Microsoft Azure

Azure Machine Learning

Azure Machine Learning

IMPT NOTICE:

- If you choose to participate in this session using Microsoft Teams, your name, email address, phone number, and/or title may be viewable by other session participants.

- **Please note that the training will not and cannot be recorded in alignment with Microsoft's policies**

Check In

Microsoft Azure

How to get Machine Learning Models to work for you organization !!

Agenda

- Maturity Model
- Azure Machine Learning
- Azure Databricks & MLflow
- Azure Synapse Analytics - AML
- Q&A (15 minutes)

Mapping Your AI Maturity

Foundational

Questioning what AI is and how to apply it

- Wrong expectations or disappointment
- Low digitization
- Basic analytical capabilities

Approaching

Hopeful on AI and its promise

- Digitization under way
- Looking to increase or optimize processes
- Cautious about disruption

Aspirational

Experimented and applied AI

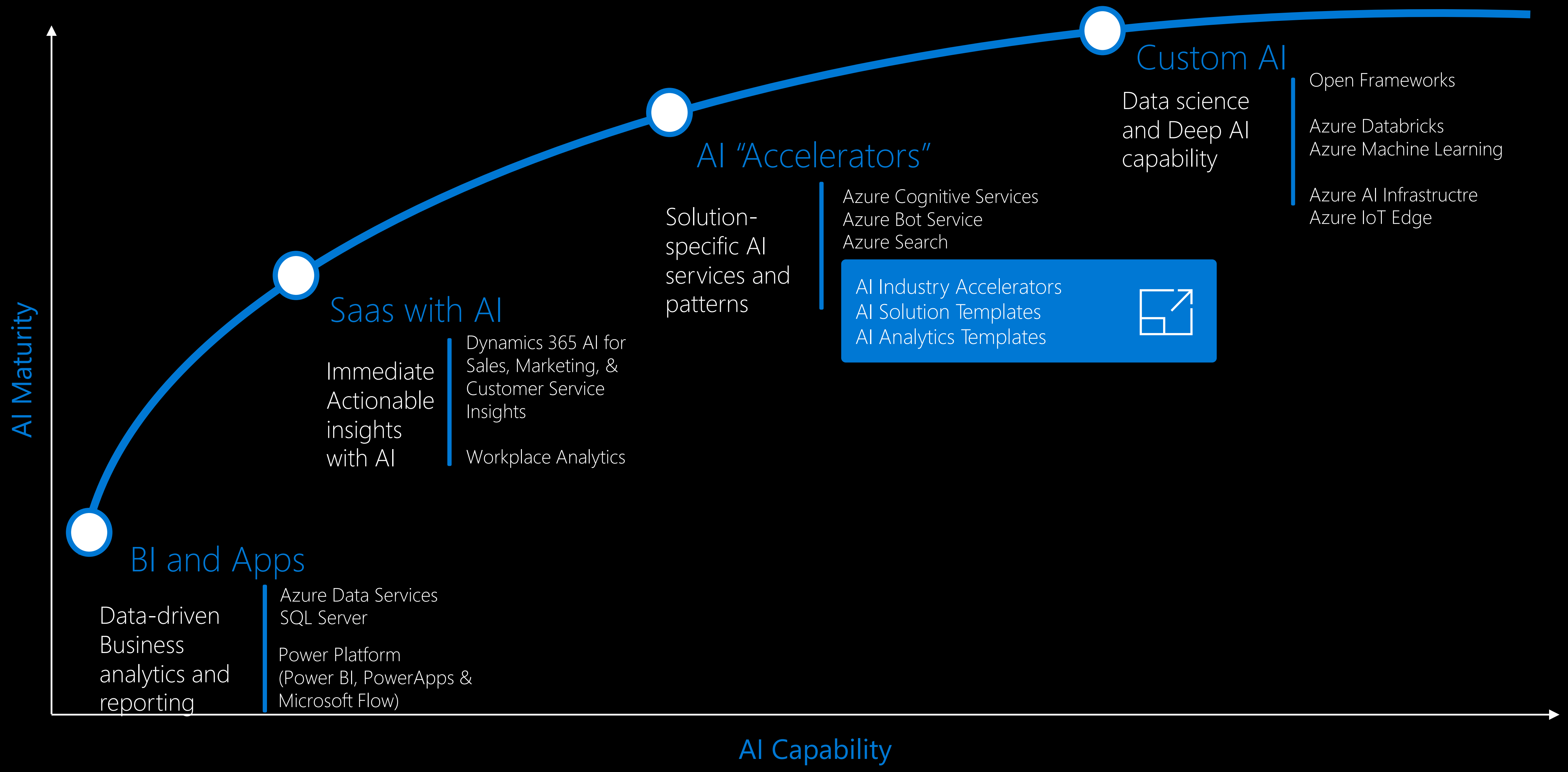
- High digitization
- Desires new business models
- Achieved a data culture

Mature

Emerging Data science and operational capability

- Understands model lifecycle and management
- Building a foundational data architecture

The AI Journey – Where to Start



Azure AI

Microsoft 365 | Dynamics 365 | Power Platform | LinkedIn | Bing

1P (Microsoft SaaS Solutions)

ISV Product | SI Solution | Customer Implementation

3P (Customer & Partner Solutions)

Azure AI

Developers & Data Scientists

Responsible Development and Use of AI

Scenario specific AI services

Bot Service
Cognitive Search
Form Recognizer
Video Indexer
Metrics Advisor
"Transcription Intelligence"

Customizable AI models

Cognitive Services

Vision
Speech
Language
Decision

ML platform

Azure Machine Learning

61 Regions
90+ Compliance Offerings
95% Fortune 500 use Azure
$1B Security investment per year

Azure

What makes up Machine Learning Lifecycle

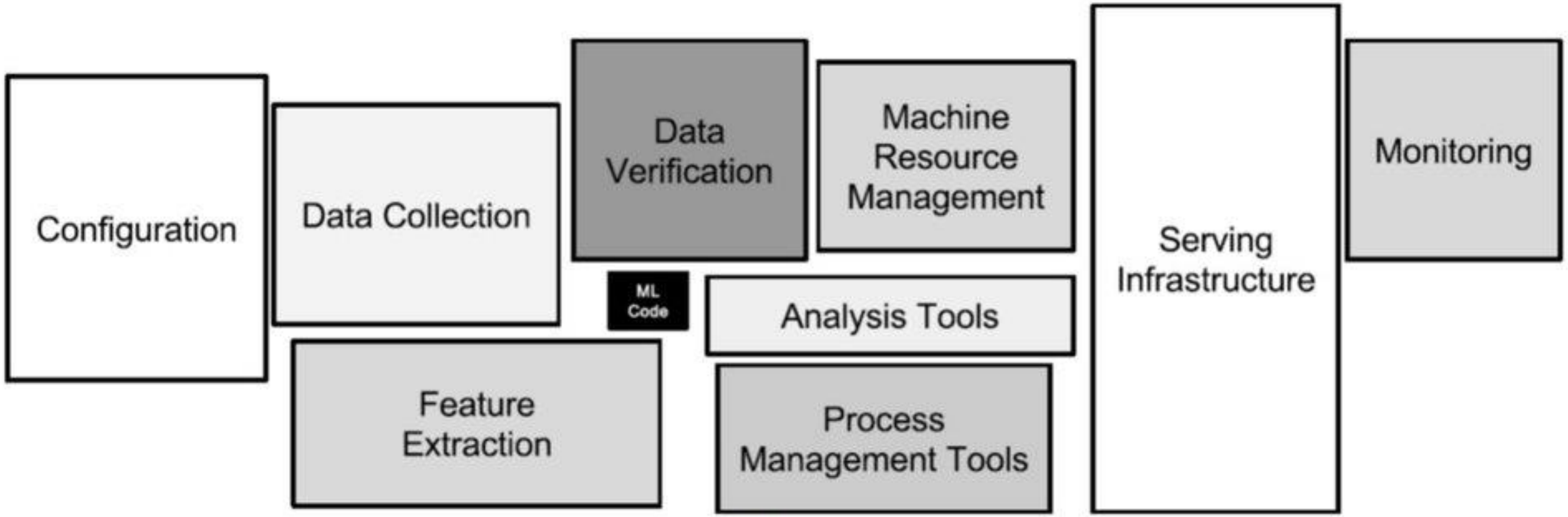


Figure 1: Only a small fraction of real-world ML systems is composed of the ML code, as shown by the small black box in the middle. The required surrounding infrastructure is vast and complex.

Azure Machine Learning

For all skill levels

Automated ML + drag & drop + code first

Industry leading MLOps

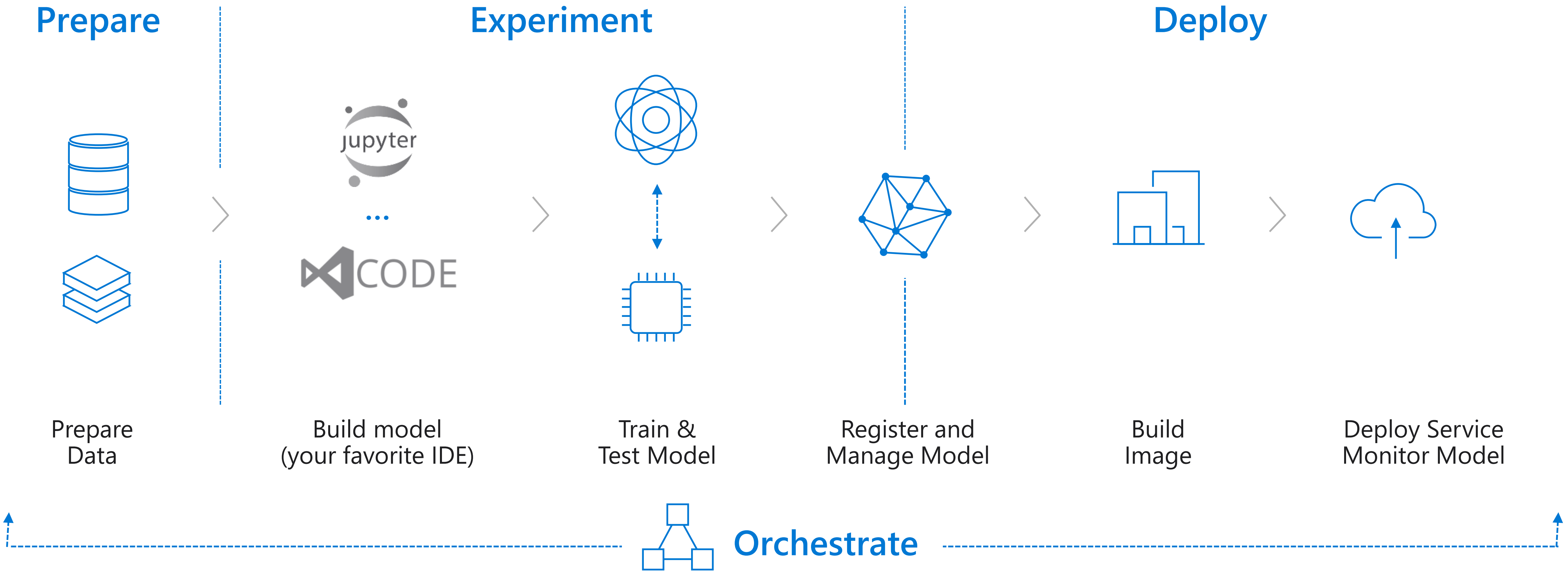
Integrated with Azure DevOps

Open

Any tool + any framework

Machine Learning

Typical E2E Process



What is Azure Machine Learning service?

Set of Azure Cloud Services + **Python SDK**

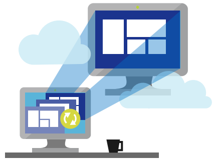
That enables you to:

- ✓ Prepare Data
- ✓ Build Models
- ✓ Train Models
- ✓ Manage Models
- ✓ Track Experiments
- ✓ Deploy Models

Azure Machine Learning service

Bring AI to everyone with an end-to-end, scalable, trusted platform

Boost your data science productivity



Increase your rate of experimentation

Deploy and manage your models everywhere

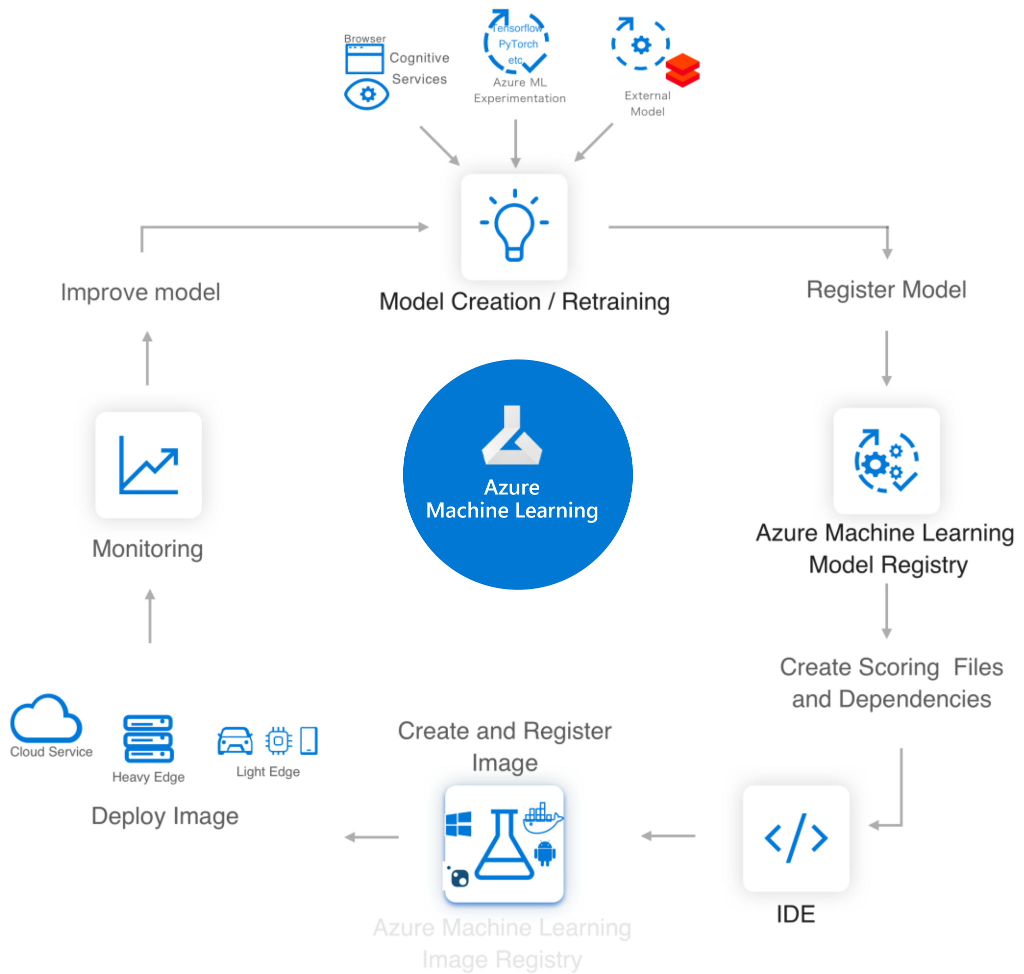
Built with your needs in mind

- Automated machine learning
- Managed compute
- Simple deployment
- DevOps for machine learning
- Support for open source frameworks
- Tool agnostic Python SDK

Seamlessly integrated with the Azure Portfolio

Azure ML service

Lets you easily implement this AI/ML Lifecycle



Workflow Steps

Develop machine learning training scripts in Python.

Create and configure a compute target.

Submit the scripts to the configured compute target to run in that environment. During training, the compute target stores run records to a datastore. There the records are saved to an experiment.

Query the experiment for logged metrics from the current and past runs. If the metrics do not indicate a desired outcome, loop back to step 1 and iterate on your scripts.

Once a satisfactory run is found, register the persisted model in the model registry.

Develop a scoring script.

Create an Image and register it in the image registry.

Deploy the image as a web service in Azure.

Azure Machine Learning: Technical Details

Azure ML service

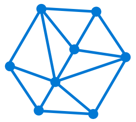
Key Artifacts

Workspace

- Models
- Experiments
- Pipelines
- Compute Target
- Images
- Deployment
- Data Stores

Azure ML service Artifacts

Models and Model Registry



Model

A machine learning model is an artifact that is created by your training process. You use a model to get predictions on new data.

A model is produced by a **run** in Azure Machine Learning.

> Note: You can also use a model trained outside of Azure Machine Learning.

Azure Machine Learning service is framework agnostic — you can use any popular machine learning framework when creating a model.

A model can be registered under an Azure Machine Learning service workspace

Model Registry

Keeps track of all the models in your Azure Machine Learning service workspace.

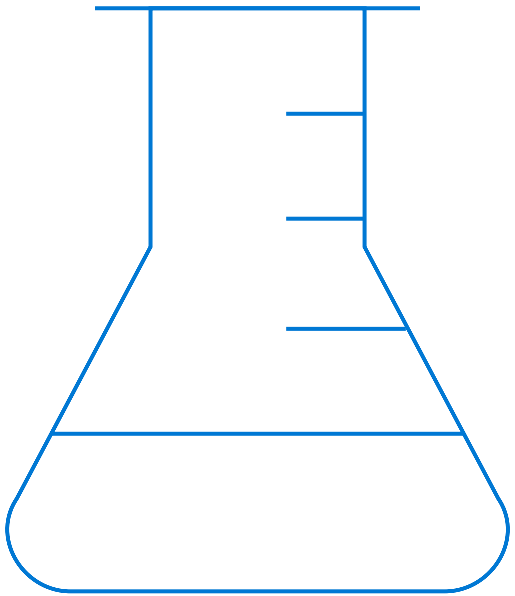
Models are identified by name and version.

You can provide additional metadata tags when you register the model, and then use these tags when searching for models.

You cannot delete models that are being used by an image.

Azure ML Artifacts

Runs and Experiments



Experiment

Grouping of many runs from a given script.

- Always belongs to a workspace.
- Stores information about runs

Run

Produced when you submit a script to train a model. Contains:

- Metadata about the run (timestamp, duration etc.)
- Metrics logged by your script.
- Output files autocollected by the experiment, or explicitly uploaded by you.
- A snapshot of the directory that contains your scripts, prior to the run.

Run configuration

A set of instructions that defines how a script should be run in a given compute target.

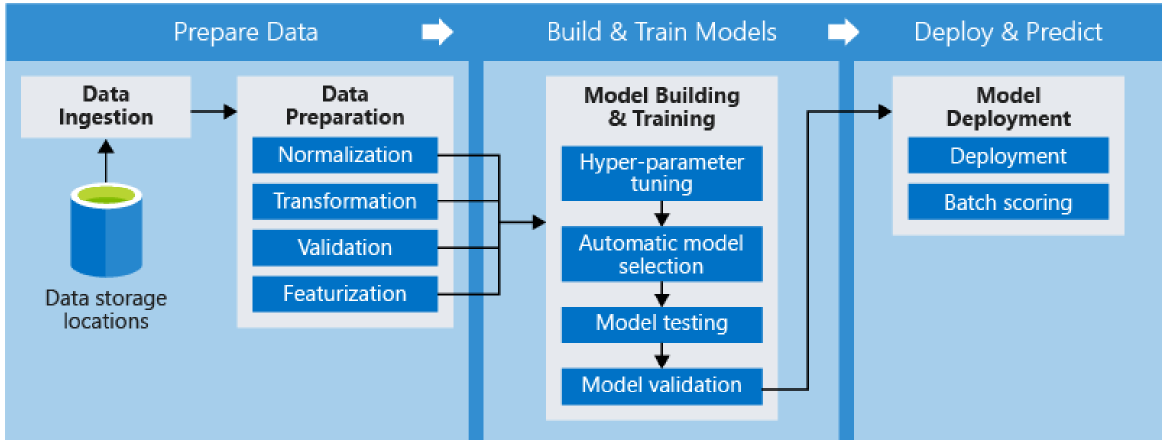
Azure ML Artifact

Pipeline

An Azure ML pipeline consists of a number of steps, where each step can be performed independently or as part of a single deployment command.

A step is a computational unit in the pipeline.

Diagram shows an example pipeline with multiple steps.



Azure ML pipelines enables data scientists, data engineers, and IT professionals to collaborate on the steps involved in: Data preparation, Model training, Model evaluation, Deployment

Azure ML Pipelines

Advantages

Advantage	Description
Unattended runs	Schedule a few steps to run in parallel or in sequence in a reliable and unattended manner. Since data prep and modeling can last days or weeks, you can now focus on other tasks while your pipeline is running.
Mixed and diverse compute	Use multiple pipelines that are reliably coordinated across heterogeneous and scalable computes and storages. Individual pipeline steps can be run on different compute targets, such as HDInsight, GPU Data Science VMs, and Databricks.
Reusability	Pipelines can be templatized for specific scenarios such as retraining and batch scoring. They can be triggered from external systems via simple REST calls.
Tracking and versioning	Instead of manually tracking data and result paths as you iterate, use the pipelines SDK to explicitly name and version your data sources, inputs, and outputs as well as manage scripts and data separately for increased productivity

Azure ML Pipeline

Python SDK

The Azure Machine Learning SDK offers imperative constructs for sequencing and parallelizing the steps in your pipelines when no data dependency is present.

Using declarative data dependencies, you can optimize your tasks.

The SDK includes a framework of pre-built modules for common tasks such as data transfer and model publishing.

The framework can be extended to model your own conventions by implementing custom steps that are reusable across pipelines.

Compute targets and storage resources can also be managed directly from the SDK.

Pipelines can be saved as templates and can be deployed to a REST endpoint so you can schedule batch-scoring or retraining jobs

Azure ML Artifact

Compute Target

Compute Targets are the compute resources used to run training scripts or host your model when deployed as a web service.

They can be created and managed using the Azure Machine Learning SDK or CLI.

You can attach to existing resources.

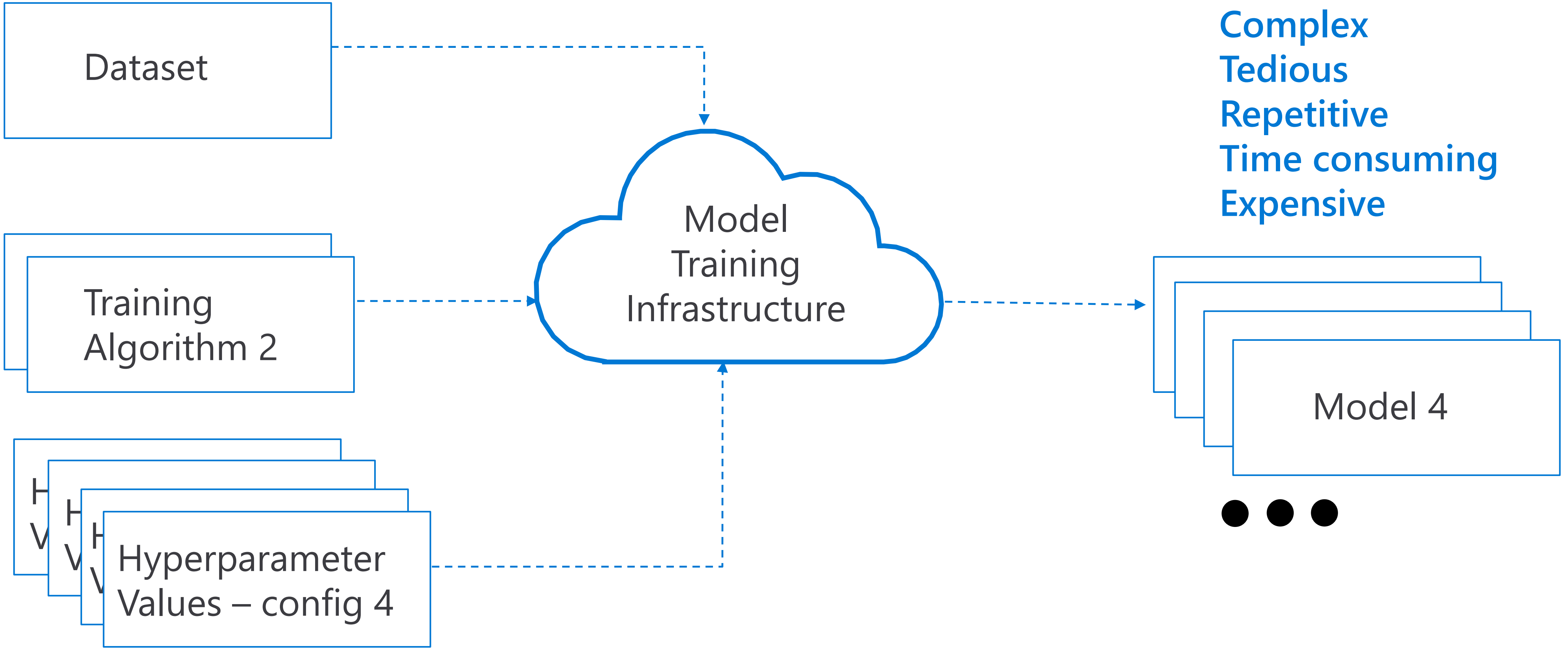
You can start with local runs on your machine, and then scale up and out to other environments.

Currently supported compute targets

Compute Target	Training	Deployment
Local Computer	✓	
A Linux VM in Azure (such as the Data Science Virtual Machine)	✓	
Azure ML Compute	✓	
Azure Databricks	✓	
Azure Data Lake Analytics	✓	
Apache Spark for HDInsight	✓	
Azure Container Instance		✓
Azure Kubernetes Service		✓
Azure IoT Edge		✓
Field-programmable gate array (FPGA)		✓

Azure Automated Machine Learning ‘simplifies’ the creation and selection of the optimal model

Typical ‘manual’ approach to hyperparameter tuning

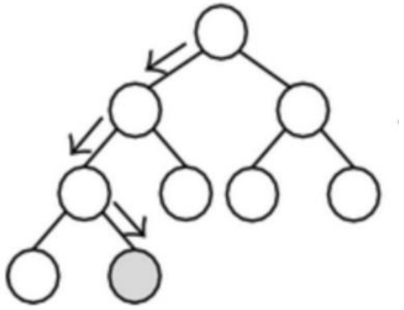


What are Hyperparameters?

Adjustable parameters that govern model training

Chosen prior to training, stay constant during training

Model performance heavily depends on hyperparameter



Setting
Number Of Leaves
Minimum Leaf Instances
Learning Rate
Number Of Trees

Number of leaves	Minimum leaf instances	Learning rate	Number of trees
8	10	0.1	500
8	1	0.05	500
8	1	0.2	100
32	1	0.05	100
8	10	0.2	100
32	1	0.025	500
8	10	0.05	500
32	1	0.1	100
8	1	0.025	500
8	50	0.05	500
32	10	0.025	500
8	50	0.025	500
32	10	0.05	100
8	10	0.025	500
32	10	0.2	20
8	1	0.1	500
32	10	0.1	100
8	1	0.1	100
8	10	0.1	100

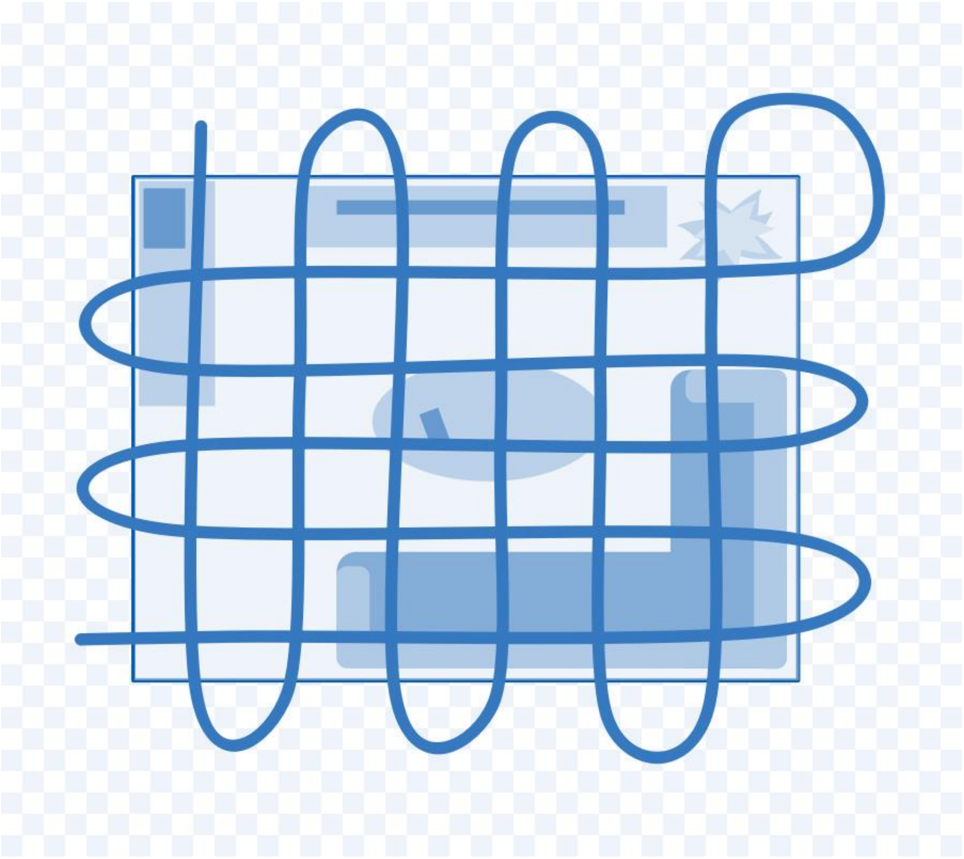
Challenges with Hyperparameter Selection

The search space to explore—i.e. evaluating all possible combinations—is huge.

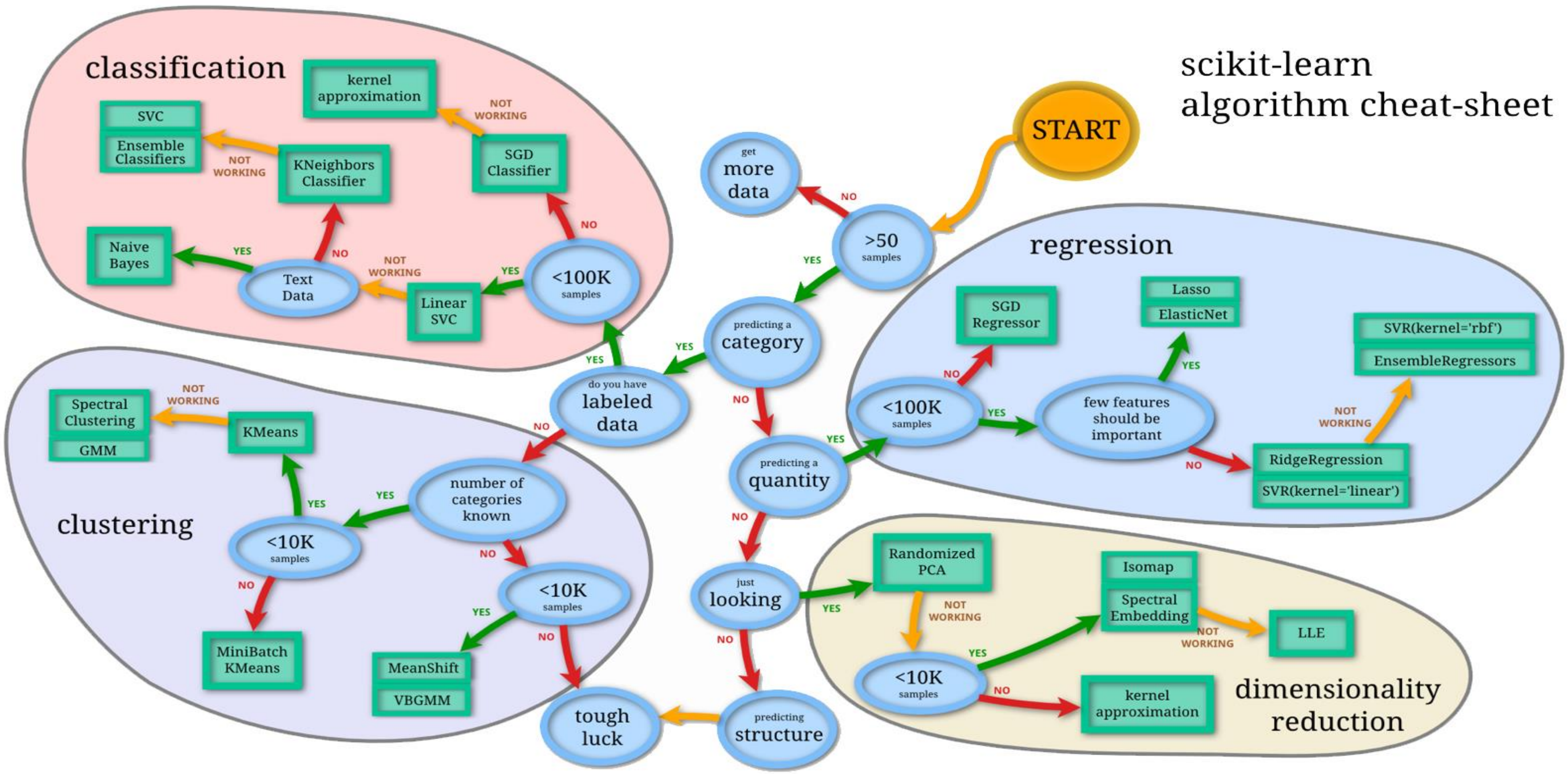
Sparsity of good configurations.
Very few of all possible configurations are optimal.

Evaluating each configuration is resource and time consuming.

Time and resources are limited.



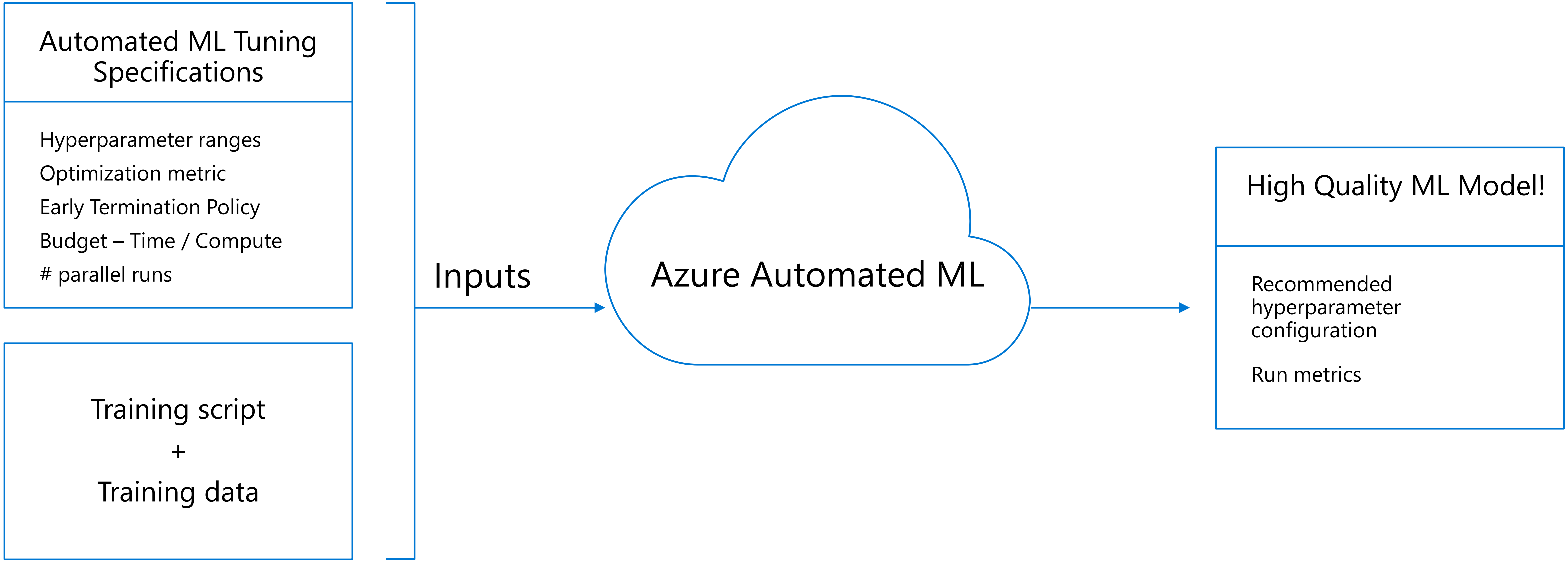
Complexity of Machine Learning



Source: http://scikit-learn.org/stable/tutorial/machine_learning_map/index.html

Automated ML

Conceptual Overview



Automated ML

Benefits Overview

Azure Automated ML lets you

Automate the exploration process

Use resources more efficiently

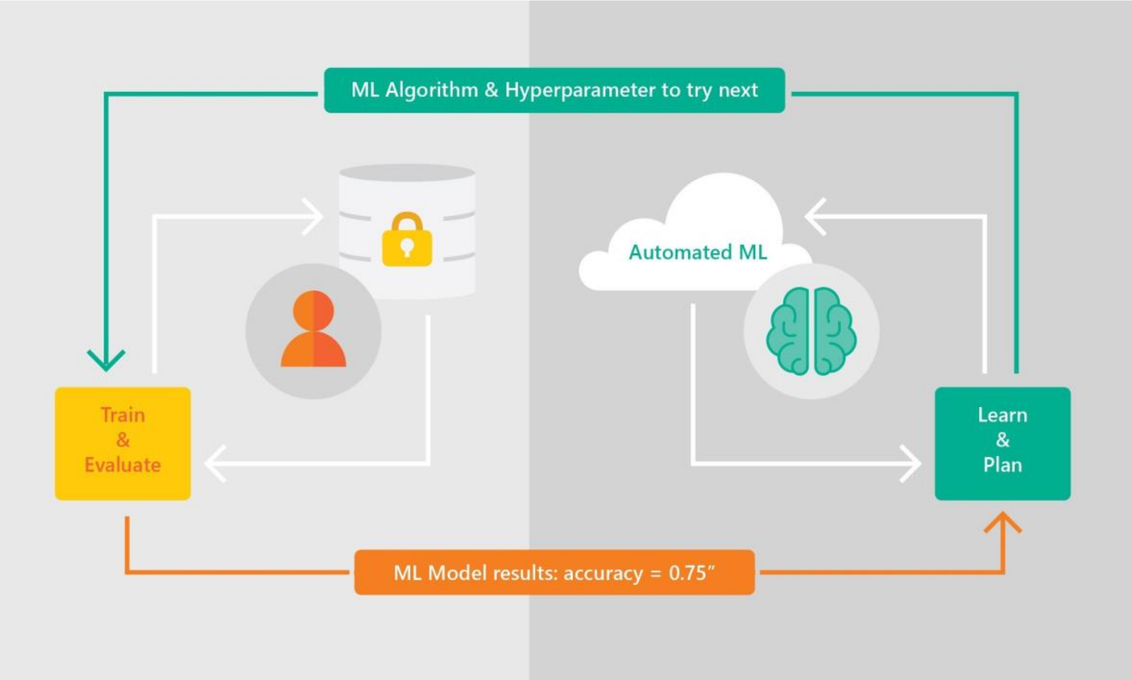
Optimize model for desired outcome

Control resource budget

Apply it to different models and learning domains

Pick training frameworks of choice

Visualize all configurations in one place



Automated ML

Current Capabilities

<table>
<tr><th colspan="2">Category</th><th>Value</th></tr>
<tr><td colspan="2">ML Problem Spaces</td><td>Classification
Regression
Forecasting</td></tr>
<tr><td colspan="2">Frameworks</td><td>Scikit Learn</td></tr>
<tr><td colspan="2">Languages</td><td>Python</td></tr>
<tr><td colspan="2">Data Type and Data Formats</td><td>Numerical
Text
Scikit-learn supported data formats (Numpy, Pandas)</td></tr>
<tr><td colspan="2">Data sources</td><td>Local Files, Azure Blob Storage</td></tr>
<tr><td rowspan="2">Compute Target</td><td>Automated Hyperparameter Tuning</td><td>Azure ML Compute (Batch AI), Azure Databricks</td></tr>
<tr><td>Automated Model Selection</td><td>Local Compute, Azure ML Compute (Batch AI), Azure Databricks</td></tr>
</table>

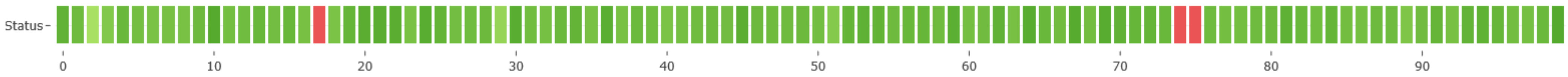
Automated ML

Algorithms Supported

Classification	Regression
sklearn.linear_model.LogisticRegression	sklearn.linear_model.ElasticNet
sklearn.linear_model.SGDClassifier	sklearn.ensemble.GradientBoostingRegressor
sklearn.naive_bayes.BernoulliNB	sklearn.tree.DecisionTreeRegressor
sklearn.naive_bayes.MultinomialNB	sklearn.neighbors.KNeighborsRegressor
sklearn.svm.SVC	sklearn.linear_model.LassoLars
sklearn.svm.LinearSVC	sklearn.linear_model.SGDRegressor
sklearn.calibration.CalibratedClassifierCV	sklearn.ensemble.RandomForestRegressor
sklearn.neighbors.KNeighborsClassifier	sklearn.ensemble.ExtraTreesRegressor
sklearn.tree.DecisionTreeClassifier	lightgbm.LGBMRegressor
sklearn.ensemble.RandomForestClassifier	
sklearn.ensemble.ExtraTreesClassifier	
sklearn.ensemble.GradientBoostingClassifier	
lightgbm.LGBMClassifier	

Azure Automated ML – Sample Output

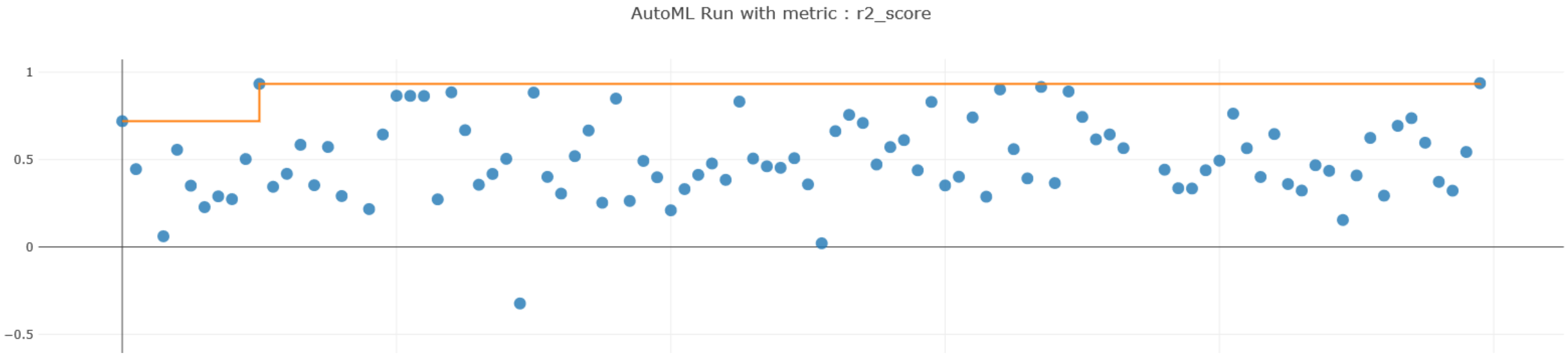
AutoML_ab755820-4bfd-4e8a-8b4b-9e0a2446b1c2:
Status: Completed



Iteration	Pipeline	Iteration metric	Best metric	Status	Duration	Started	Run Id
99	Ensemble	0.93702349	0.93702349	Completed	0:02:18	Dec 4, 2018 12:18 AM	
10	MaxAbsScaler, LightGBM	0.93289307	0.93289307	Completed	0:01:22	Dec 3, 2018 7:49 PM	
67	SparseNormalizer, LightGBM	0.9154763	0.93289307	Completed	0:01:31	Dec 3, 2018 10:19 PM	
64	MaxAbsScaler, LightGBM	0.90148724	0.93289307	Completed	0:01:24	Dec 3, 2018 10:09 PM	
69	MaxAbsScaler, LightGBM	0.88975241	0.93289307	Completed	0:00:55	Dec 3, 2018 10:22 PM	

Pages: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 ... Next Last 5 per page

r2_score



Automated ML

Use via the Python SDK

jupyter 102.auto-ml-regression (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python [default]

Run | Code

Instantiate Auto ML Regressor

Instantiate a AutoML Object This creates an Experiment in Azure ML. You can reuse this objects to trigger multiple runs. Each run will be part of the same experiment.

Property	Description
primary_metric	This is the metric that you want to optimize. Auto ML Regressor supports the following primary metrics *spearman_correlation* *normalized_root_mean_squared_error* *r2_score*
max_time_sec	Time limit in seconds for each iterations
iterations	Number of iterations. In each iteration Auto ML Classifier trains the data with a specific pipeline
num_cross_folds	Cross Validation split

In [5]:

```
from azureml.train.automl import AutoMLConfig

automl_config = AutoMLConfig(task = 'regression',
                             debug_log = 'automl_errors.log',
                             primary_metric = 'spearman_correlation',
                             max_time_sec = 12000,
                             iterations = 10,
                             n_cross_validations = 3,
                             verbosity = logging.INFO,
                             X = X,
                             y = y,
                             path=project_folder)
```

Training the Model

You can call the fit method on the AutoML instance and pass the run configuration. For Local runs the execution is synchronous. Depending on the data and number of iterations this can run for while. You will see the currently running iterations printing to the console.

fit method on Auto ML Regressor triggers the training of the model. It can be called with the following parameters

Parameter	Description
X	(sparse) array-like, shape = [n_samples, n_features]
y	(sparse) array-like, shape = [n_samples,], [n_samples, n_classes] Multi-class targets. An indicator matrix turns on multilabel classification.
compute_target	Indicates the compute used for training. *local* indicates train on the same compute which hosts the jupyter notebook. For DSVM and Batch AI please refer to the relevant notebooks.
show_output	True/False to turn on/off console output

In [6]:

```
local_run = experiment.submit(automl_config, show_output=True)
```

```
Parent Run ID: AutoML_e7a4236e-8935-4e93-888d-1ea8310a6b22
*********************************************************************************************
ITERATION: The iteration being evaluated.
PIPELINE: A summary description of the pipeline being evaluated.
DURATION: Time taken for the current iteration.
METRIC: The result of computing score on the fitted pipeline.
BEST: The best observed score thus far.
*********************************************************************************************

 ITERATION   PIPELINE                               DURATION              METRIC      BEST
         0    Normalize extra trees regressor       0:00:12.069893         0.688     0.688
         1    Normalize lightGBM regressor          0:00:11.192919         0.597     0.688
         2    Normalize Elastic net                 0:00:09.866233         0.689     0.689
         3    Scale 0/1 lightGBM regressor          0:00:10.069764         0.656     0.689
         4    Robust Scaler kNN regressor           0:00:09.090668         0.598     0.689
         5    Normalize lightGBM regressor          0:00:12.562876         0.649     0.689
         6    Robust Scaler kNN regressor           0:00:09.361137         0.600     0.689
         7    Normalize SGD regressor               0:00:09.010672         0.070     0.689
         8    Scale 0/1 extra trees regressor       0:00:10.442752         0.685     0.689
         9    Robust Scaler Gradient boosting regres0:00:09.567582         0.651     0.689
```

Automated Hyperparameter Tuning

Automated Hyperparameter Tuning

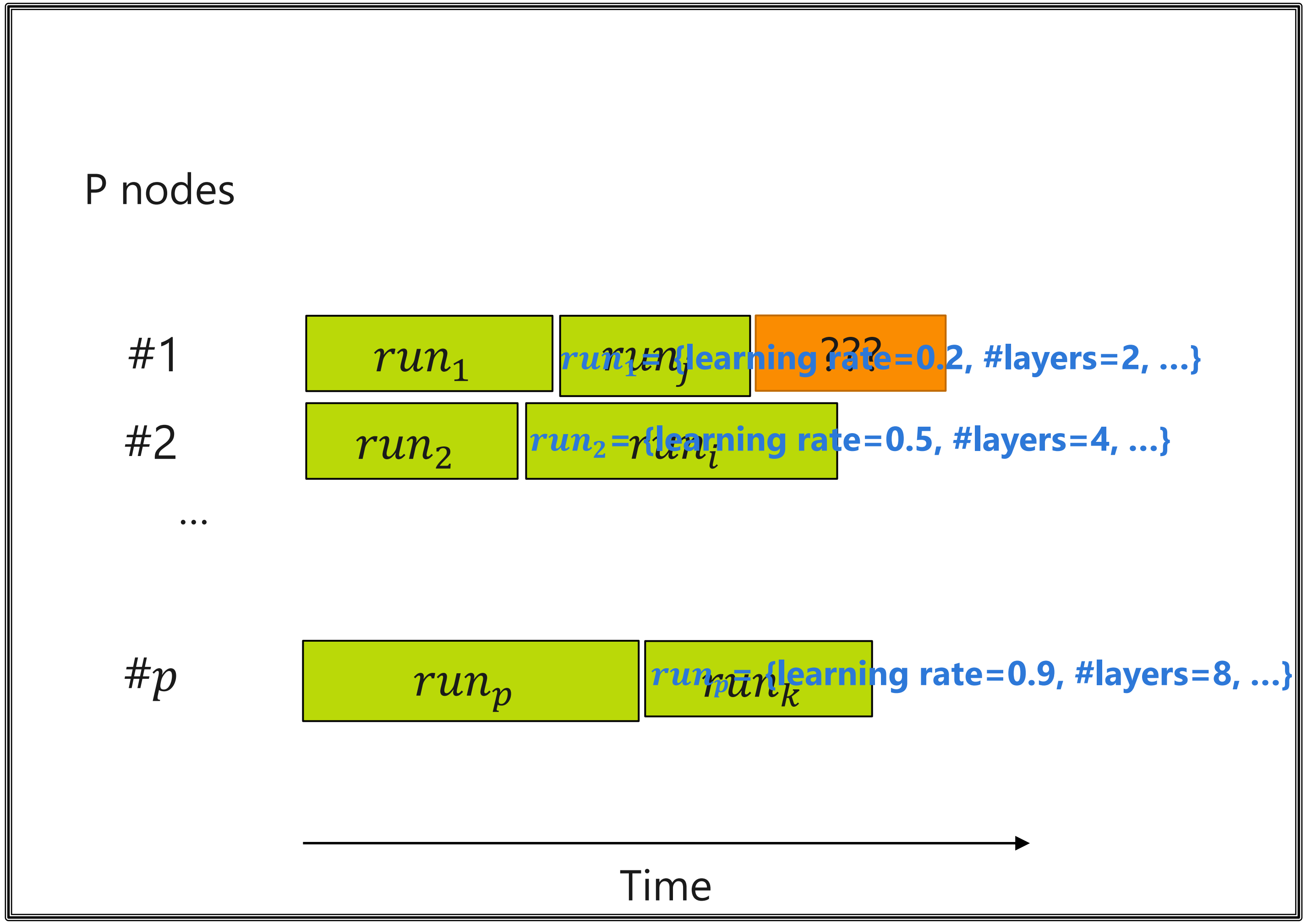
How it works

Launch multiple parallel training runs

(A) Generate new runs
- Which parameter configuration to explore?

(B) Manage resource usage of active runs
- How long to execute a run?

Hyperparameter Tuning runs in Azure ML



Automated Hyperparameter Tuning

Sampling to generate new runs

Define hyperparameter search space

```
{
    “learning_rate”: uniform(0, 1),
    “num_layers”: choice(2, 4, 8)
    …
}
```

Sampling algorithm

```
Config1= {“learning_rate”: 0.2,
“num_layers”: 2, …}

Config2= {“learning_rate”: 0.5,
“num_layers”: 4, …}

Config3= {“learning_rate”: 0.9,
“num_layers”: 8, …}

…
```

Supported sampling algorithms:
Grid Sampling
Random Sampling
Bayesian Optimization

Automated Hyperparameter Tuning

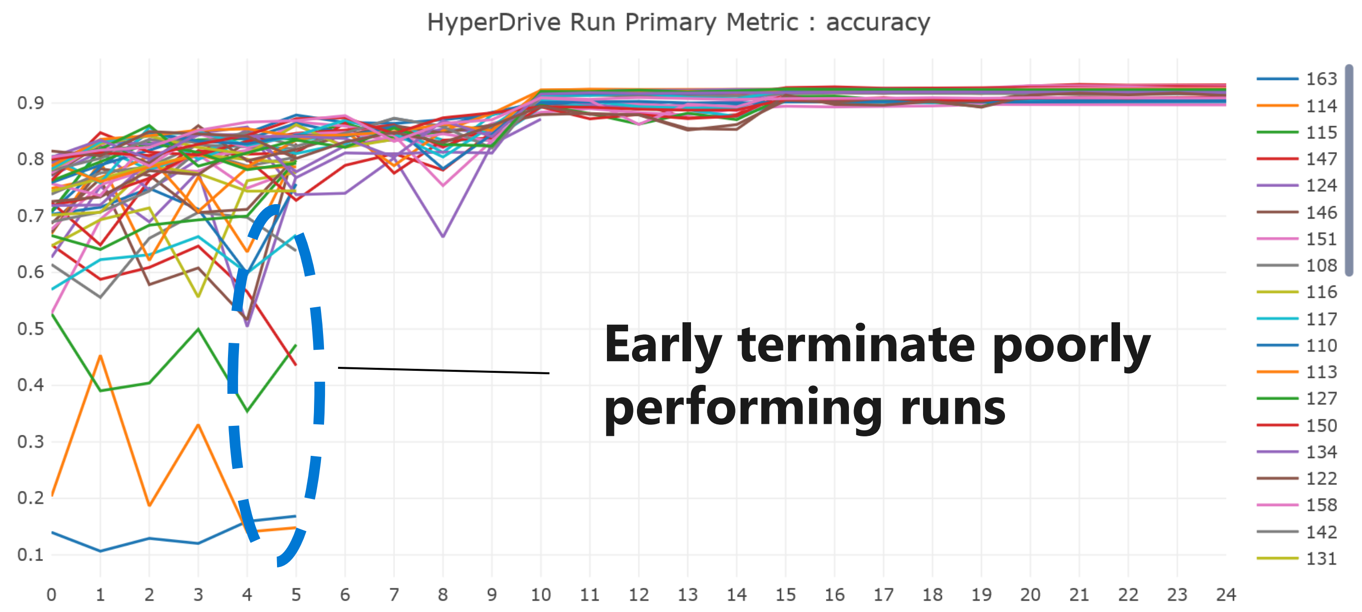
Manage Active Jobs

Evaluate training runs for specified primary metric

Use resources to explore new configurations

Early terminate poor performing training runs. Early termination policies include:

- Bandit policy
- Median Stopping policy
- Truncation Selection policy



Distributed Training with Azure ML Compute

Distributed Training with Azure ML Compute

You submit a model training 'job' – the infrastructure is managed for you.

- Jobs run on a native VM or Docker container.
- Supports Low priority (Cheaper) or Dedicated (Reliable) VMS.

Auto-scales: Just specify min and max number of nodes.

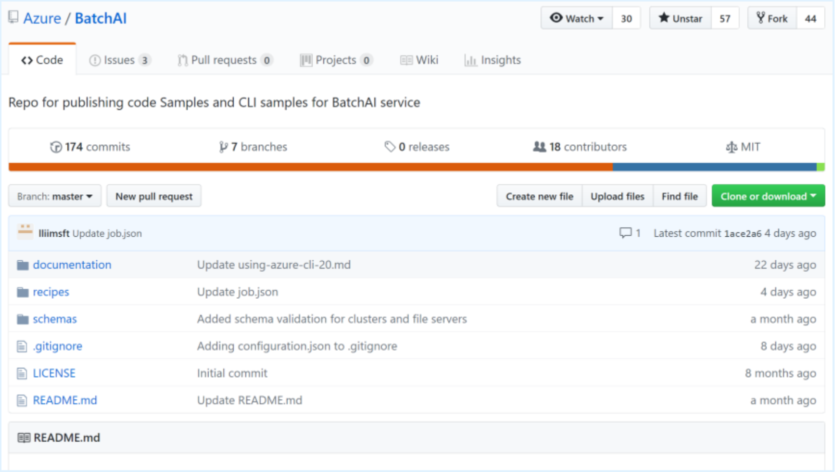
- If min is set to zero, *cluster is deleted when no jobs are running; so pay only for job duration.*

Works with most popular frameworks and multiple languages.

Supports [distributed training with Horovod](#).

Cluster can be shared; multiple experiments can be run in parallel.

Supports most VM Families, including latest NVidia GPUs for DL model training.



Deploy to IoT Edge

IoT Edge Modules

An Azure IoT Edge device is a Linux or Windows-based device that runs the Azure IoT Edge runtime.
Machine learning models can be deployed to these devices as **IoT Edge Modules**.

Benefits: Deploying a model to an IoT Edge device allows the device to use the model directly, instead of having to send data to the cloud for processing. You get faster response times and less data transfer

IoT Edge Modules

Azure IoT Edge modules are the smallest unit of computation managed by IoT Edge, and can contain Azure services or your own solution-specific code.

IoT Edge module images contain applications that take advantage of the management, security, and communication features of the IoT Edge runtime.

In implementation, modules images exist as container images in a repository, and module instances are containers on devices.

Deploying to IoT Edge Devices

Prerequisites

IoT Hub
IoT Edge Device with the IoT Egde runtime installed.
Docker Image based on the ML model and image configuration stored in the container registry.
This can be done as follows

```
from azureml.core.image import Image, ContainerImage

#Image configuration
image_config = ContainerImage.image_configuration (
                                                  runtime = "python", execution_script
="score.py",
                                                  conda_file = "myenv.yml",
                                                  tags = {"attributes", "classification"},
                                                  description = "Image with my model")

image = ContainerImage.create (name = "myimage",
                                          models = [model], #this is the model object
                                          image_config = image_config, workspace = ws )
```

Deploying to the IoT Edge

Steps

1. **Get the container registry credentials**: Azure IoT needs the credentials for the container registry that Azure Machine Learning service stores docker images in. You can get via the Azure Portal
2. Configure deployment manifest, a JSON document that describes which modules to deploy, how data flows between the modules, and desired properties of the module twins. You can use wizard in the Azure Portal to create this. The Wizard has 4 steps:
    1. Add Modules
    2. Specify Routes
    3. Review Deployment
    4. Submit
3. View Modules on device: Once you've deployed modules to your device, you can view all of them in the Device details page of the portal. This page displays the name of each deployed module, as well as useful information like the deployment status and exit code.

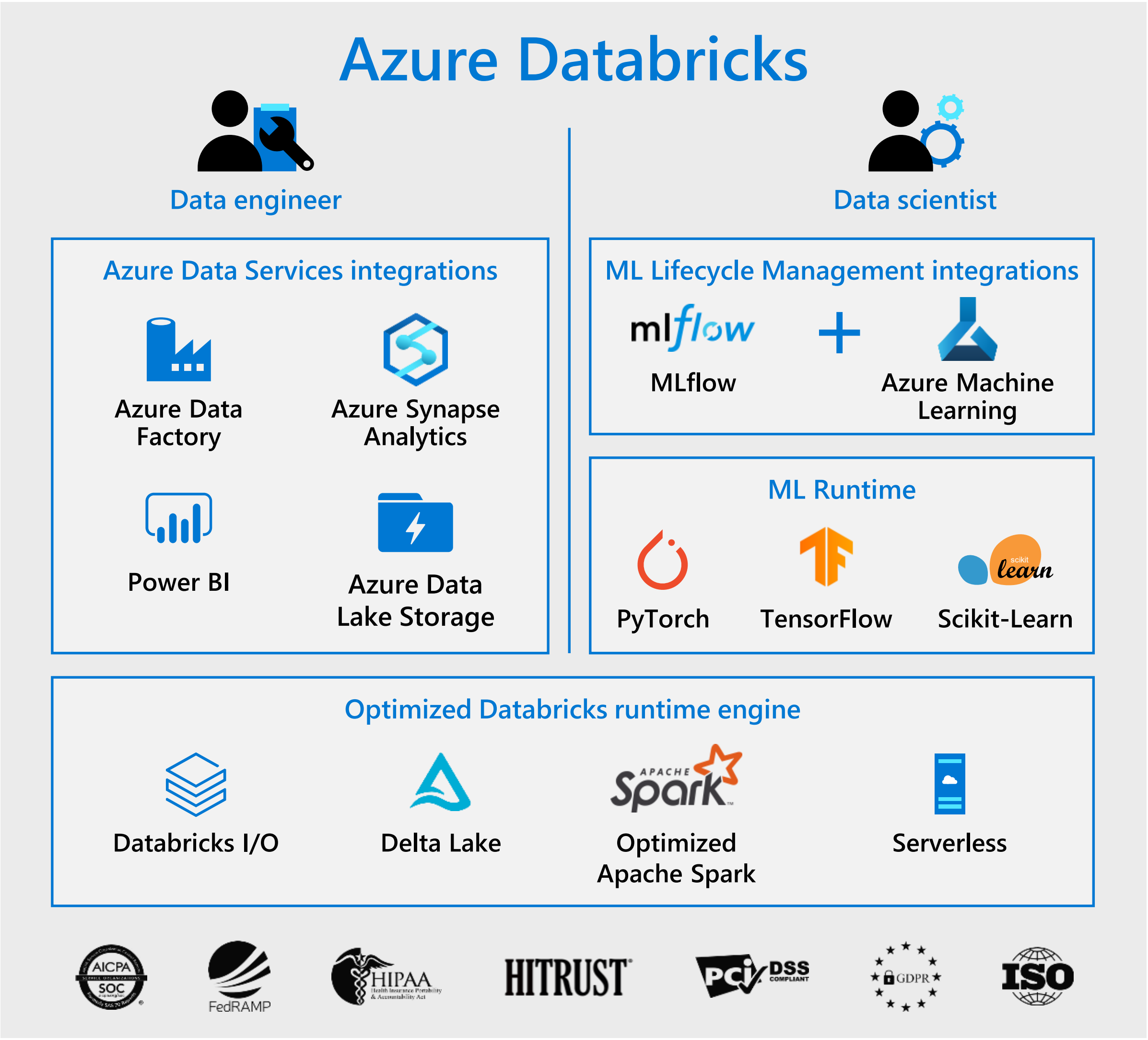
Edge Deployment

Light and Heavy

		Heavy Edge				Light Edge		
Description	An Azure host that spans from CPU to GPU and FPGA VMs	A server with slots to insert CPUs, GPUs, and FPGAs or a X64 or ARM system that needs to be plugged in to work **Cloud Consistent Hybrid Server**	**Servers**	**PC class devices**	**Gateway**	A Sensor with a SOC (ARM CPU, NNA, MCU) and memory that can operate on batteries **Smart Sensors + Ambient AI**		**Sensors**
Example	DSVM / ACI / AKS / Batch AI	- DataBox Edge - HPE - Azure Stack	- DataBox Edge	- Industrial PC	-Video Gateway -DVR	-Mobile Phones -VAIDK	-Mobile Phones -IP Cameras	-Azure Sphere - Appliances
What runs the model	CPU,GPU or FPGA	CPU,GPU or FPGA	CPU, GPU	x64 CPU	Multi-ARM CPU	Hw accelerated NNA	CPU/GPU	MCU

Machine learning with Azure Databricks and Azure Machine Learning

Databricks accelerates data-driven innovation



Collaborative workspace for data teams across the full lifecycle

Native integration with AML and Azure Data Services

Scalable, reliable, and fast data - built on your existing data lake powered by most optimized Spark Engine

One fully-integrated security model for production infrastructure

MLflow Components

mlflow
Tracking

Record and query experiments: code, data, config, results

mlflow
Projects

Packaging format for reproducible runs on any platform

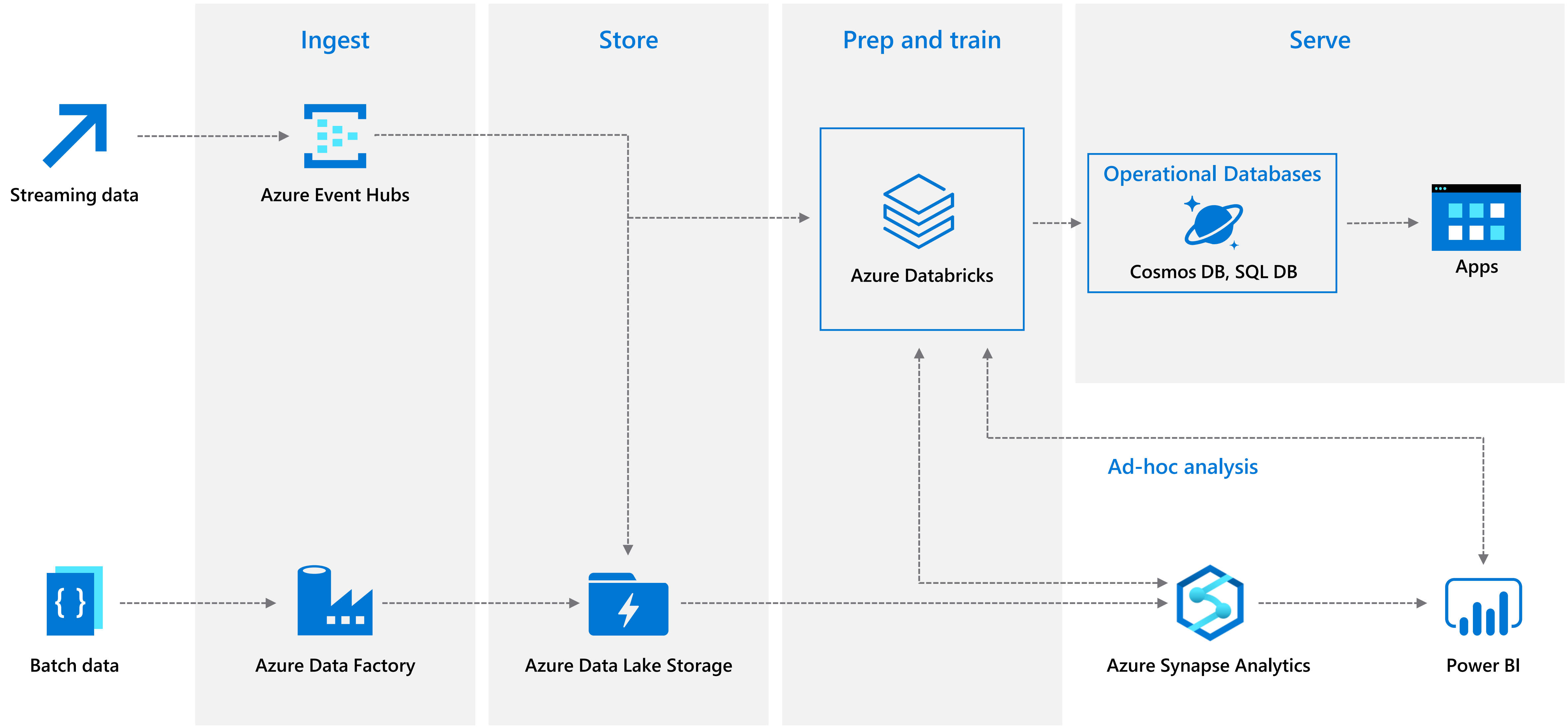
mlflow
Models

General model format that supports diverse deployment tools

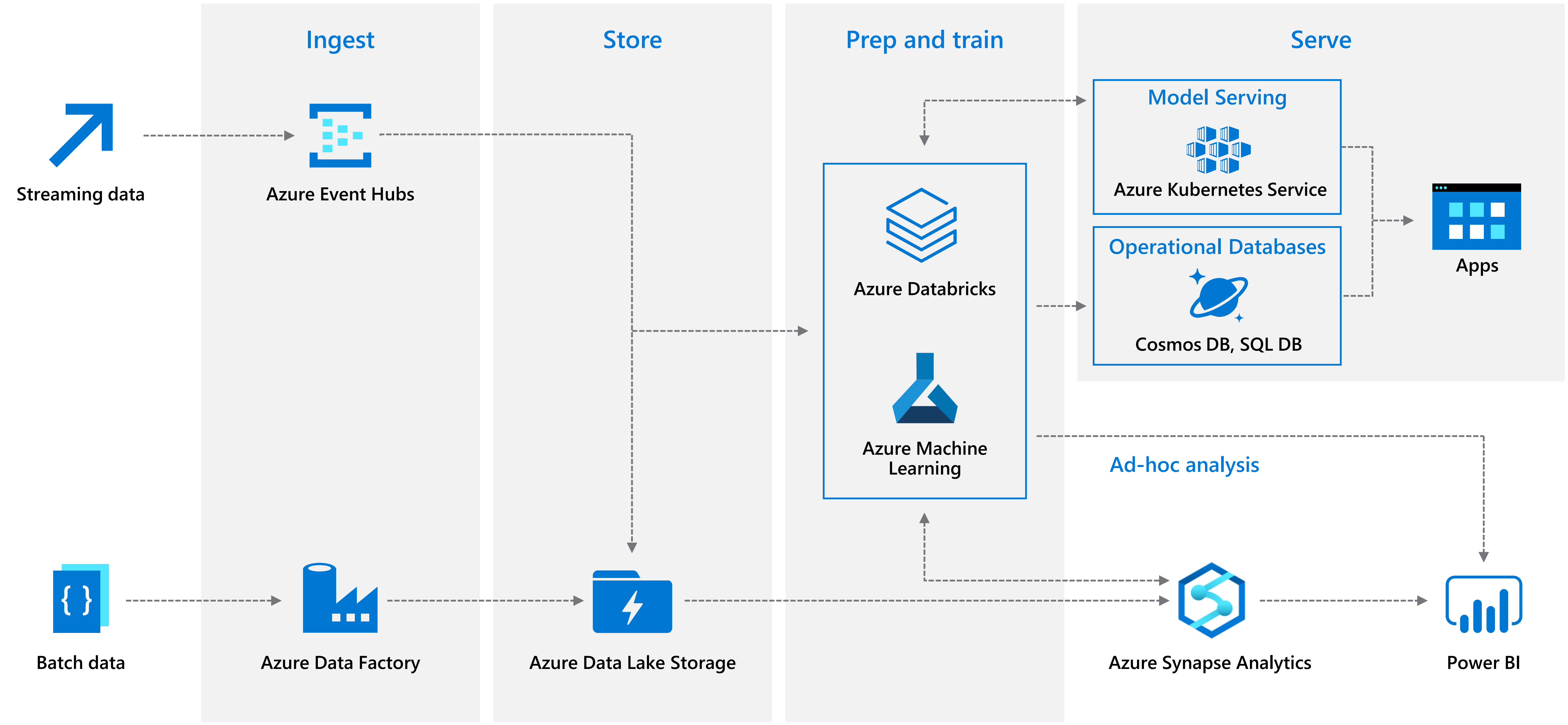
mlflow
Model Registry

Centralized and collaborative model lifecycle management

Machine learning



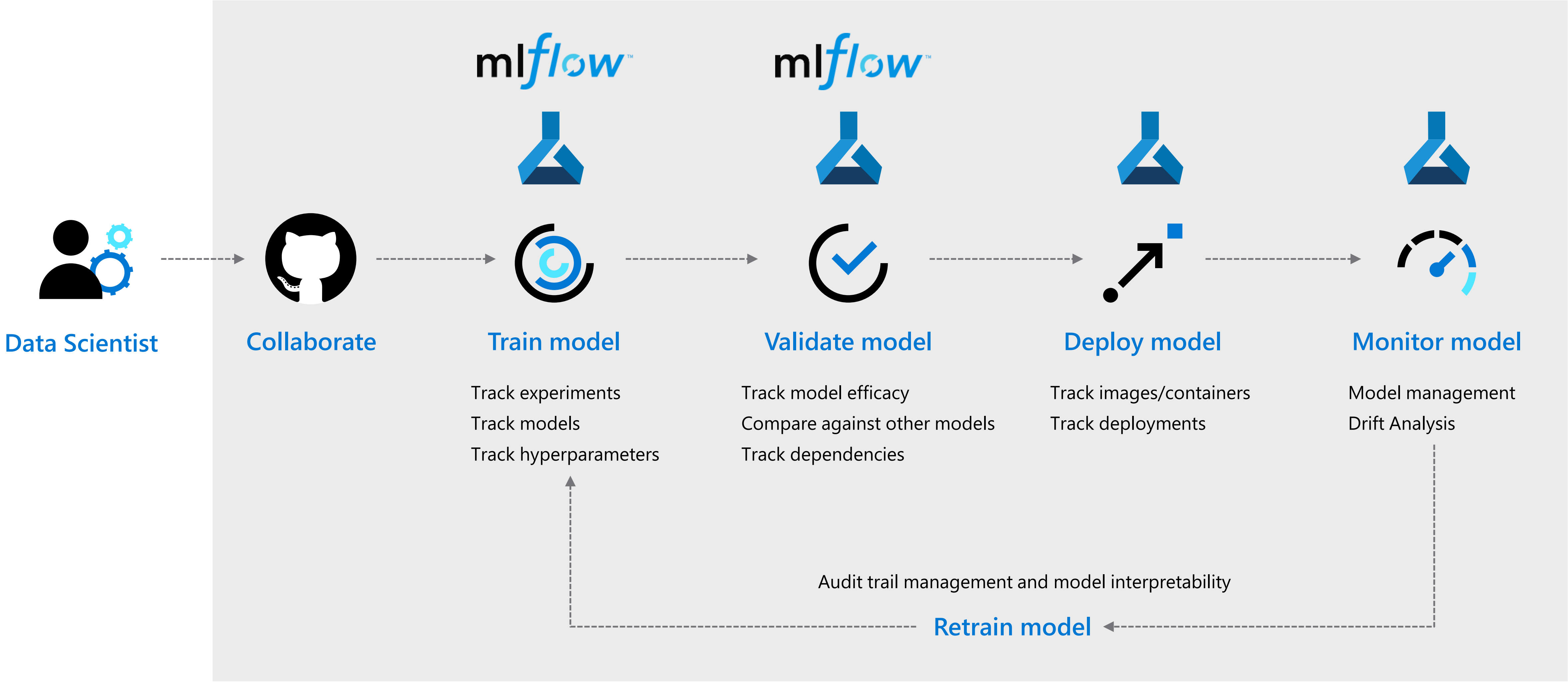
Machine learning



Azure Databricks and Azure ML are better together

- Log experiments and models in a central place
- Maintain audit trails centrally
- Deploy models seamlessly via Azure ML
- Implement robust MLOps

MLFlow and Azure Machine Learning



Azure Databricks with Azure Machine Learning

Azure Databricks

Open & extensible

- Leverage the latest libraries and frameworks
- Perform distributed training across CPUs and GPUs
- Dedicated ML runtime with pre-built optimizations

Engineered integration

MLflow integration

- Common experiment tracking and results backend
- Store models in a central model registry across Azure
- Combined view of all ML activity within Azure

Azure Machine Learning

ML & ML management

- Package and deploy models for inferencing at scale
- Leverage automated ML to design a model factory
- Create CI/CD pipelines for retraining with drift tracking and audit trails

Azure DevOps

Implement MLOps with Azure DevOps

Powerful infrastructure with latest CPUs and GPUs

Demo Time

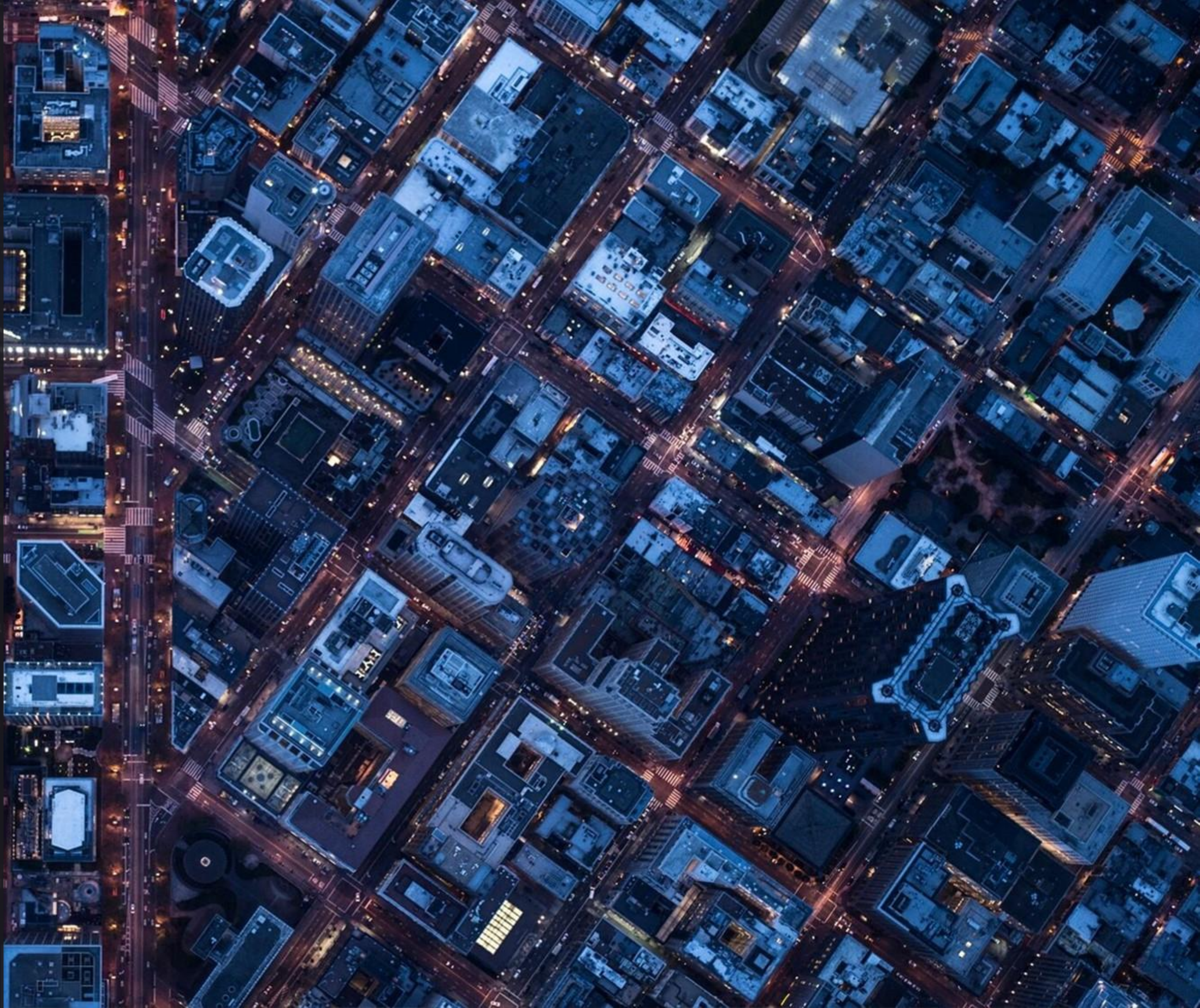
MLFlow

AutoML with Databricks and Azure Machine Learning

Machine Learning Within Synapse Analytics

Microsoft Azure

Q&A



Feedback Please!