FEATURE STORE SUMMIT

12-13 OCTOBER | 08:30 AM - 4:00 PM PT

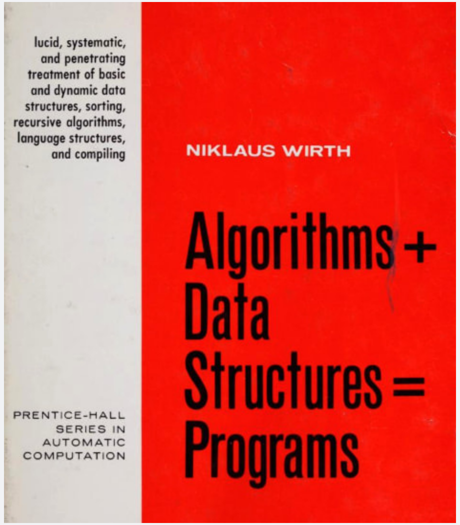
ORGANIZED BY HOPSWORKS

The SAME Project:
A Cloud Native Approach to Reproducible Machine Learning

David Aronchick
PM, Office of the CTO
Azure

Definitions

- Terms in AI/ML are weird
  - **AI** = Artificial Intelligence
  - **ML** = Machine Learning
  - Experiment, Run, Pipeline, Job all have many varied definition (but often are synonymous)
- For this deck:
  - **Program** == Algorithms + Data Structures + Workflow
  - **Experiment** == Program + parameters for a *specific* execution (an instance of a program)
  - Any other terms are legacy
- <u>I promise you I will get this wrong</u> – feel free to raise your hand if/when I do ☺



lucid, systematic, and penetrating treatment of basic and dynamic data structures, sorting, recursive algorithms, language structures, and compiling

NIKLAUS WIRTH

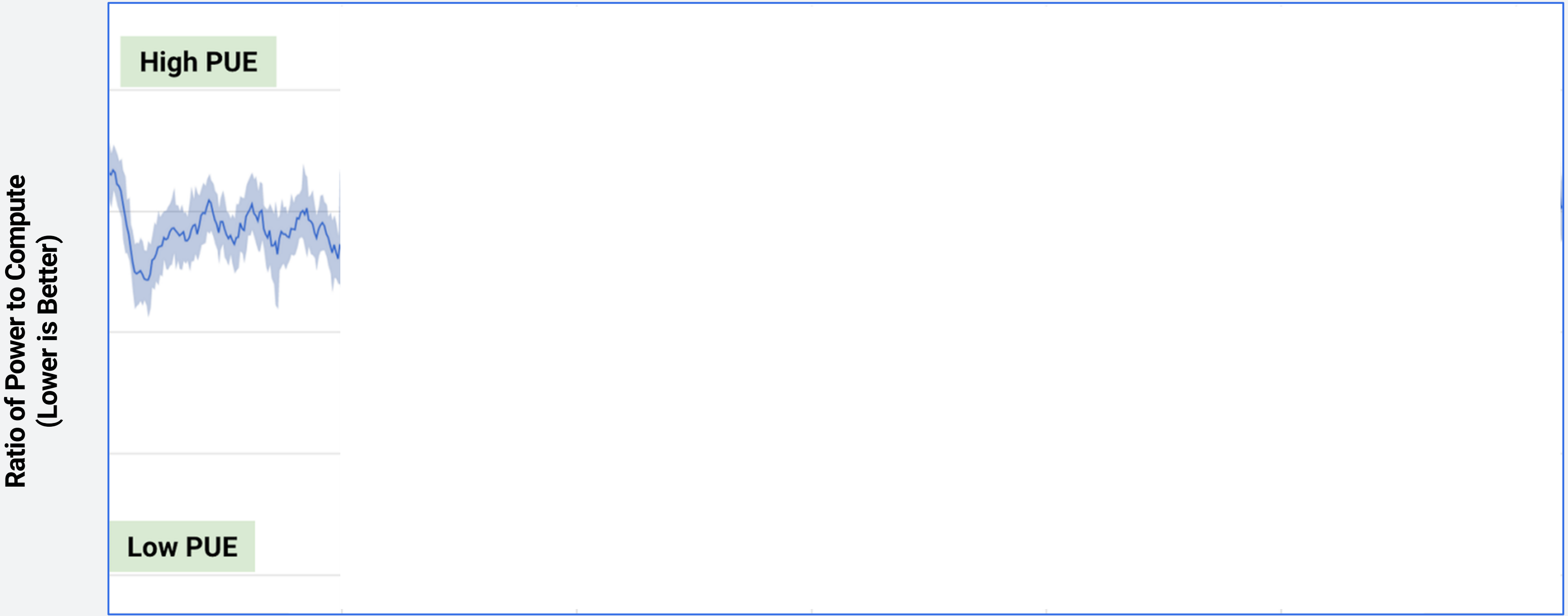
Algorithms + Data Structures = Programs

PRENTICE-HALL SERIES IN AUTOMATIC COMPUTATION

ML is great!

PUE

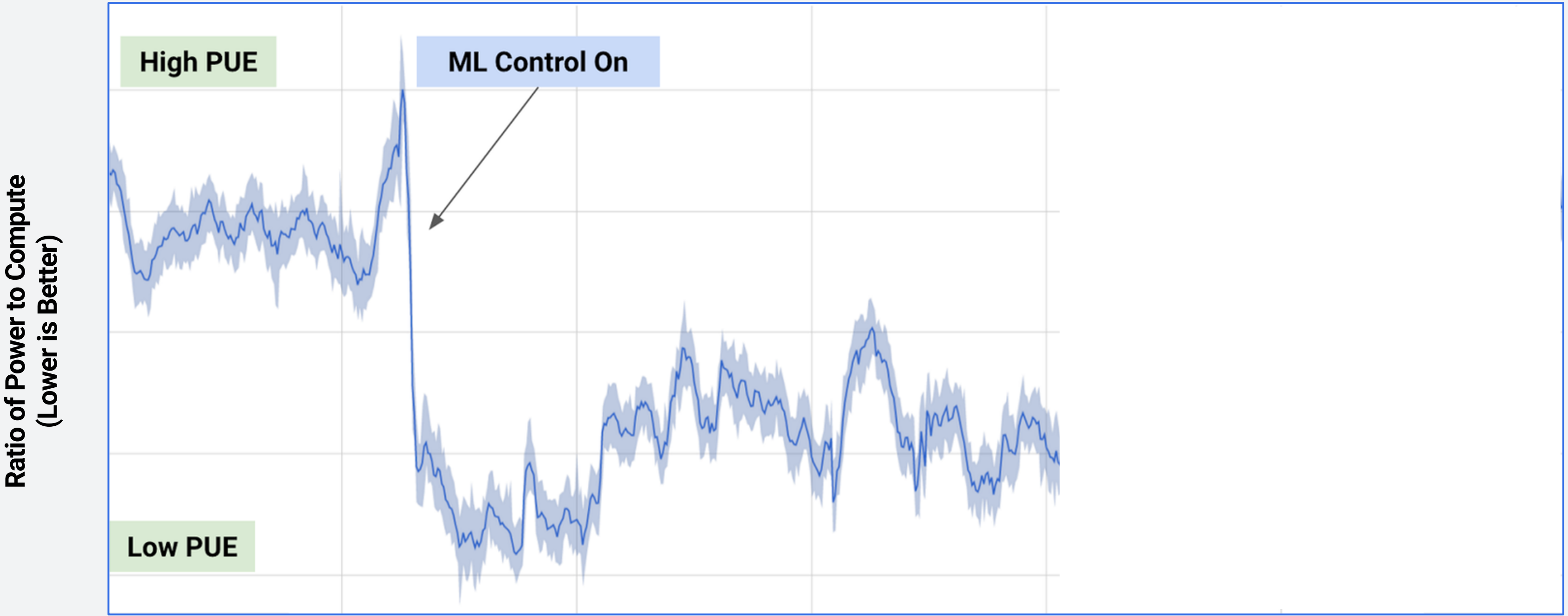
power usage effectiveness



Feature Stores for ML

PUE

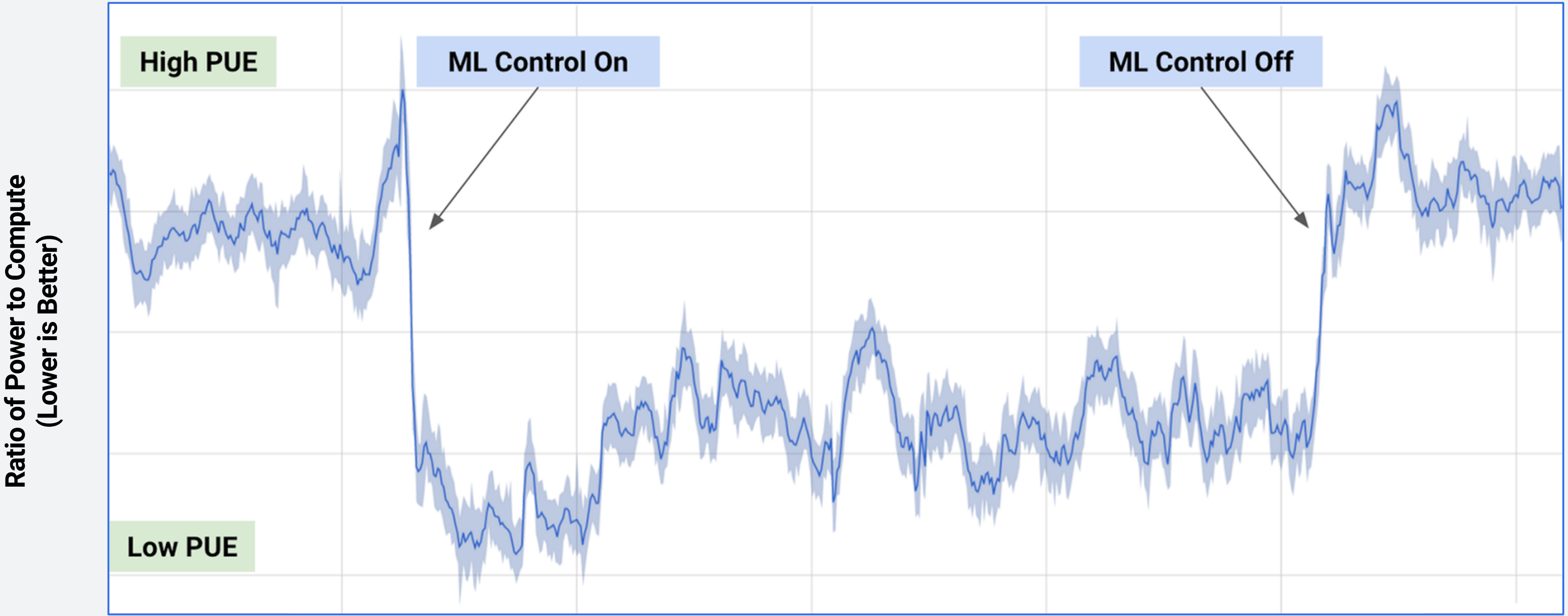
power usage effectiveness



Feature Stores for ML

PUE

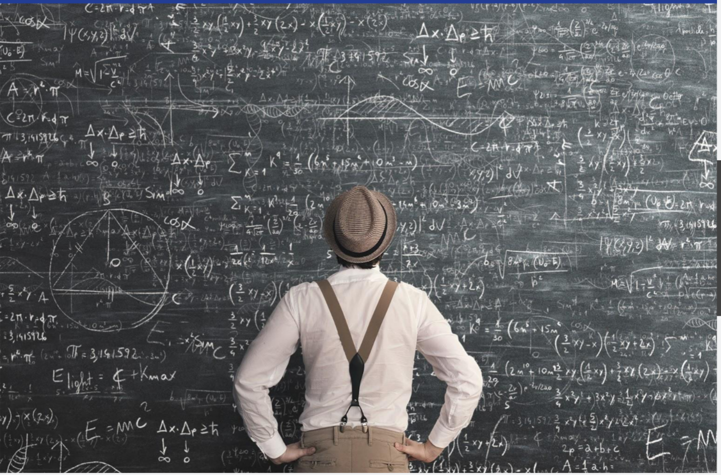
power usage effectiveness



Feature Stores for ML

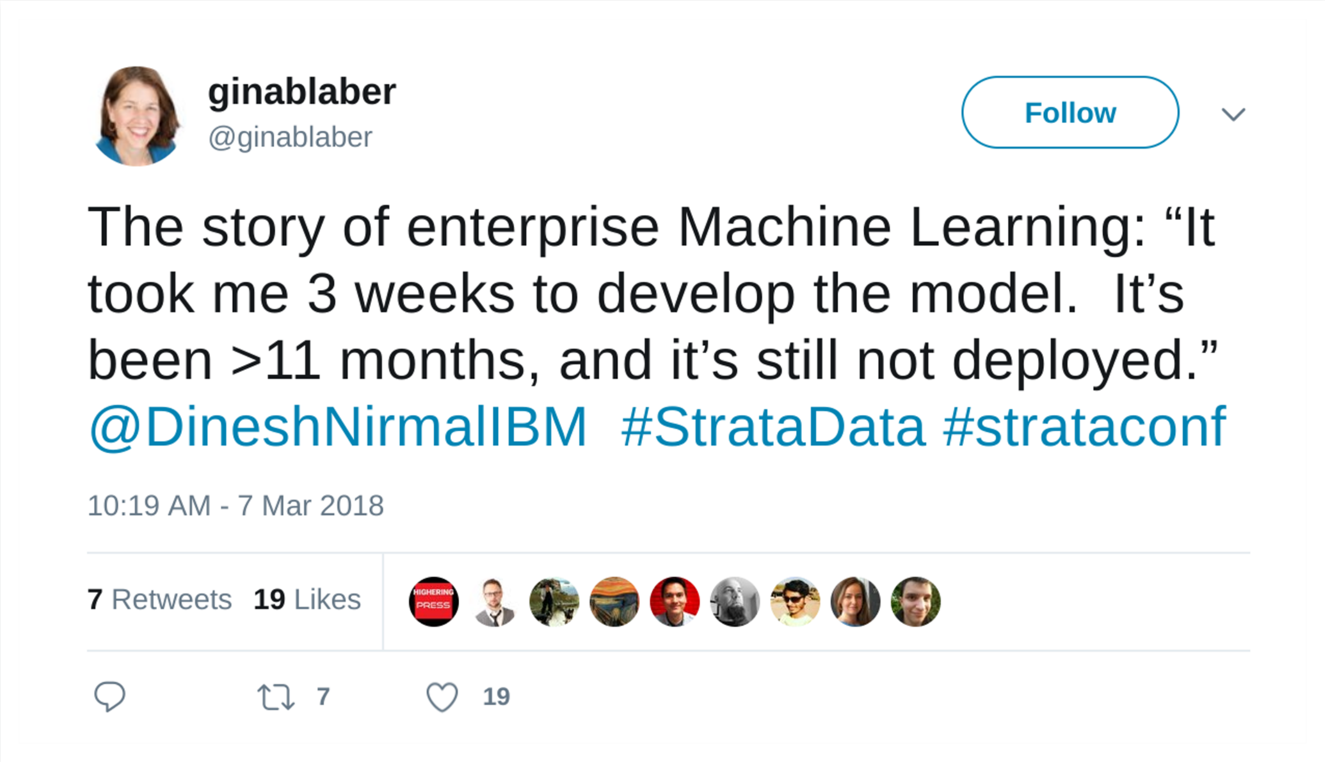
ML is hard!

Most folks



Lots of pain

Magical AI goodness



ginablaber
@ginablaber

The story of enterprise Machine Learning: “It took me 3 weeks to develop the model. It’s been >11 months, and it’s still not deployed.” @DineshNirmalIBM #StrataData #strataconf

10:19 AM - 7 Mar 2018

7 Retweets **19** Likes

Users Have Two Options

DIY

- Set up from scratch
- Integrate with existing legacy systems
- Eventual need to migrate based on business needs

Proprietary Solution

- First 5 minutes? **YES**.
- **Next 5 years** …
  - Set it up from scratch
  - Integrate with legacy
  - Migrate based on business needs
- … plus lock-in!

Feature Stores for ML

Haven’t we heard this story before?

Containers & Kubernetes

Cloud Native Apps

So…

We Need
Cloud Native ML!

We Need
Cloud Native ML?

- Composability
- Portability
- Scalability

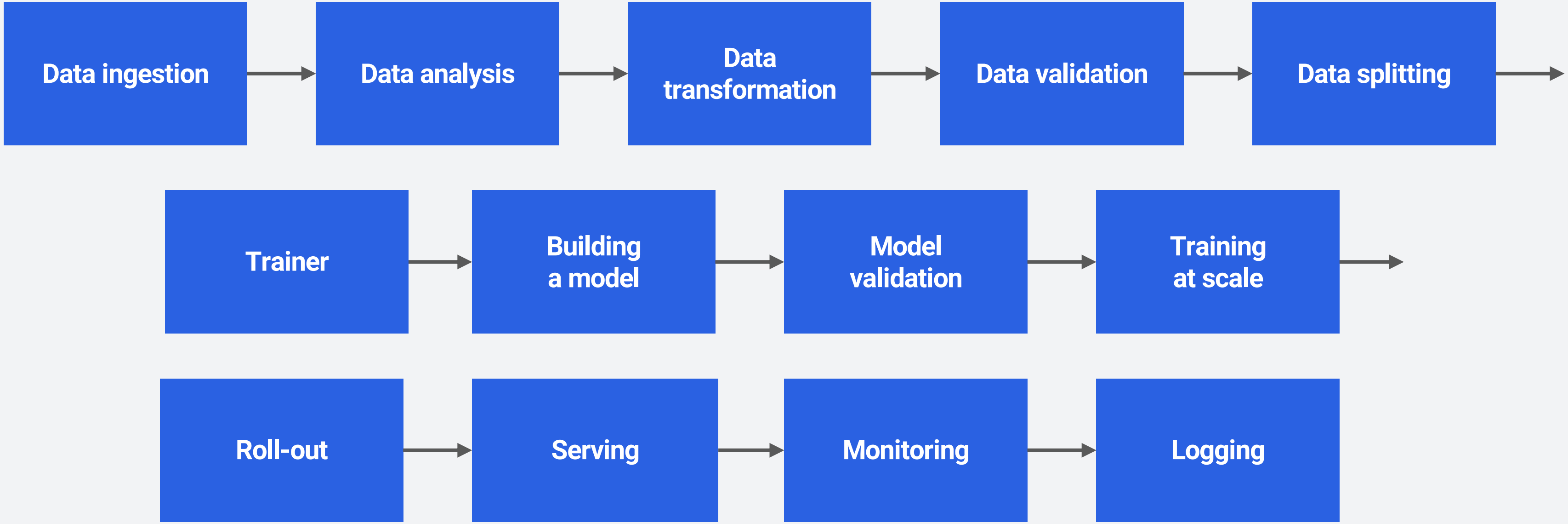
Composability

Building a model

Composability

Building
a model

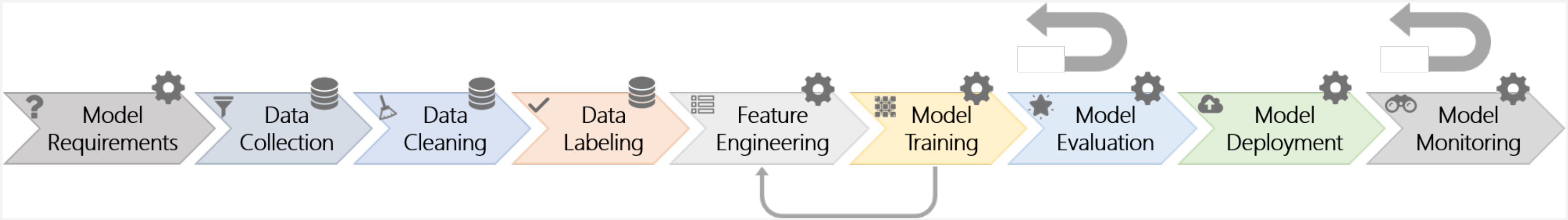
Composability



Data ingestion → Data analysis → Data transformation → Data validation → Data splitting →

Trainer → Building a model → Model validation → Training at scale →

Roll-out → Serving → Monitoring → Logging



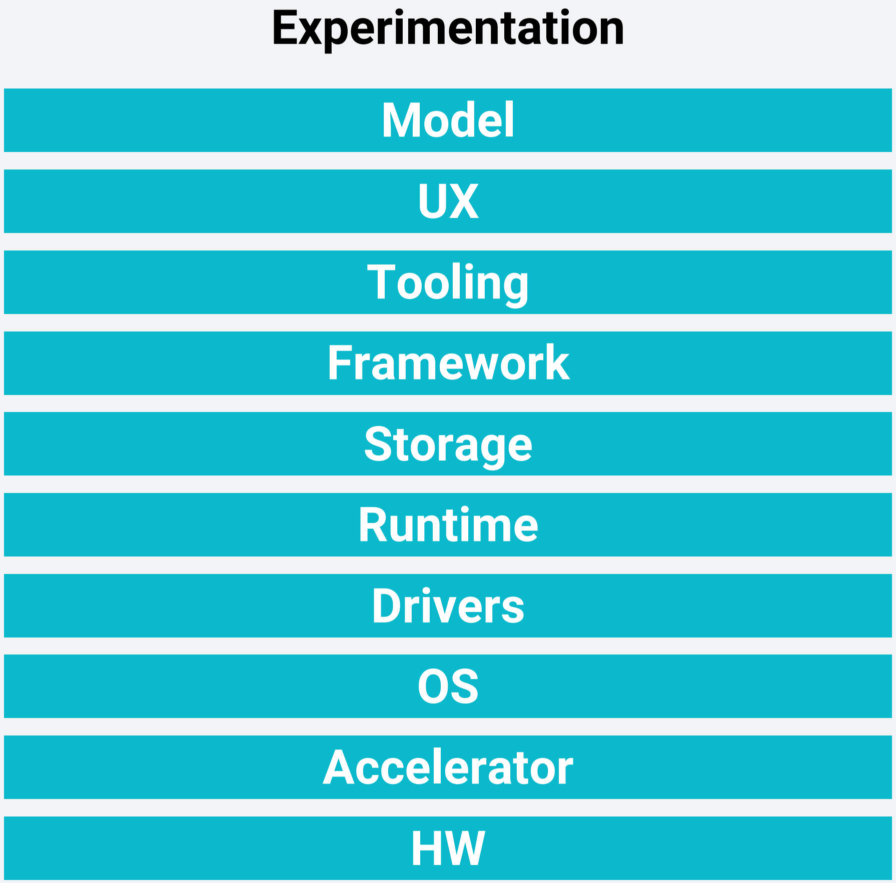
Number of unique tools reported per activity

18	44	35	9	34	58	24	48	13
OneNote (7)	Cosmos (15)	Python (14)	Custom (11)	Python (14)	TLC (17)	TLC (6)	Custom (6)	Custom (6)
Word (7)	Custom (8)	R (11)	UHRS (7)	R (9)	Python (14)	Aether (5)	AzureML (6)	AzureML (1)
PowerPoint (3)	Kusto (7)	Cosmos (10)	Excel (6)	AzureML (5)	CNTK (11)	Python (4)	Azure (4)	Docker (1)
Excel (2)	Scope (6)	SQL (8)	Python (1)	SQL (4)	Tensorflow (11)	Custom (3)	Docker (4)	TLC (1)
Pytorch (2)	SQL (4)	Scope (6)	R (1)	Scope (3)	Aether (8)	Excel (3)	Visual Studio (4)	Aether (1)
Jupyter (2)	Python (2)	Custom (3)	Cosmos (1)	C# (3)	Custom (8)	R (2)	TLC (3)	Cosmos (1)
Python (1)	Aether (2)	Excel (3)	Matlab (1)	Aether (3)	Pytorch (7)	Tensorflow (1)	Aether (3)	C# (1)
TLC (1)	Azure (2)	Databricks (3)	Datagrid (1)	TLC (3)	Cosmos (5)	Cosmos (1)	R (3)	CNTK (1)
Aether (1)	Datagrid (2)	NLTK (3)	PICL (1)	Excel (2)	Keras (5)	SkLearn (1)	VSTS (3)	TensorFlow (1)
CNTK (1)	SSIS (2)	Kusto (2)		VSO (2)	R (4)	Jupyter (1)	QAS (3)	PowerBI (1)

2018 MSR Fragmentation Study: 159 Unique Tools

Portability

Portability



Experimentation

Model

UX

Tooling

Framework

Storage

Runtime

Drivers

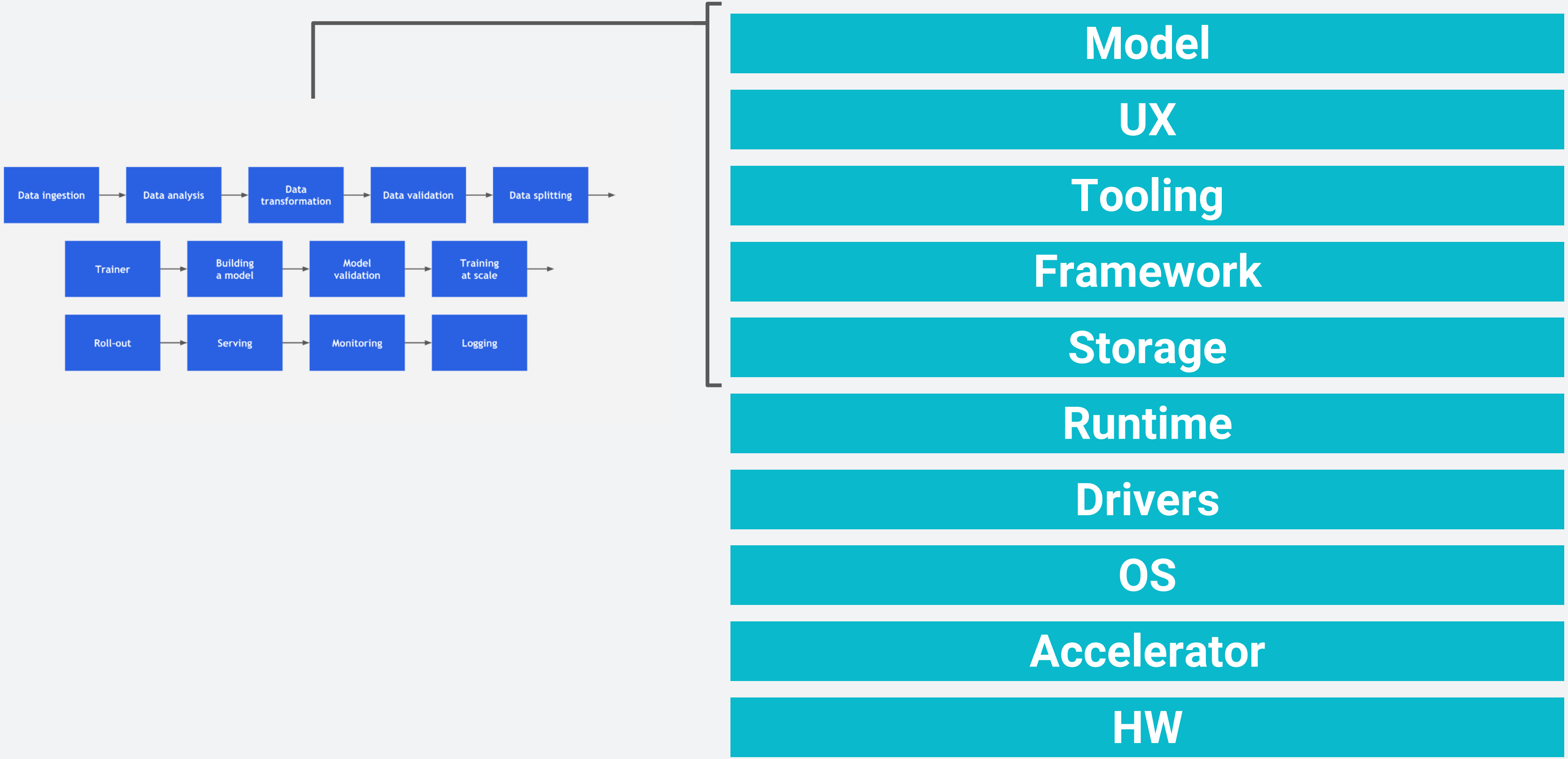
OS

Accelerator

HW

Portability

Experimentation



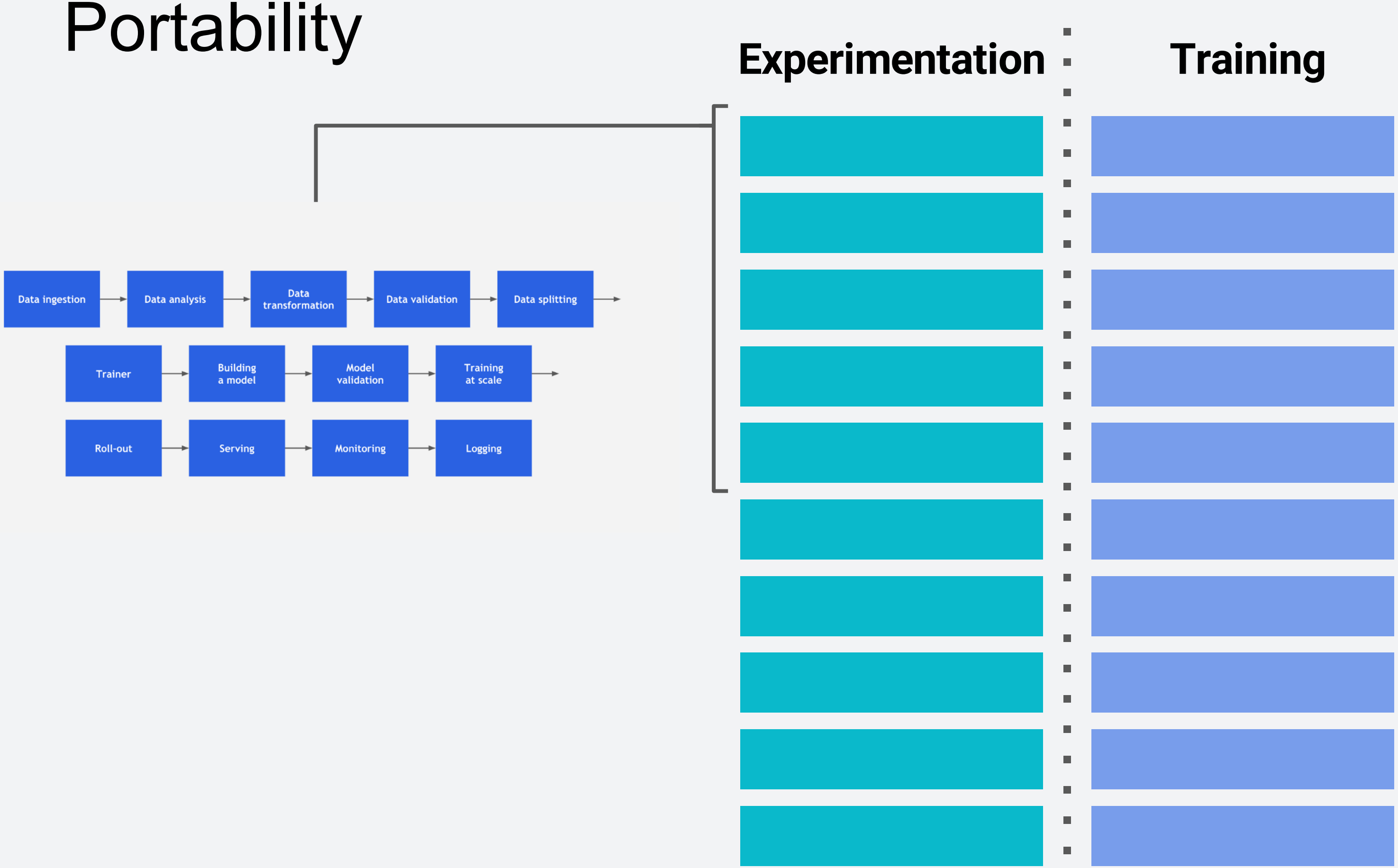
Data ingestion → Data analysis → Data transformation → Data validation → Data splitting →

Trainer → Building a model → Model validation → Training at scale →

Roll-out → Serving → Monitoring → Logging

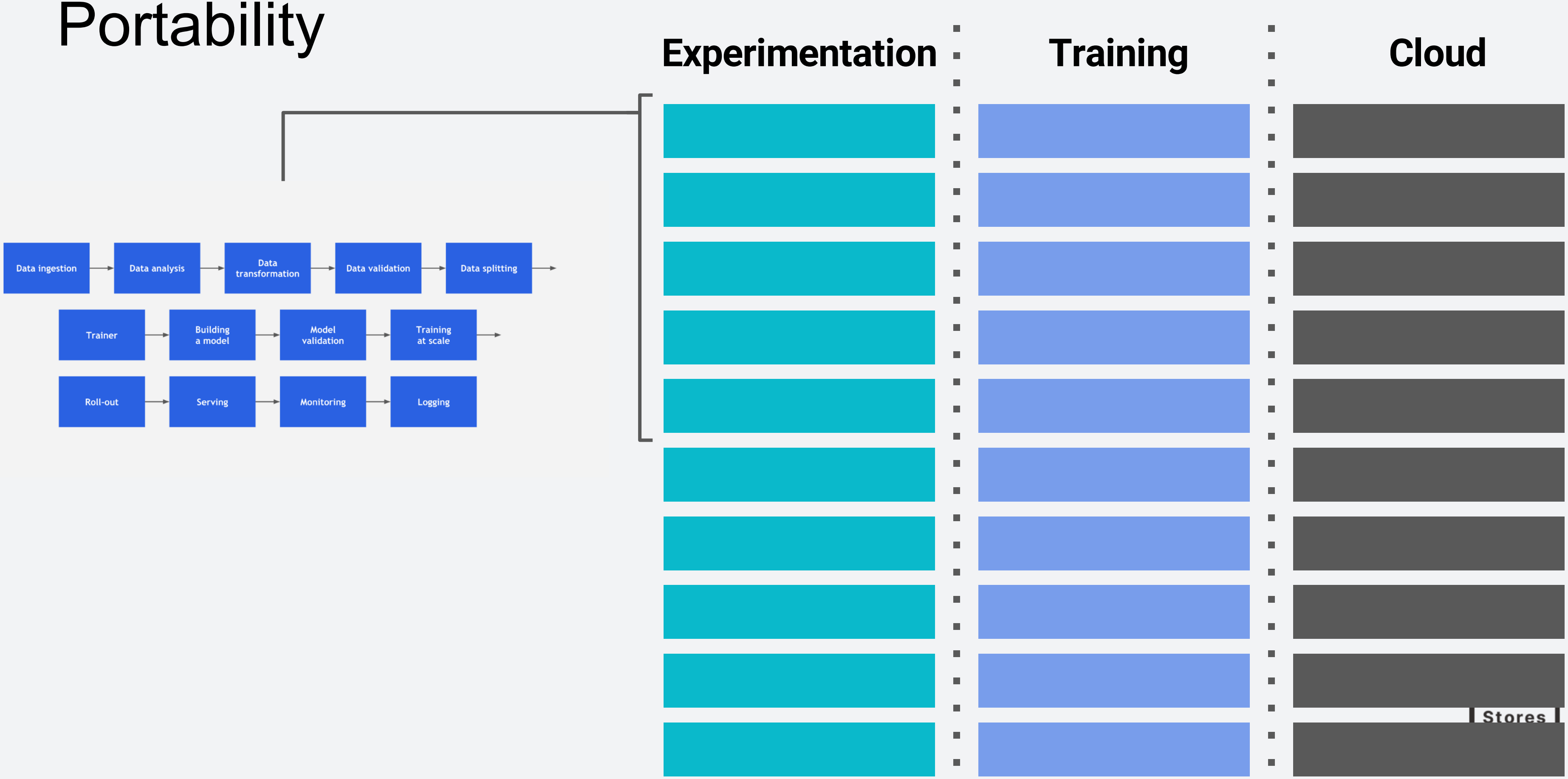
- Model
- UX
- Tooling
- Framework
- Storage
- Runtime
- Drivers
- OS
- Accelerator
- HW

Portability



Feature Stores for ML

Portability



Experimentation

Training

Cloud

Data ingestion → Data analysis → Data transformation → Data validation → Data splitting →

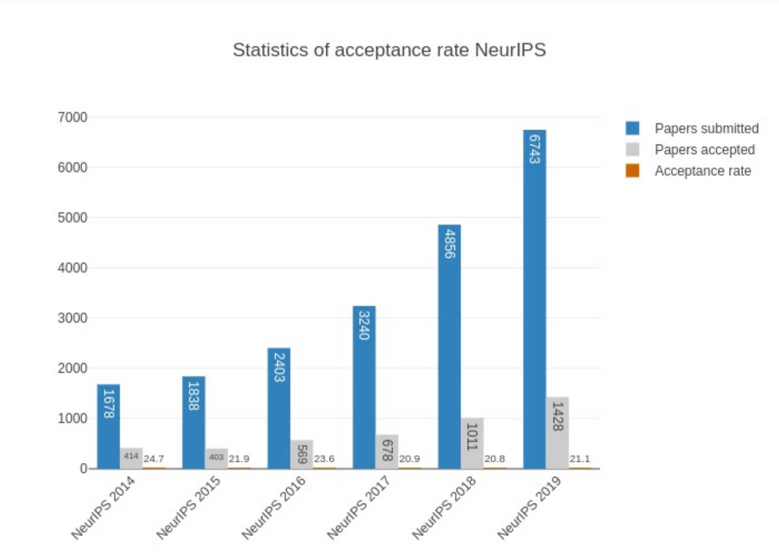
Trainer → Building a model → Model validation → Training at scale →

Roll-out → Serving → Monitoring → Logging

Stores

Scalability

- More infrastructure
  - Accelerators (GPU, TPU)
  - Cores/CPUs
  - Disk/networking
- More programs
  - 1000s+ of programs run simultaneously
  - 1000s+ historical runs to compare
  - 1Ms+ of papers published
- More collaboration
  - Skillsets (SWEs, data scientists)
  - Teams
  - Organizations



More programs, faster

Composability

Portability

Scalability

You know what’s really good at composability, portability, and scalability?

Containers & Kubernetes

Containers & Kubernetes

except...

Oh, You Want to Use ML?

First, become an expert in:

- Containers
- Packaging
- Kubernetes service endpoints
- Persistent volumes
- Scaling
- Immutable deployments
- Service meshes
- GPUs, Drivers & the GPL
- Cloud APIs
- DevOps
- ...

Let's Build Something To Give Everyone Cloud Native ML

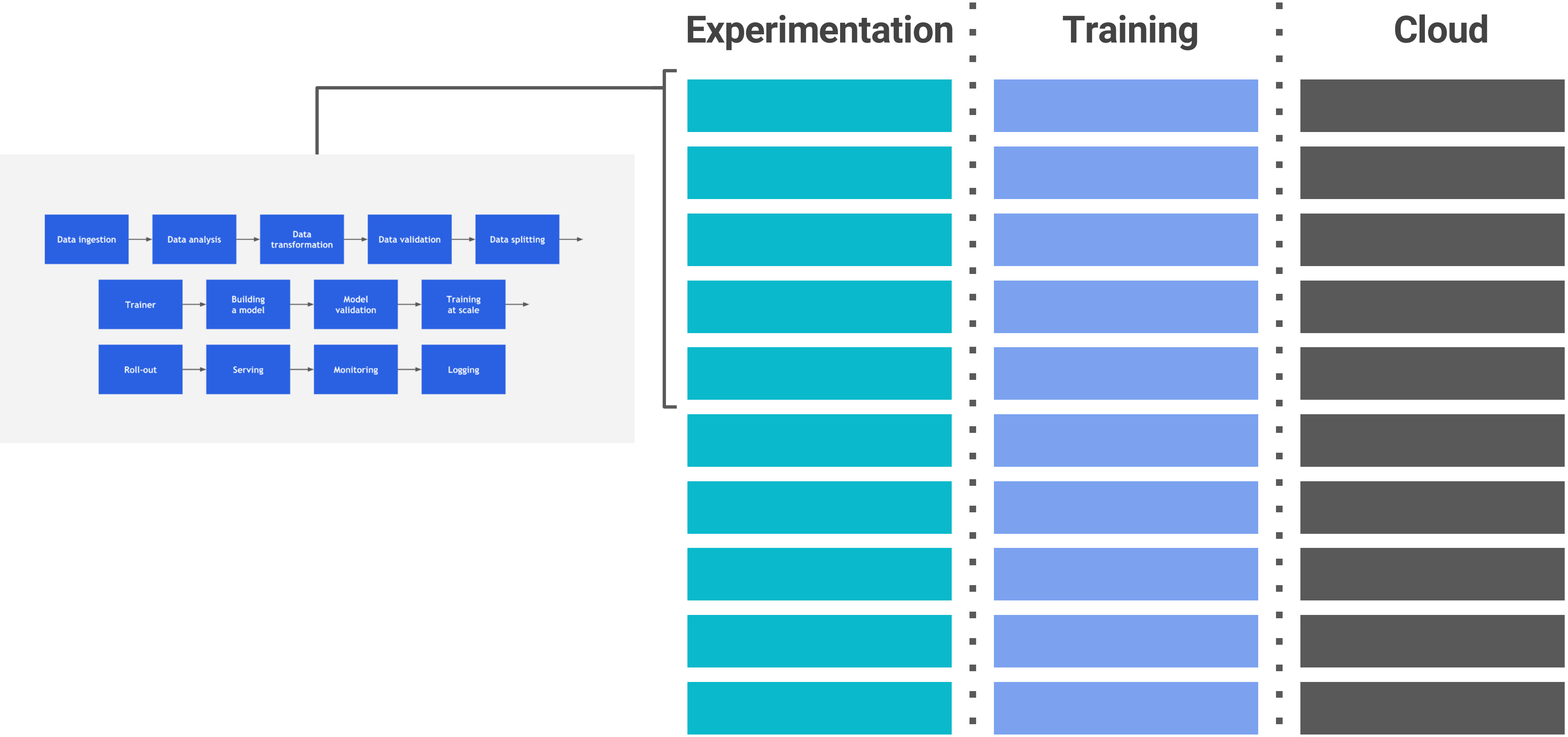
The SAME Project

The SAME Project

Self Assembling
Machine Learning
Environments

The SAME Project

Self Assembling
Machine Learning
Environments



Experimentation

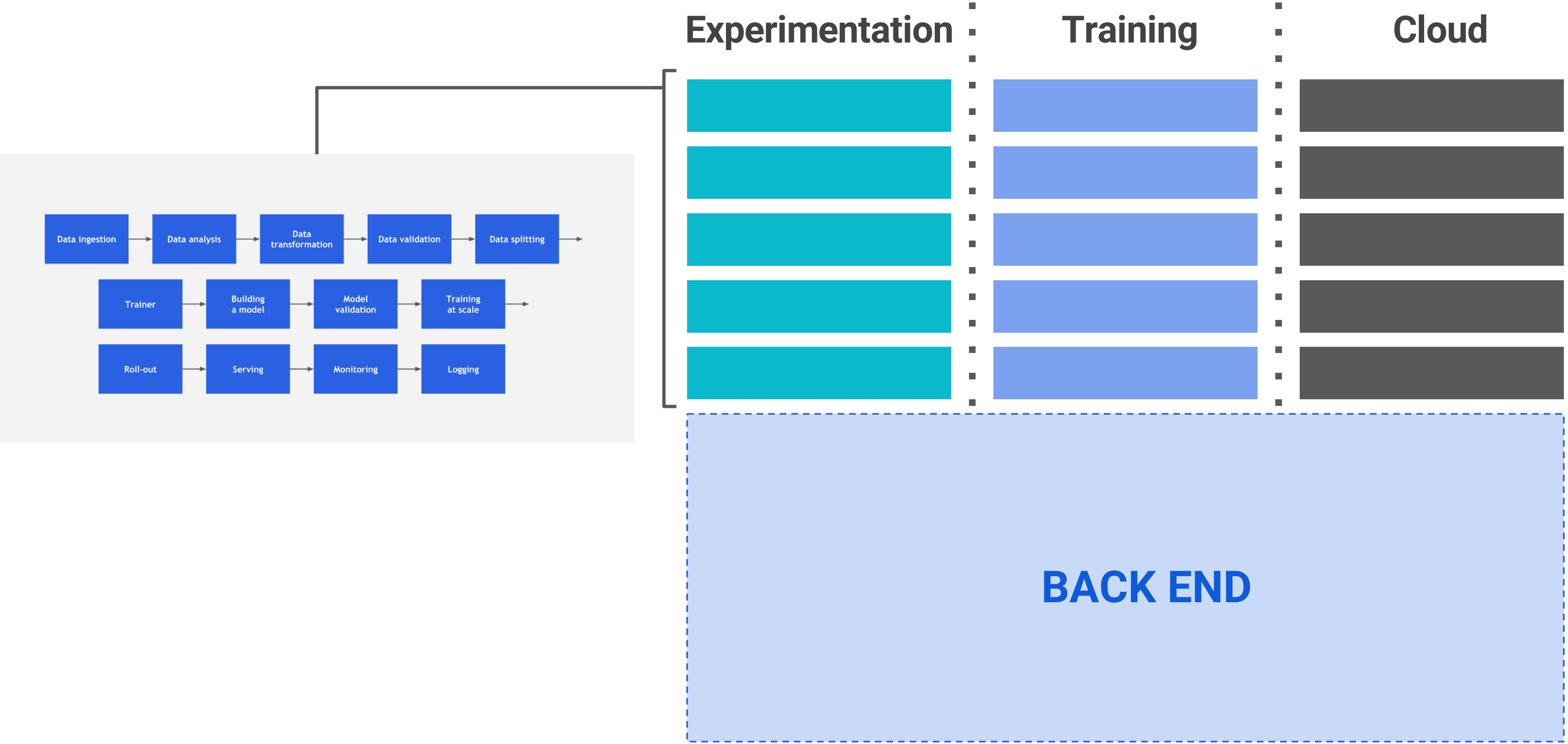
Training

Cloud

Data ingestion → Data analysis → Data transformation → Data validation → Data splitting →

Trainer → Building a model → Model validation → Training at scale →

Roll-out → Serving → Monitoring → Logging



Data ingestion → Data analysis → Data transformation → Data validation → Data splitting →

Trainer → Building a model → Model validation → Training at scale →

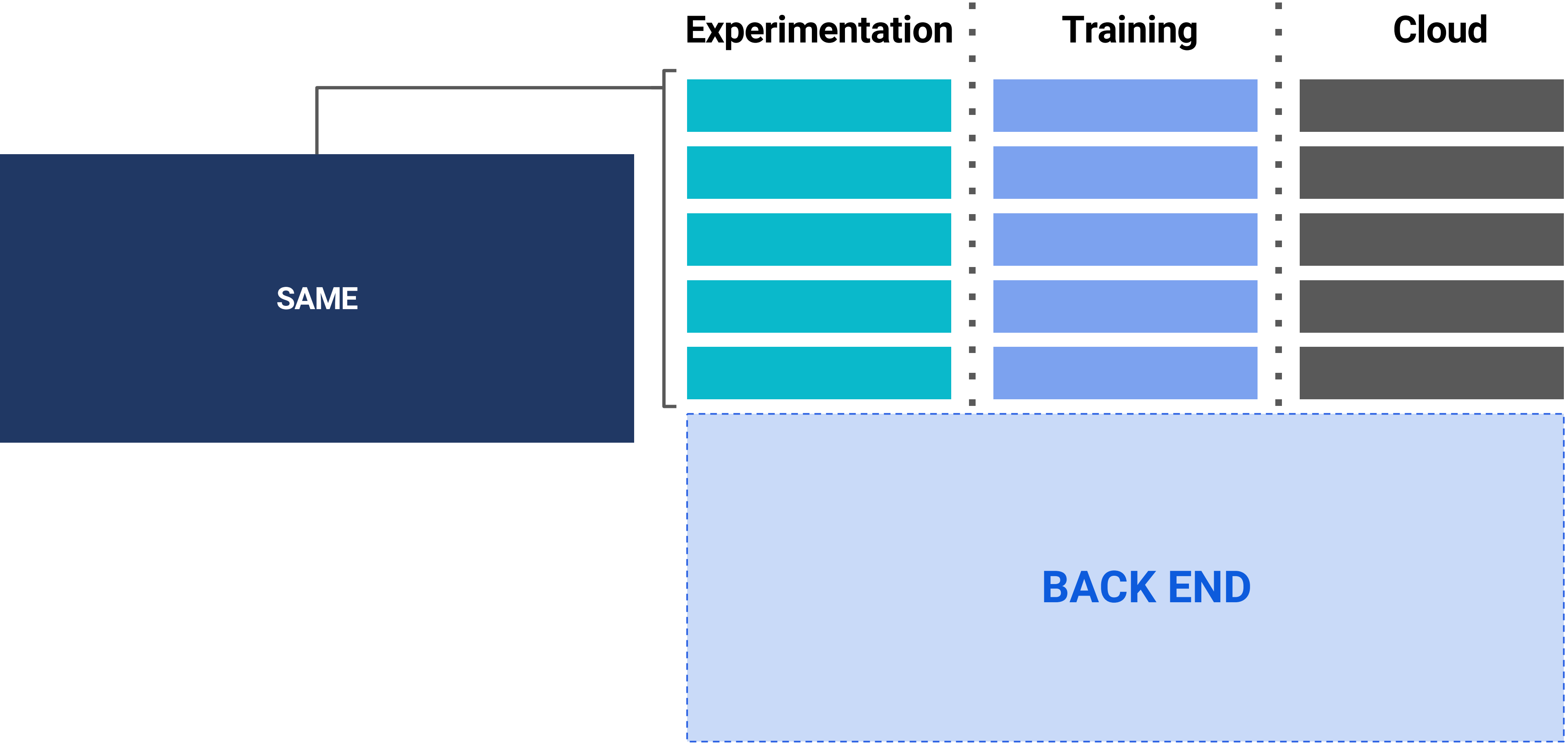
Roll-out → Serving → Monitoring → Logging

Experimentation

Training

Cloud

BACK END



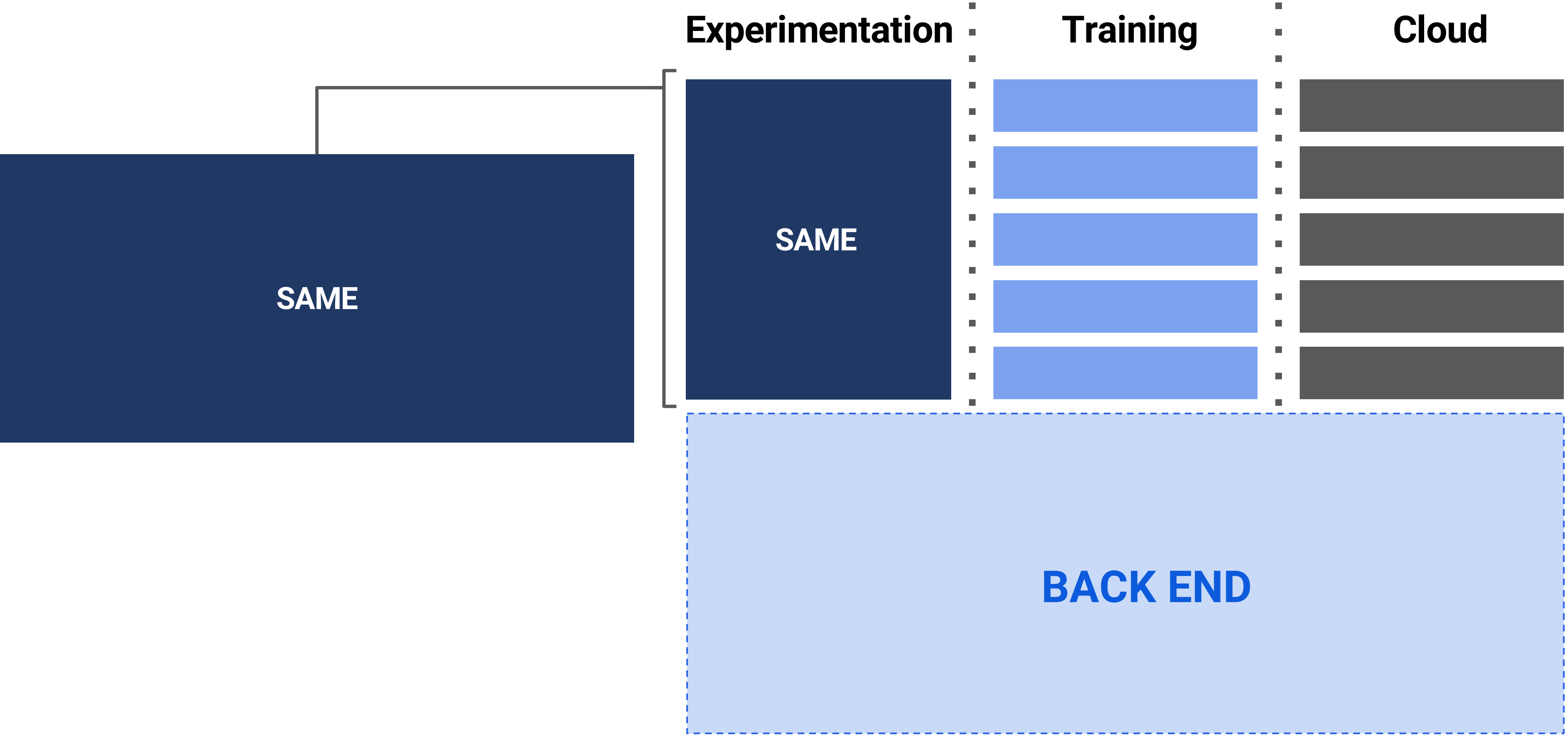
Experimentation

Training

Cloud

SAME

BACK END



Experimentation

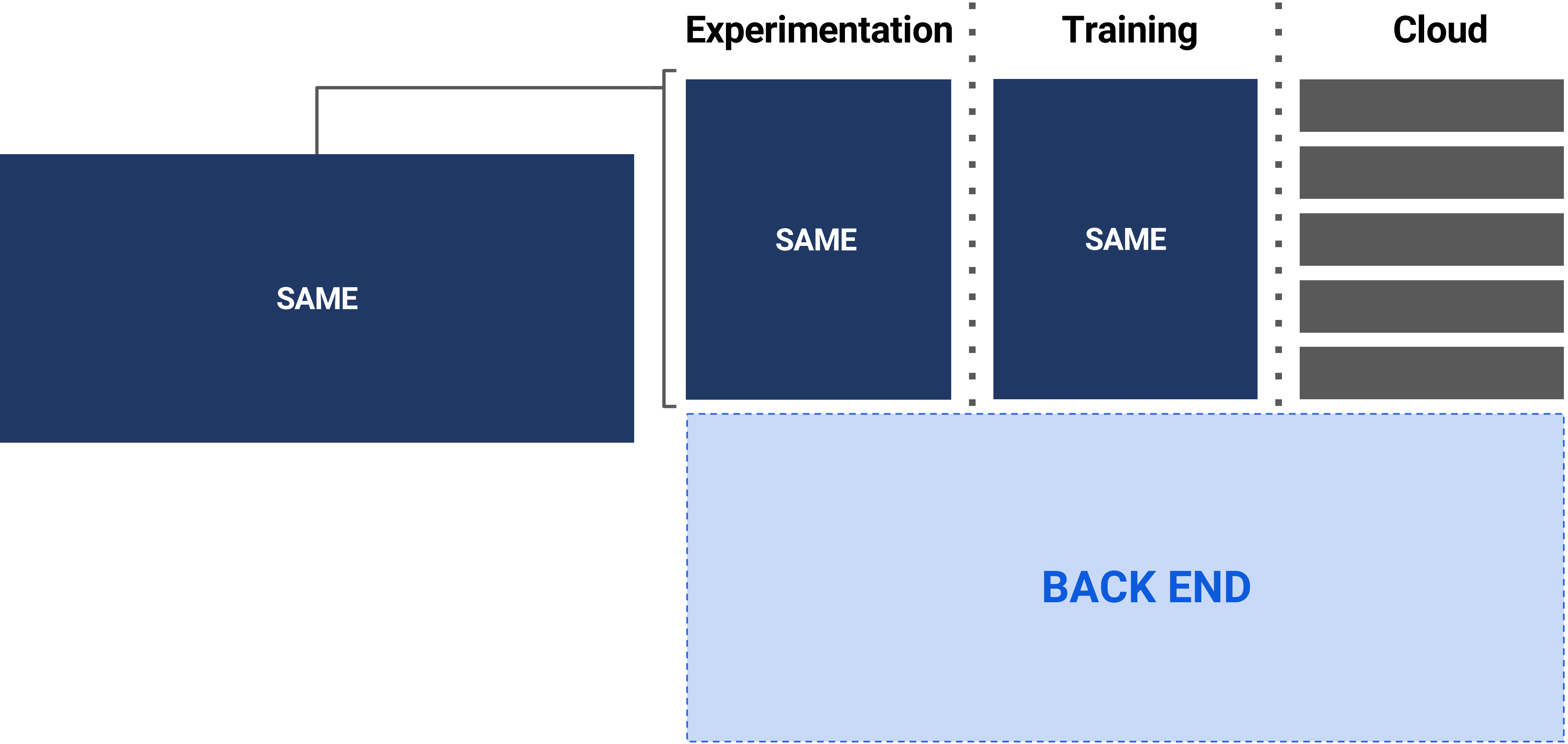
Training

Cloud

SAME

SAME

BACK END



Experimentation

Training

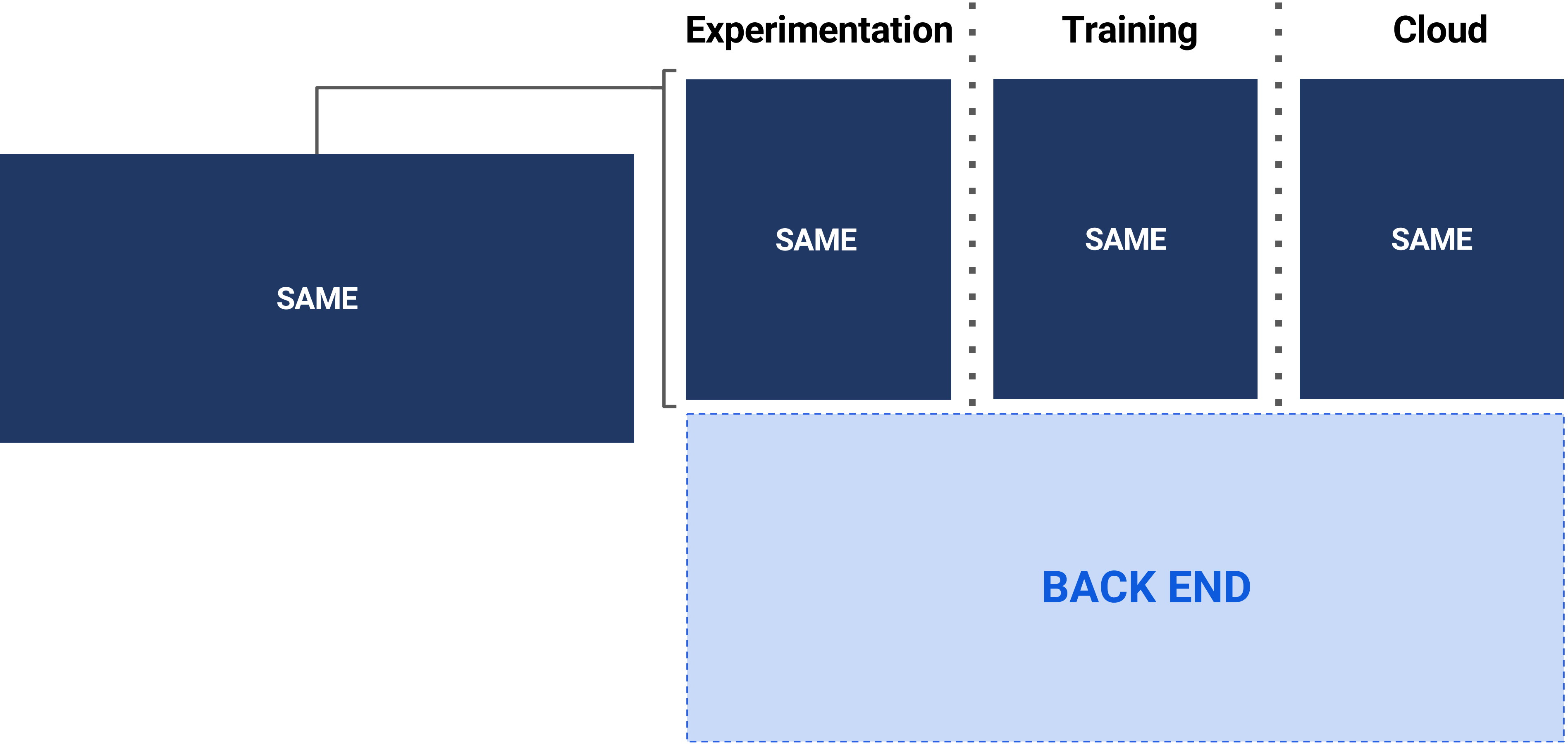
Cloud

SAME

SAME

SAME

BACK END



Experimentation

Training

Cloud

SAME

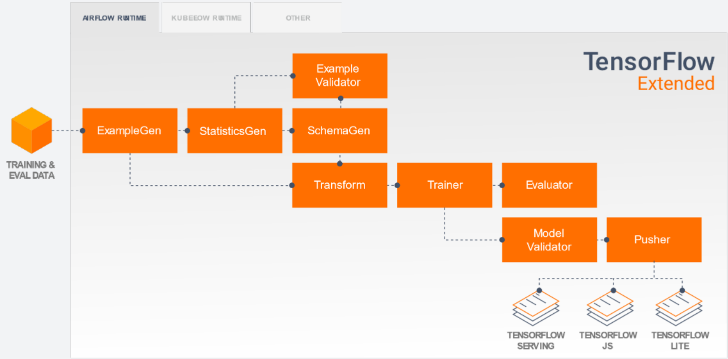
SAME

SAME

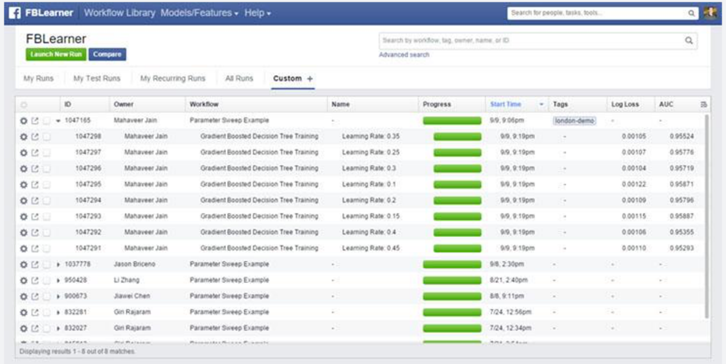
SAME

BACK END

How Big Companies Solve This… Sort Of



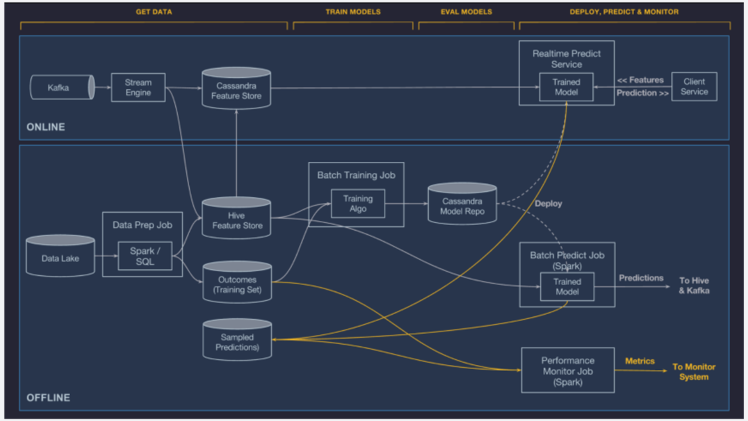
TensorFlow Extended



ID	Owner	Workflow	Name	Progress	Start Time	Tags	Log Loss	AUC
1047165	Mahaveer Jain	Parameter Sweep Example	-	🟩	9/9, 9:06pm	london-demo	-	-
1047298	Mahaveer Jain	Gradient Boosted Decision Tree Training	Learning Rate: 0.35	🟩	9/9, 9:19pm	-	0.00105	0.95524
1047297	Mahaveer Jain	Gradient Boosted Decision Tree Training	Learning Rate: 0.25	🟩	9/9, 9:19pm	-	0.00107	0.95776
1047296	Mahaveer Jain	Gradient Boosted Decision Tree Training	Learning Rate: 0.3	🟩	9/9, 9:19pm	-	0.00104	0.95719
1047295	Mahaveer Jain	Gradient Boosted Decision Tree Training	Learning Rate: 0.1	🟩	9/9, 9:19pm	-	0.00122	0.95871
1047294	Mahaveer Jain	Gradient Boosted Decision Tree Training	Learning Rate: 0.2	🟩	9/9, 9:19pm	-	0.00109	0.95796
1047293	Mahaveer Jain	Gradient Boosted Decision Tree Training	Learning Rate: 0.15	🟩	9/9, 9:19pm	-	0.00115	0.95887
1047292	Mahaveer Jain	Gradient Boosted Decision Tree Training	Learning Rate: 0.4	🟩	9/9, 9:19pm	-	0.00106	0.95355
1047291	Mahaveer Jain	Gradient Boosted Decision Tree Training	Learning Rate: 0.45	🟩	9/9, 9:19pm	-	0.00110	0.95293
1037778	Jason Briceno	Parameter Sweep Example	-	🟩	9/8, 2:30pm	-	-	-
950428	Li Zhang	Parameter Sweep Example	-	🟩	8/21, 2:40pm	-	-	-
900673	Jiawei Chen	Parameter Sweep Example	-	🟩	8/8, 9:11pm	-	-	-
832281	Giri Rajaram	Parameter Sweep Example	-	🟩	7/24, 12:56pm	-	-	-
832027	Giri Rajaram	Parameter Sweep Example	-	🟩	7/24, 12:34pm	-	-	-

Displaying results 1 - 8 out of 8 matches.

FBLearner Flow



Uber's Michelangelo

Microsoft®
Research

Microsoft Aether

Feature Stores for ML

Let's Solve It
For Everyone!

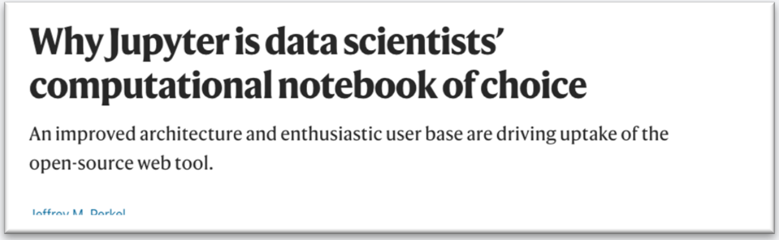
Let's Up the Level of Difficulty

The Growth of Jupyter Notebooks

- Increasingly popular and broadly adopted tool for building applications

- Made up of:
  - A file format
  - An IDE
  - An execution kernel
  - An ecosystem of associated tools

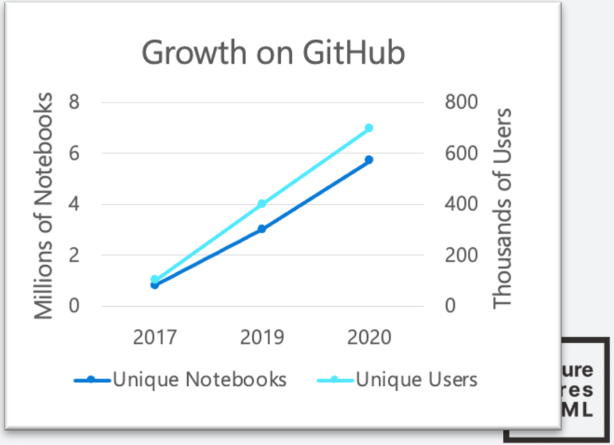
- Significant growth and footprint
  - Initial opportunity: ~75% of ~6M "simple" notebooks written by ~700k+ users on GitHub
  - Next opportunity: ~2.1M data scientists and ML engineers



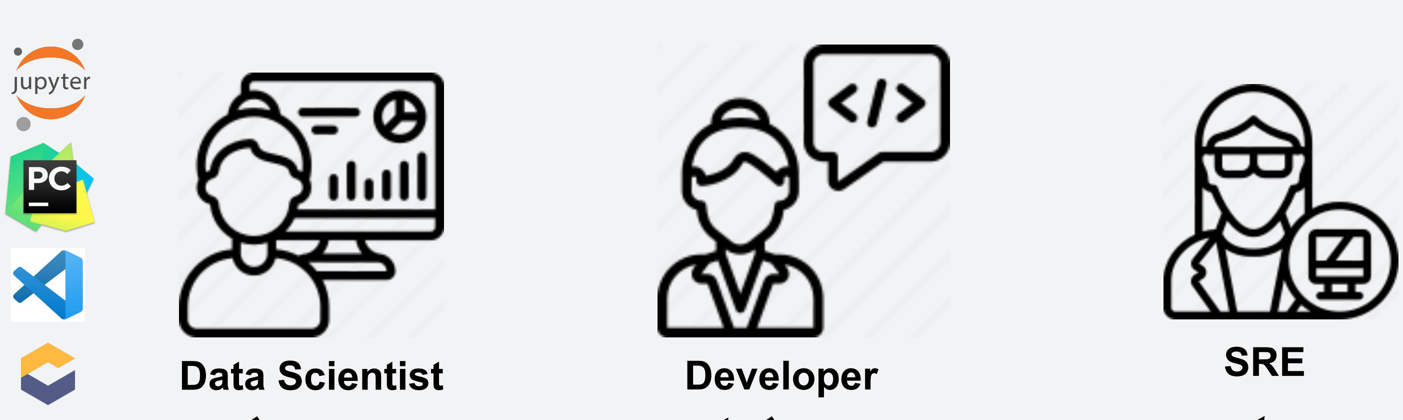
Why Jupyter is data scientists' computational notebook of choice

An improved architecture and enthusiastic user base are driving uptake of the open-source web tool.

Jeffrey M. Perkel



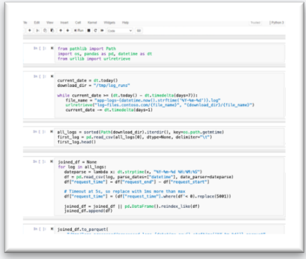
The Model Development Process



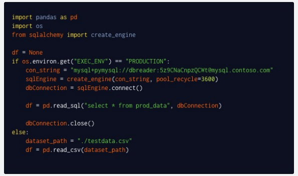
Data Scientist

Developer

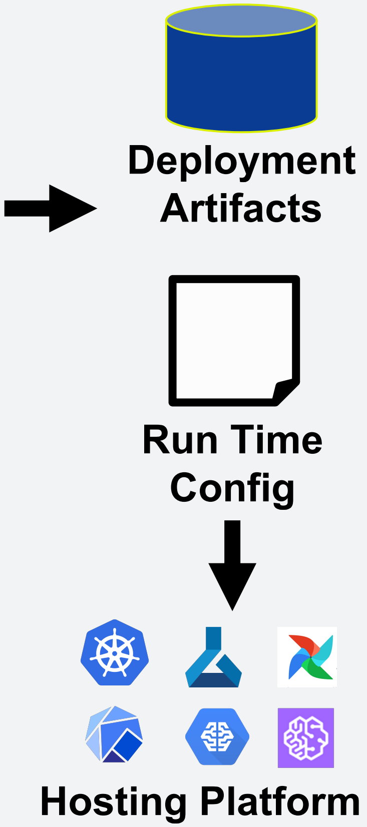
SRE



```
In [ ]: from pathlib import Path
        import os, pandas as pd, datetime as dt
        from urllib import urlretrieve

In [ ]: current_date = dt.today()
        download_dir = "/tmp/log_runs"

        while current_date >= (dt.today() - dt.timedelta(days=7)):
            file_name = "app-logs-{datetime.now().strftime('%Y-%m-%d')}.log"
            urlretrieve("log-files.contoso.com/{file_name}", "{download_dir}/{file_name}")
            current_date -= dt.timedelta(days=1)

In [ ]: all_logs = sorted(Path(download_dir).iterdir(), key=os.path.getmtime)
        first_log = pd.read_csv(all_logs[0], dtype=None, delimiter="\t")
        first_log.head()

In [ ]: joined_df = None
        for log in all_logs:
           dateparse = lambda x: dt.strptime(x, "%Y-%m-%d %H:%M:%S")
           df = pd.read_csv(log, parse_dates=["datetime"], date_parser=dateparse)
           df["request_time"] = df["request_end"] - df["request_start"]

           # Timeout at 5s, so replace with 1ms more than max
           df["request_time"] = (df["request_time"].where(df'< 0).replace(5001))

           joined_df = joined_df || pd.DataFrame().reindex_like(df)
           joined_df.append(df)

In [ ]: joined_df.to_parquet(
```



```
import pandas as pd
import os
from sqlalchemy import create_engine

df = None
if os.environ.get("EXEC_ENV") == "PRODUCTION":
    con_string = "mysql+pymysql://dbreader:5z9CNaCnpzQCWt@mysql.contoso.com"
    sqlEngine = create_engine(con_string, pool_recycle=3600)
    dbConnection = sqlEngine.connect()

    df = pd.read_sql("select * from prod_data", dbConnection)

    dbConnection.close()
else:
    dataset_path = "./testdata.csv"
    df = pd.read_csv(dataset_path)
```



Deployment Artifacts

Run Time Config

Hosting Platform

SAME Vision

Make is easy for notebook developers to build reliable workflow applications that can be developed locally and continuously deployed to production

Pillars

Make Notebooks Portable

Enable **environment portability** by serializing requirements and capturing environment details.

Support **platform portability** from laptop to hosted with adapters for popular backends.

Accelerated Path to Production via an SDK

Reduce boiler plate code reducing errors and improving performance

Simplify using **best practices** and **improve coding** for notebook developers.

Improve Pipeline Execution

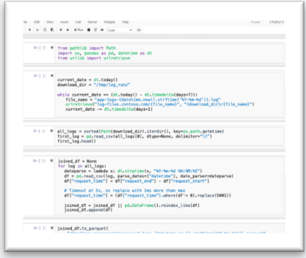
Provide **declarative setup and execution** of end-to-end pipeline.

Deliver low dependency tooling that enable **continuous deployment.**

Leverage **managed services** when for cloud deployments.

The Model Development Process

Data Scientist



```
In [ ]: from pathlib import Path
        import os, pandas as pd, datetime as dt
        from urllib import urlretrieve
```

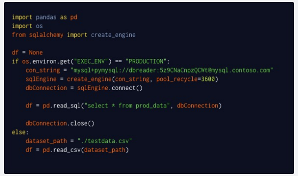
```
In [ ]: 
        current_date = dt.today()
        download_dir = "/tmp/log_runs"

        while current_date >= (dt.today() - dt.timedelta(days=7)):
            file_name = "app-logs-{datetime.now().strftime('%Y-%m-%d')}.log"
            urlretrieve("log-files.contoso.com/{file_name}", "{download_dir}/{file_name}")
            current_date -= dt.timedelta(days=1)
```

```
In [ ]: all_logs = sorted(Path(download_dir).iterdir(), key=os.path.getmtime)
        first_log = pd.read_csv(all_logs[0], dtype=None, delimiter="\t")
        first_log.head()
```

```
In [ ]: joined_df = None
        for log in all_logs:
          dateparse = lambda x: dt.strptime(x, "%Y-%m-%d %H:%M:%S")
          df = pd.read_csv(log, parse_dates=["datetime"], date_parser=dateparse)
          df["request_time"] = df["request_end"] - df["request_start"]

          # Timeout at 5s, so replace with 1ms more than max
          df["request_time"] = (df["request_time"].where(df'< 0).replace(5001))

          joined_df = joined_df || pd.DataFrame().reindex_like(df)
          joined_df.append(df)
```

```
In [ ]: joined_df.to_parquet(
```

Developer



```
import pandas as pd
import os
from sqlalchemy import create_engine

df = None
if os.environ.get("EXEC_ENV") == "PRODUCTION":
    con_string = "mysql+pymysql://dbreader:5z9CNaCnpzQCWt@mysql.contoso.com"
    sqlEngine = create_engine(con_string, pool_recycle=3600)
    dbConnection = sqlEngine.connect()

    df = pd.read_sql("select * from prod_data", dbConnection)

    dbConnection.close()
else:
    dataset_path = "./testdata.csv"
    df = pd.read_csv(dataset_path)
```

SRE

Deployment Artifacts

Run Time Config

Hosting Platform

The Model Development Process

Data Scientist

- Faster inner loop
- Less boiler plate
- Simplified best practices

- Limiting manual rewrites
- Captures data scientist intent
- Higher velocity deployment

```
from pathlib import Path
import os, pandas as pd, datetime as dt
from urllib import urlretrieve
```

```
current_date = dt.today()
download_dir = "/tmp/log_runs"

while current_date >= (dt.today() - dt.timedelta(days=7)):
    file_name = "app-logs-{datetime.now().strftime('%Y-%m-%d')}.log"
    urlretrieve("log-files.contoso.com/{file_name}", "{download_dir}/{file_name}")
    current_date -= dt.timedelta(days=1)
```

```
all_logs = sorted(Path(download_dir).iterdir(), key=os.path.getmtime)
first_log = pd.read_csv(all_logs[0], dtype=None, delimiter="\t")
first_log.head()
```

```
joined_df = None
for log in all_logs:
    dateparse = lambda x: dt.strptime(x, "%Y-%m-%d %H:%M:%S")
    df = pd.read_csv(log, parse_dates=["datetime"], date_parser=dateparse)
    df["request_time"] = df["request_end"] - df["request_start"]

    # Timeout at 5s, so replace with 1ms more than max
    df["request_time"] = (df["request_time"].where(df'= 0).replace(500)))

    joined_df = joined_df || pd.DataFrame().reindex_like(df)
    joined_df.append(df)
```

```
joined_df.to_parquet
```

- One toolchain for multiple platforms
- Native 12-factor app support
- Immutable deployments enable security/cost controls

Deployment Artifacts

Run Time Config

Hosting Platform

DEMO

Demo Summary

Make Notebooks Portable

- Prevented missing packages & errors
- Serialized environment
- Captured settings which are not included in any existing package solution

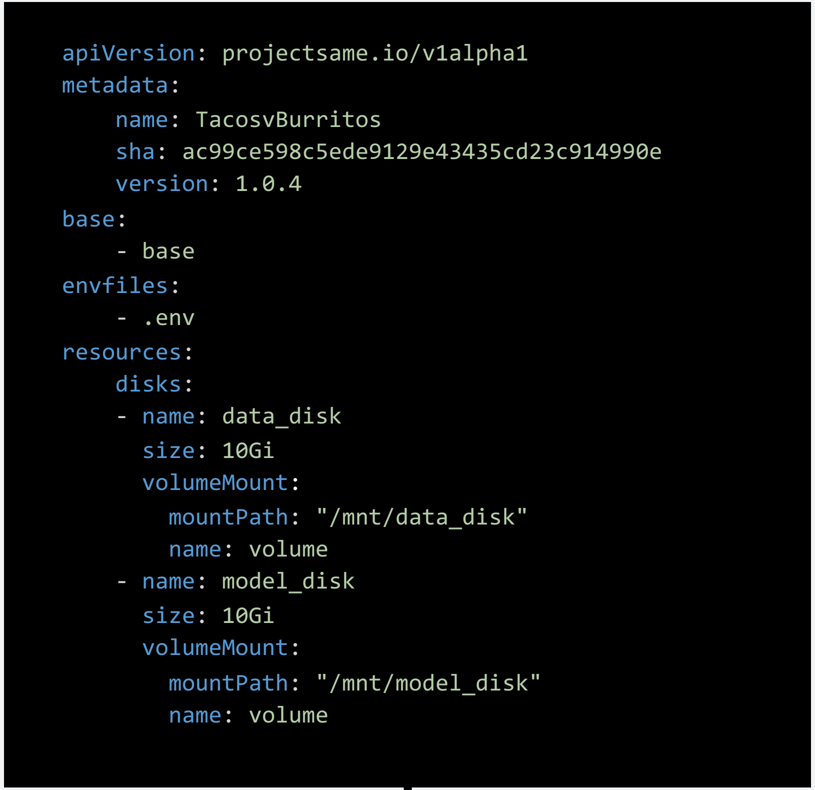
Accelerated Path to Production via an SDK

- Declarative configuration checked in with code
- No complicated (and hard to debug) control flow
- Dependency-less CLI that can be installed and used in any DevOps platform

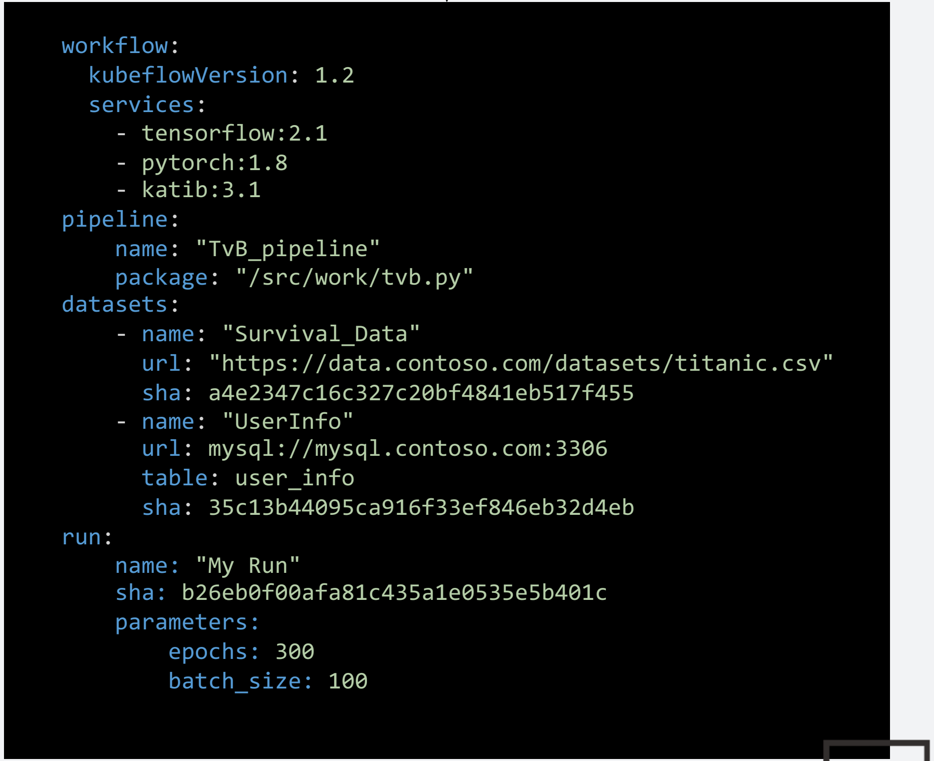
Improve Pipeline Execution

- Simplified monolithic breakup enabling caching and individual resource requests
- Use pre-built environments without dealing with Dockerfiles
- Portable fan-out/fan-in without dealing with futures/concurrency

Sample SAME Manifest



```
apiVersion: projectsame.io/v1alpha1
metadata:
    name: TacosvBurritos
    sha: ac99ce598c5ede9129e43435cd23c914990e
    version: 1.0.4
base:
    - base
envfiles:
    - .env
resources:
    disks:
    - name: data_disk
      size: 10Gi
      volumeMount:
        mountPath: "/mnt/data_disk"
        name: volume
    - name: model_disk
      size: 10Gi
      volumeMount:
        mountPath: "/mnt/model_disk"
        name: volume
```



```
workflow:
  kubeflowVersion: 1.2
  services:
    - tensorflow:2.1
    - pytorch:1.8
    - katib:3.1
pipeline:
    name: "TvB_pipeline"
    package: "/src/work/tvb.py"
datasets:
    - name: "Survival_Data"
      url: "https://data.contoso.com/datasets/titanic.csv"
      sha: a4e2347c16c327c20bf4841eb517f455
    - name: "UserInfo"
      url: mysql://mysql.contoso.com:3306
      table: user_info
      sha: 35c13b44095ca916f33ef846eb32d4eb
run:
    name: "My Run"
    sha: b26eb0f00afa81c435a1e0535e5b401c
    parameters:
        epochs: 300
        batch_size: 100
```

Feature Stores for ML

Customer Feedback

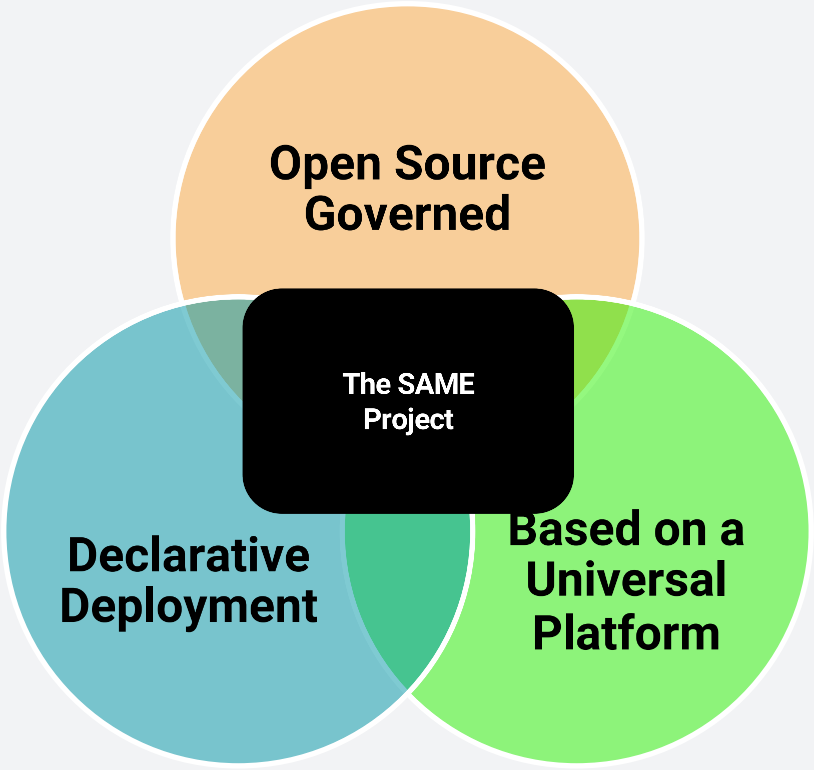
- "We still have to do a lot of work to stitch together all of the artifacts that make up that point in time version of the model (raw data, annotated data, python libraries and versions, training parameters, training metrics, model binaries, etc.)." – **Head of ML at Large Consulting Company**

- "SAME isn't just useful across organizations, it's useful INSIDE an organization for tracking over time" – **SWE, Cloud Engineering**

- "We're looking for reusable experiments; infrastructure and workflows as code. We'll build it ourselves, but we don't want to." – **Head of ML, Retail**

- "I don't want the data scientist to think about infrastructure or platforms. Based on the experiment, everything should automatically provision" – **ML Lead, Storage Infra Co**

Center of the Venn Diagram



Feature Stores for ML

What YOU Can Do

1. Come join our community
   - Website – https://sameproject.org (or https://s9t.io if you're lazy)
   - Mail - https://aka.ms/same-group
   - Slack - https://aka.ms/SAMEProjectSlack
2. Come talk to us about YOUR needs
   - We're looking for key folks to co-develop this with
   - It's not going to be production ready for O(months), but the underlying components ARE production ready
   - Just gathering info helps too!
3. Come join our repo!
   - https://github.com/same-project/same-cli
   - Try it out (build instructions included)
   - Complain about missing features
   - EXPERTS ONLY: Add your own

Thank you!

Do you have any questions?

David Aronchick
aronchick@gmail.com

Linkedin URL

Twitter handle