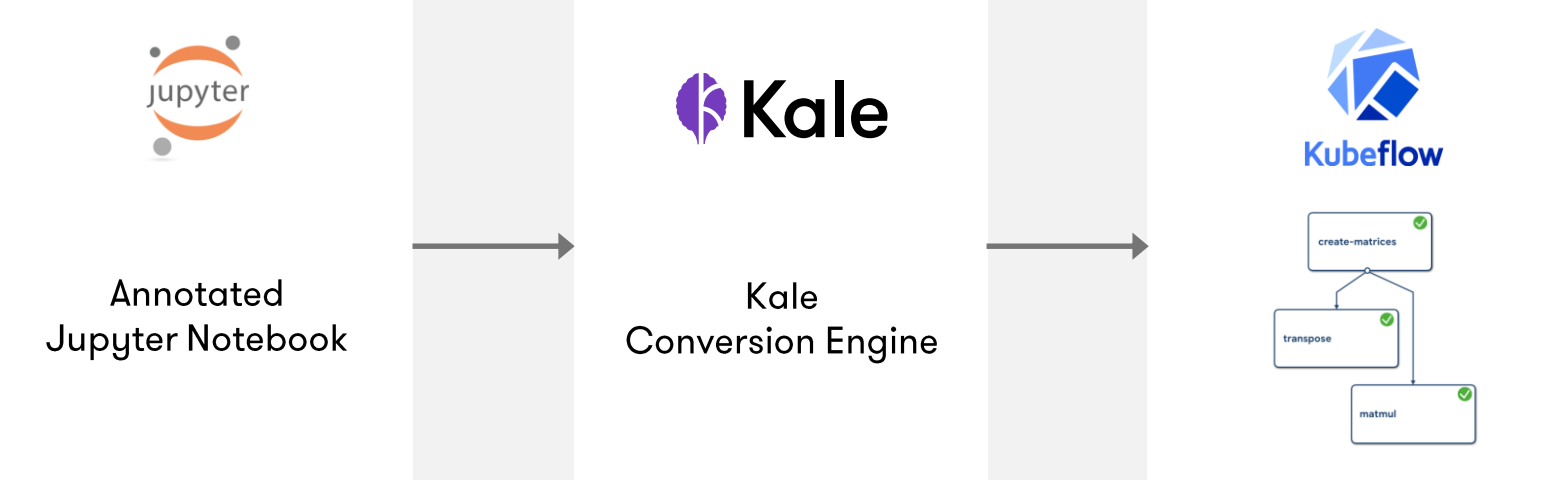
Arrikto

1. Install MiniKF
2. Explore the ML code of the dog breed identification example
3. Convert notebook to a Kubeflow pipeline
4. Explore the accuracy of the various models
5. Optimize a model with hyperparameter tuning
6. Clean up

KALE – Kubeflow Automated Pipelines Engine

Arrikto

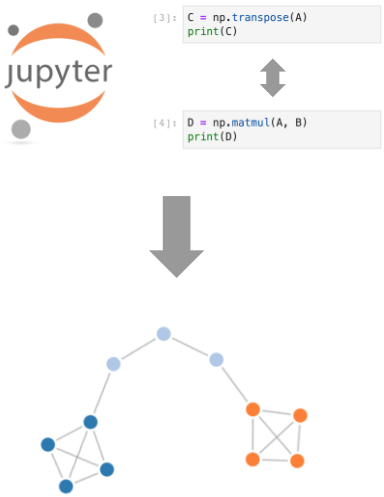
- Python package + JupyterLab extension
- Convert a Jupyter Notebook to a KFP workflow
- No need for Kubeflow SDK



Kale Modules

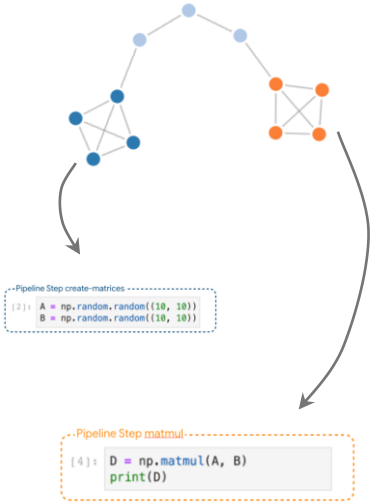
Arrikto

nbparser



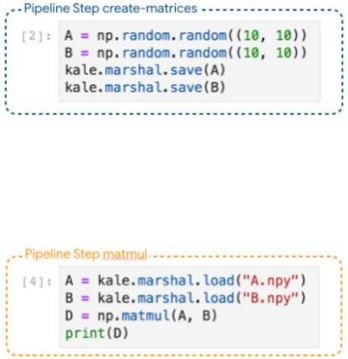
Derive pipeline structure

static_analyzer



Identify dependencies

marshal



Inject data objects

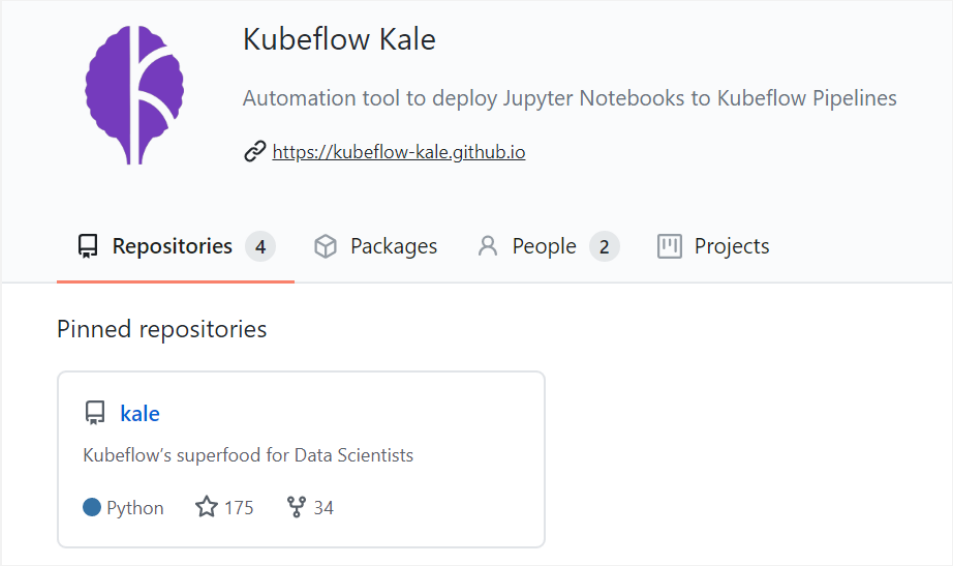
codegen

```
def {{ function_name }}({{ function_args|join(', ') }}):

    from kale.converter.odo import resource_save, resource_load

    _odo_data_directory = "/data/{{ pipeline_name }}/_odo_data/"
    _input_data_folder = "/data/{{ pipeline_name }}/"

    # ------------------------DATA LOADING------------------------
{%- for in_var in in_variables %}

    [...]

    {{ in_var }} = resource_save(
        _odo_data_directory + _odo_load_file_name)
{%- endfor %}
    # ------------------------DATA LOADING------------------------

{% for block in function_blocks %}
{{block|indent(4, True)}}
{% endfor %}

    # ------------------------DATA SAVING------------------------
{%- for out_var in out_variables %}

    [...]

    resource_load(
        {{ out_var }}, _odo_data_directory + "{{ out_var }}")
{%- endfor %}
    # ------------------------DATA SAVING------------------------
```

Generate & deploy pipeline

Contribute

Arrikto



github.com/kubeflow-kale

Agenda

Arrikto

1. Install MiniKF
2. Explore the ML code of the dog breed identification example
3. Convert notebook to a Kubeflow pipeline
4. Explore the accuracy of the various models
5. Optimize a model with hyperparameter tuning
6. Clean up

Agenda

Arrikto

1. Install MiniKF
2. Explore the ML code of the dog breed identification example
3. Convert notebook to a Kubeflow pipeline
4. Explore the accuracy of the various models
5. Optimize a model with hyperparameter tuning
6. Clean up

Agenda

Arrikto

1

Install MiniKF

2

Explore the ML code of the dog breed identification example

3

Convert notebook to a Kubeflow pipeline

4

Explore the accuracy of the various models

5

Optimize a model with hyperparameter tuning

6

Clean up

Summary

Arrikto

What have we achieved in this tutorial?

- Run a pipeline-based hyperparameter tuning workflow starting from your Jupyter Notebook
- Use Kale as a workflow tool to orchestrate Katib and Kubeflow Pipelines experiments
- **Simplify** your ML workflows using intuitive UIs
- Exploit the caching feature so that you ***accelerate*** your pipeline runs
- **Collaborate** faster and more easily, and have complete visibility of your training

Just a small sample of community contributions

Arrikto

- Jupyter manager UI
- Pipelines volume support
- MiniKF
- Auth with Istio + Dex
- On-premise installation
- Linux Kernel

Community

Kubeflow is open

- Open community
- Open design
- Open source
- Open to ideas

Get involved

- github.com/kubeflow
- kubeflow.slack.com
- @kubeflow
- kubeflow-discuss@googlegroups.com
- Community call on Tuesdays

Thank You!

Arrikto

More Info
arrik.to/kubeconAMS

company/arrikto

Arrikto

Email Address:
stefano@arrikto.com
elikatsis@arrikto.com

Arrikto

Arrikto