Kubeflow

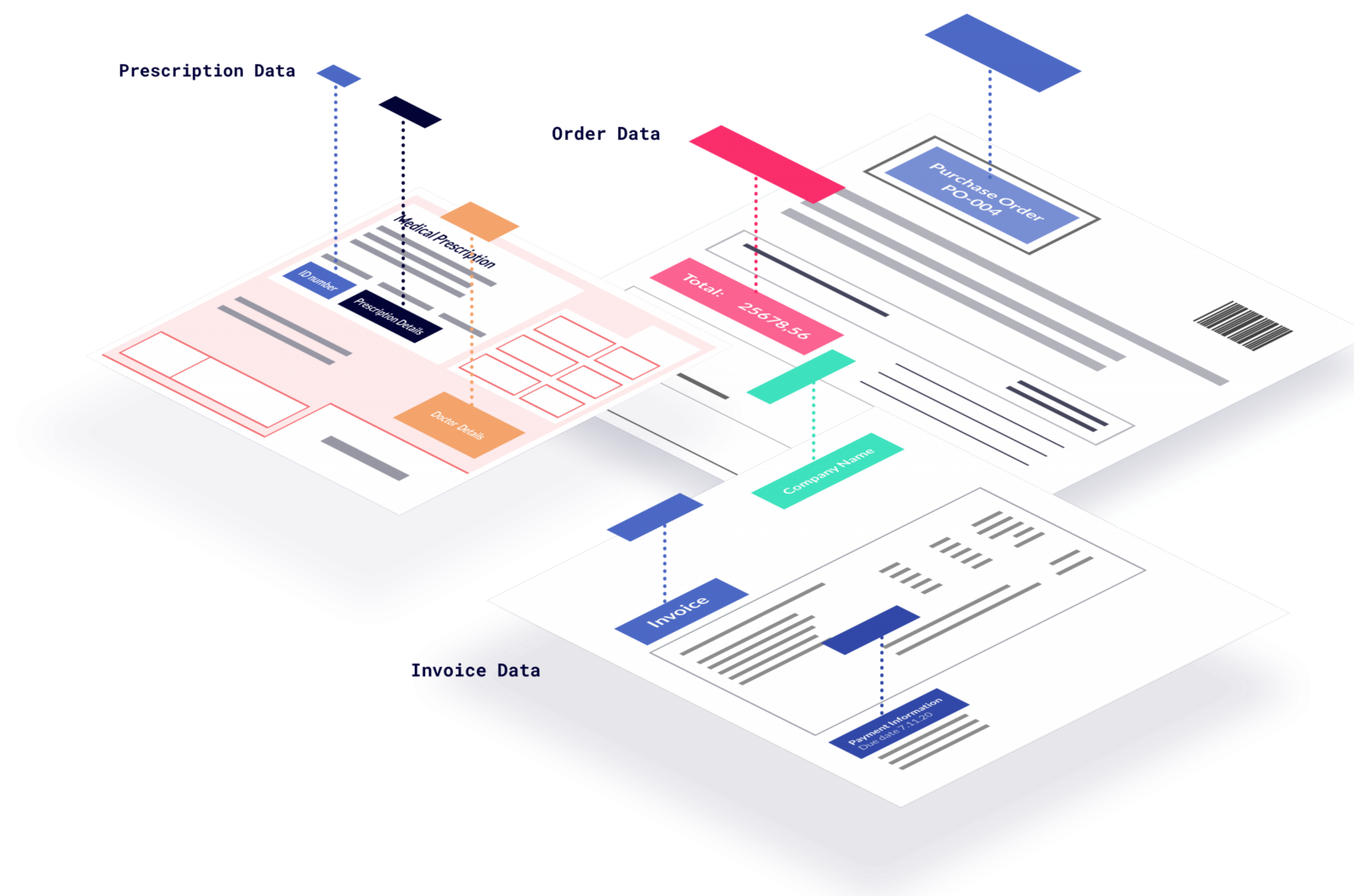
Wojciech Barczyński [Head of Engineering]
SMACC.io | Hypatos.ai

Wojtek Barczynski

- Software Developer
- System Engineer
- Head of Engineering at hypatos.ai and SMACC.io

Hypatos / SMACC.io

- Fintech Machine Learing
- Data capturing from document
- Validation
- Automation
- Deep learning



Hypatos / SMACC.io

Go

PYTORCH

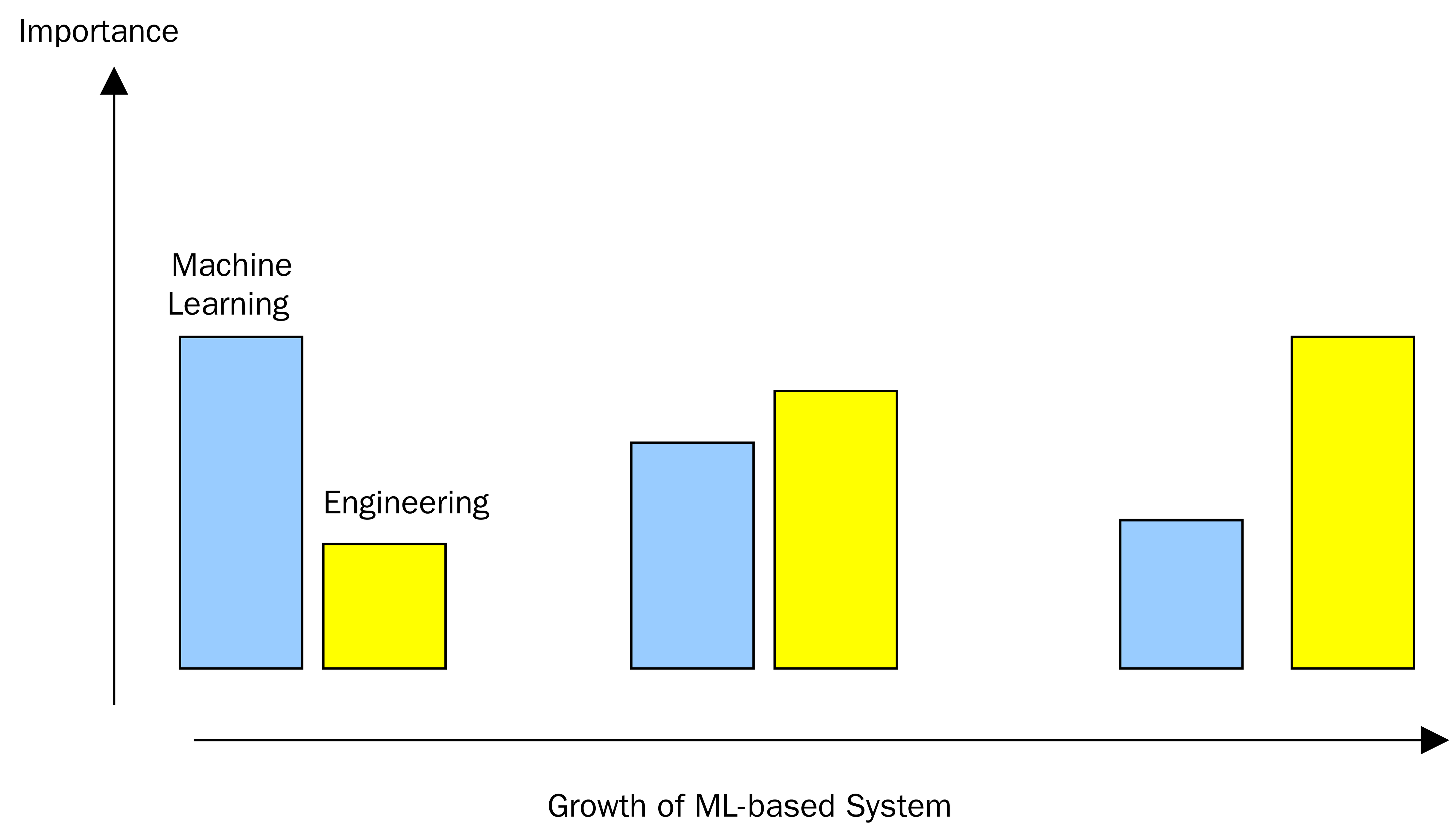
TensorFlow

amazon web services

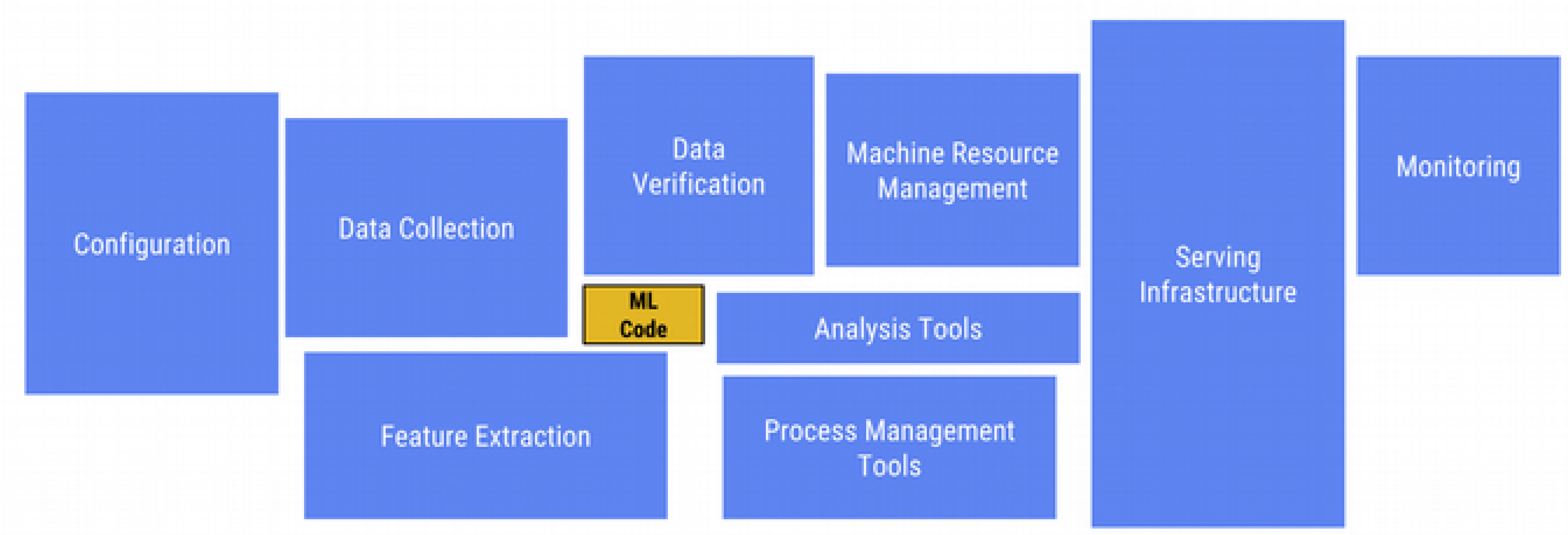
Azure

CloudNative: Prometheus, Grafana, Fluentd, Grafana Loki

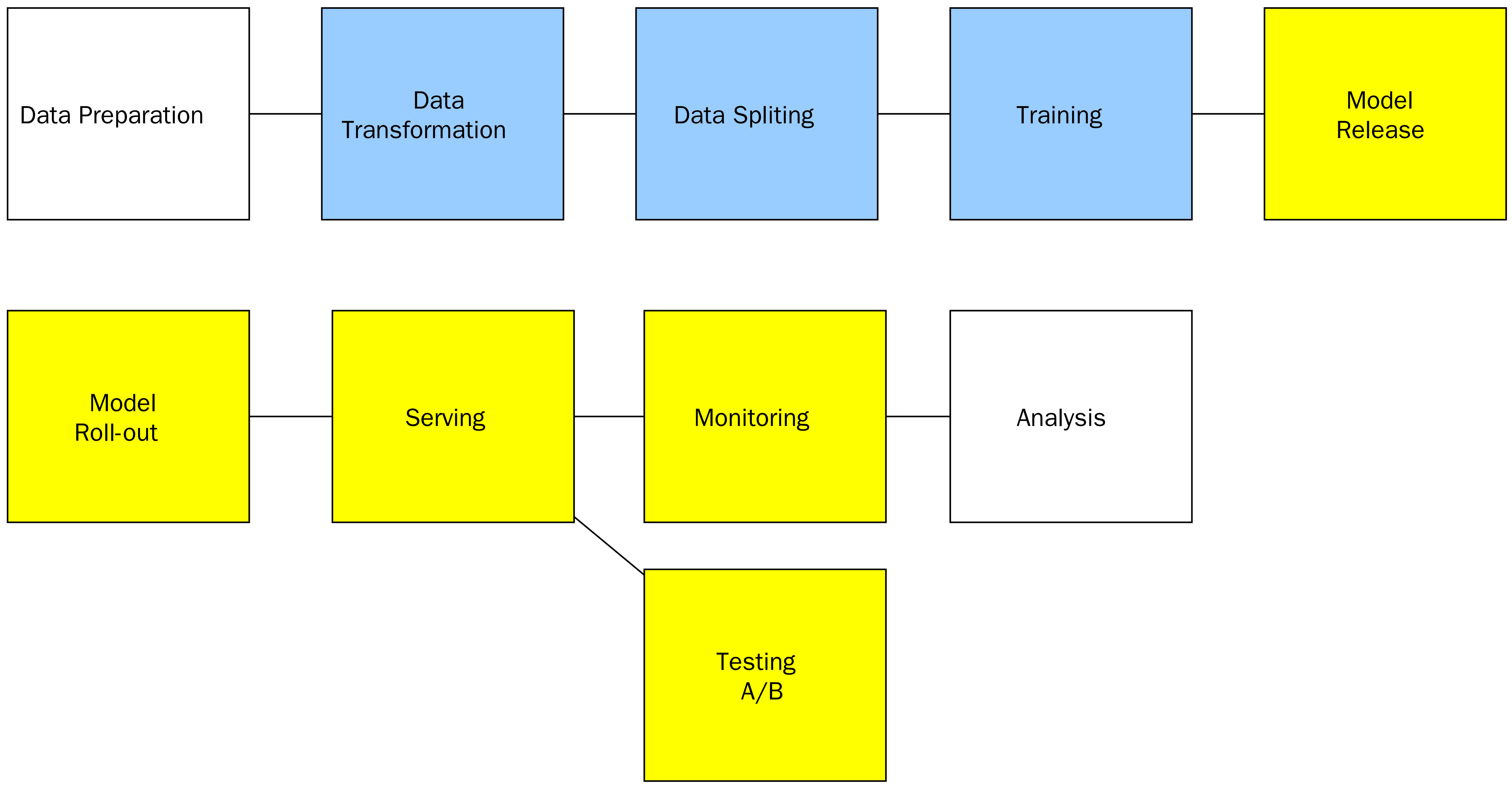
Machine Learning



Big Companies



Machine Learning



ML pipeline

Look a lot like
Continuous Integration / Deployment

ML pipeline

What did we lean from XX years of CD/CI?

What we leant from XX years of CD/CI?

Nobody likes it

Kubeflow

- Easy the pain
- Unified experience
- One to rule them all
- Low bar; High ceiling

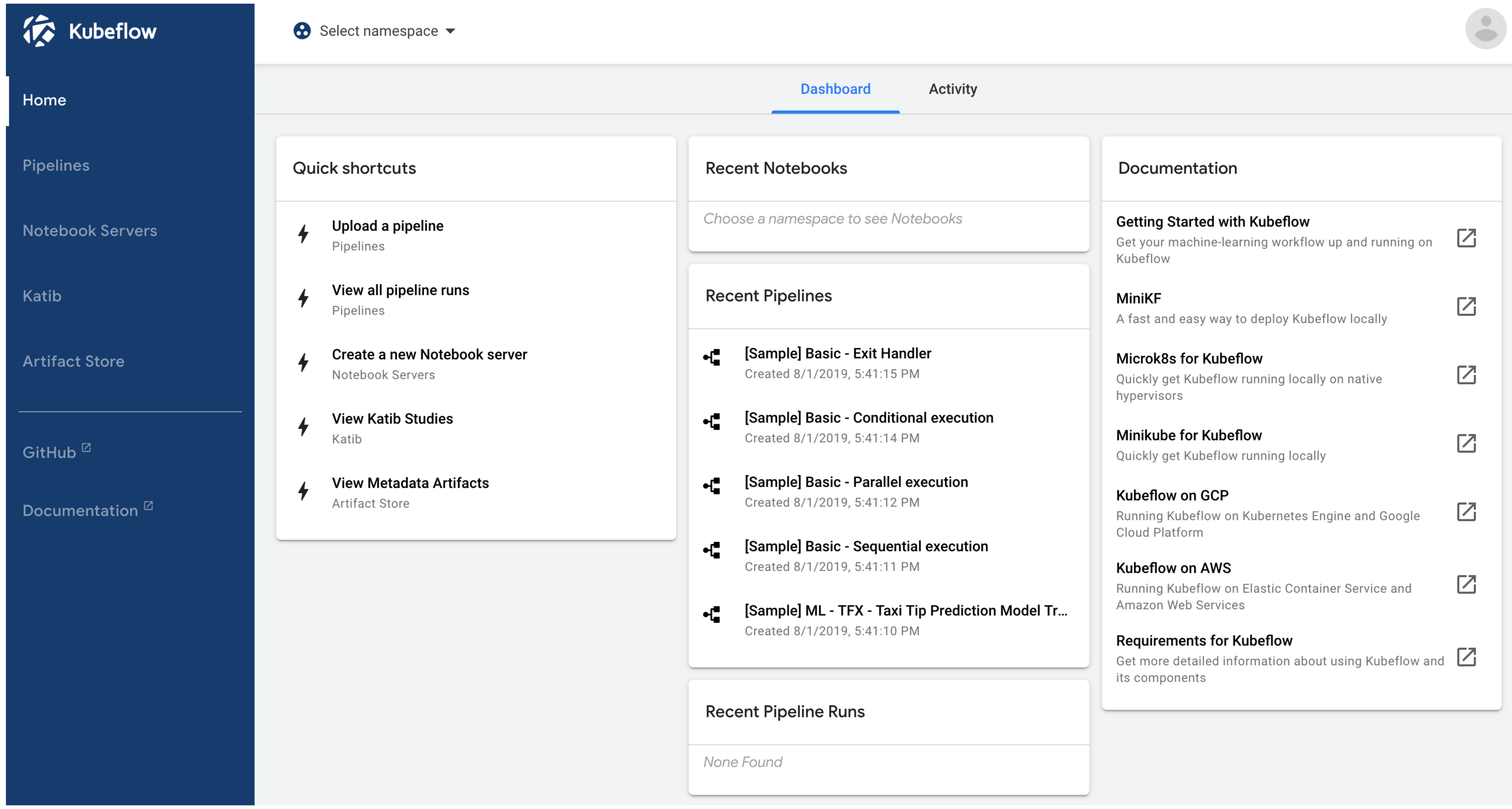
Focus

- Scalability
- Composition
- Portability

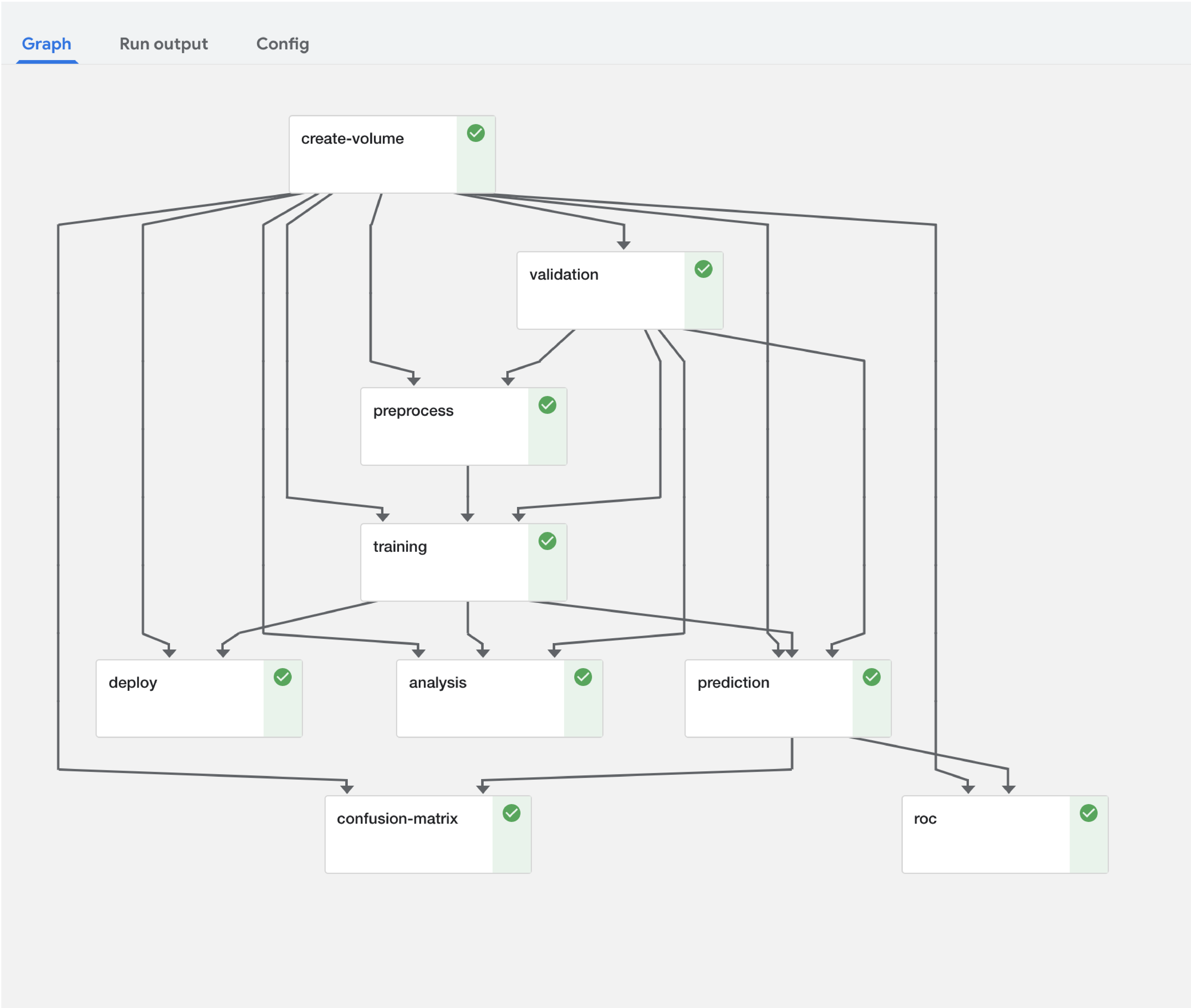
Focus

- Enable Dev(ML)ops culture

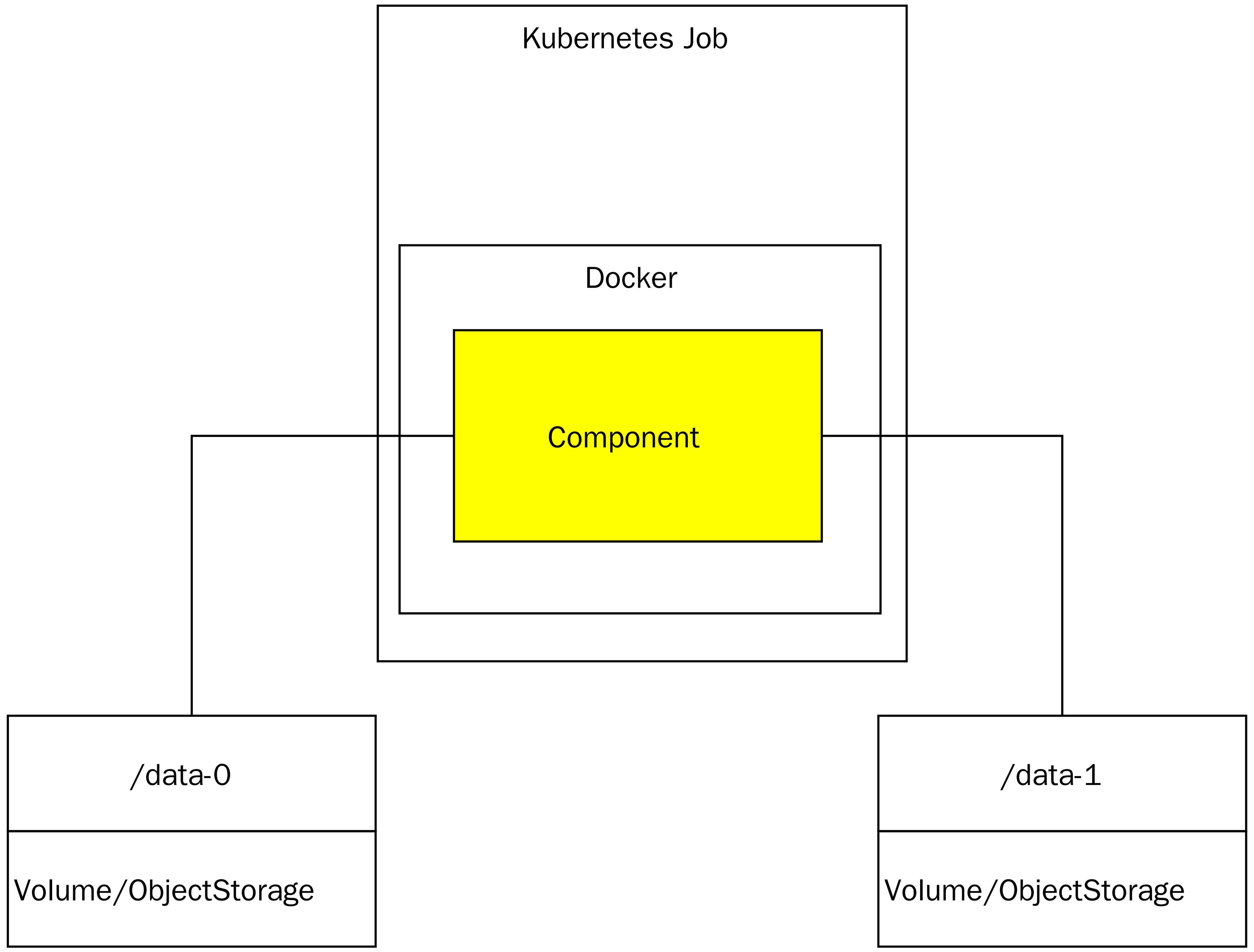
DEMO


Kubeflow
Select namespace
Home
Pipelines
Notebook Servers
Katib
Artifact Store
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
Choose a namespace to see Notebooks
Recent Pipelines
[Sample] Basic - Exit Handler
Created 8/1/2019, 5:41:15 PM
[Sample] Basic - Conditional execution
Created 8/1/2019, 5:41:14 PM
[Sample] Basic - Parallel execution
Created 8/1/2019, 5:41:12 PM
[Sample] Basic - Sequential execution
Created 8/1/2019, 5:41:11 PM
[Sample] ML - TFX - Taxi Tip Prediction Model Tr...
Created 8/1/2019, 5:41:10 PM
Recent Pipeline Runs
None Found
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

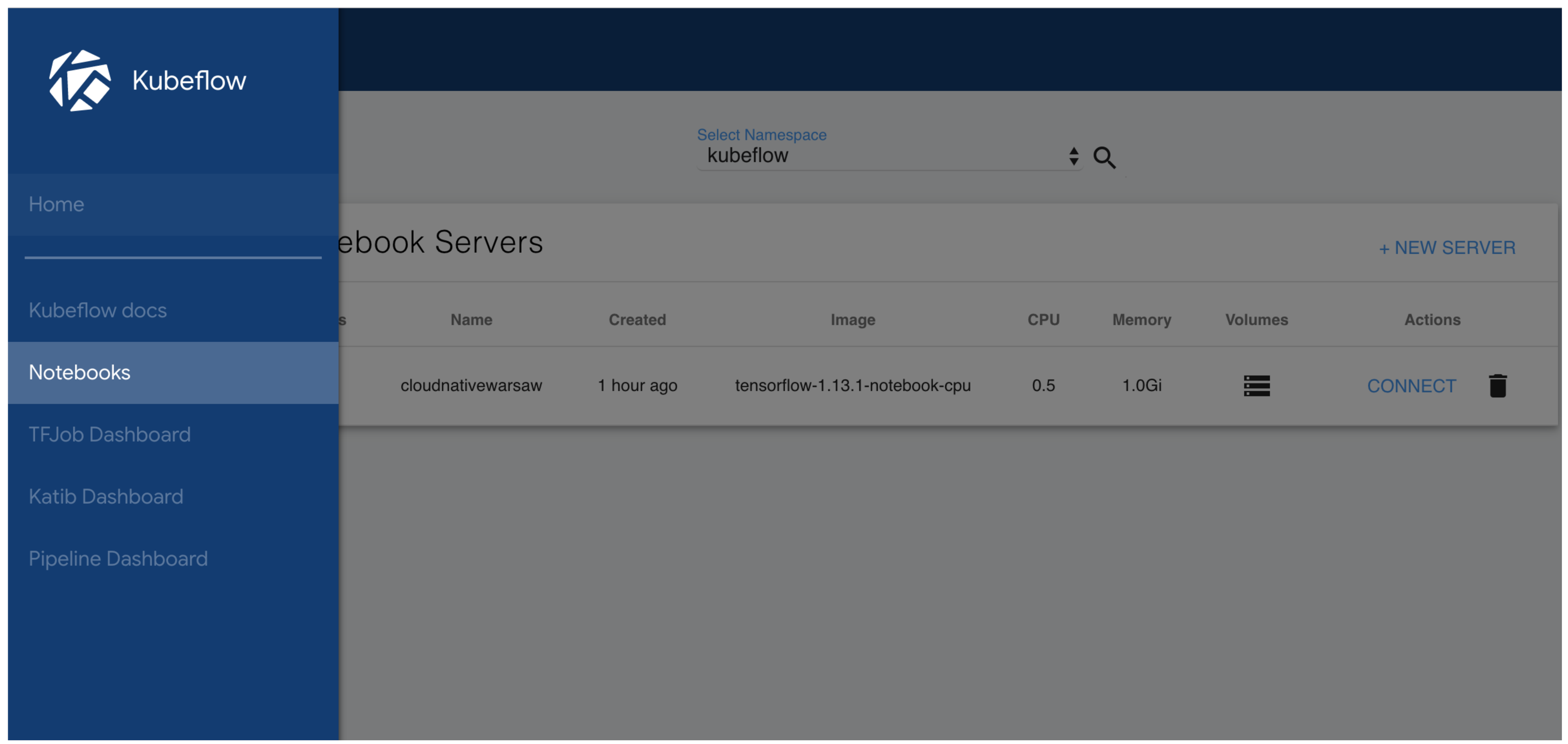
Heart: pipelines



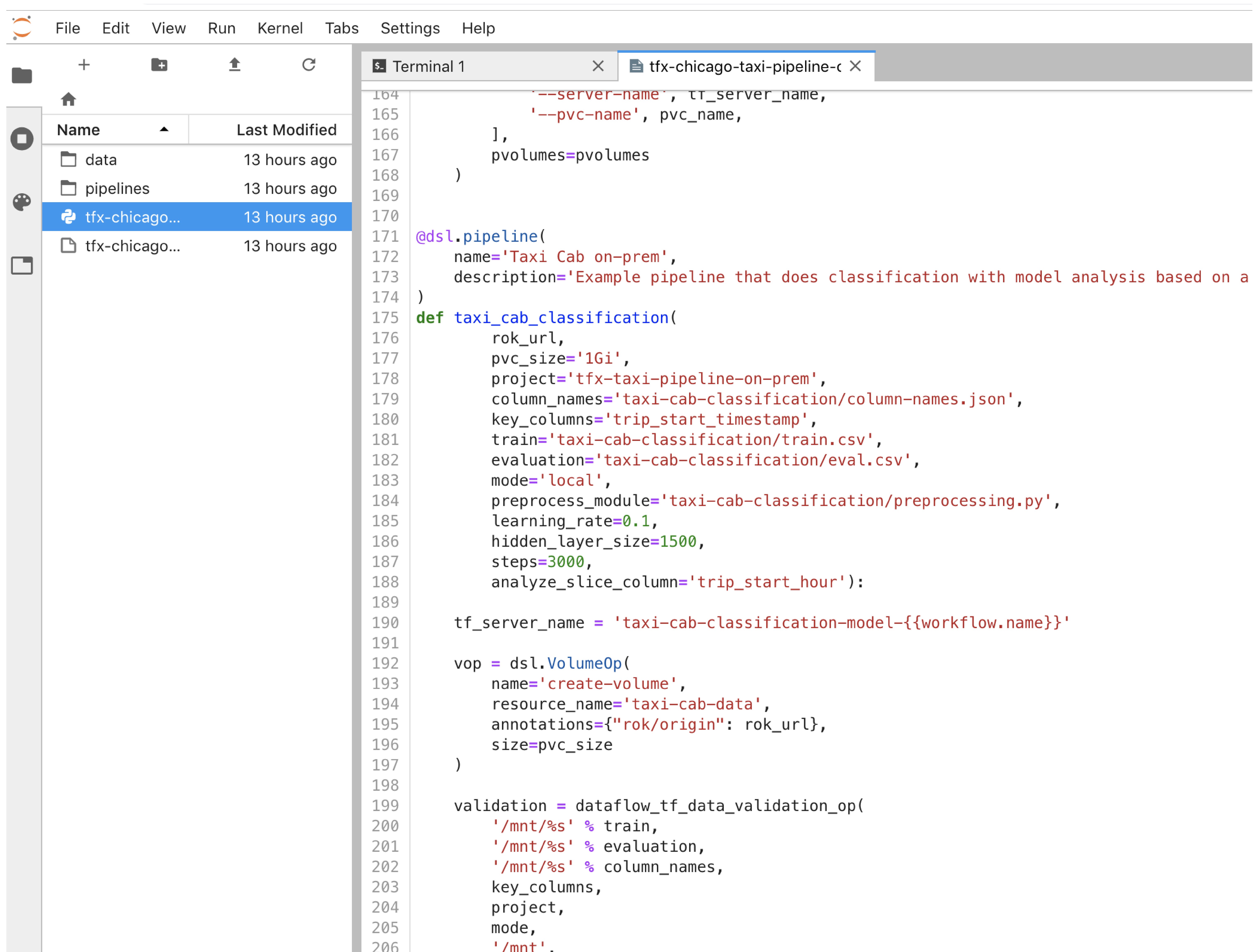
Gears: component



User Experience: notebook



User Experience: notebook



Python SDK

- Let data scientist and engineers work together

Kubeflow User story

YAML vs Python SDK

```yaml
apiVersion: argoproj.io/v1alpha1
kind: Workflow
metadata:
  generateName: charts-of-accounts-
spec:
  arguments:
    parameters:
      - name: aws-cli-image
        value: "pbsmacc/aws-cli:latest"
      - name: prepare-dataset-image
        value: "smaccio/accounting-charts-prepare-dataset:v0.2
      - name: trainer-image
        value: "smaccio/accounting-charts-classifier:v0.2.0"
      - name: minio-client-image
```

Python SDK

```
@dsl.pipeline(
    name='Taxi Cab on-prem',
    description='Exar.'
)
def taxi_cab_classification(
    training = tf_train_op(
        preprocess.output,
        validation.outputs['schema'],
        learning_rate,
        hidden_layer_size,
        steps,
        'tips',
        '/mnt/%s' % preprocess_module,
        '/mnt',
```

Python SDK

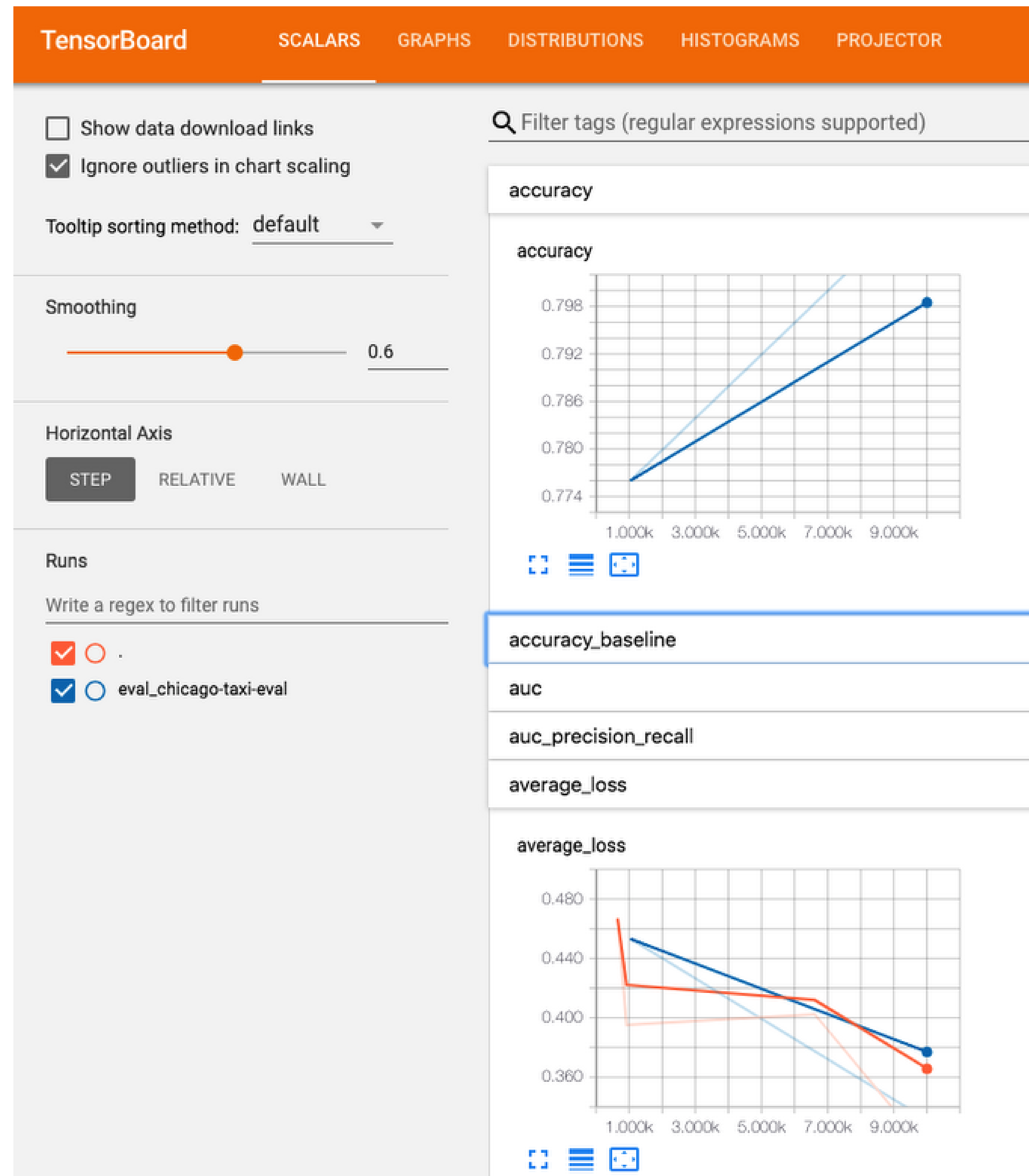
```
def kubeflow_deploy_op(model: 'TensorFlow model', tf_server_na
                       pvolumes, step_name='deploy'):
    return dsl.ContainerOp(
        name=step_name,
        image='gcr.io/ml-pipeline/ml-pipeline-kubeflow-deploye
        arguments=[
            '--cluster-name', 'tfx-taxi-pipeline-on-prem',
            '--model-export-path', model,
            '--server-name', tf_server_name,
            '--pvc-name', pvc_name,
        ],
        pvolumes=pvolumes
    )
```

More Python: Fairing SDK

- All power of kubeflow
  from your local jupyter notebook
- For hybird cloud

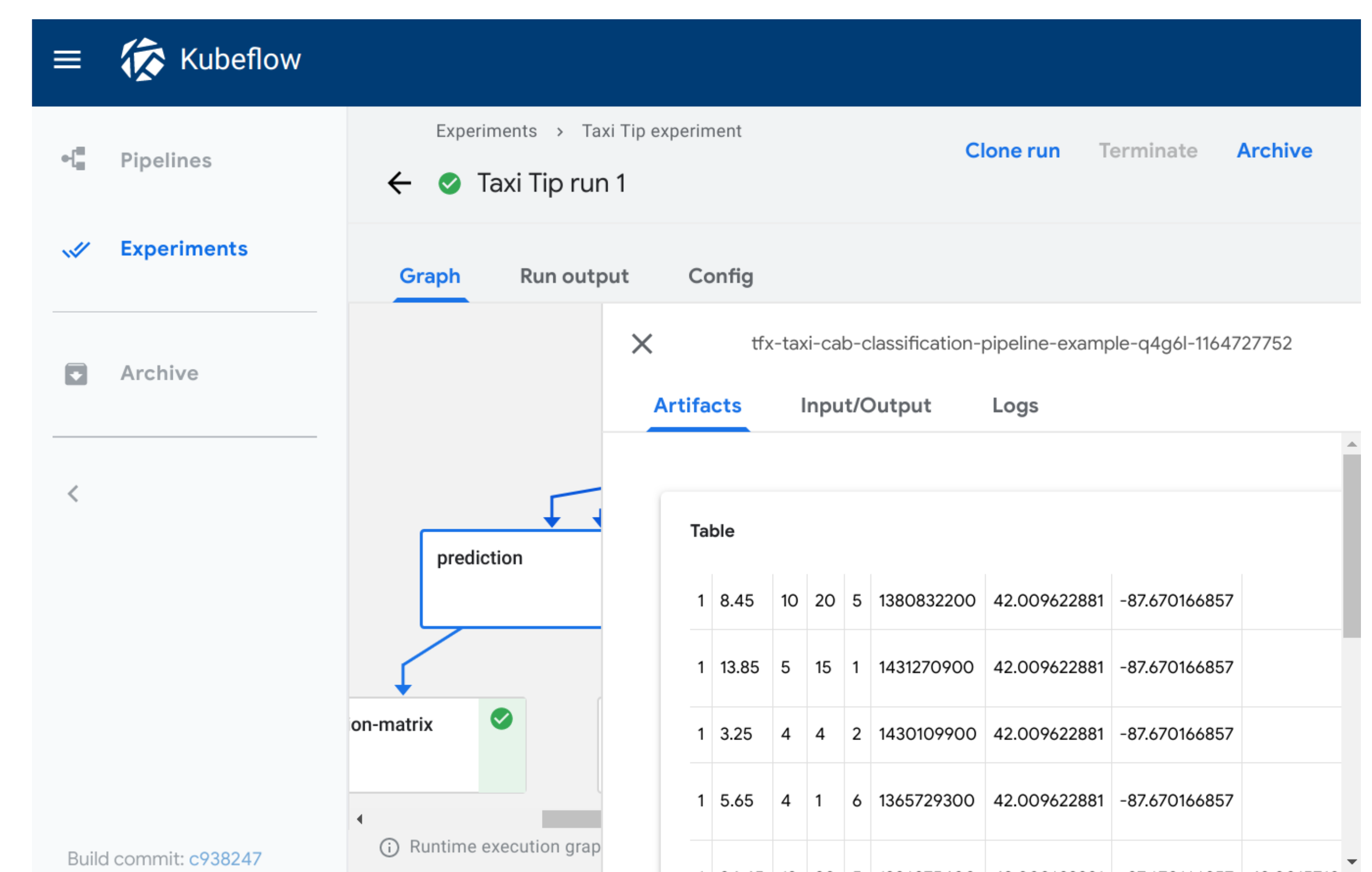
https://github.com/kubeflow/fairing

User Experience: Tensorboard



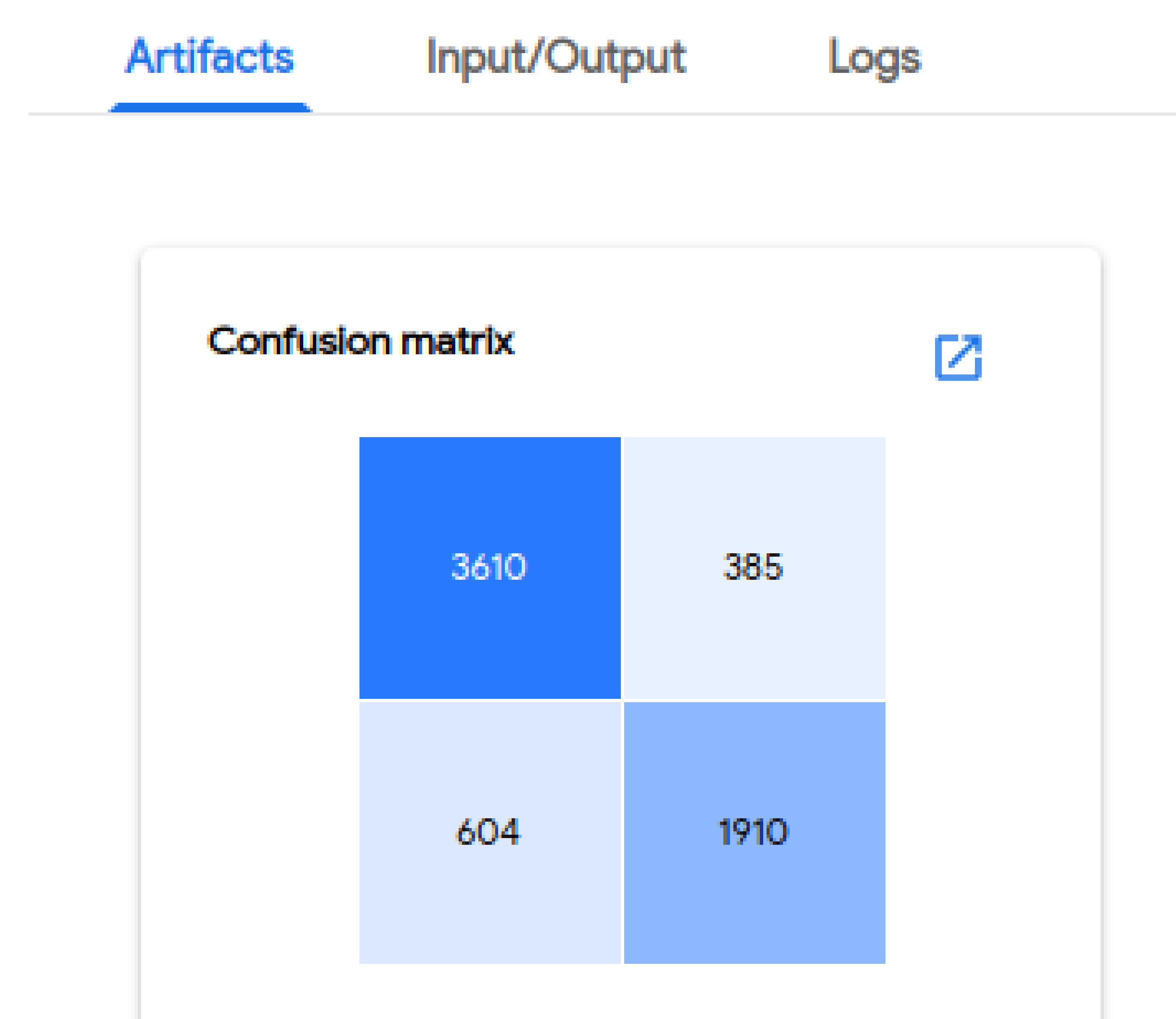
Tracking: Artifacts

- Emitted by steps as metadata



Tracking: Artifacts

- Emitted by steps as metadata
- `mlpipeline-ui-metadata.json`



Focus on Data Scientist

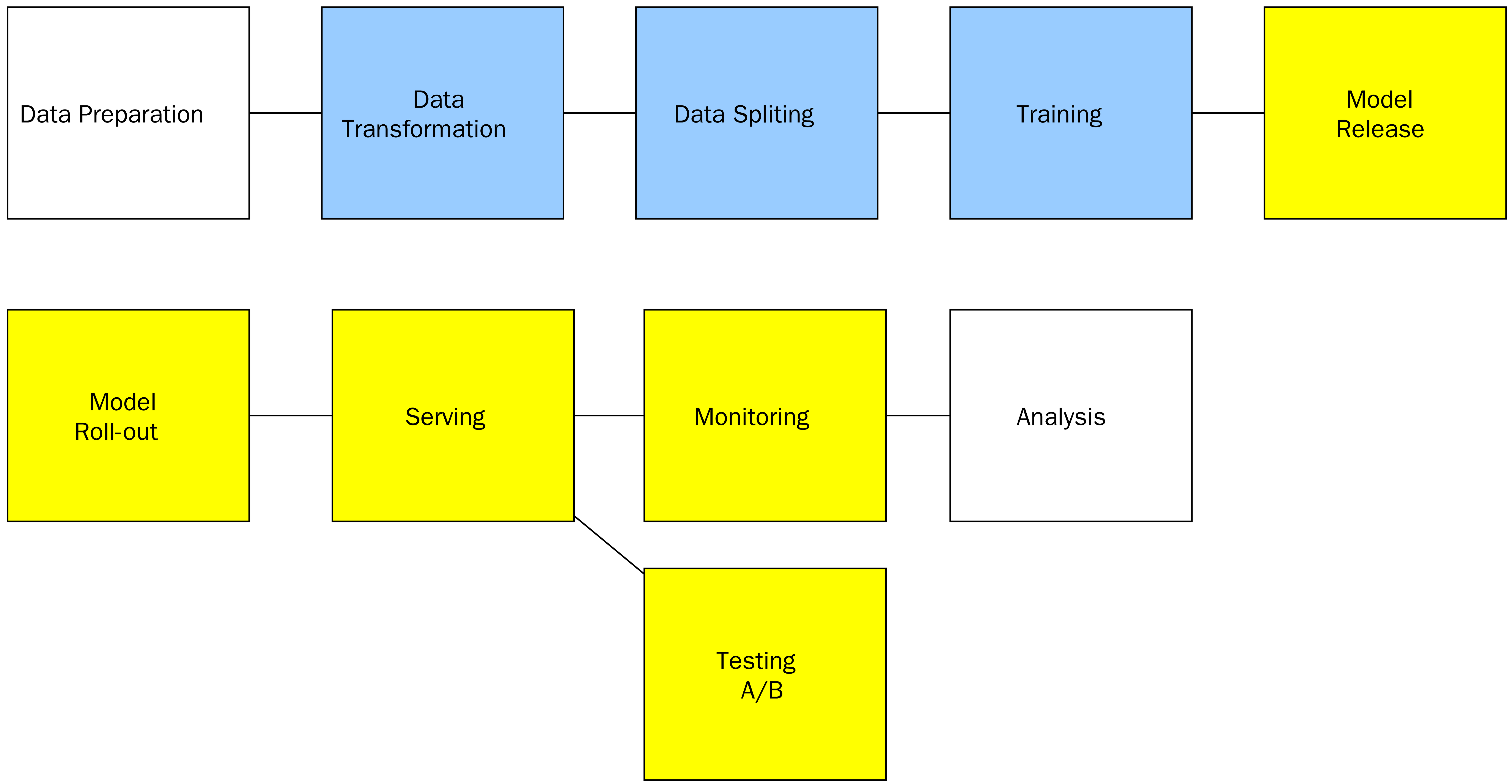
- Self-service
- Provide familiar user experience and tools
- Share the knowledge
- Hide the engineering complexity

Batteries Included

Scale trainings

- TFJobs
- MPI Training
- PyTorch Training
- MXNet Training

Machine Learning



Kubeflow

- **How to serve the model**
- Operating service
- Observability, e.g., metric collection

Kubeflow

- Operating service
- Observability, e.g., metric collection
- Deployment strategies
- ...

One CloudNative project comes to mind - Istio.

Kubeflow

Serving

- ML Model servers
- seldon.io
- kfserving

ML Model Servers

- TFserving
- PyTorch Serving
- ...

ML Model Servers

- Minimum configuration
- Serve a given trained ML model

with Istio integration if needed

Seldon.io

- More complex use cases

seldon.io

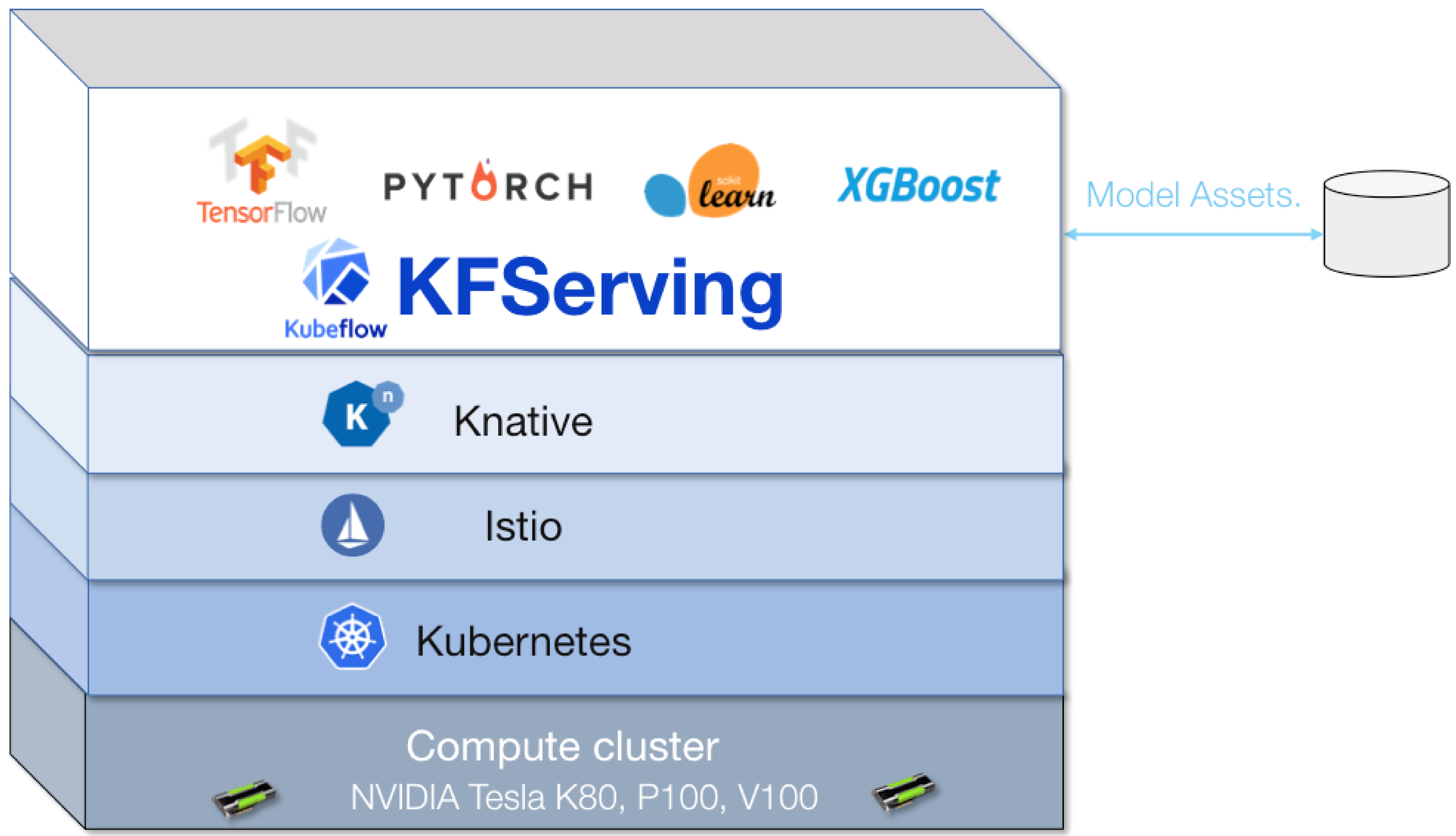
Istio

- Observability Grafana
- Deployment strategies

Istio

- All served models are available with Istio

kfserving

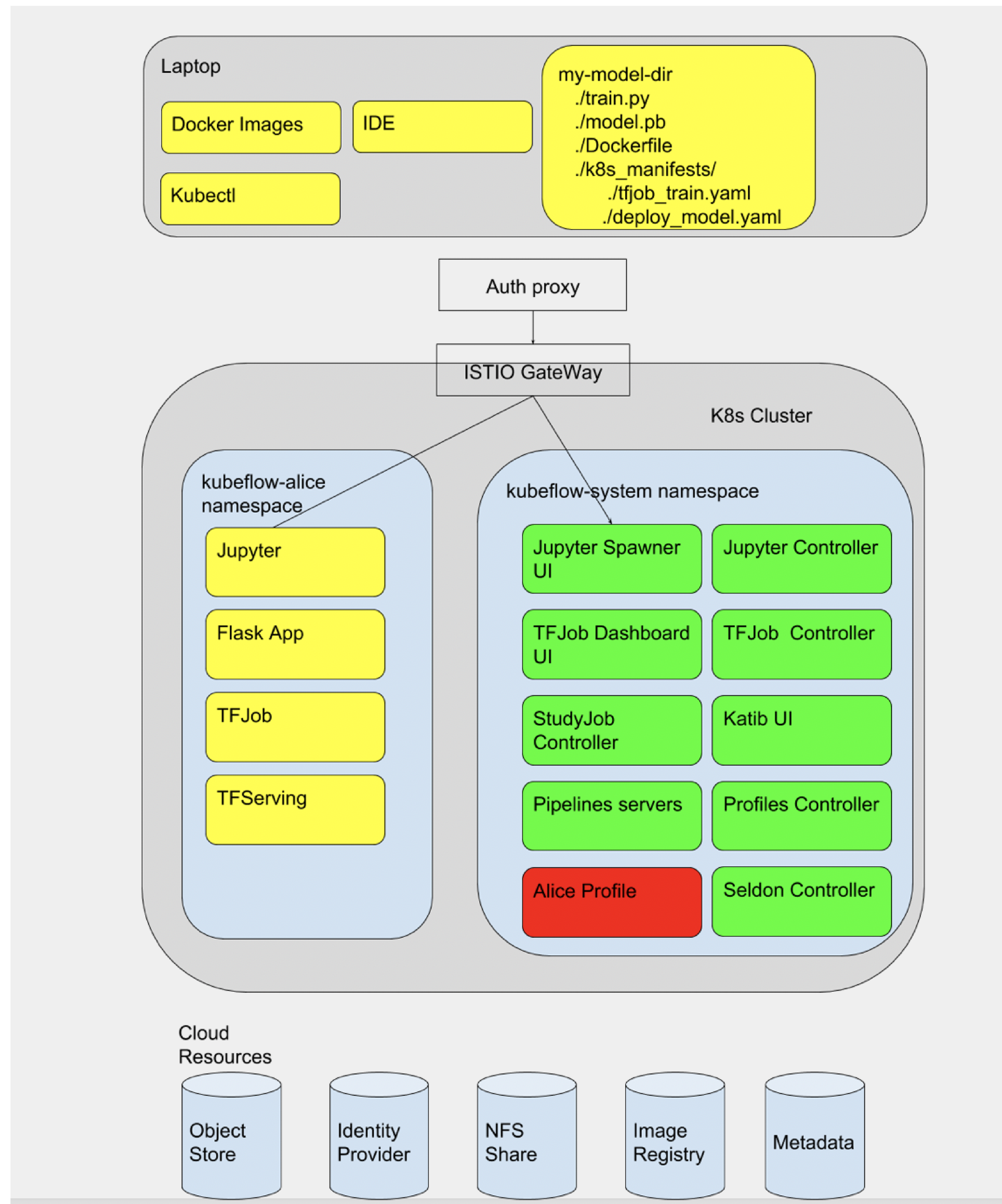


TensorFlow
PYTORCH
learn
XGBoost
Kubeflow
KFServing
Model Assets.
Knative
Istio
Kubernetes
Compute cluster
NVIDIA Tesla K80, P100, V100

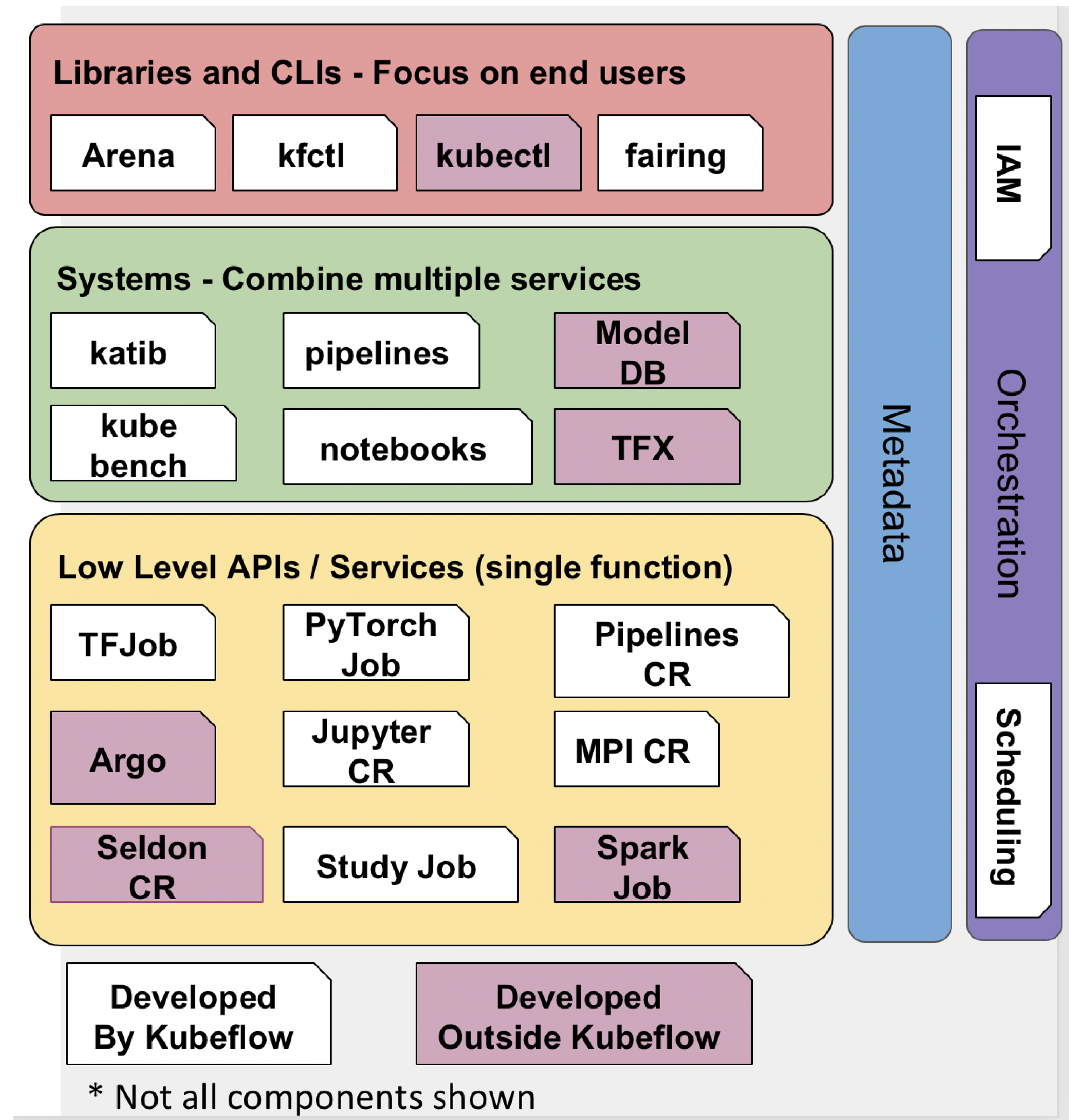
kfserving

```
apiVersion: "serving.kubeflow.org/v1alpha2"
kind: "KFService"
metadata:
  name: "xgboost-iris"
spec:
  default:
    predictor:
      xgboost:
        storageUri: "gs://kfserving-samples/models/xgboost/iri
```

Architecture



Architecture



What did we decided?

- Large project with many moving parts
- Take bits that we need and keep delivering
- Invest more time into observability

We do not have such a large team

What did we pick?

1. Mostly Model + Code as Software Components
2. Automation project:
    - Argo in YAML
    - tfserving
    - tensorflow_transform.beam

Deployment?

- Git-driven deployment
- Version the model and the code

we might use argo here as well

Observability

- Prometheus + Grafana
- Dedicated metric collector

Keep an eye on kubeflow

- Enterprise client projects on-prem
- Growing team

Summary

- Enable Dev(ML)Ops culture
- Hide the engineering complexity

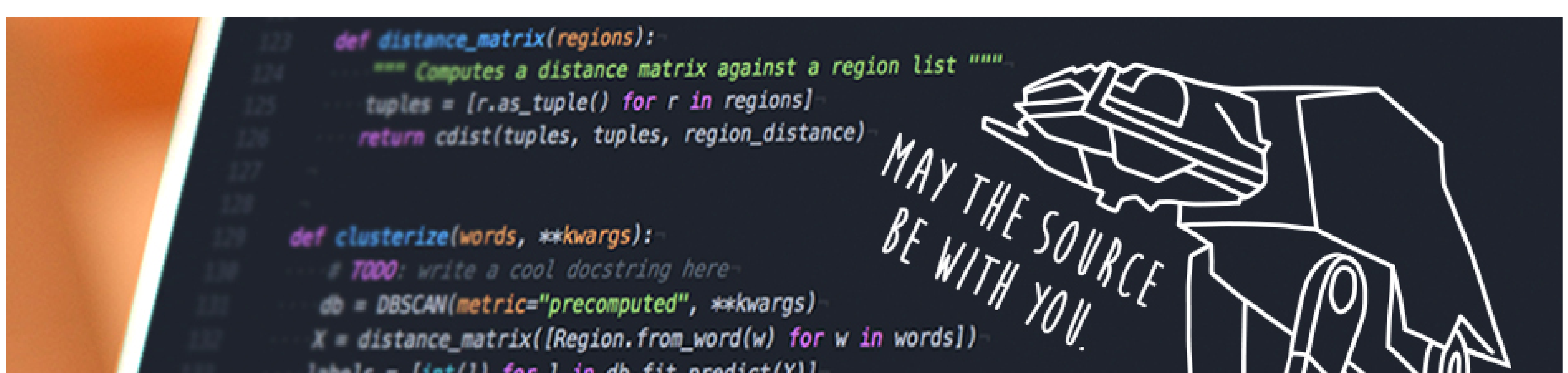
Summary

- The learning part is the most compelling
- and self-service

QUESTIONS?


def distance_matrix(regions):
""" Computes a distance matrix against a region list """
tuples = [r.as_tuple() for r in regions]
return cdist(tuples, tuples, region_distance)
def clusterize(words, **kwargs):
# TODO: write a cool docstring here
db = DBSCAN(metric="precomputed", **kwargs)
X = distance_matrix([Region.from_word(w) for w in words])
labels = [int(l) for l in db.fit_predict(X)]
MAY THE SOURCE
BE WITH YOU.

Big thanks to Piotr Brzostowski and whole BER+WAW team.



BACKUP

Development

- How to handover to engineering?
- How did I trained the model X?
- Lineage and Metadata

Operation

- How the model performs in production
- Is it better?
- Which data should I add to the next training?
- Low performance → roll back
- Keep the TCO reasonable

Operation

- Observability: Prometheus, Grafana
- A/B testing: service mesh
- Serving model?