

# Real World Machine Learning at Enterprise Scale

Azure Data Conference

POWERED BY

Azure Data Community & Azure Data Conf

---



**Carey Payette**  
Azure MVP & MCT  
Sr. Software Engineer, Solliance



**Lino Tadros**  
Microsoft Regional Director & MCT  
Technical Fellow, Solliance.

---

---

# Get Whova for

DEVintersection, Microsoft Azure + AI  
Conference, Azure Data and M365  
Conference - Fall 2022



DEVintersection, Microsoft Azure + AI Conference,  
Azure Data and M365 Conference - Fall 2022 Official  
Event App

- Explore the **professional profiles** of event speakers and attendees
- Send **in-app messages** and **exchange contact info**
- **Network and find attendees** with common affiliations, educations, shared networks, and social profiles
- Receive **update notifications** from organizers
- Access the **event agenda**, GPS guidance, maps, and parking directions at your fingertips



The event invitation code is:  
**FallConf22**

# Agenda

9:00 to 9:10: Welcome & Introduction

9:10 to 10:00: AML Architecture and Data Science Process

10:00 to 10:20: Live Demo

10:20 to 10:45: Responsible AI

10:45 to 11:00: Break

11:00 to 12:00: Lab 1

12:00 to 1:00: Lunch

1:00 to 1:15: AML Resources and Security

1:15 to 1:45: AML with Data Drift and Synapse

1:45 to 2:15: Lab 2

2:15 to 2:30: Break

2:30 to 3:00: MLOps

3:00 to 4:00: Lab 3

# Machine Learning on Azure

## Domain specific pretrained models

To simplify solution development



Vision



Speech



Language



Web search



Decision

## Familiar data science tools

To simplify model development



Visual Studio Code



Azure Notebooks



Jupyter



Command line

## Popular frameworks

To build advanced deep learning solutions



PyTorch



TensorFlow



Scikit-Learn



ONNX

## Productive services

To empower data science and development teams



Azure Machine Learning



Azure Databricks



Machine Learning VMs

## Powerful infrastructure

To accelerate deep learning



CPU



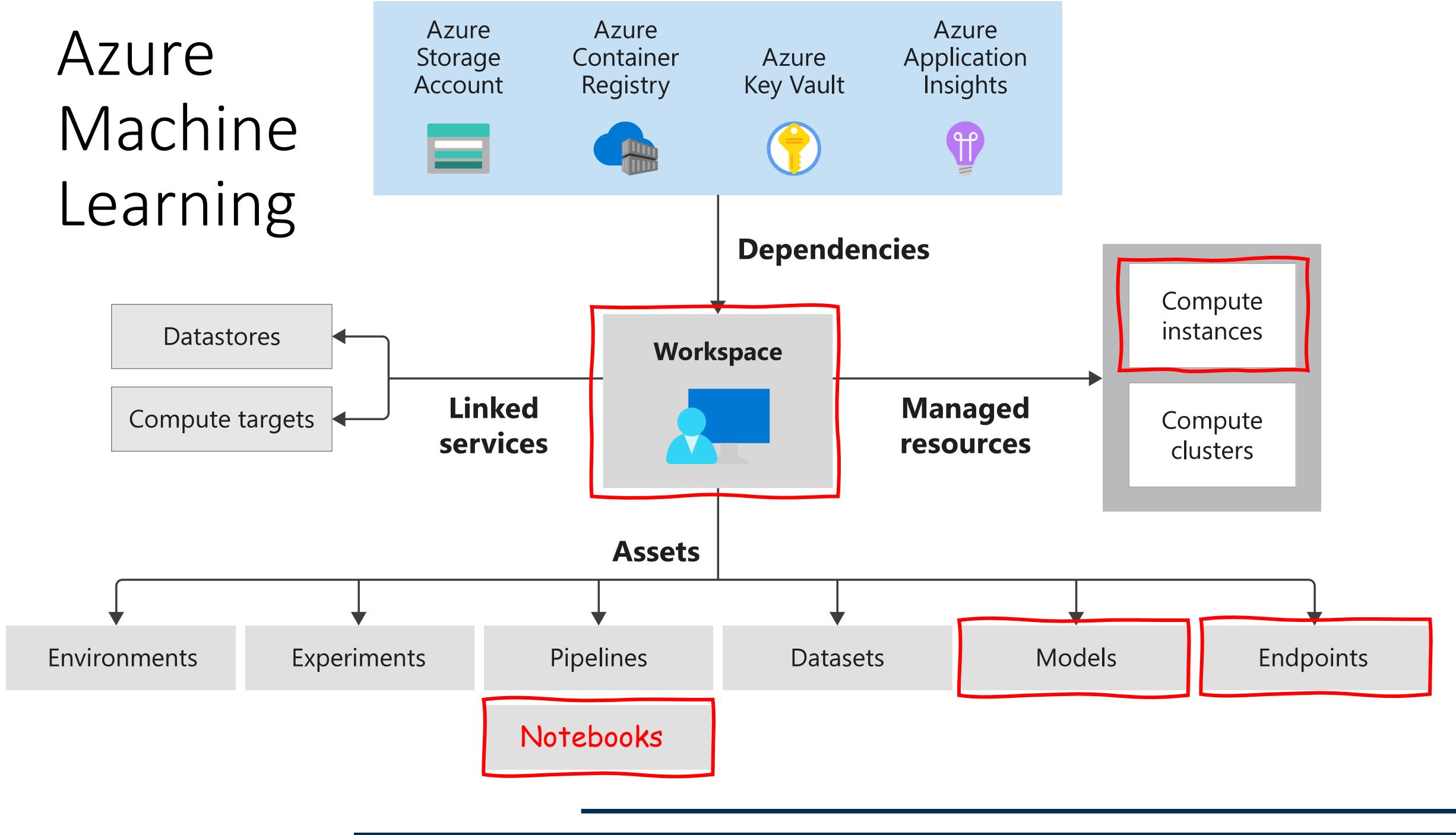
GPU



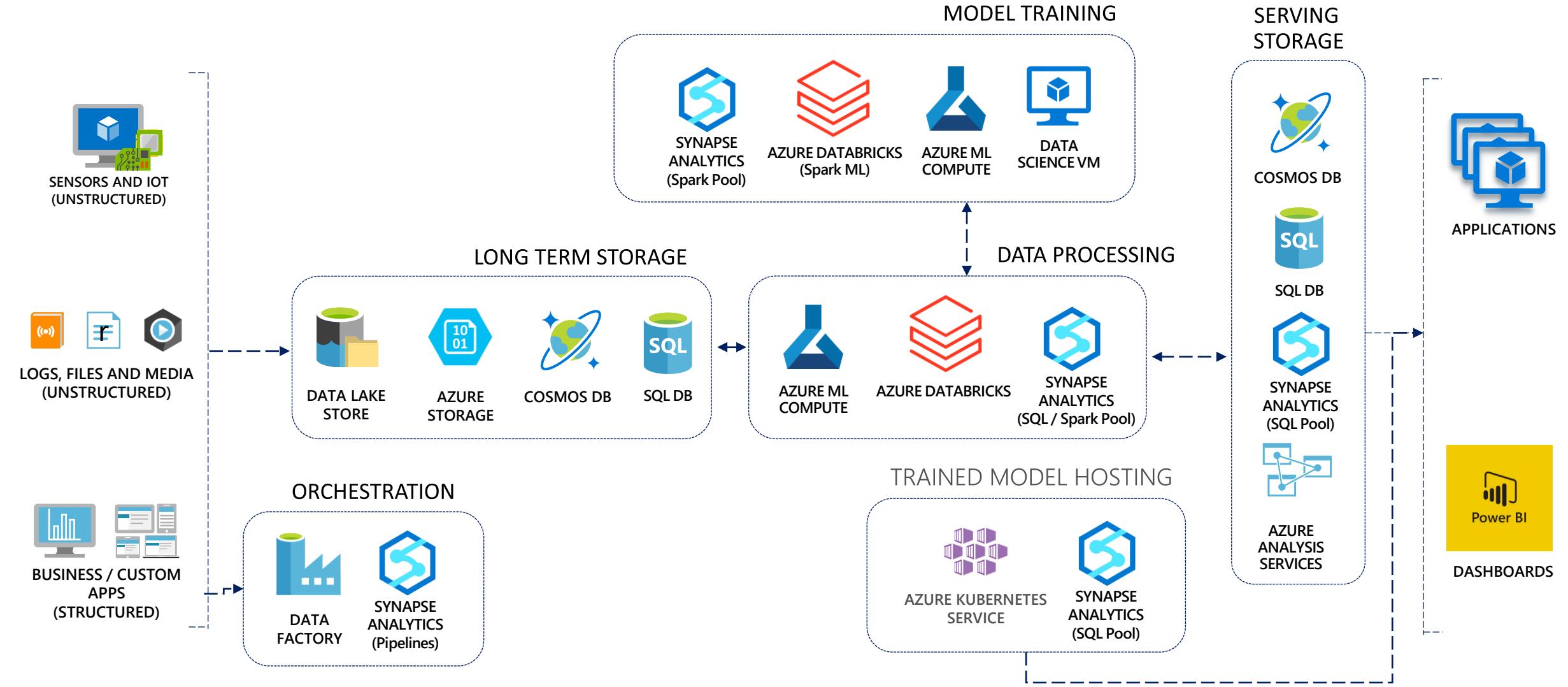
FPGA

From the Intelligent Cloud to the Intelligent Edge

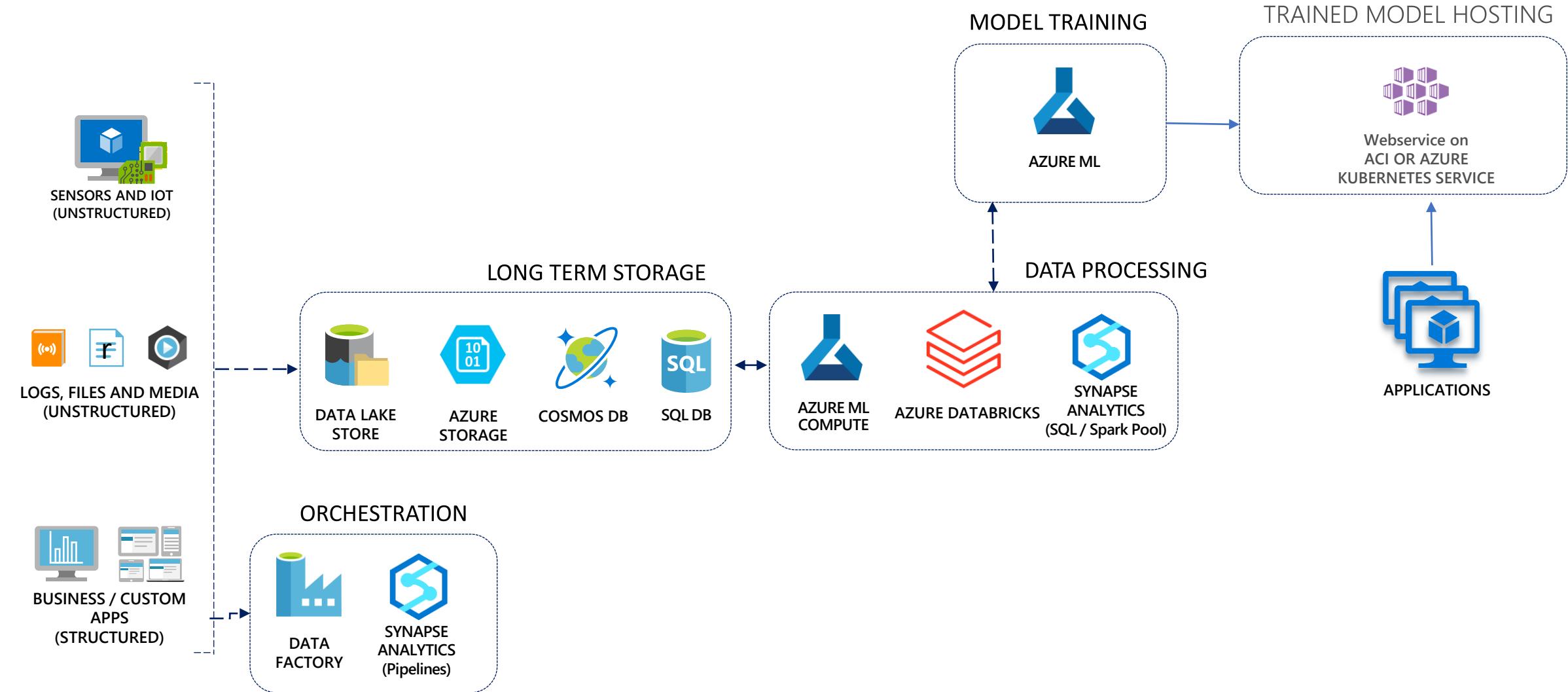
# Azure Machine Learning



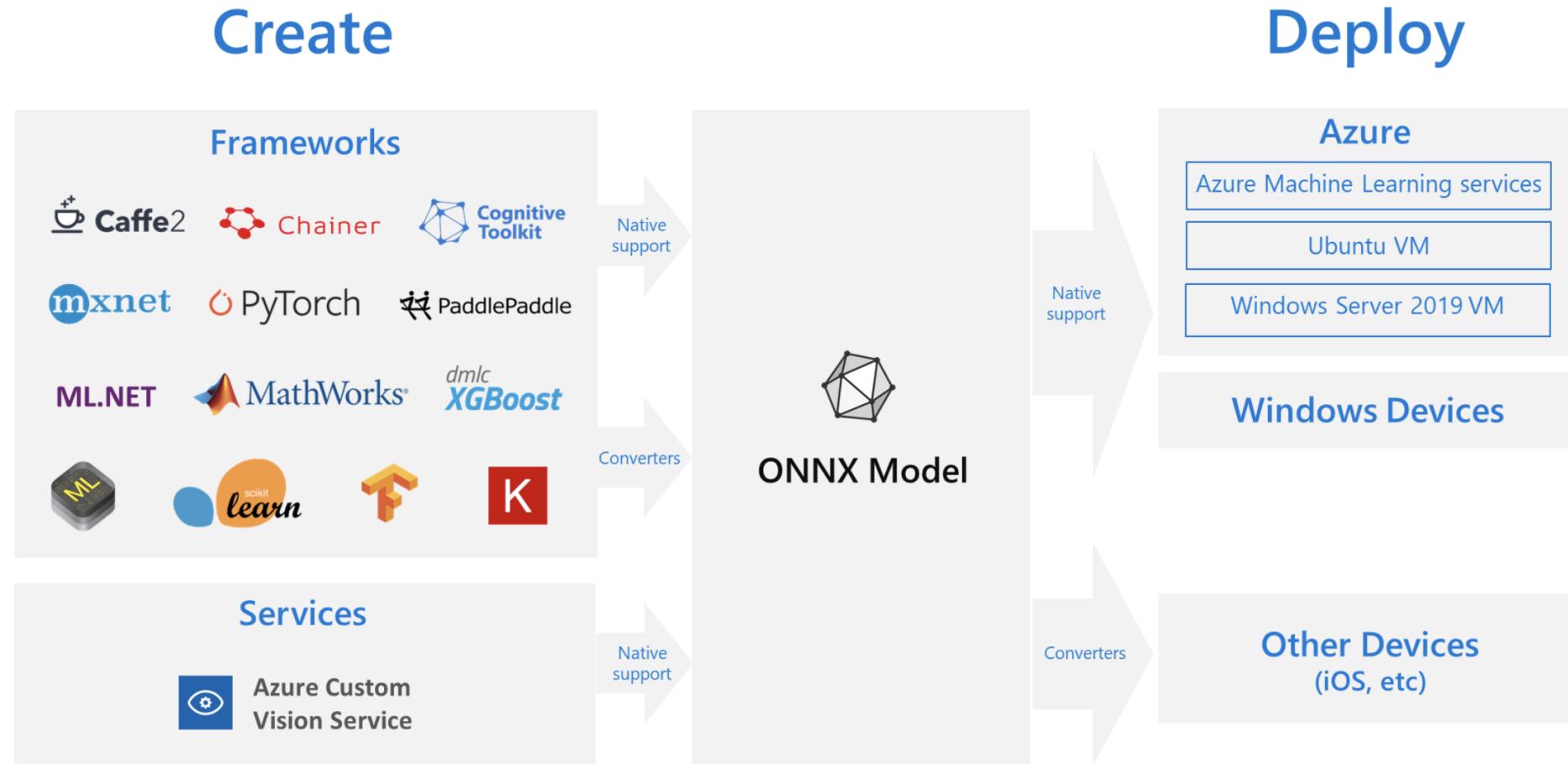
# Advanced analytics pattern in Azure



# Model deployment patterns in Azure



# What is ONNX?



# Tools for Machine Learning

- Azure Machine Learning Studio
  - Visual Studio Code
  - Python SDK 2
  - Azure CLI 2
-

# Azure Machine Learning CLI v2

- The CLI v2 is useful in the following scenarios:
  - On board to Azure ML without the need to learn a **specific programming language**
  - Ease of deployment and automation
  - Azure ML offers endpoints to streamline model deployments for both real-time and batch inference deployments. This functionality is available only via CLI v2 and SDK v2.
  - Azure ML introduces components for managing and reusing common logic across pipelines. This functionality is available only via CLI v2 and SDK v2.

# Azure Machine Learning Python SDK v2

- The SDK v2 is useful in the following scenarios:
  - SDK v2 allows you to build a single command or a chain of commands like Python functions - the command has a name, parameters, expects input, and returns output.
  - SDK v2 allows you to:
    - Construct a single command.
    - Add a hyperparameter sweep on top of that command,
    - Add the command with various others into a pipeline one after the other.
  - Azure ML introduces components for managing and reusing common logic across pipelines. This functionality is available only via CLI v2 and SDK v2.
  - Azure ML offers endpoints to streamline model deployments for both real-time and batch inference deployments. This functionality is available only via CLI v2 and SDK v2.

# The Data Science process

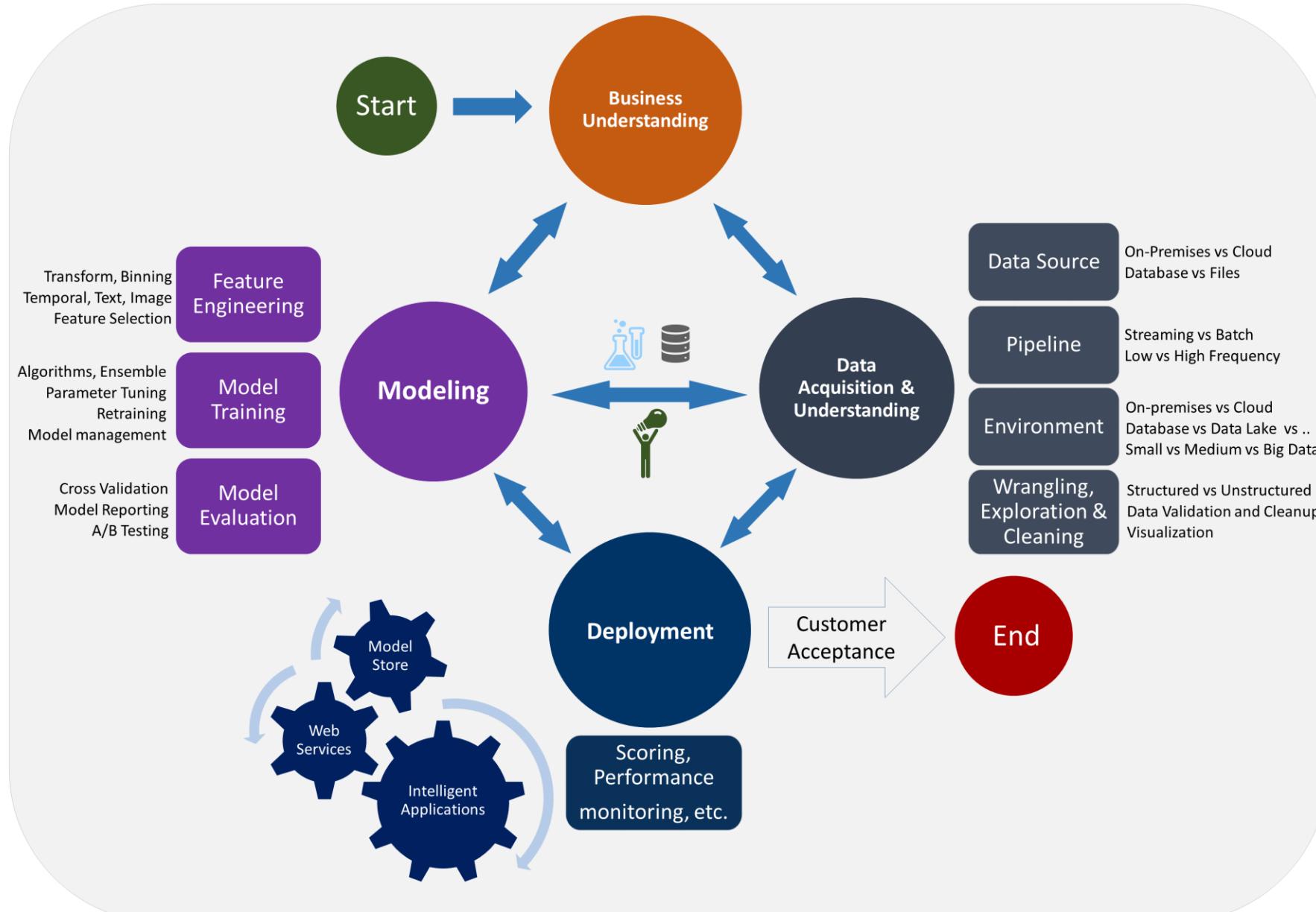
Azure Data Conference

POWERED BY

Azure Data Community & Azure Data Conf

---

# Data Science Lifecycle



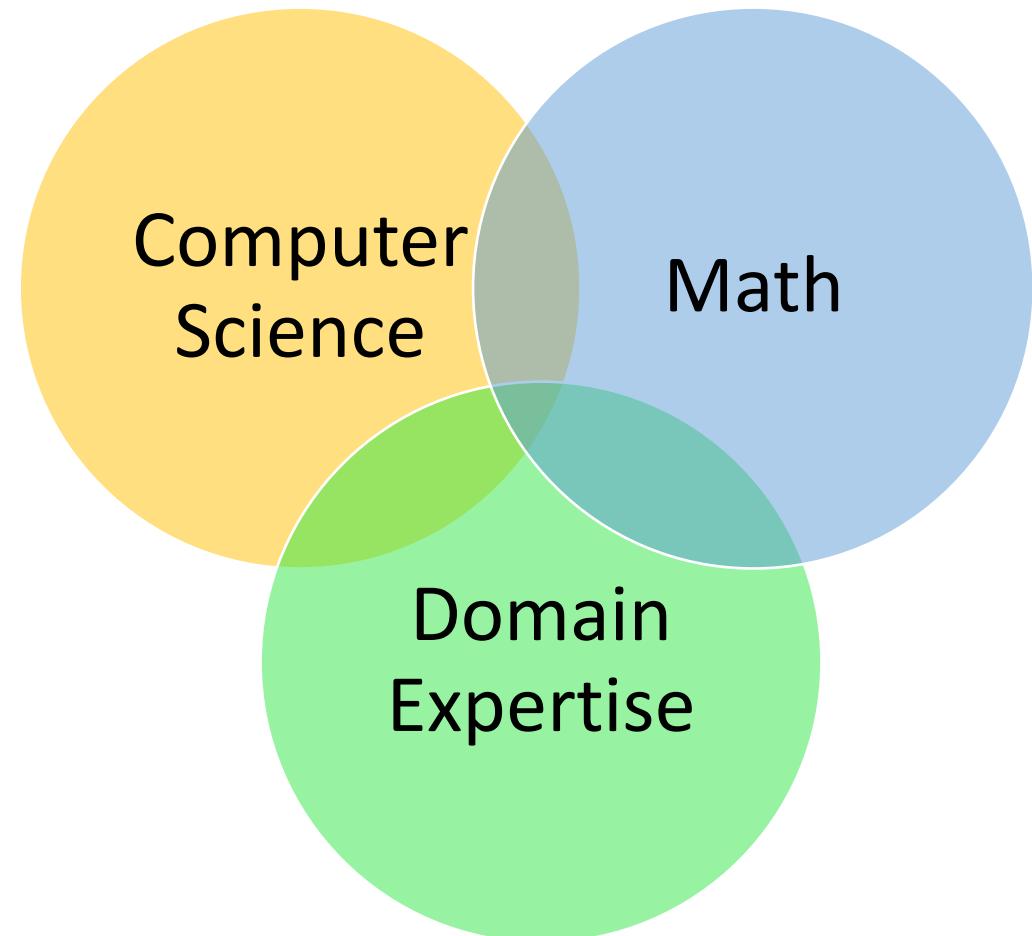
# As a developer is deep learning for you?

*In the future, deep learning will not only be used by the specialists-researchers, graduate students and engineers with an academic profile-but will also be **a tool in the toolbox of every developer**, much like web technology today.*

- Francois Chollet, Creator of Keras

# What do you really need to know to do machine learning?

- There's an argument being made that only data scientists can do ML because they come from the math side.
- How many developers do you know that are actually bad at math?



# Machine/Deep learning and math

*Machine learning and deep learning exhibits comparatively little mathematical theory [compared to statistical analysis] and is engineering oriented.*

*It's a hands-on discipline in which ideas are proven empirically more often than theoretically.*

- Francois Chollet, Creator of Keras

---

# How much math did we write in this code?

```
from keras import models  
from keras import layers  
  
network = models.Sequential()  
network.add(layers.Dense(512, activation='relu',  
input_shape=(784,)))  
network.add(layers.Dense(10, activation='softmax'))  
network.compile(optimizer='rmsprop',  
    loss='categorical_crossentropy',  
    metrics=['accuracy'])  
network.fit(train_images, train_labels, epochs=5,  
batch_size=128)
```



As a developer, what technologies do you need to learn?

Python

Azure Compute

Azure Storage

Azure Machine Learning

# The Pre-Built and Custom AI Spectrum

Pre-Built AI

Custom AI

**Azure Cognitive Services**

**Azure Machine Learning**

Models are pre-trained using  
Microsoft supplied data

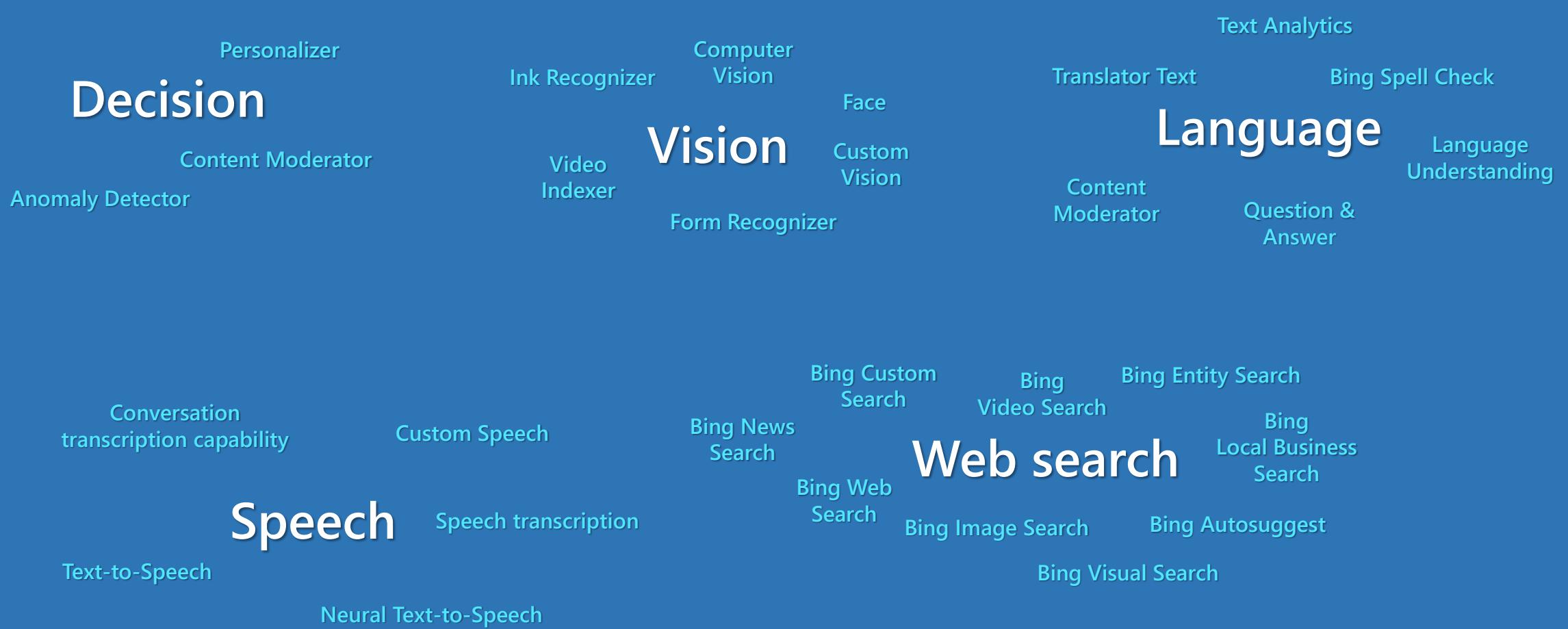
Models “customized” with your data

Pre-built models

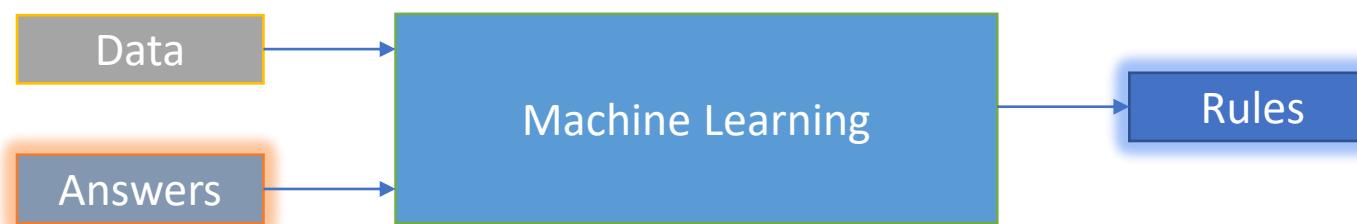
Custom models

Models tailored to your  
scenario and your data

# Azure Cognitive Services

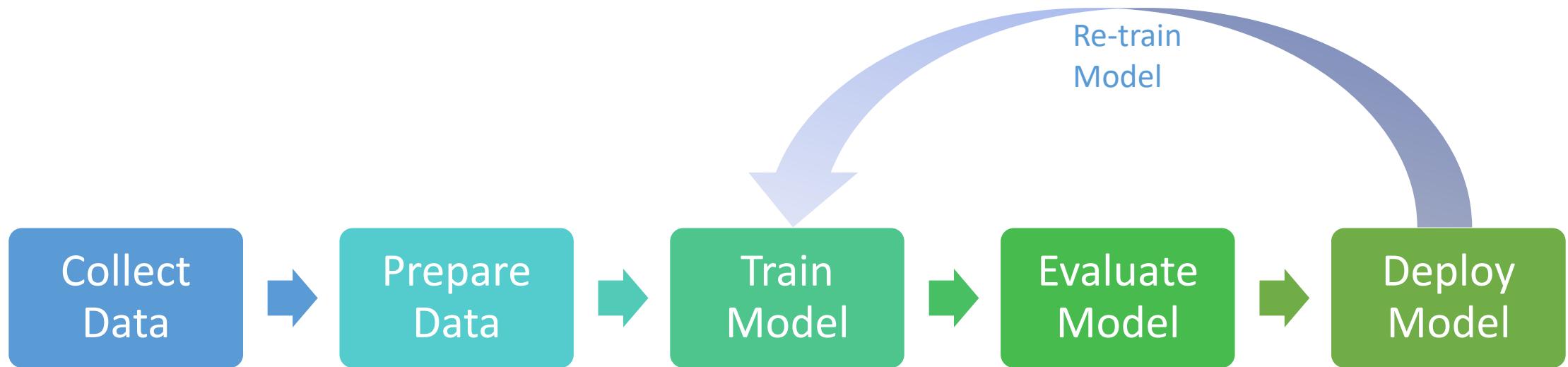


# Machine Learning as the new programming paradigm

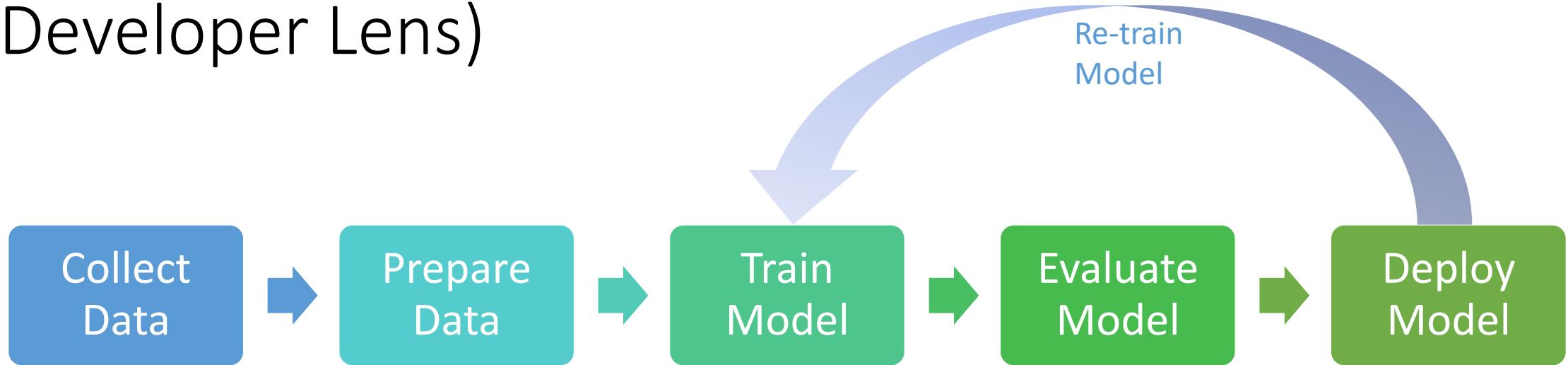


Let the machine figure out the rules based on history!

# Behind Machine Learning is the Data Science Process



# The Data Science Process (Developer Lens)



Write Code

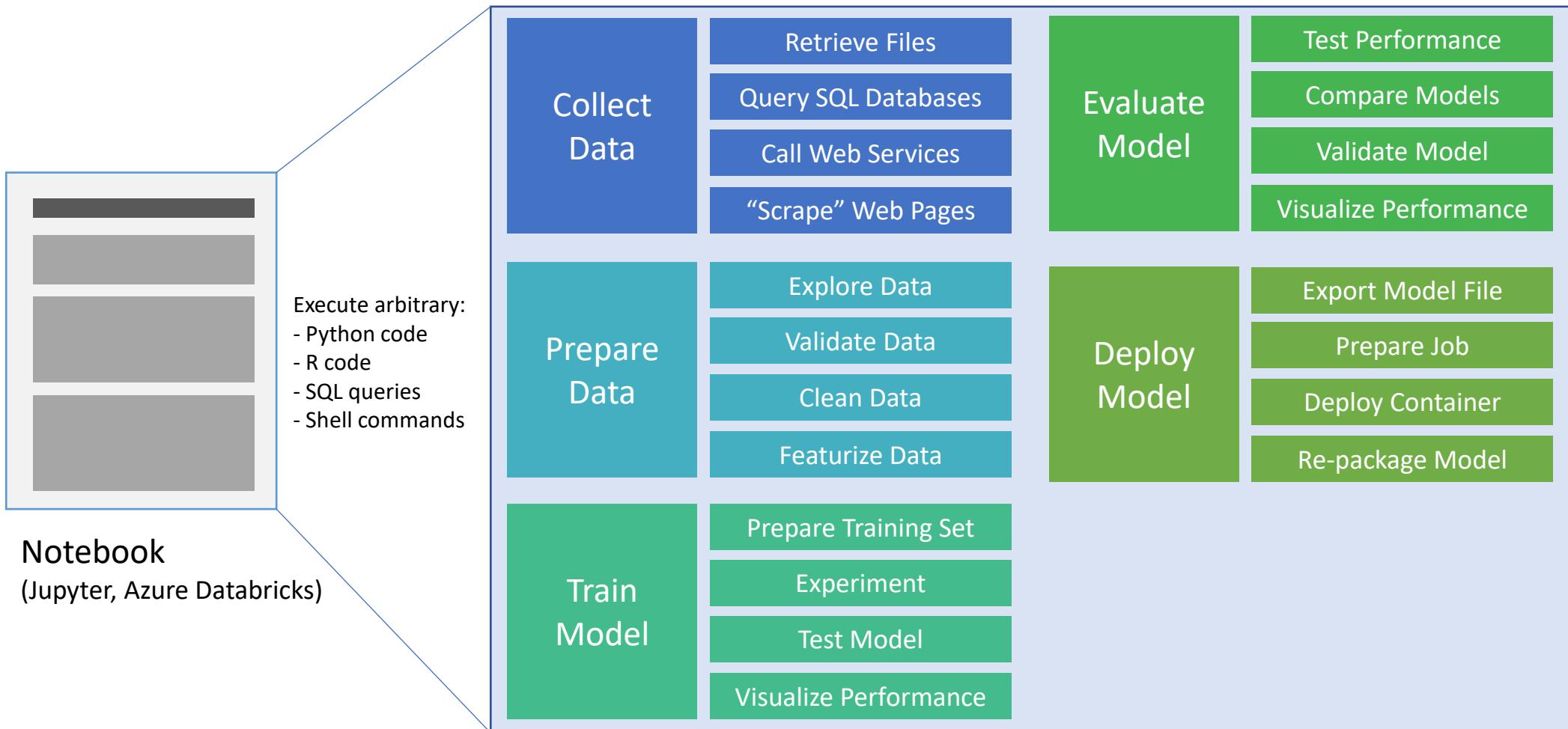
Write Queries  
& Code

Write Code  
Do some math

Write Code  
Do some math

Do DevOps

# The Notebook Paradigm

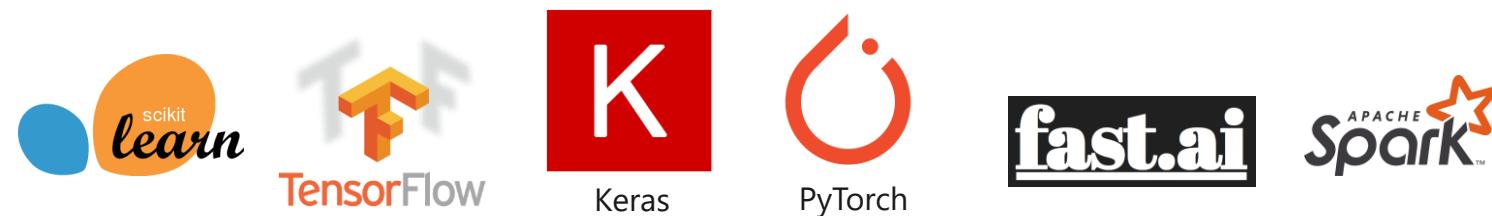


# Examples of Tools for Data Science in the Wild

Core frameworks  
and tools



Machine Learning  
& Deep Learning



Visualization



Natural Language  
Processing

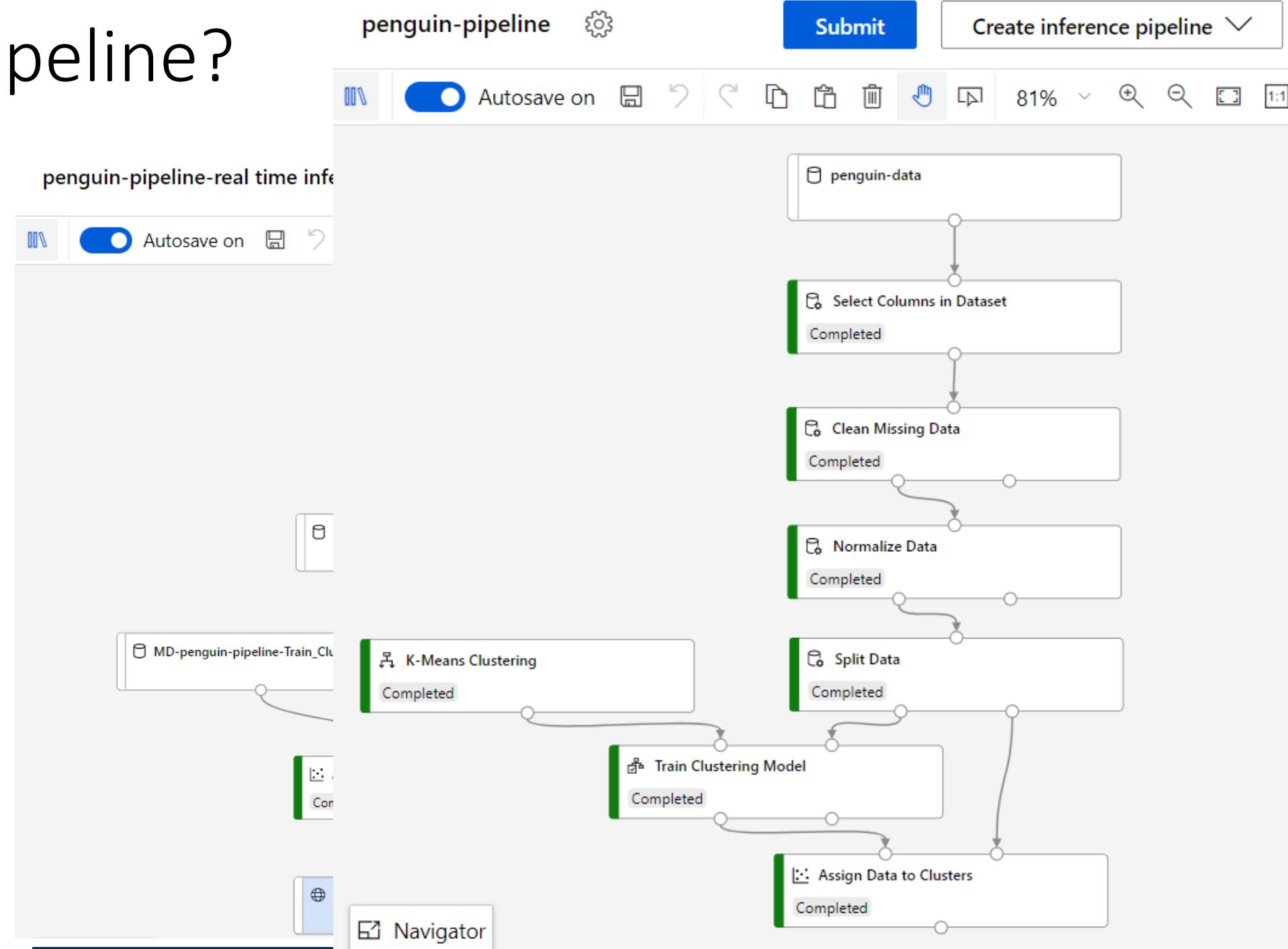


# What is a Pipeline?

# The working metaphor for Azure Machine Learning

# Training pipelines to train models

- Inference pipelines to generate predictions
  - Real-time inference
  - Batch inference



# Creating a New Pipeline

Create a pipeline through the Azure ML studio or via API using a language like Python, R, C#, or F#.

Microsoft Azure Machine Learning

CSAzureML > Designer

Designer

New pipeline

+

Easy-to-use prebuilt modules ⓘ

Image Classification using DenseNet ⓘ

Pipelines

+ New

Home

Author

Notebooks

Automated ML

Designer

Assets

Datasets

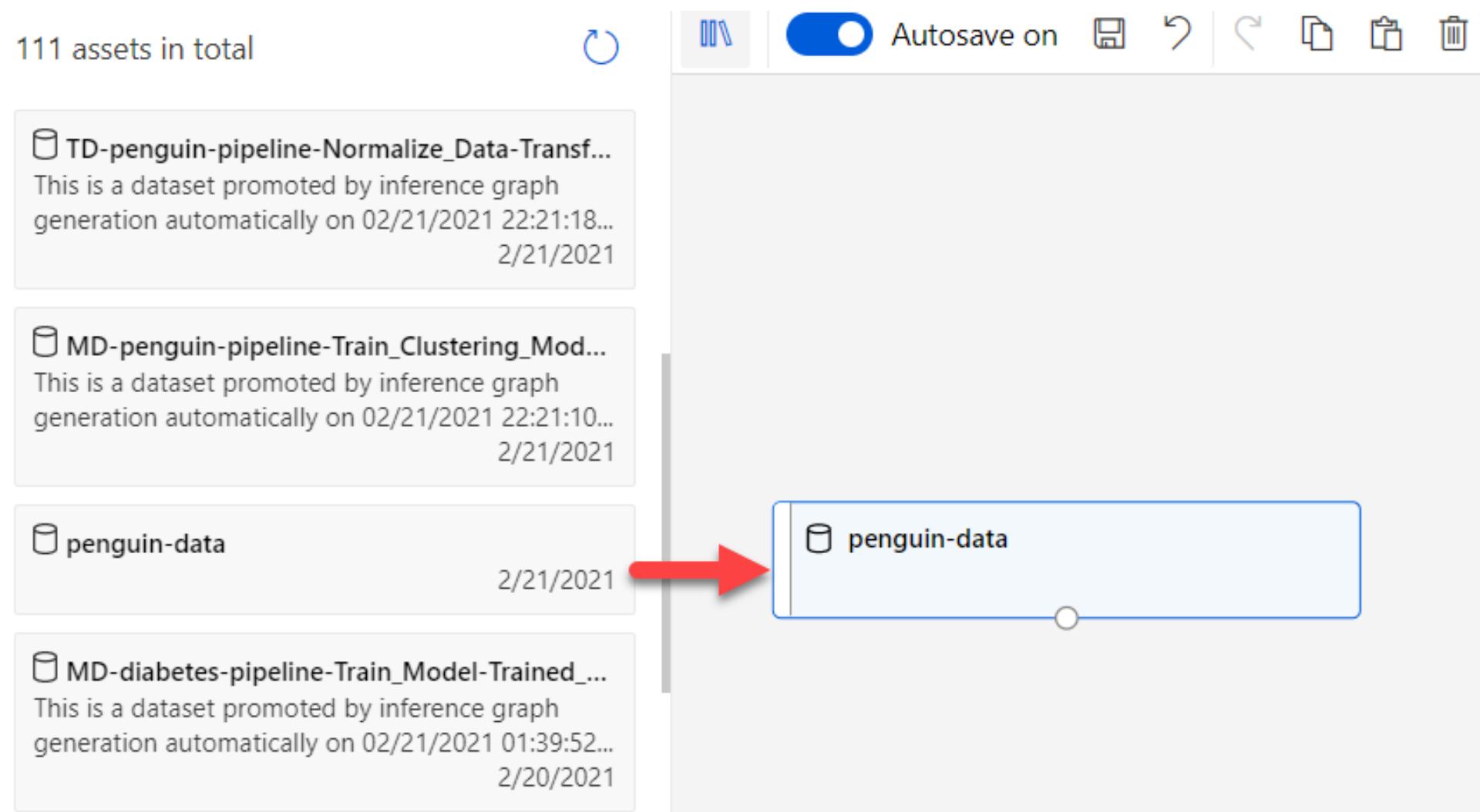
Experiments

Pipelines

Models

# Modifying a Pipeline

The Azure Machine Learning Studio includes a drag-and-drop interface. Bring pipeline components onto the canvas.



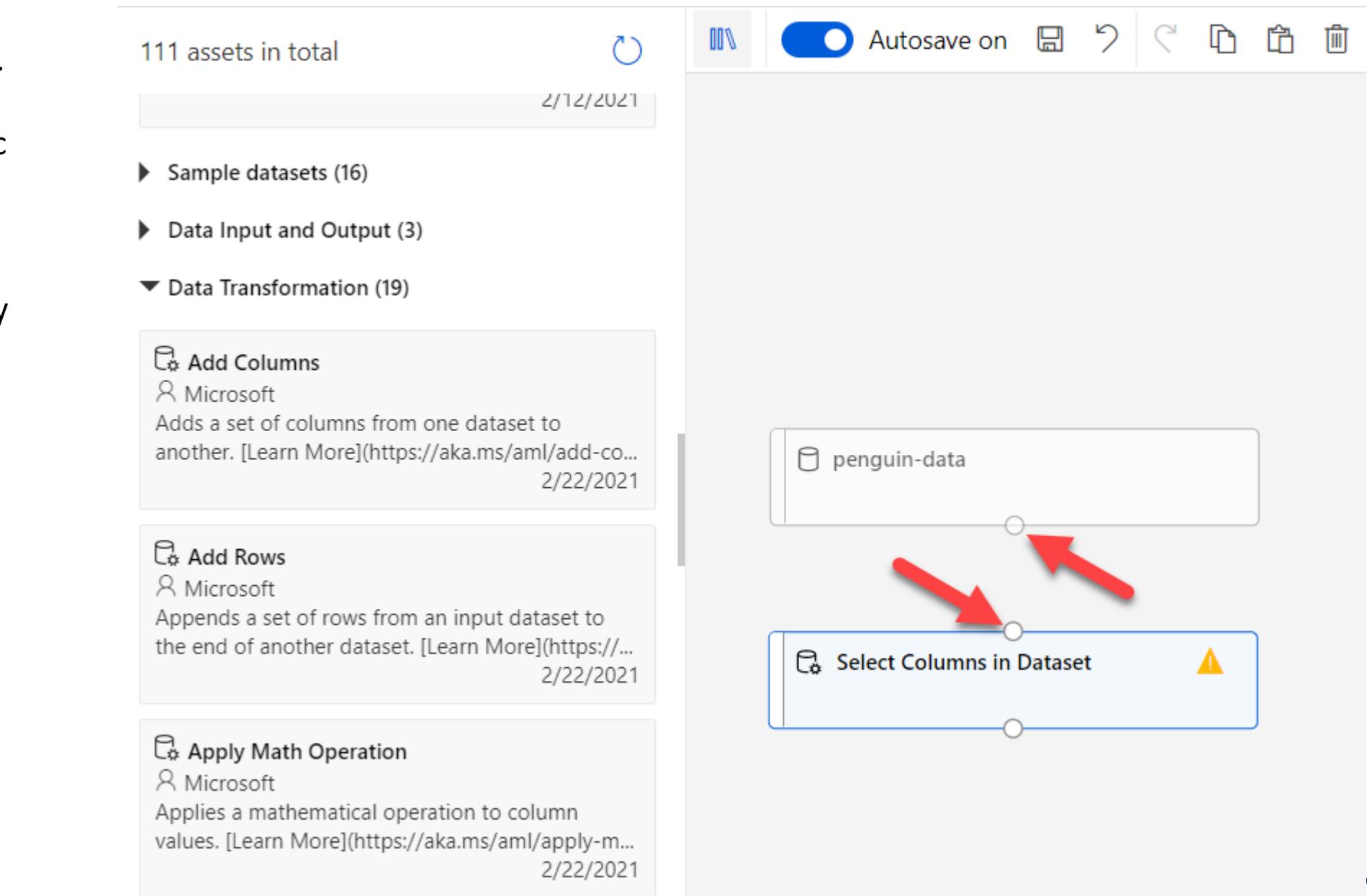
# Modifying a Pipeline

Then, connect them together.

Each connector is of a specific class, such as

**DataFrameDirectory** or  
**TransformationDirectory**.

Pipeline components can only connect if the classes match.





# Built-In Components

Azure Machine Learning features a variety of built-in components.

- ▶ Datasets (19)
- ▶ Sample datasets (16)
- ▶ Data Input and Output (3)
- ▶ Data Transformation (19)
- ▶ Feature Selection (2)
- ▶ Statistical Functions (1)
- ▶ Machine Learning Algorithms (19)
- ▶ Model Training (4)
- ▶ Model Scoring & Evaluation (6)
- ▶ Python Language (2)
- ▶ R Language (1)
- ▶ Text Analytics (7)
- ▶ Computer Vision (6)
- ▶ Recommendation (5)
- ▶ Anomaly Detection (2)
- ▶ Web Service (2)

# Built-In Components

Model training includes training the following sorts of models:

- Regression
- Classification
- Clustering
- Image classification
- Anomaly detection
- Recommendation (SVD and Wide & Deep)

## ▼ Model Training (4)

### Train Clustering Model

 Microsoft

Train clustering model and assign data to clusters.

[Learn More](<https://aka.ms/aml/train-clustering-...>)

3/4/2021

### Train Model

 Microsoft

Trains a classification or regression model in a supervised manner. [Learn More](<https://aka.ms/...>)

3/4/2021

### Train PyTorch Model

 Microsoft

Train pytorch model from scratch or finetune it.

[Learn More](<https://aka.ms/aml/train-pytorch-m...>)

3/4/2021

### Tune Model Hyperparameters

 Microsoft

Perform a parameter sweep on the model to determine the optimum parameter settings. [Learn More](<https://aka.ms/aml/tune-model-hyperparameters-...>)

3/4/2021

# Custom Components

Build out components in R or Python. This includes installing custom packages.

## ▼ Python Language (2)

### Create Python Model

 Microsoft

Creates Python model using custom script. [Learn More](<https://aka.ms/aml/create-python-model>)

3/4/2021

### Execute Python Script

 Microsoft

Executes a Python script from an Azure Machine Learning designer pipeline. [Learn More](<https://...>)

3/4/2021

## ▼ R Language (1)

### Execute R Script

 Microsoft

Executes an R script from an Azure Machine Learning designer pipeline. [Learn More](<https://...>)

3/4/2021

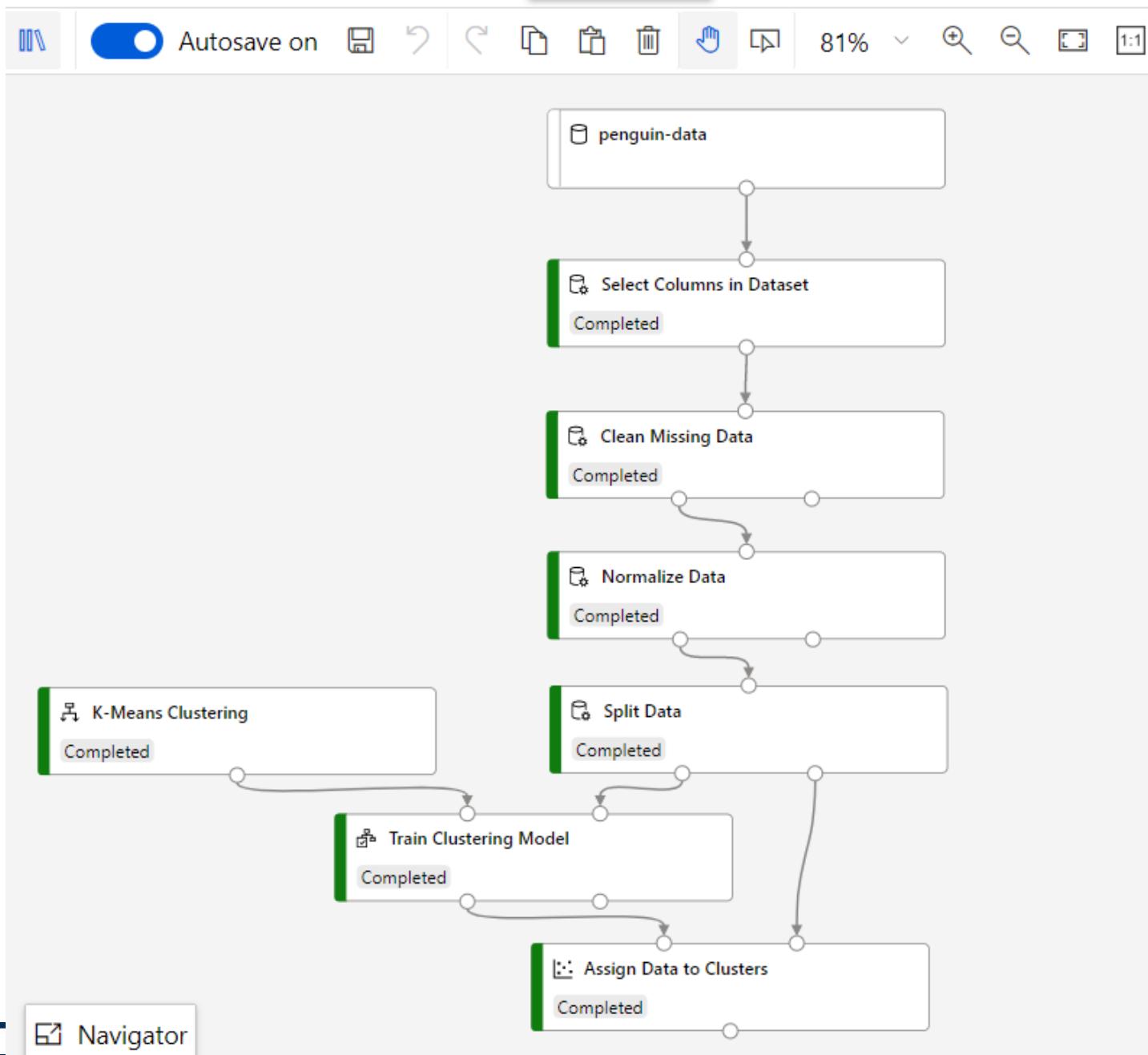
# Training a Model

Train models at the click of a button.

penguin-pipeline 

**Submit**

Create inference pipeline 



# Evaluating a Model

Regression and classification models include UI components for evaluation.

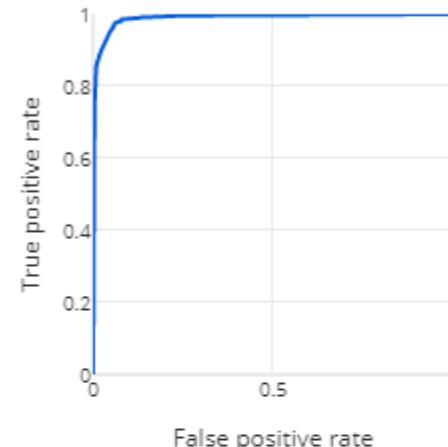
**Regression** includes statistics such as R<sup>2</sup>, MAE, MAPE, and RMSE.

**Classification** includes the confusion matrix and calculations off of it, such as accuracy, precision, and recall.

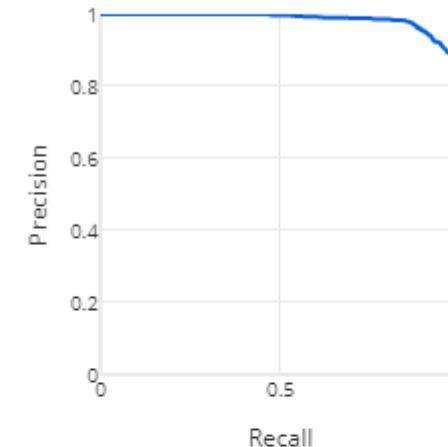
**Two-class classification** models include ROC curves and Area Under the Curve.

## Evaluate Model result visualization

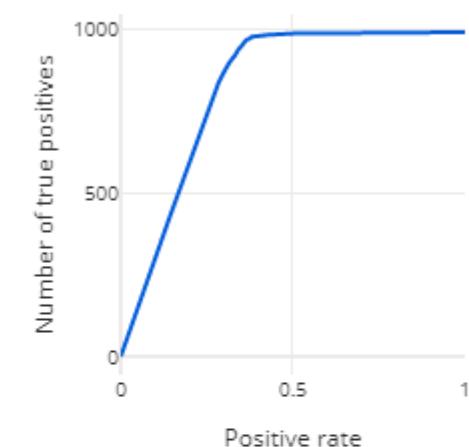
ROC curve



Precision-recall curve



Lift curve



Threshold

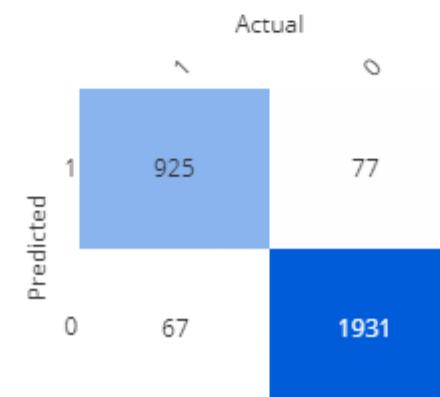
Accuracy 0.952

Precision 0.923

Recall 0.932

F1 Score 0.928

AUC 0.991



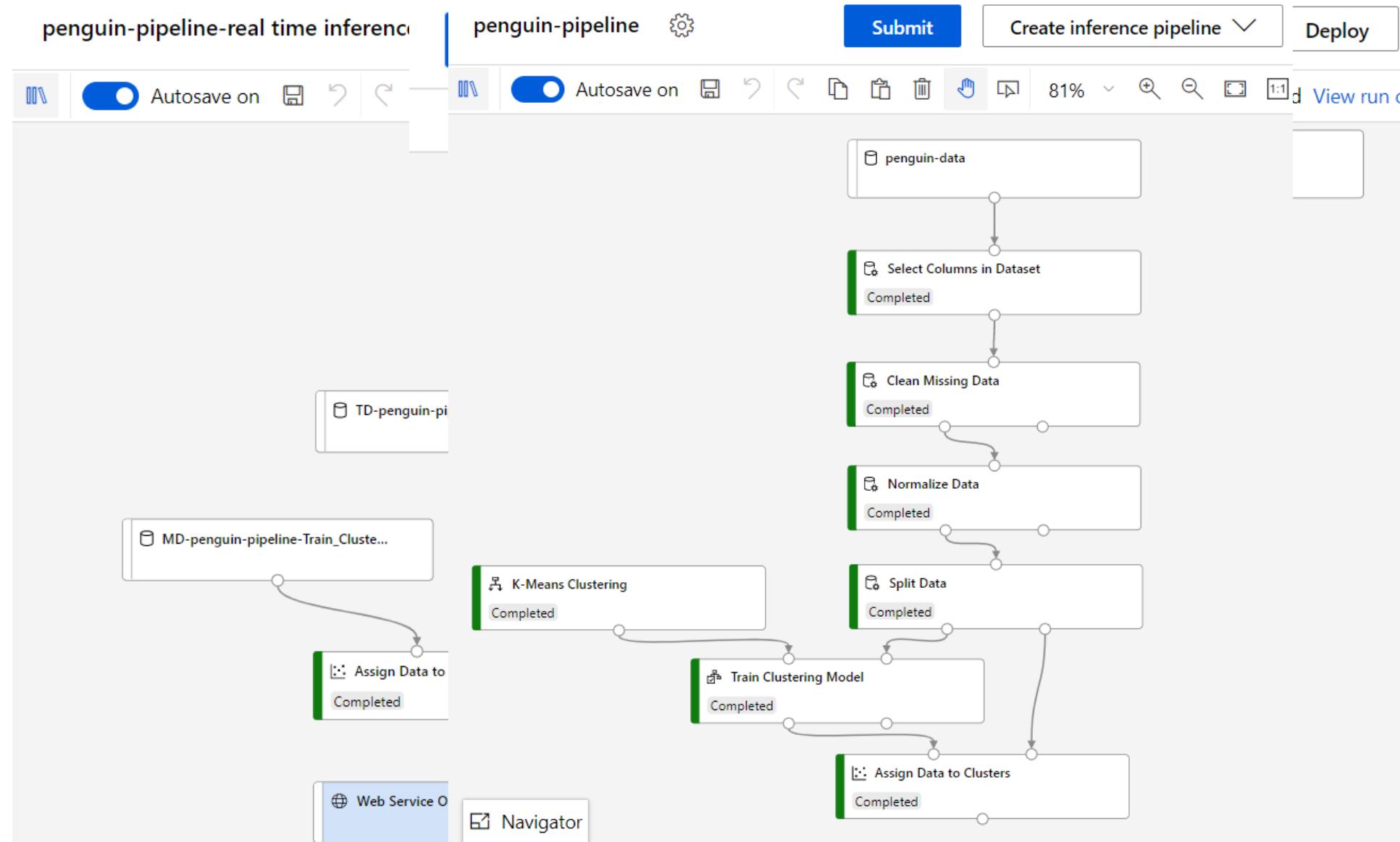
# Creating an Inference Pipeline

Create an inference pipeline with a few clicks. Two types are available:

- Real-time inference pipeline
- Batch inference pipeline

Real-time inference pipelines operate via web service and return results for one or a few observations.

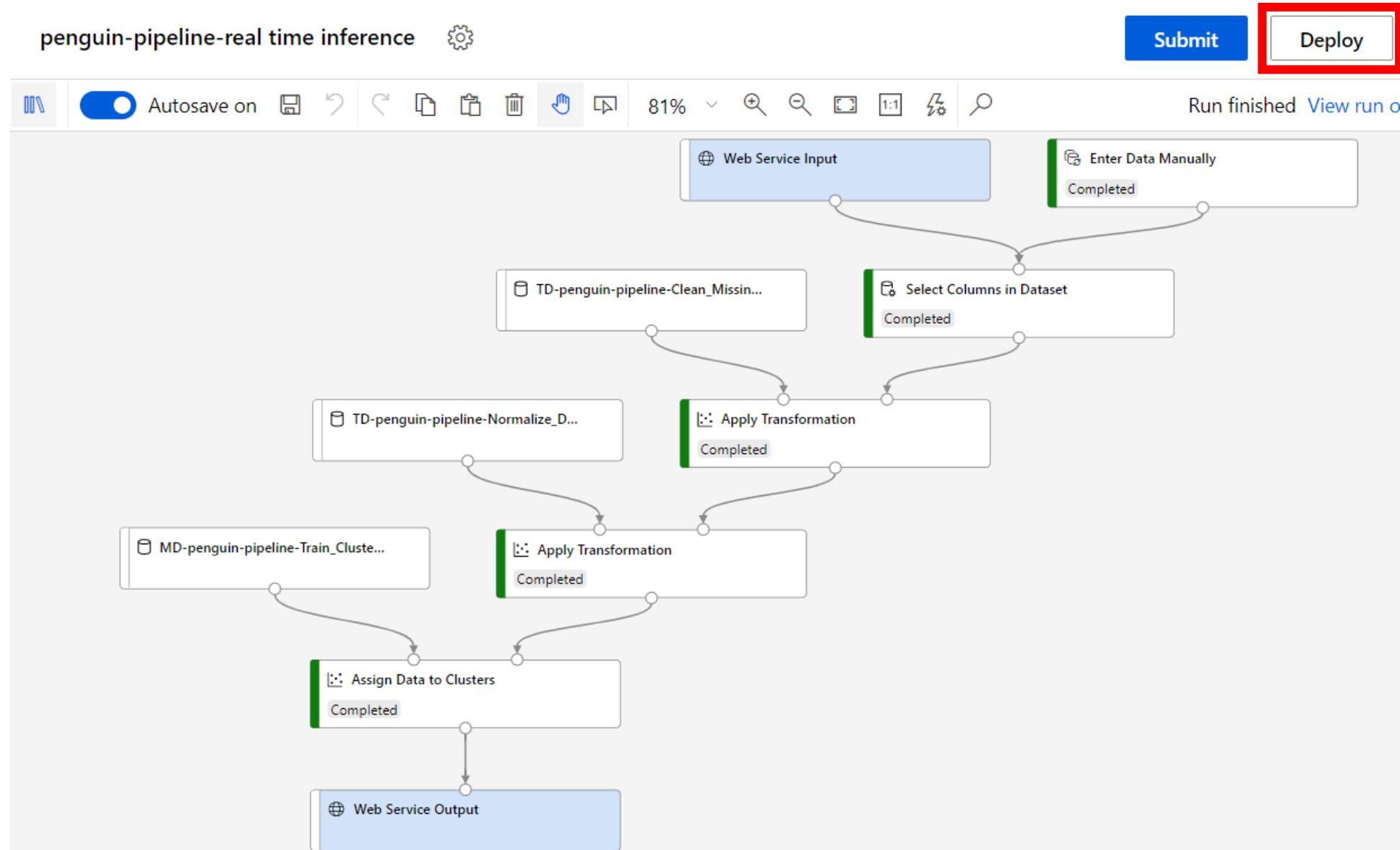
Batch inference pipelines typically write results to files or databases and operate asynchronously.



# Deploying an Inference Pipeline

Deploy a pipeline with a click of a button.

The end result is a web service available via REST call.



# Consuming the Endpoint

Consume the endpoint via HTTP. Instructions are available for C#, Python, and R, but other languages work too.

Consumption option

Consumption types

C# Python R

```
1 import urllib.request
2 import json
3 import os
4 import ssl
5
6 def allowSelfSignedHttps(allowed):
7     # bypass the server certificate verification on client side
8     if allowed and not os.environ.get('PYTHONHTTPSVERIFY', '') == '1':
9         ssl._create_default_https_context = ssl._create_unverifi
10
11 allowSelfSignedHttps(True) # this line is needed if you use self-signed cert
12
13 data = {
14     "Inputs": {
15         "WebServiceInput0": [
16             [
17                 {
18                     "CulmenLength": "39.1",
19                     "CulmenDepth": "18.7",
20                     "FlipperLength": "181",
21                     "BodyMass": "3750",
22                 },
23             ],
24         ]
25     }
26 }
```

# Automated ML

Automates the exploration of feature engineering approaches, algorithm selection and model parameters to produce the best performing ML model.

Saves you time and gives you a head-start.

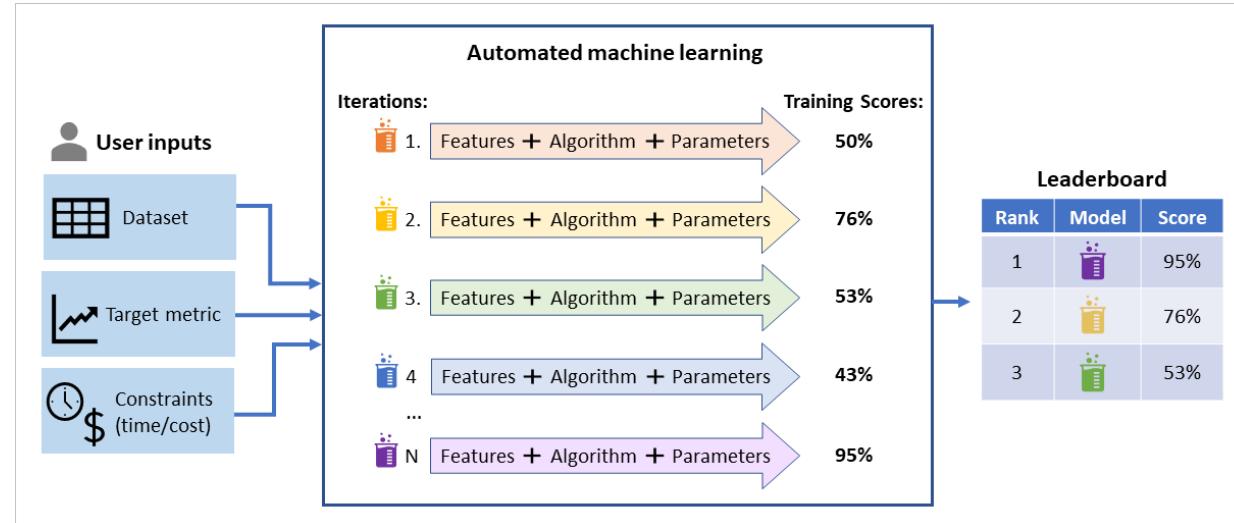
## Key Scenarios:

- Classification
- Regression
- Forecasting

\*Not suitable outside of these scenarios.

## Data Shape Required:

- Tabular (Text + Numeric)
- Tabular (Numeric)
- Time Series



## Deployment Options:

- Notebook
- Web Service
- ONNX

Best model

\*Not all algorithms support ONNX export.

## Feature Engineering:

- Scaling/Normalization
- Dimensionality Reduction
- Encoding

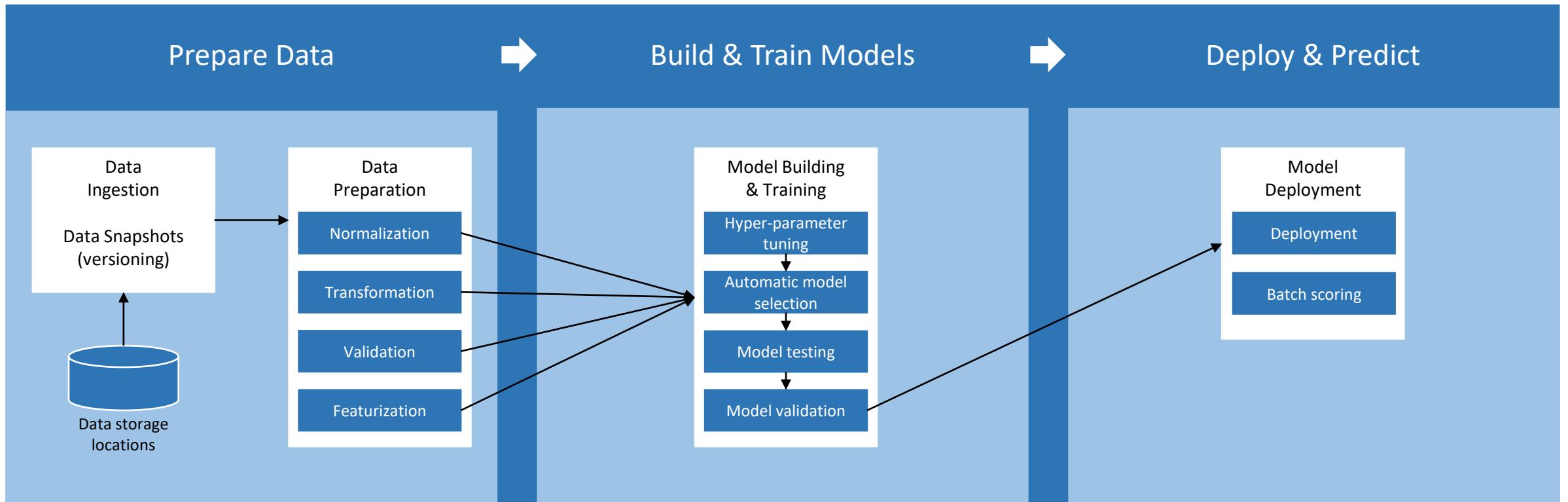
## Algorithms (count):

- Classification (14)
- Regression (12)
- Forecasting (17)

\*Includes Regressors, Trees, Ensembles & Deep Neural Networks

# Azure Machine Learning pipelines (Overview)

- Pipeline: distinct steps to facilitate model training, deployment, and inference



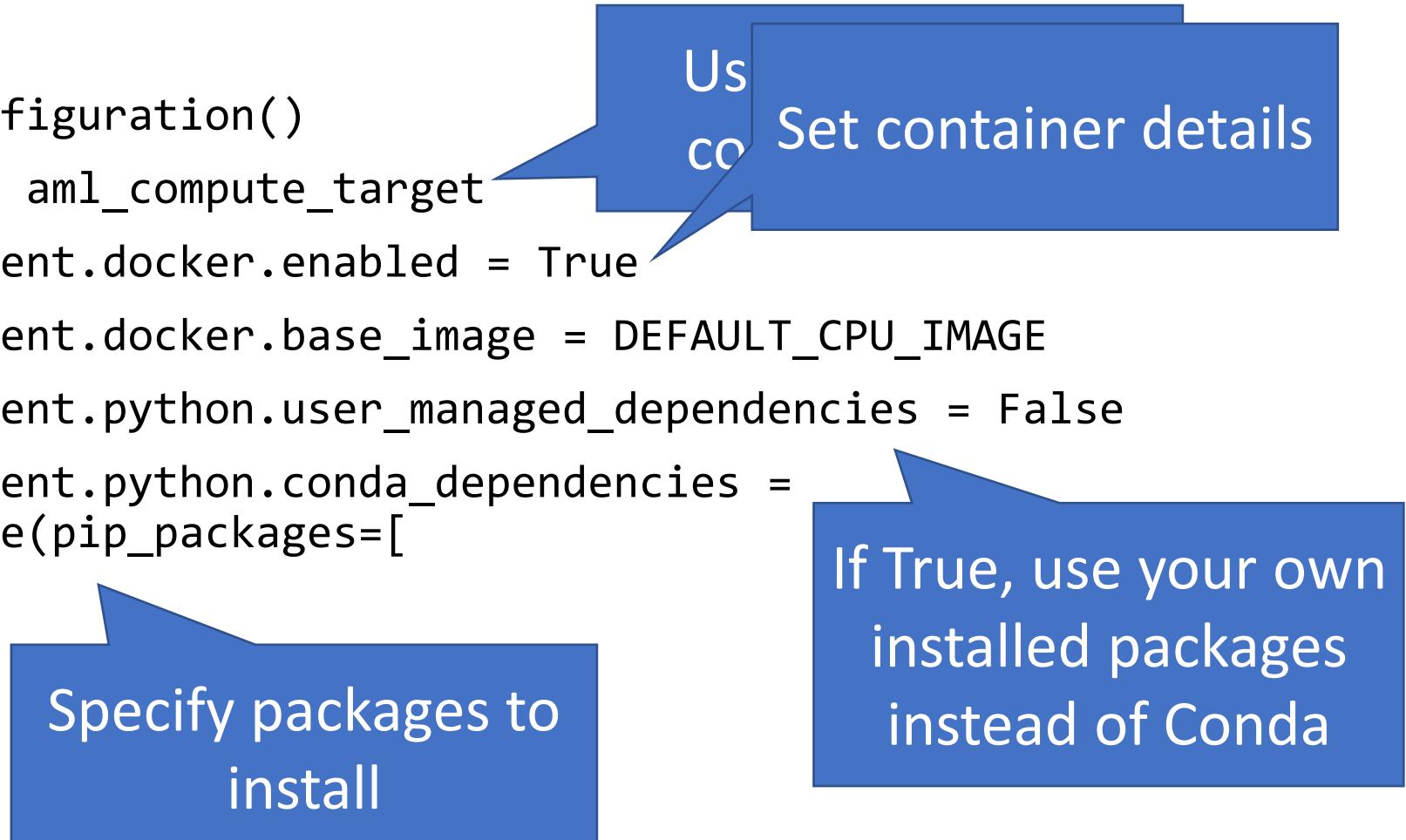
# Create AML Compute

- Provision and manage clusters of Azure VMs for running ML workloads
- Needed to execute pipelines
- Can re-use compute clusters across ML jobs or create new clusters per job

```
provisioning_config =  
    AmlCompute.provisioning_configuration(  
        vm_size = "STANDARD_D2_V2",  
        min_nodes = 1,  
        max_nodes = 1)  
  
aml_compute =  
    ComputeTarget.create(  
        ws,  
        aml_compute_target,  
        provisioning_config)
```

# Create Run Configuration

```
run_amlcompute = RunConfiguration()  
run_amlcompute.target = aml_compute_target  
run_amlcompute.environment.docker.enabled = True  
run_amlcompute.environment.docker.base_image = DEFAULT_CPU_IMAGE  
run_amlcompute.environment.python.user_managed_dependencies = False  
run_amlcompute.environment.python.conda_dependencies =  
    CondaDependencies.create(pip_packages=[  
        'numpy',  
        'pandas',  
        'scikit-learn',  
        'sklearn_pandas'  
    ])
```



Use  
co

Set container details

If True, use your own  
installed packages  
instead of Conda

Specify packages to  
install

# Walkthrough: Creating a Pipeline

In [28]: `RunDetails(batch_scoring_pipeline_run).show()`

Run Properties	
Status	Completed
Start Time	7/11/2019 5:10:18 PM
Duration	0:01:49
Run Id	28c53c32-cbc3-4f52-931e-574cdc686141
End Time	7/11/2019 5:12:08 PM

Output Logs  logs/azureml/executionlogs.txt  Auto-switch

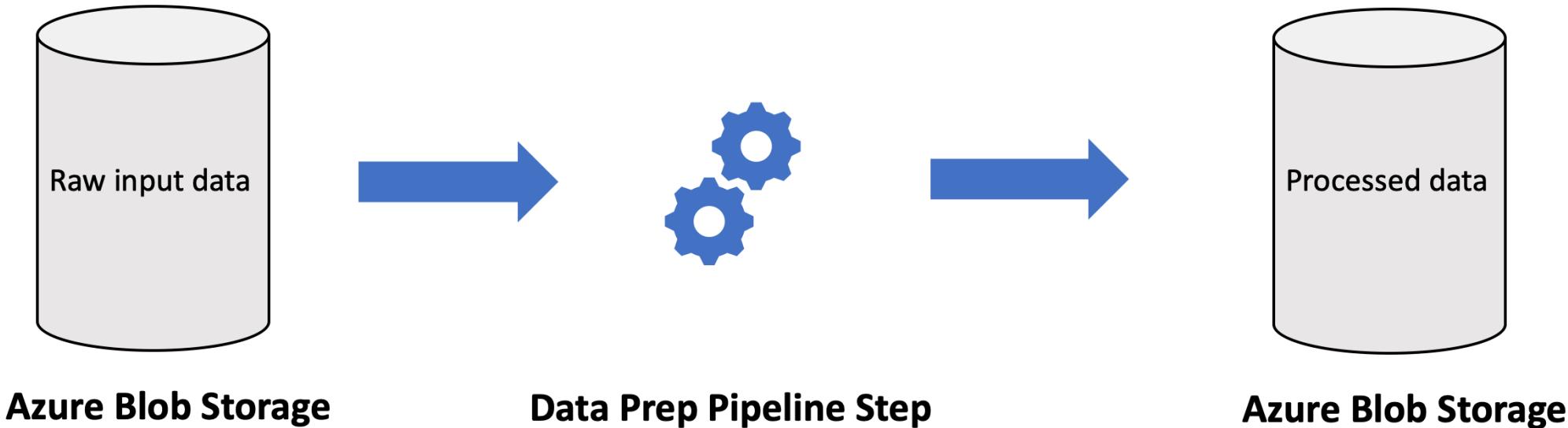
```
[2019-07-11 21:10:23Z] Completing processing run id 6bf70fc6-d7cc-4b21-a556-e9acaccd101c.  
[2019-07-11 21:10:24Z] Completing processing run id 721316d5-79ad-4275-9c31-30e6964d0463.  
[2019-07-11 21:10:24Z] Submitting run id b9a04594-f169-4ce8-adc4-5d03a9f2bd25 in experiment Sample-ML-Pipeline  
[2019-07-11 21:11:22Z] Completing processing run id b9a04594-f169-4ce8-adc4-5d03a9f2bd25.  
[2019-07-11 21:11:22Z] Submitting run id 3263b271-f9ae-4462-9c93-56a0f7975c23 in experiment Sample-ML-Pipeline  
[2019-07-11 21:12:04Z] Completing processing run id 3263b271-f9ae-4462-9c93-56a0f7975c23.  
  
Run is completed.
```

Step	Status	Reused	Created	Ended	Duration	Run Id
inference	Finished		Jul 11, 2019 5:11 PM	Jul 11, 2019 5:12 PM	0:00:38	3263b271-f9ae-4462-9c93-56a0f7975c23
process_batch_scoring_data	Running		Jul 11, 2019 5:10 PM	Jul 11, 2019 5:11 PM	0:00:54	b9a04594-f169-4ce8-adc4-5d03a9f2bd25
train	Finished	Yes	Jul 11, 2019 5:10 PM	Jul 11, 2019 5:10 PM	0:00:00	721316d5-79ad-4275-9c31-30e6964d0463
process_data	Finished	Yes	Jul 11, 2019 5:10 PM	Jul 11, 2019 5:10 PM	0:00:00	6bf70fc6-d7cc-4b21-a556-e9acaccd101c

```
graph TD; raw_data[raw_data] --> raw_batch_scoring_data[raw_batch_scoring_data]; raw_batch_scoring_data --> process_data[process_data - Finished]; process_data --> train[train - Finished]; train --> inference[inference - Finished]; inference --> inference_data[inference data]; process_data --> process_data_data[process_data data]; train --> trained_model[trained_model]; process_data_data --> batch_scoring_processed_data[batch_scoring_processed_data]; trained_model --> batch_scoring_processed_data; inference --> batch_scoring_processed_data;
```

The diagram illustrates the data flow of a machine learning pipeline. It starts with a blue box labeled "raw\_data". An arrow points from "raw\_data" to a blue box labeled "raw\_batch\_scoring\_data". From "raw\_batch\_scoring\_data", an arrow points to a green box labeled "process\_data - Finished". From "process\_data - Finished", an arrow points to a green box labeled "train - Finished". From "train - Finished", an arrow points to a green box labeled "inference - Finished". Additionally, there is a direct arrow from "raw\_data" to a green box labeled "process\_data - Finished". From "process\_data - Finished", an arrow points to a green box labeled "process\_data data". From "train - Finished", an arrow points to a green box labeled "trained\_model". Finally, there are two arrows pointing from "process\_data data" and "trained\_model" to a green box labeled "batch\_scoring\_processed\_data".

# Create Data Prep Step



- Normalization
- Transformation
- Validation
- Featurization

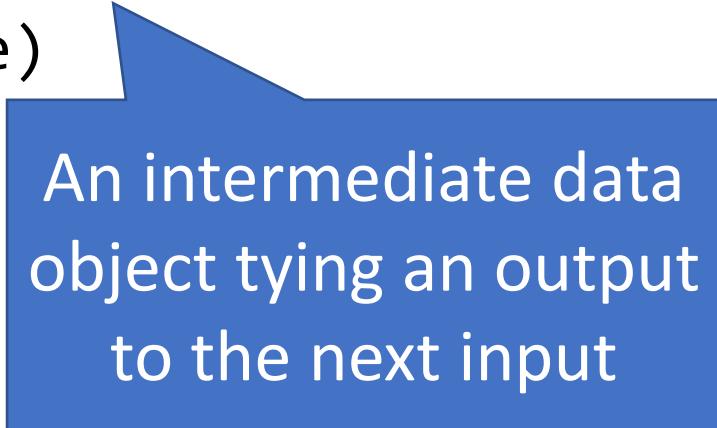
# Create Data Reference

```
from azureml.data.data_reference import DataReference  
  
def_blob_store = ws.get_default_datastore()  
raw_data = DataReference(  
    datastore=def_blob_store,  
    data_reference_name="raw_data",  
    path_on_datastore=".../...")
```

The input to our Data  
Prep Pipeline step

# Create Pipeline Data for Prepared Data

```
from azureml.pipeline.core import PipelineData  
  
processed_data = PipelineData(  
    'processed_data',  
    datastore=def_blob_store)
```



An intermediate data object tying an output to the next input

# Create Data Prep Step

```
dataPrepStep = PythonScriptStep(  
    name="process_data",  
    source_directory="...",  
    script_name="process.py",  
    arguments=[>--process_mode", 'train',  
              "--input", raw_data,  
              "--output", processed_data],  
    inputs=[raw_data],  
    outputs=[processed_data],  
    compute_target=aml_compute,  
    runconfig=run_amldata)
```

Path

Three input  
arguments:  
train/inference, input  
loc, output loc

raw\_data,

Specify the input and  
output objects

aml\_compute

Specify the compute  
and config choices

# Viewing process.py

```
# Parse arguments: process_mode, input, output
data = pd.read_csv(args.input)
if(args.process_mode == 'train'): ...
elif(args.process_mode == 'inference'): ...
else:
    print('Invalid process_mode!')
os.makedirs(args.output, exist_ok=True)
data.to_csv(os.path.join(args.output, "processed-data.csv"),
header=True, index=False)
```

Read in input data given a file path with parameter values

Raw inference data won't include labels, so behavior changes!

# Create Model Training Step

```
trained_model = PipelineData('trained_model',  
datastore=def_blob_store)  
trainStep = PythonScriptStep(  
    name="train",  
    source_directory="...",  
    script_name="train.py",  
    arguments=["--input", processed_data, "--output", trained_model],  
    inputs=[processed_train_data],  
    outputs=[trained_model],  
    compute_target=aml_compute,  
    runconfig=run_amlcompute  
)
```

Build a new

Locate and process  
train.py

processed\_data was  
our prior step's  
output

Use the same  
compute and config

# Run the Training Pipeline

```
from azureml.pipeline.core import Pipeline  
from azureml.core import Experiment
```

Build a pipeline, tying together the prior steps

```
pipeline = Pipeline(workspace=ws, steps=[trainStep1])  
pipeline.validate()  
experiment_name = 'Sample-ML-Pipeline'  
pipeline_run = Experiment(ws, experiment_name)  
    .submit(pipeline)  
RunDetails(pipeline_run).show()
```

Simple validation: check for unconnected inputs, etc.

Create an experiment and execute the pipeline

Show the pipeline

```
RunDetails(pipeline_run).show()
```

### Run Properties

Status	Completed
Start Time	7/11/2019 9:29:39 AM
Duration	0:06:13
Run Id	ae3ac08d-7b15-4d35-bb0c-23a8ff01c2e8
End Time	7/11/2019 9:35:53 AM

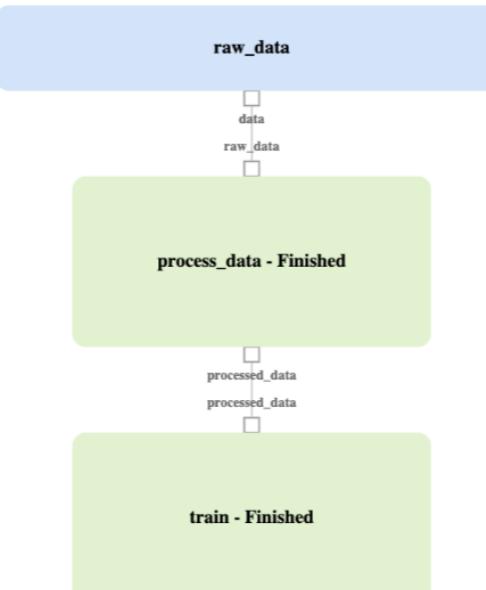
### Output Logs

logs/azureml/executionlogs.txt

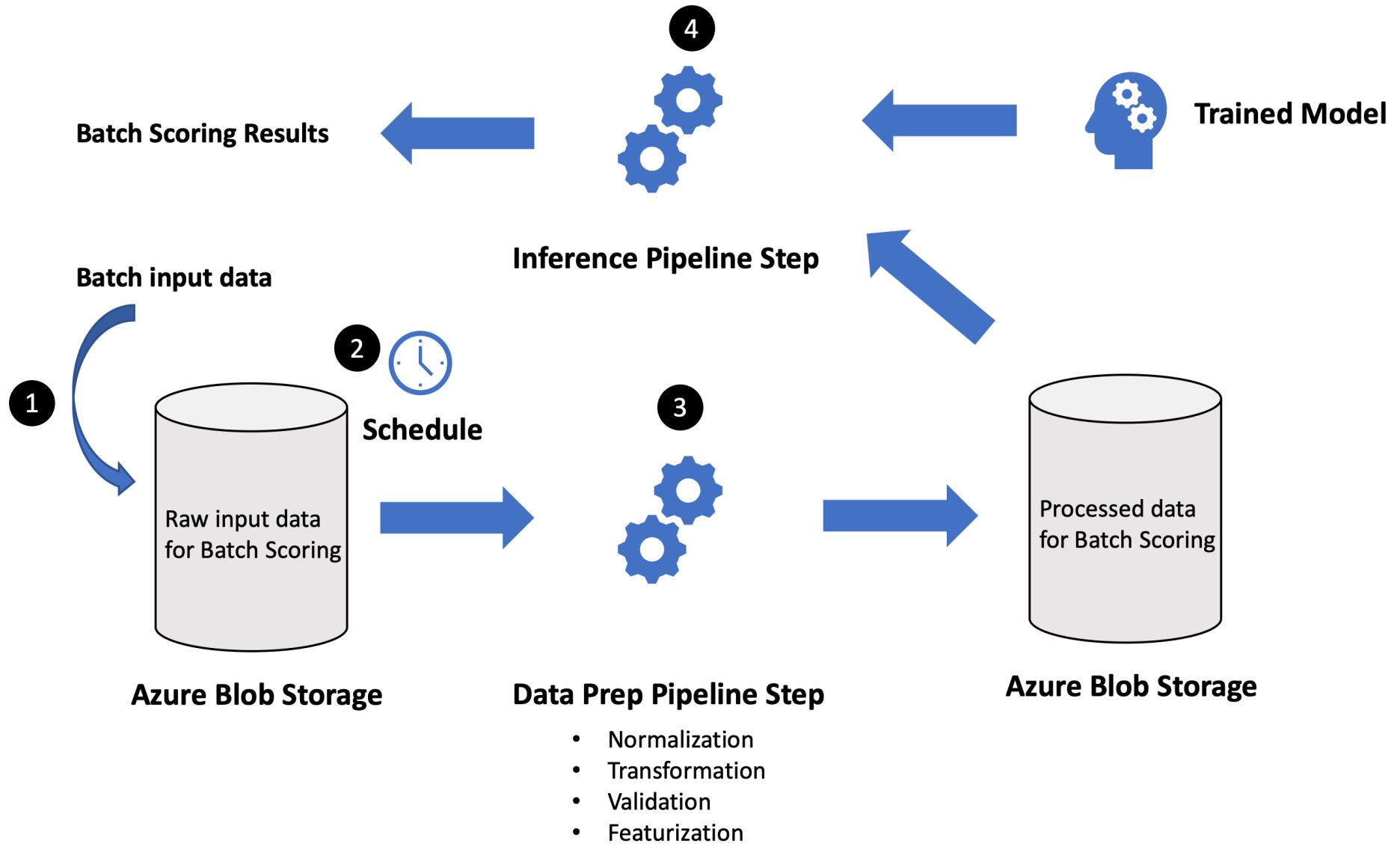
Auto-switch

```
[2019-07-11 13:29:44Z] Submitting run id 5f329b0e-0a62-48a6-a1be-424468a5a79a in experiment  
Sample-ML-Pipeline  
[2019-07-11 13:35:15Z] Completing processing run id 5f329b0e-0a62-48a6-a1be-424468a5a79a.  
[2019-07-11 13:35:15Z] Submitting run id de0f8fd1-4e0d-4ec3-9f64-b208a75acb32 in experiment  
Sample-ML-Pipeline  
[2019-07-11 13:35:52Z] Completing processing run id de0f8fd1-4e0d-4ec3-9f64-b208a75acb32.  
  
Run is completed.
```

Step	Status	Reused	Created	Ended	Duration	Run Id
train	Finished		Jul 11, 2019 9:35 AM	Jul 11, 2019 9:35 AM	0:00:31	de0f8fd1-4e0d-4ec3-9f64-b208a75acb32
process_data	Finished		Jul 11, 2019 9:29 AM	Jul 11, 2019 9:35 AM	0:05:29	5f329b0e-0a62-48a6-a1be-424468a5a79a



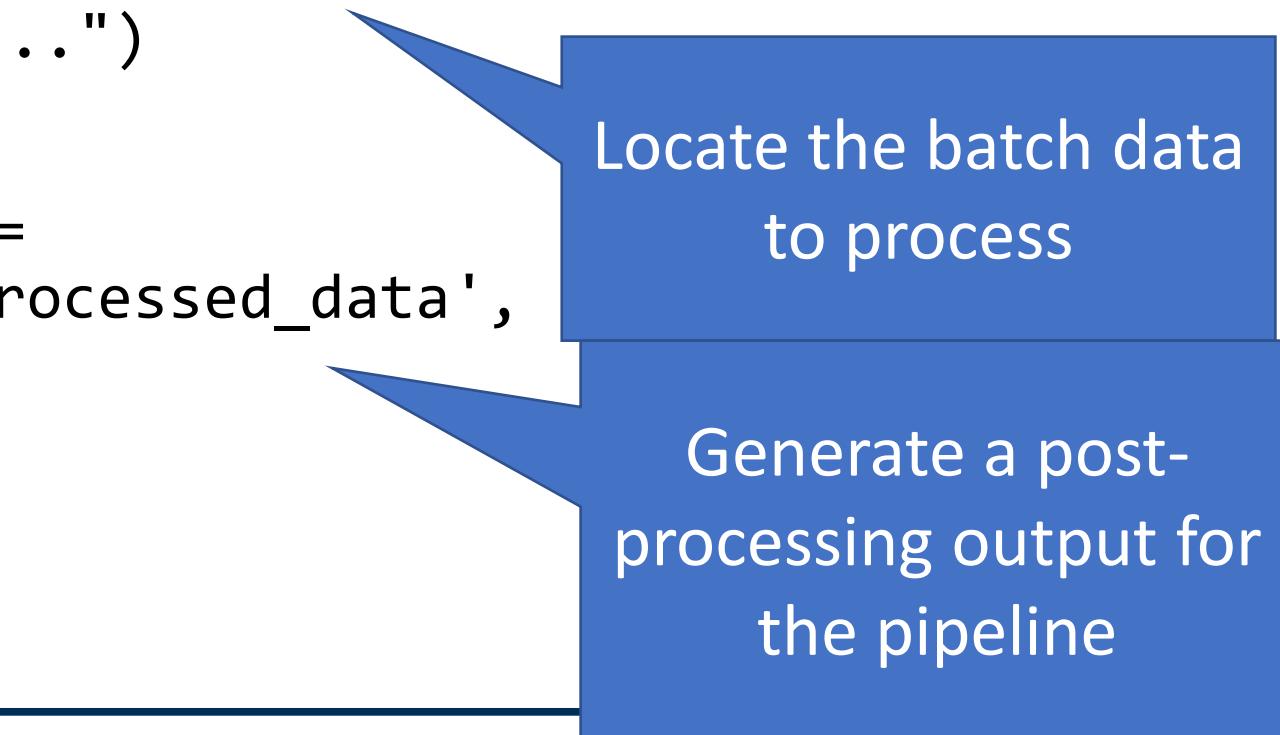
# Create the Batch Scoring Pipeline



# Create the Batch Scoring Pipeline

```
raw_batch_scoring_data = DataReference(  
    datastore=def_blob_store,  
    data_reference_name="raw_batch_scoring_data",  
    path_on_datastore=".../...")
```

```
batch_scoring_processed_data =  
PipelineData('batch_scoring_processed_data',  
datastore=def_blob_store)
```



Locate the batch data to process

Generate a post-processing output for the pipeline

# Create the Batch Scoring Pipeline

```
batchDataPrepStep = PythonScriptStep(  
    name="process_batch_scoring_data",  
    source_directory="...",  
    script_name="process.py",  
    arguments=[  
        "--process_mode", 'inference',  
        "--input", raw_batch_scoring_data,  
        "--output", batch_scoring_processed_data],  
    inputs=[raw_batch_scoring_data],  
    outputs=[batch_scoring_processed_data],  
    allow_reuse = False,  
    compute_target=aml_compute, runconfig=run_
```

The diagram illustrates three annotations pointing to specific parts of the Python code:

- A blue callout points to the first three arguments: `--process_mode`, `'inference'`, and `--input`. The text inside says: "Three input arguments: train/inference, input".
- A second blue callout points to the `inputs` and `outputs` parameters. The text inside says: "Specify the input and output objects".
- A third blue callout points to the `allow_reuse` parameter. The text inside says: "Can the step use previous results when re-run with the same settings?".

# Create the Batch Scoring Pipeline

```
batch_scoring_results = PipelineData('batch_scoring_results',  
datastore=def_blob_store)
```

```
inferenceStep = PythonScriptStep(
```

```
    name="inference",
```

```
    source_directory="...>,
```

```
    script_name="inference.py",
```

```
    arguments=["--input", batch_scoring_processed_data,
```

```
                "--model", trained_model,
```

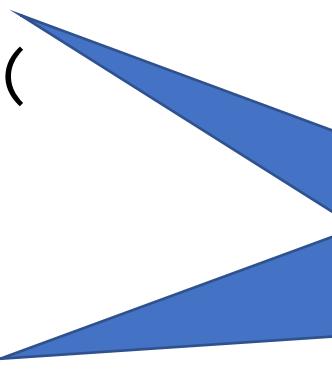
```
                "--output", batch_scoring_results],
```

```
    inputs=[batch_scoring_processed_data, trained_model],
```

```
    outputs=[batch_scoring_results],
```

```
    compute_target=aml_compute,
```

```
    runconfig=run_amldatastore)
```



Inference file (not shown) loads model & data, predicts, and saves in  
batch\_scoring\_results

# Publish Batch Scoring Pipeline

```
batch_scoring_pipeline = Pipeline(workspace=ws,  
steps=[inferenceStep])
```

```
batch_scoring_pipeline.validate()
```

```
pipeline_name = 'Batch Scoring Pipeline'
```

```
published_pipeline = batch_scoring_pipeline.publish(name =  
pipeline_name)
```

Simple validation:  
check for  
unconnected inputs,  
etc.

Publish the pipeline,  
making it available to  
run on demand

# Schedule Batch Scoring Pipeline

```
from azureml.pipeline.core.schedule import Schedule  
schedule = Schedule.create(  
    workspace=ws,  
    name=pipeline_name + "_sch",  
    pipeline_id=published_pipeline.id,  
    experiment_name=experiment_name,  
    datastore=def_blob_store,  
    wait_for_provisioning=True,  
    description="Scheduler for Pipeline: " + pipeline_name,  
    path_on_datastore="...",  
    polling_interval=1  
)
```

Create a schedule for an experiment to run

Monitor the datastore for changes

Folder where the new input data is uploaded

Polling interval minutes

# Schedule Batch Scoring Pipeline

```
batch_scoring_pipeline_run = schedule.get_last_pipeline_run()  
RunDetails(batch_scoring_pipeline_run).show()
```

```
In [28]: RunDetails(batch_scoring_pipeline_run).show()
```

### Run Properties

Status	Completed
Start Time	7/11/2019 5:10:18 PM
Duration	0:01:49
Run Id	28c53c32-cbc3-4f52-931e-574cdc686141
End Time	7/11/2019 5:12:08 PM

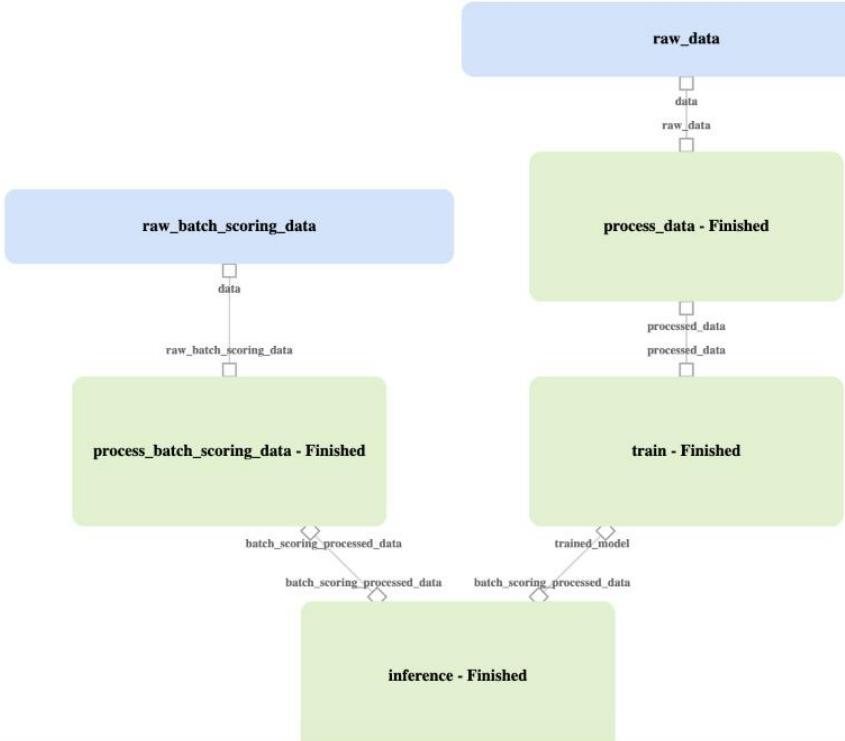
### Output Logs

logs/azureml/executionlogs.txt

Auto-switch

```
[2019-07-11 21:10:23Z] Completing processing run id 6bf70fc6-d7cc-4b21-a556-e9acaccd101c.  
[2019-07-11 21:10:24Z] Completing processing run id 721316d5-79ad-4275-9c31-30e6964d0463.  
[2019-07-11 21:10:24Z] Submitting run id b9a04594-f169-4ce8-adc4-5d03a9f2bd25 in experiment  
Sample-ML-Pipeline  
[2019-07-11 21:11:22Z] Completing processing run id b9a04594-f169-4ce8-adc4-5d03a9f2bd25.  
[2019-07-11 21:11:22Z] Submitting run id 3263b271-f9ae-4462-9c93-56a0f7975c23 in experiment  
Sample-ML-Pipeline  
[2019-07-11 21:12:04Z] Completing processing run id 3263b271-f9ae-4462-9c93-56a0f7975c23.  
  
Run is completed.
```

Step	Status	Reused	Created	Ended	Duration	Run Id
inference	Finished		Jul 11, 2019 5:11 PM	Jul 11, 2019 5:12 PM	0:00:38	3263b271-f9ae-4462-9c93-56a0f7975c23
process_batch_scoring_data	Finished		Jul 11, 2019 5:10 PM	Jul 11, 2019 5:11 PM	0:00:54	b9a04594-f169-4ce8-adc4-5d03a9f2bd25
train	Finished	Yes	Jul 11, 2019 5:10 PM	Jul 11, 2019 5:10 PM	0:00:00	721316d5-79ad-4275-9c31-30e6964d0463
process_data	Finished	Yes	Jul 11, 2019 5:10 PM	Jul 11, 2019 5:10 PM	0:00:00	6bf70fc6-d7cc-4b21-a556-e9acaccd101c



# DEMO: The Azure Machine Learning Workspace

Azure Data Conference

POWERED BY

Azure Data Community & Azure Data Conf

---

# Responsible AI and Machine Learning

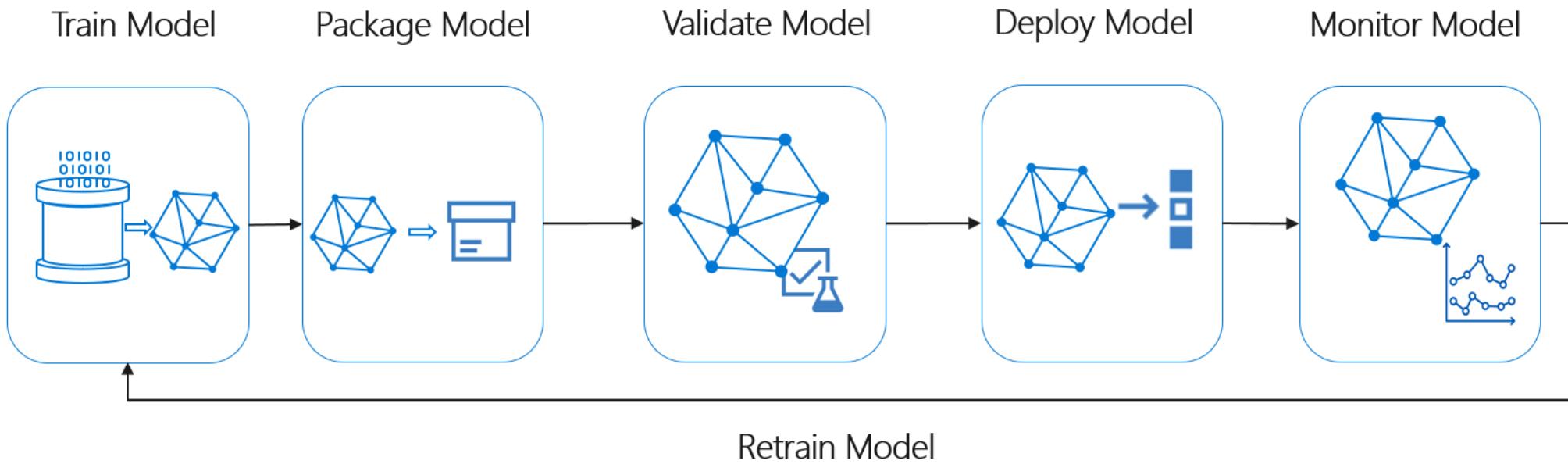
Azure Data Conference

POWERED BY

Azure Data Community & Azure Data Conf

---

# The Machine Learning Lifecycle



# Responsible AI



Fairness



Reliability  
& Safety



Privacy &  
Security



Inclusiveness



Transparency



Accountability

# Fairness

“AI systems should treat all people fairly”

# Reliability & Safety

“AI systems should perform reliably and safely”



# Privacy & Security

“AI systems should be secure and respect privacy”



# Inclusiveness

“AI systems should empower everyone and engage people”



# Transparency

“AI systems should be understandable”

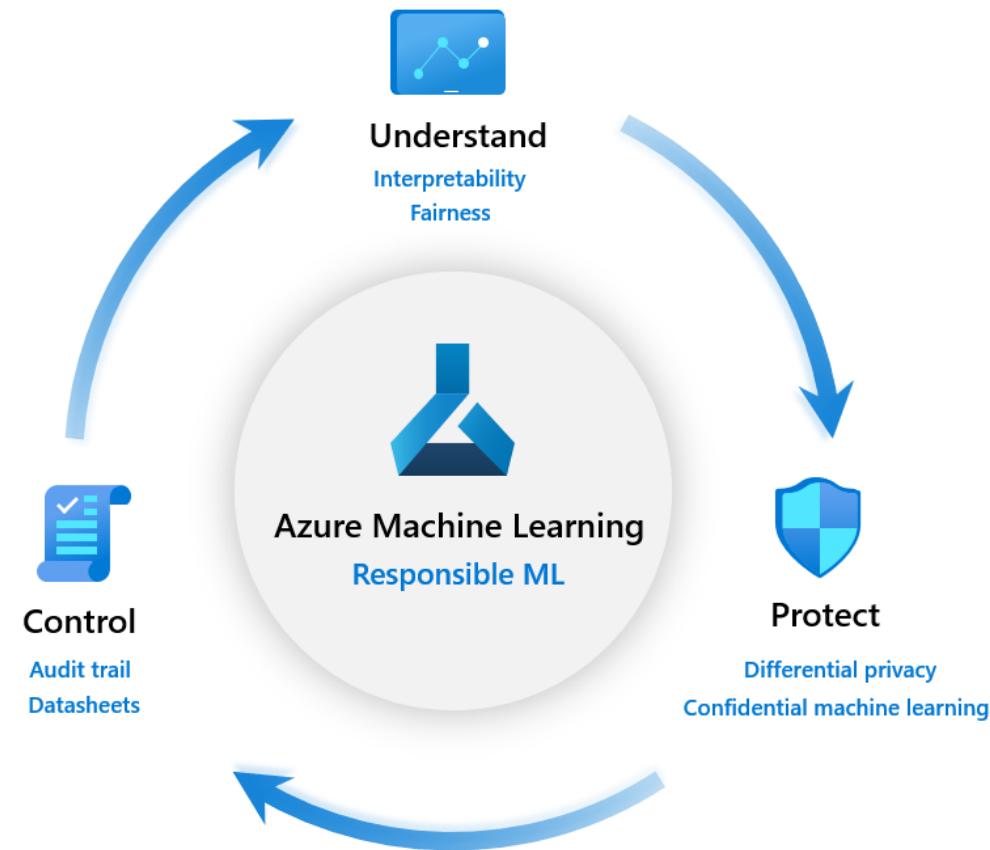


# Accountability

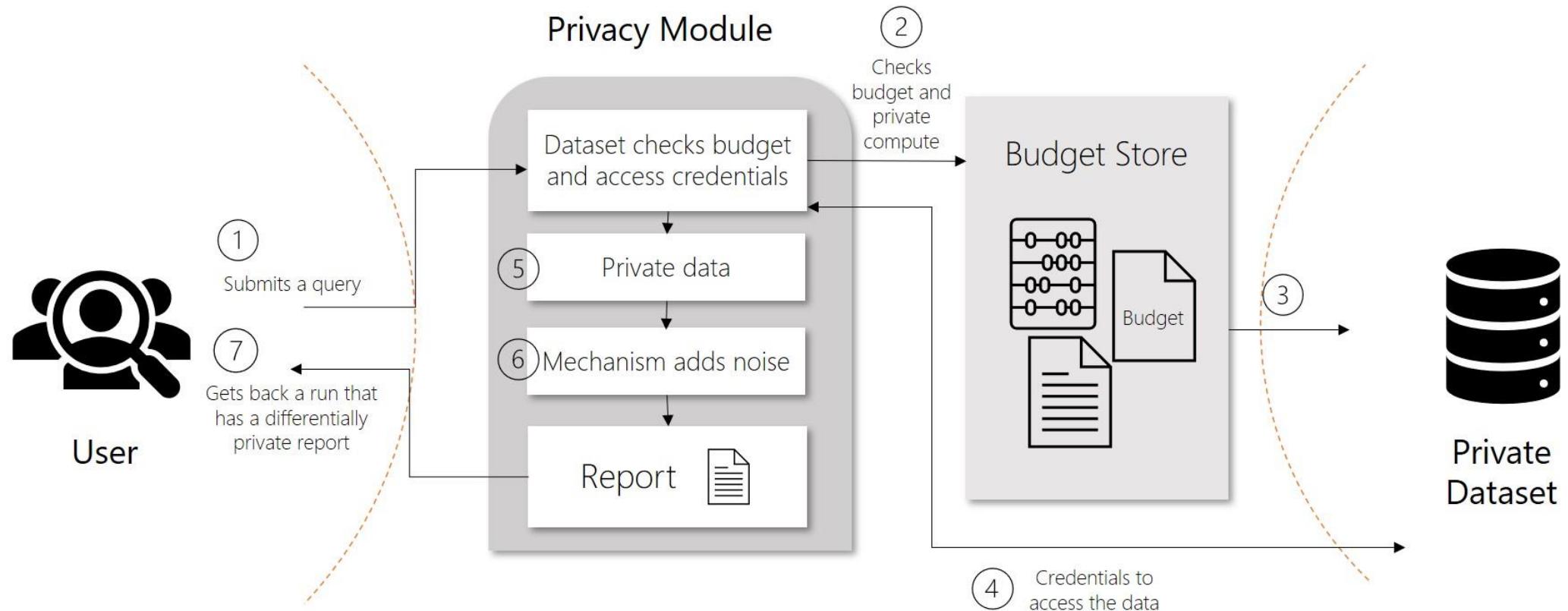
“People should be accountable for AI systems”



# Responsible Machine Learning

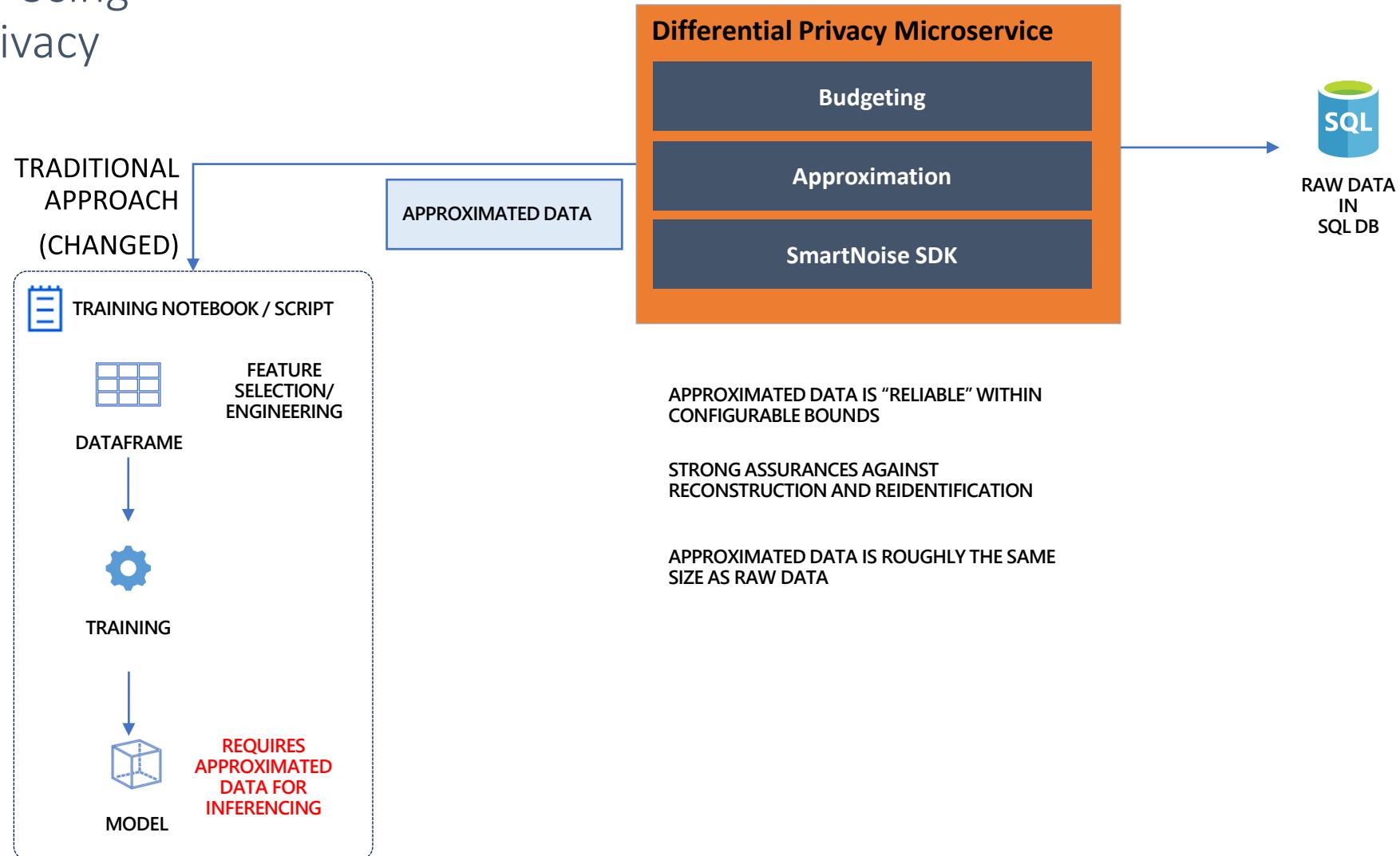


# Differential Privacy

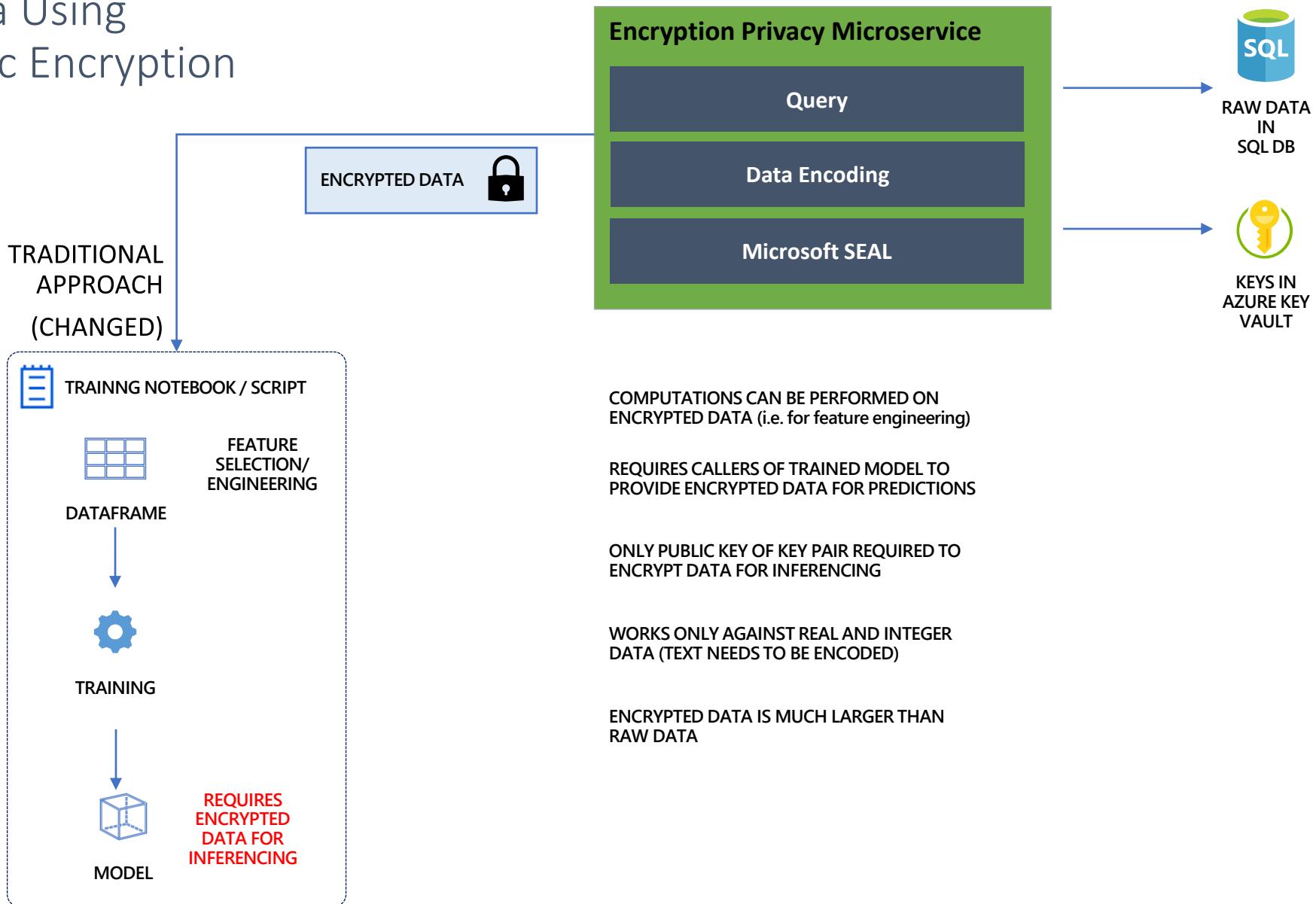


SmartNoise SDK

# Model Training with Sensitive Data Using Differential Privacy



# Model Training with Sensitive Data Using Homomorphic Encryption



# Using compute resources in Azure Machine Learning

Azure Data Conference  
POWERED BY  
[Azure Data Community & Azure Data Conf](#)

---

# Agenda

Compute Instance

Compute Clusters

Inference Clusters

Managed Compute

Environments

Pipelines

GPU and FPGA Compute

Serverless/Azure Functions/App Service Integration

Support for NVIDIA Triton

# Compute Instances

The screenshot shows the Microsoft Azure Machine Learning interface for creating a new compute instance. The left sidebar lists various ML components like Notebooks, Datasets, and Models. The main area shows a 'Compute' section with tabs for 'Compute instances', 'Compute clusters', and 'Inference clusters'. A note about COVID-19 service prioritization is displayed. The right side is a 'New compute instance' dialog with fields for name ('mlcompute'), region ('eastus'), virtual machine type ('CPU (Central Processing Unit)'), size ('Standard\_DS3\_v2'), and SSH access. Buttons for 'Create' and 'Cancel' are at the bottom.

Microsoft Azure Machine Learning

AML\_Whitepaper > Compute

Compute

Compute instances Compute clusters Inference clusters

In the wake of COVID-19, we are prioritizing maintaining service availability.

New compute instance

Compute name \* mlcompute

Region \* eastus

Virtual machine type \* CPU (Central Processing Unit)

Virtual machine size \* Standard\_DS3\_v2 4 Cores, 14 GB (RAM), 28 GB (Disk)

Enable SSH access

Get started with scripts by Choose from tools such as frameworks.

Create View Azure

Download a template for automation

# Compute Clusters

The screenshot shows the Microsoft Azure Machine Learning interface for creating a new compute cluster. The left sidebar navigation includes 'New', 'Home', 'Author', 'Notebooks', 'Automated ML (preview)', 'Designer (preview)', 'Assets', 'Datasets', 'Experiments', 'Pipelines', 'Models', 'Endpoints', 'Manage', 'Compute' (which is selected), 'Datastores', and 'Data Labeling'. The main content area is titled 'Compute' and shows 'Compute instances' and 'Compute clusters' tabs, with 'Compute clusters' selected. A sub-dialog titled 'New compute cluster' is open, containing the following fields:

- Compute name \***: iAMLCuster
- Region \***: eastus
- Virtual machine type \***: CPU (Central Processing Unit)
- Virtual machine priority \***: Dedicated (selected)
- Virtual machine size \***: Standard\_DS3\_v2 (4 Cores, 14 GB (RAM), 28 GB (Disk))
- Minimum number of nodes \***: 0
- Maximum number of nodes \***: 6
- Idle seconds before scale down \***: 120

At the bottom of the dialog are links for 'Advanced settings', 'Download a template for automation', a 'Create' button (which has a mouse cursor hovering over it), and a 'Cancel' button.

# Inference Clusters

The screenshot shows the Microsoft Azure Machine Learning interface. On the left, there's a navigation sidebar with options like New, Home, Author, Notebooks, Automated ML (preview), Designer (preview), Datasets, Experiments, Pipelines, Models, Endpoints, Manage, Compute, Datastores, and Data Labeling. The Compute option is currently selected.

The main area has a breadcrumb trail: AML\_Whitepaper > Compute. Below it, there are tabs for Compute instances, Compute clusters, Inference clusters (which is underlined, indicating it's the active tab), and Attached clusters. A modal dialog titled "New inference cluster" is open on the right.

The "New inference cluster" dialog contains the following fields:

- Compute name \***: linoinferencecaeks
- Kubernetes Service**:
- Region \***: East US
- Virtual machine size \***: Standard\_D3\_v2 (4 Cores, 14 GB (RAM), 200 GB (Disk))
- Cluster purpose**:  Production  Dev-test
- Number of nodes \***: 3
- Network configuration**:
- Enable SSL configuration

At the bottom of the dialog, there are buttons for "Download a template for automation", "Create", and "Cancel".

# Azure Machine Learning environments

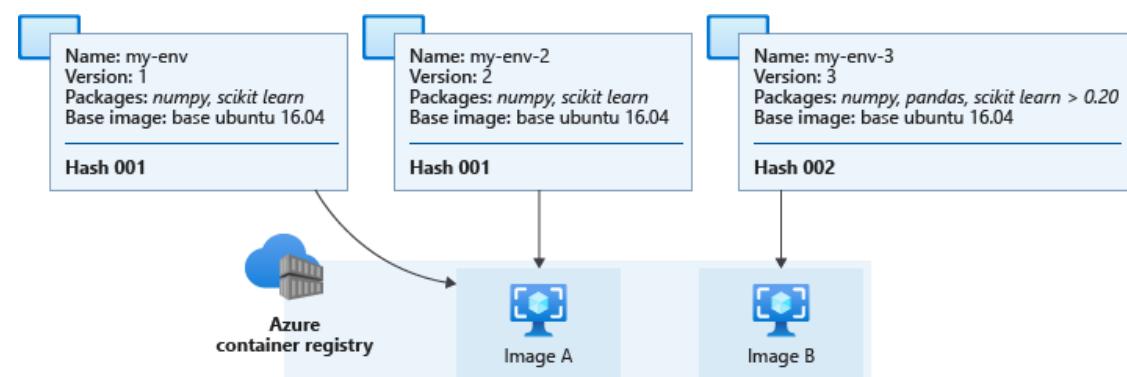
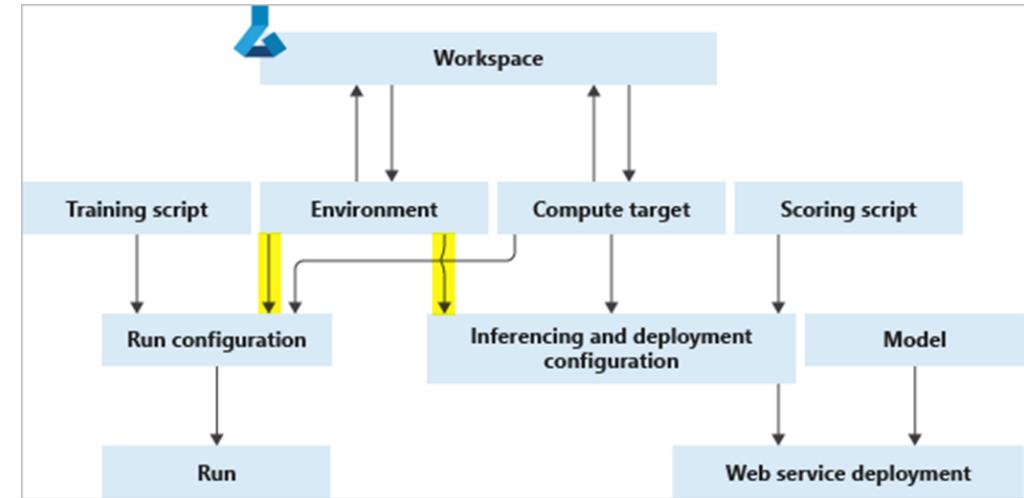
Reusable across training and scoring

Three types:

- Curated
- User-managed
- System-managed (default)

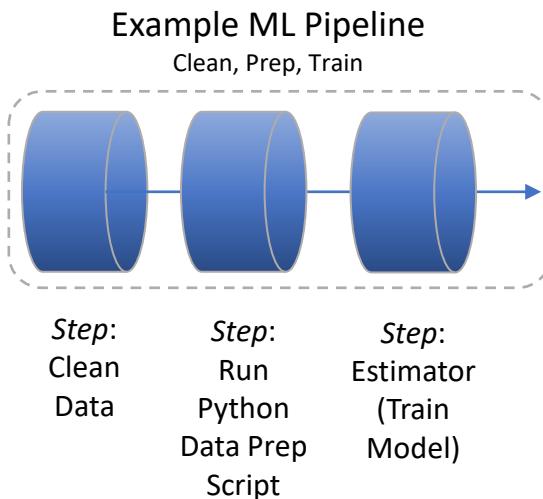
Materialized as Docker images in the workspace ACR

Cached and reused



# ML Pipelines

Define a repeatable, parameterized workflows for a machine learning task



- Each step can execute pre-built modules or custom scripts
- Each step of a pipeline can be configured to allow reuse of its previous run results

## Built-in PipelineStep Examples

Auto ML

Command (command line)

Data Transfer

Estimators (model training)

Hyperparameter Tuning

Python Script

R Script

Synapse Spark (Pyspark script)

...others

# Securing an Azure Machine Learning Workspace

Azure Data Conference

POWERED BY

Azure Data Community & Azure Data Conf

---

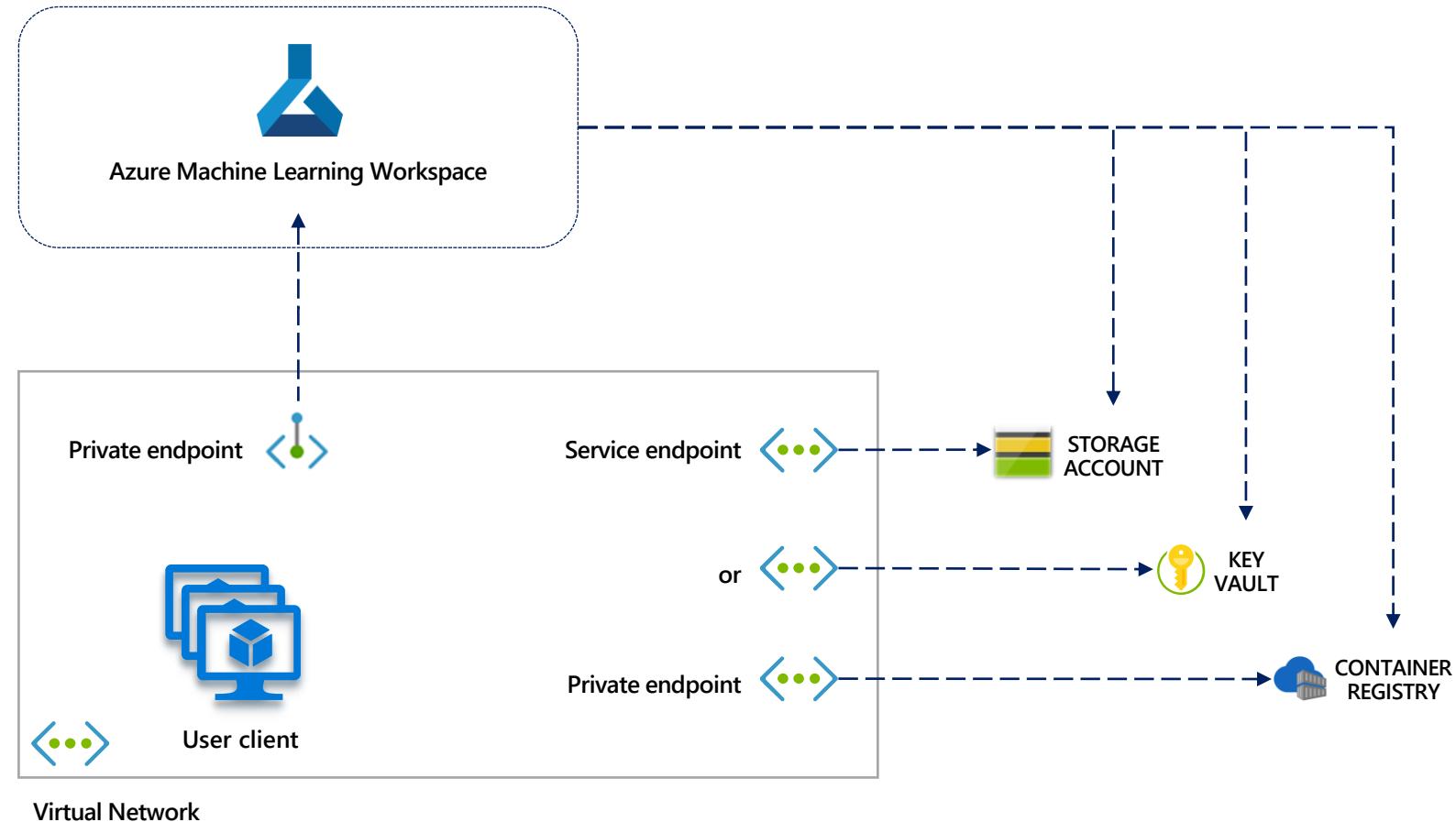
# Security checklist

Name	Description
Store training data	Ensure training data is stored with appropriate security.
Load training data	Ensure proper security is applied when connecting and reading from various data sources.
Prepare data	Ensure various intermediate forms and/or formats of data used in data preparation processes are stored securely (even if storage is temporary).
Manage datastores	In general, ensure access to various datastores is performed in a secure way. Avoid spreading datastore connecting credentials in source code, notebooks, config files, and the like. The recommended approach for Azure Machine Learning service is to use the register datastore capabilities for secure management of connections. Also, make sure you understand the data encryption features of Azure Machine Learning service.
Manage trained models	Ensure trained machine learning models are stored in a secure environment (e.g. the Azure Machine Learning service model registry with versioning). Ensure trained models are not saved (and left there) in various compute environments used for training workloads.

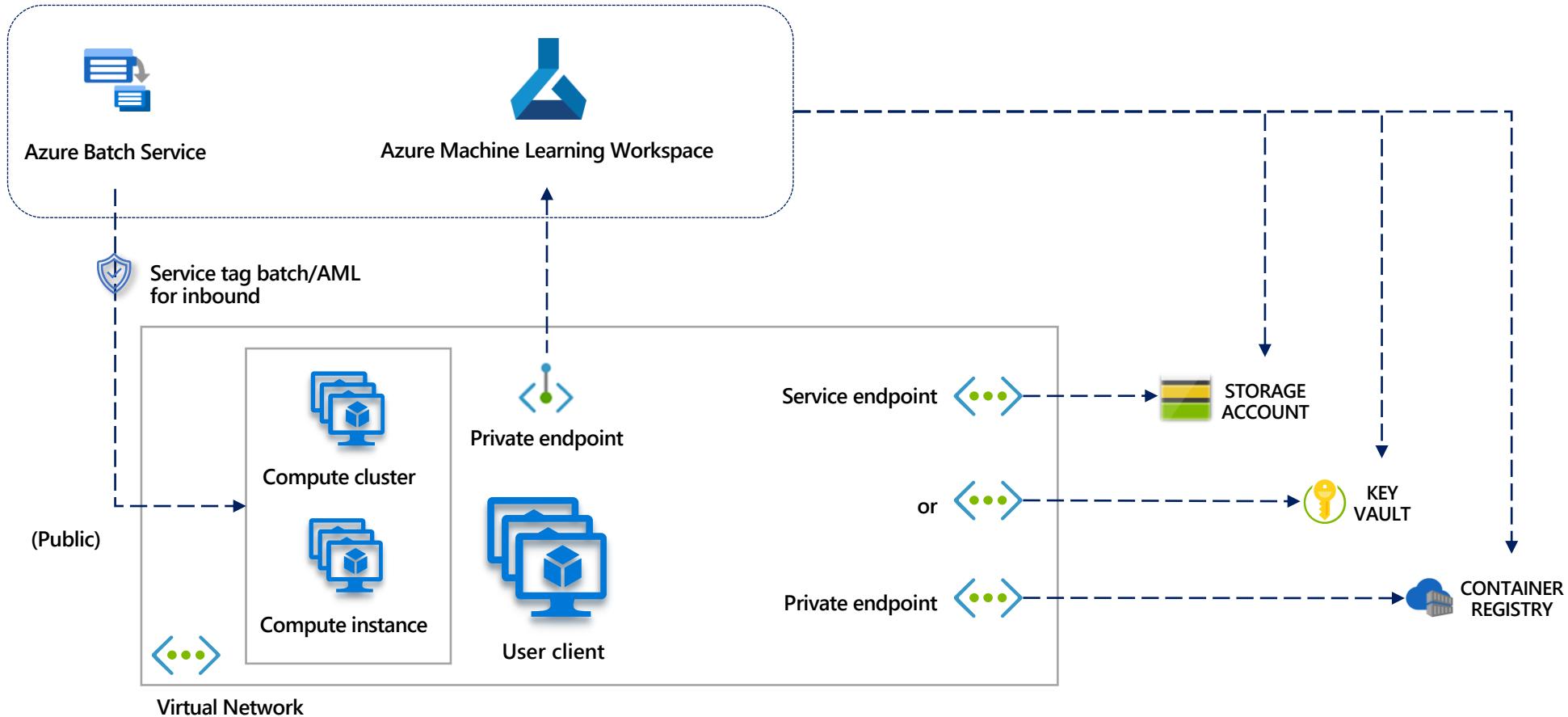
# Security checklist - continued

Name	Description
Deploy models for real-time scoring	Ensure models are deployed in secure environments. Ensure web service endpoints exposed for real-time scoring are properly secured and monitored for attempts to breach security (e.g. by using HTTPS, security keys vs. tokens, Azure AD integration etc...). Make sure you follow proper security guidelines for ACI or AKS.
Deploy models for batch scoring	Ensure models are deployed in secure environments. Ensure proper security of all actions capable of starting a batch processing job.
Manage secrets	Ensure secrets are not left in easily accessible places (source code files, source code repositories, notebooks, scripts etc...). Wherever possible, use approaches that do not require specifying credentials at run-time. Perform the task of regenerating storage account keys at regular intervals of time.
Manage workspaces	The Azure Machine Learning service is the main security boundary for controlling access to machine learning resources. Make sure you understand the security impact of an Azure machine learning workspace.
Manage networking	Secure AML training and scoring jobs using virtual networks.

# Network security – AML workspace



# Network security - training and job submission environment

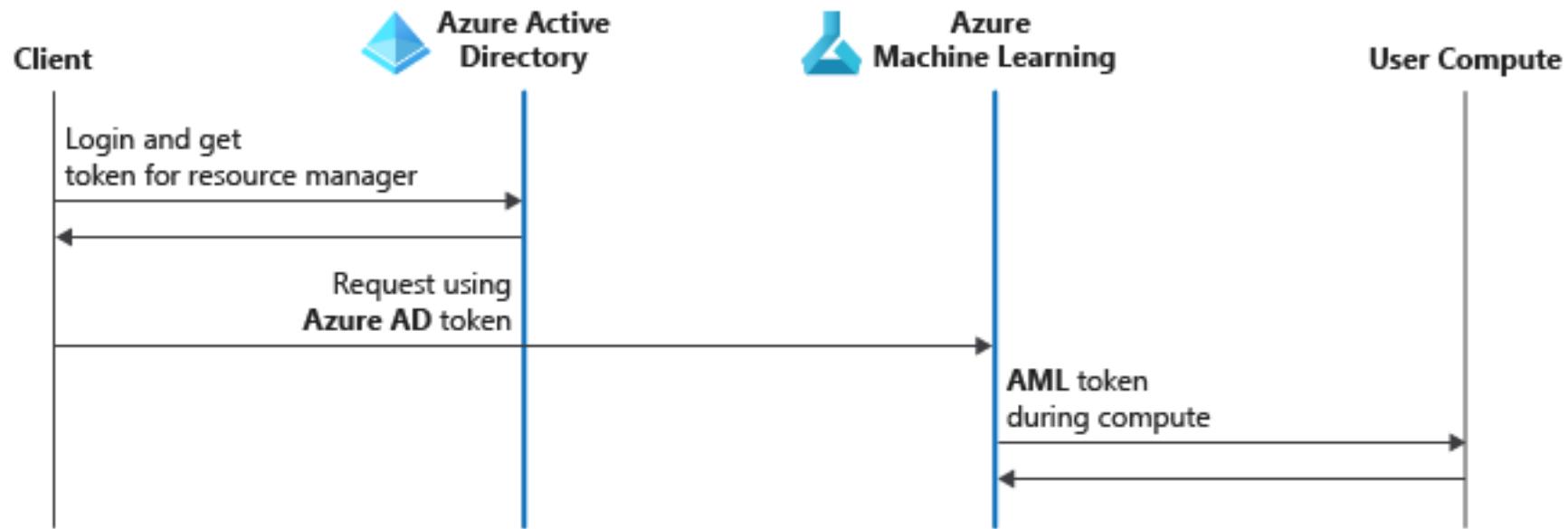


# Network security checklist

- ✓ Workspace secured
- ✓ Training environment secured
- ✓ Inferencing environment secured
- ✓ Studio functionality secured
- ✓ Firewall settings are in place
- ✓ DNS name resolution is in place



# Access control - authentication



# Access control – way to authenticate

- Interactive

```
from azureml.core.authentication import InteractiveLoginAuthentication  
interactive_auth = InteractiveLoginAuthentication(tenant_id="your-tenant-id")
```

- Service principal

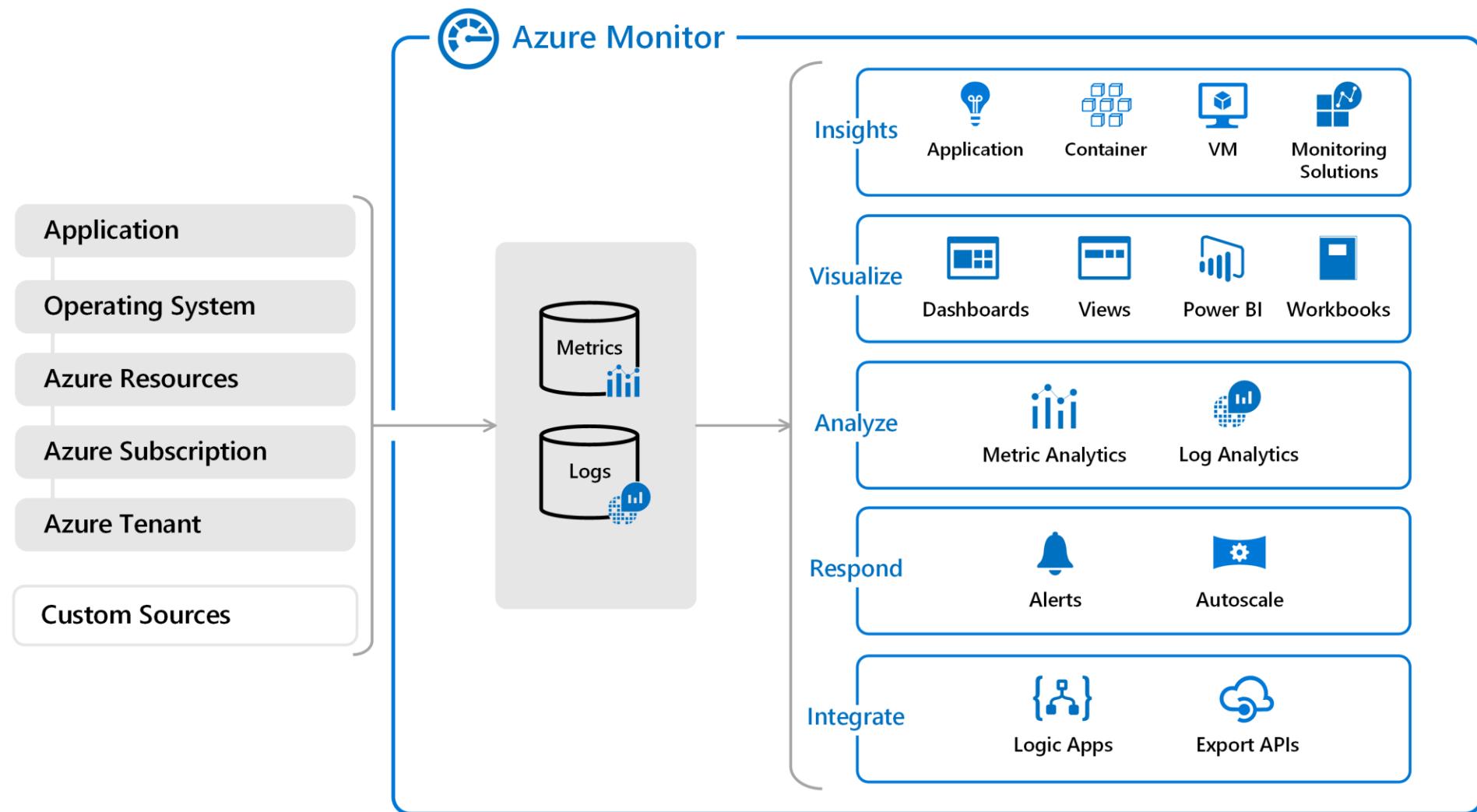
```
from azureml.core.authentication import ServicePrincipalAuthentication  
  
sp = ServicePrincipalAuthentication(tenant_id="your-tenant-id", # tenantID  
                                     service_principal_id="your-client-id", # clientId  
                                     service_principal_password="your-client-secret") # clientSecret
```

- Managed identity (via VM and SDK)

# Access control - roles

Role	Access level
Reader	Read-only actions in the workspace. Readers can list and view assets, including datastore credentials, in a workspace. Readers can't create or update these assets.
Contributor	View, create, edit, or delete (where applicable) assets in a workspace. For example, contributors can create an experiment, create or attach a compute cluster, submit a run, and deploy a web service.
Owner	Full access to the workspace, including the ability to view, create, edit, or delete (where applicable) assets in a workspace. Additionally, you can change role assignments.
Custom Role	Allows you to customize access to specific control or data plane operations within a workspace. For example, submitting a run, creating a compute, deploying a model or registering a dataset.

# Audit - Monitoring



# Security baseline for Azure Machine Learning

<https://docs.microsoft.com/en-us/azure/machine-learning/security-baseline>

## Azure security baseline for Azure Machine Learning

03/16/2021 • 37 minutes to read •  +12

This security baseline applies guidance from the [Azure Security Benchmark version 1.0](#) to Microsoft Azure Machine Learning. The Azure Security Benchmark provides recommendations on how you can secure your cloud solutions on Azure. The content is grouped by the **security controls** defined by the Azure Security Benchmark and the related guidance applicable to Azure Machine Learning. [Controls](#) not applicable to Azure Machine Learning have been excluded.

To see how Azure Machine Learning completely maps to the Azure Security Benchmark, see the [full Azure Machine Learning security baseline mapping file ↗](#).

### Network Security

*For more information, see the [Azure Security Benchmark: Network Security](#).*

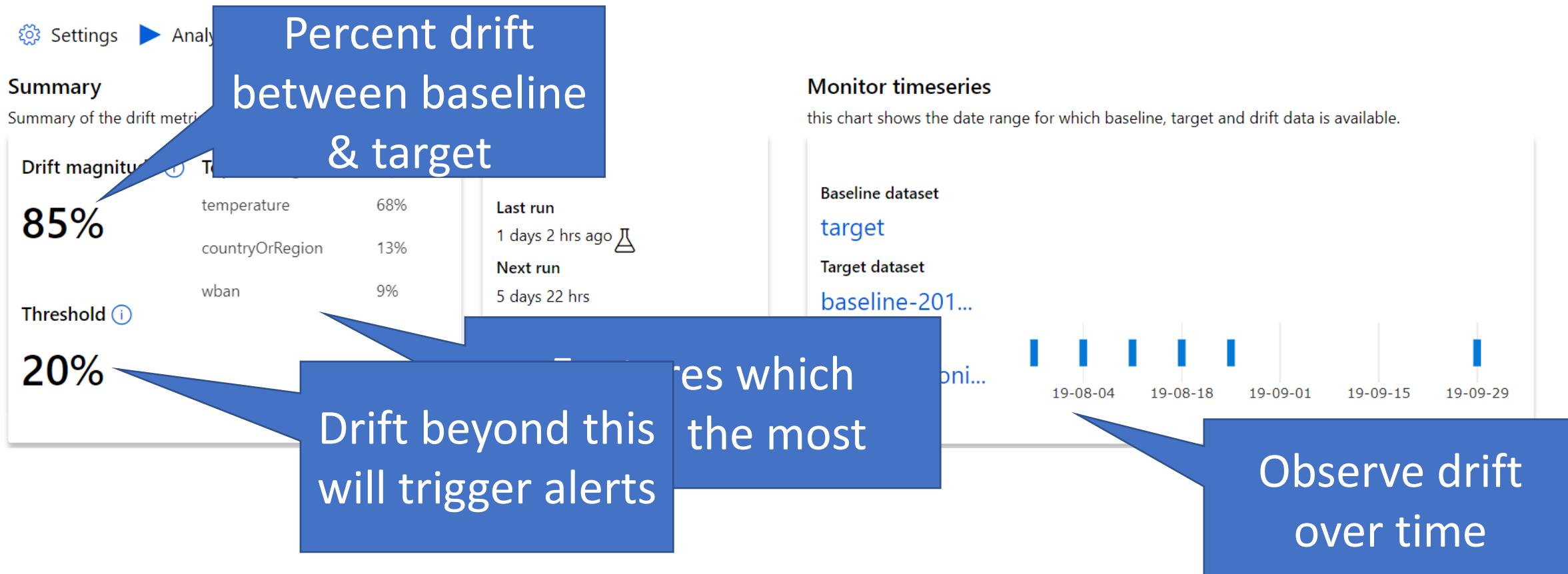
#### 1.1: Protect Azure resources within virtual networks

# Data Drift



# Detecting Data Drift with AML dataset monitors

## Review results in AML Studio



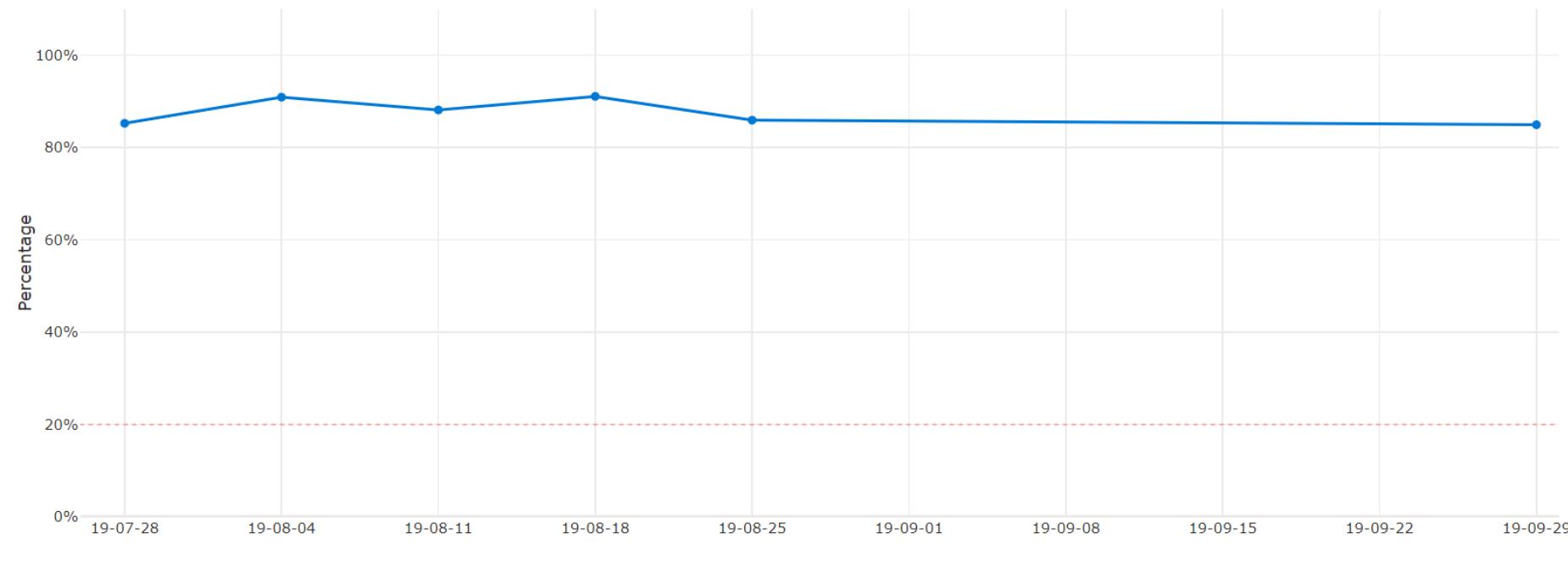
# Detecting Data Drift with AML dataset monitors

## Review results in AML Studio

Start date: 7/27/2019      End date: 9/29/2019

### Drift magnitude trend

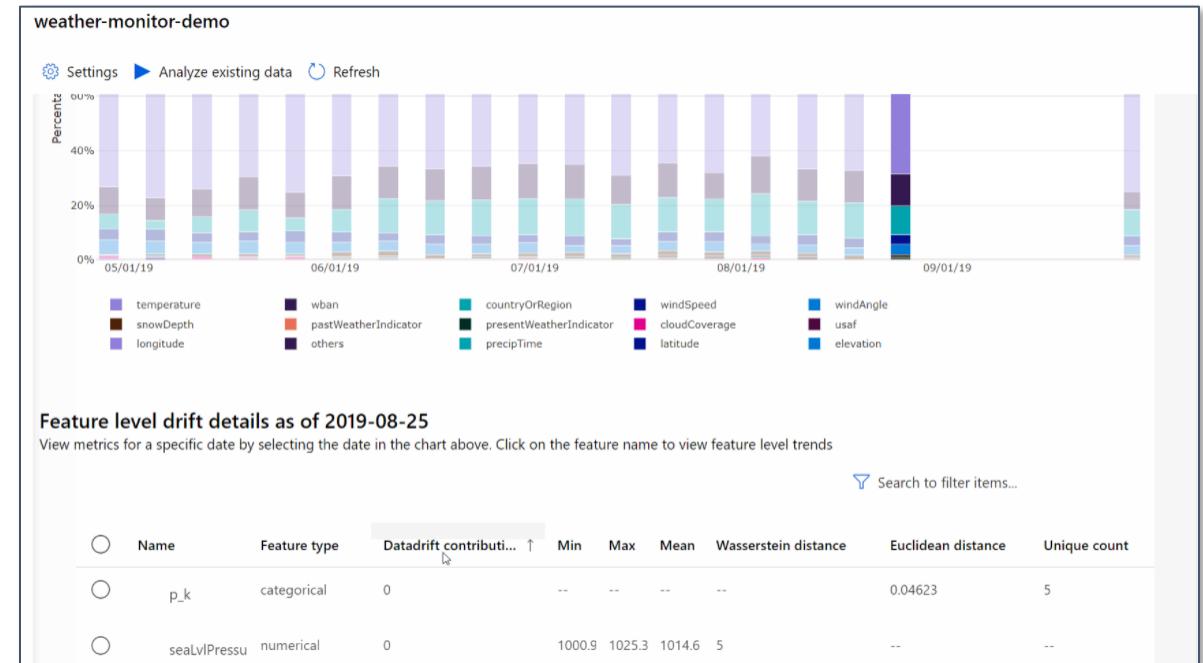
Drift magnitude metric measures the difference between target dataset for the time interval and baseline dataset. 0% implies that the target data is identical to the baseline data and 100% implies the target data is completely different from the baseline data.



# Detecting Data Drift with AML dataset monitors

## Review results by feature

Numeric features	Categorical features
Wasserstein distance	Euclidian distance
Mean value	Cardinality (unique values)
Min value	
Max value	



# Understanding Data Drift

Data drift happens over time.

Data drift affects models—they become less accurate as underlying circumstances change.

One solution: constant retraining. Downside: we're constantly retraining!

Another solution: detect drift and retrain when needed.

---

# Causes of Data Drift

- Upstream process changes (e.g., placing sensors in new locations)
  - Data quality issues (e.g., broken sensor)
  - Natural drift in data (e.g., seasonality, change in tastes)
  - Change in relationship between features (i.e., covariate shift)
-

# Detecting Data Drift with AML dataset monitors

Azure Machine Learning dataset monitors help track drift.

- Create using AML Studio or the AML SDK
  - Supports both SQL and flat file data stores
  - Defines a baseline dataset that represents the original training data
  - Defines a target dataset that represents the scoring data
  - Requires timeseries dataset with timestamp column
-

# Detecting Data Drift with AML dataset monitors

**Step 1:** Create a target dataset. **Must** have the timeseries

- Timestamp column in the data
  - Virtual column derived from the path pattern of files

```
dstore = Datastore.get(ws, 'your datastore name')
```

```
dstore_paths = [(dstore, 'weather/*/*/*/*/data.parquet')]
```

```
partition format = 'weather/{state}/{date}::yyyy/mm/dd'
```

```
dset = Dataset.Tabular.from_parquet_files(path)
```

`partition_format=partition_format)`

```
dset = dset.with_timestamp_columns('date').Specify
```

```
dset = dset.register(ws, 'target')
```

# Define a variable

Define a variable  
named “date” based  
on folder names

# Load data using the defined partition

Specify  
a timestamp column

# Detecting Data Drift with AML dataset monitors

## Step 2: Create a dataset monitor

```
dset = Dataset.get_by_name(ws, 'target')
baseline = target.time_before(datetime(2019, 2, 1))
features = ['latitude', 'longitude', ... ]
monitor = DataDriftDetector.create_from_datasets(ws,
    'drift-monitor', baseline, target,
    compute_target='cpu-cluster',
    frequency='Week',
    feature_list=None,
    drift_threshold=.6,
    latency=24)
```

Set a baseline of  
expected activity

Create a data drift  
detector object

Define frequency of  
scheduled tasks

acceptable

# Detecting Data Drift with AML dataset monitors

Step 3: Run a backfill

```
monitor = DataDriftDetector.get_by_name(ws, 'drift-monitor')  
monitor = monitor.update(feature_list=features)  
backfill1 = monitor.backfill(datetime(2019, 1, 1),  
                           datetime(2019, 5, 1))  
backfill1 = monitor.backfill(datetime(2019, 5, 1),  
                           datetimedelta=datetime.now())  
monitor = monitor.disable_schedule()  
monitor = monitor.enable_schedule()
```

Backfill from  
January to May,  
2019

Backfill from May,  
now

Stop performing  
drift detection

Start per  
drift detection

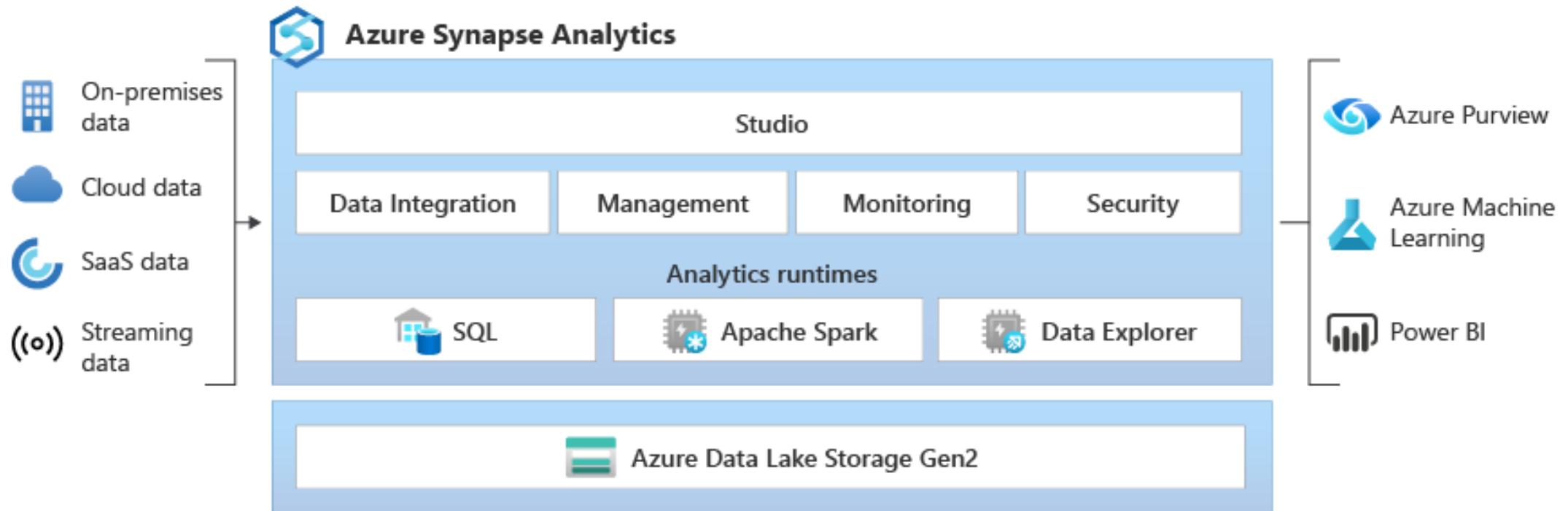
# Azure Machine Learning integration with Azure Synapse Analytics

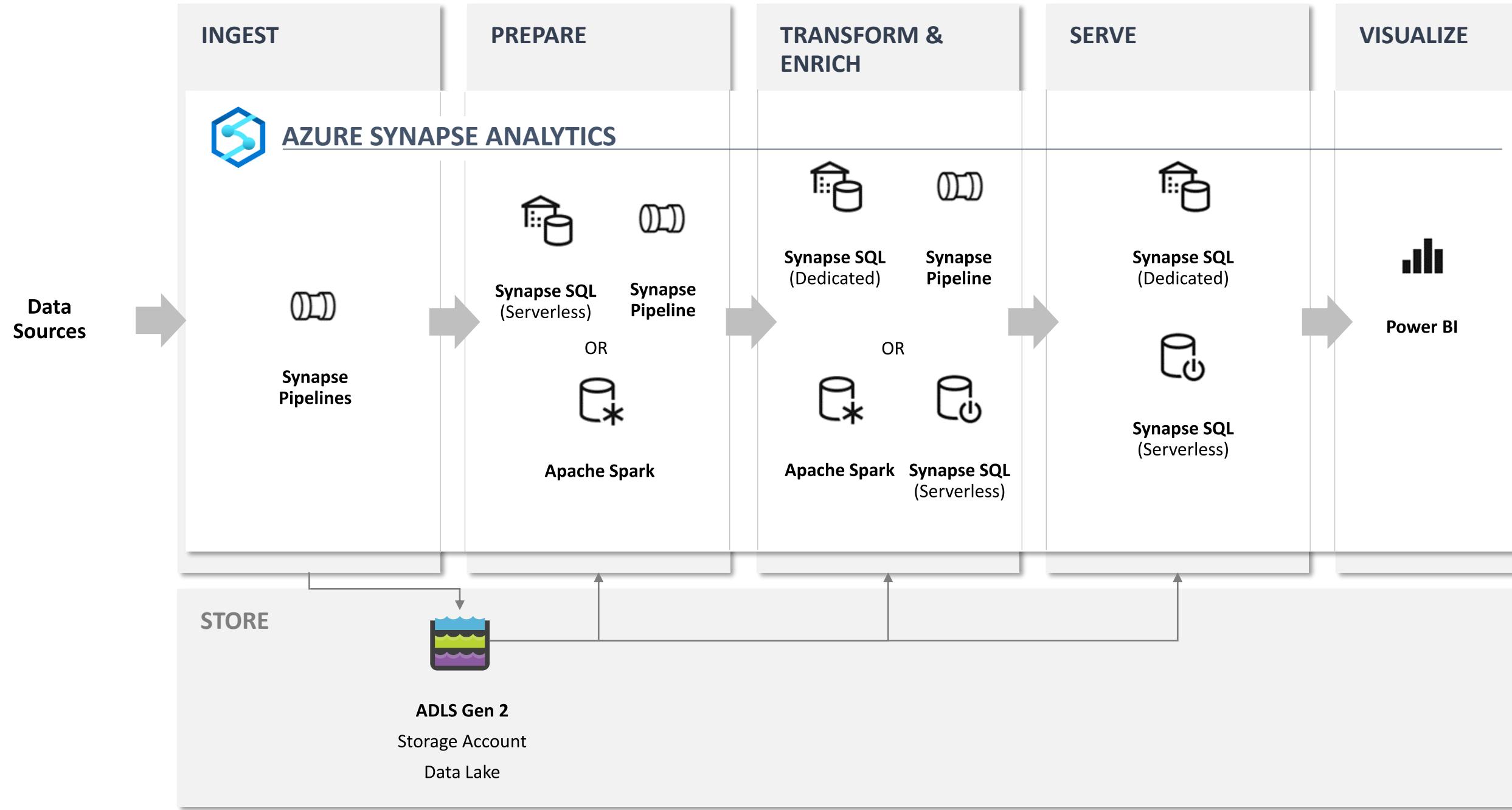
Azure Data Conference  
POWERED BY  
[Azure Data Community & Azure Data Conf](#)

---

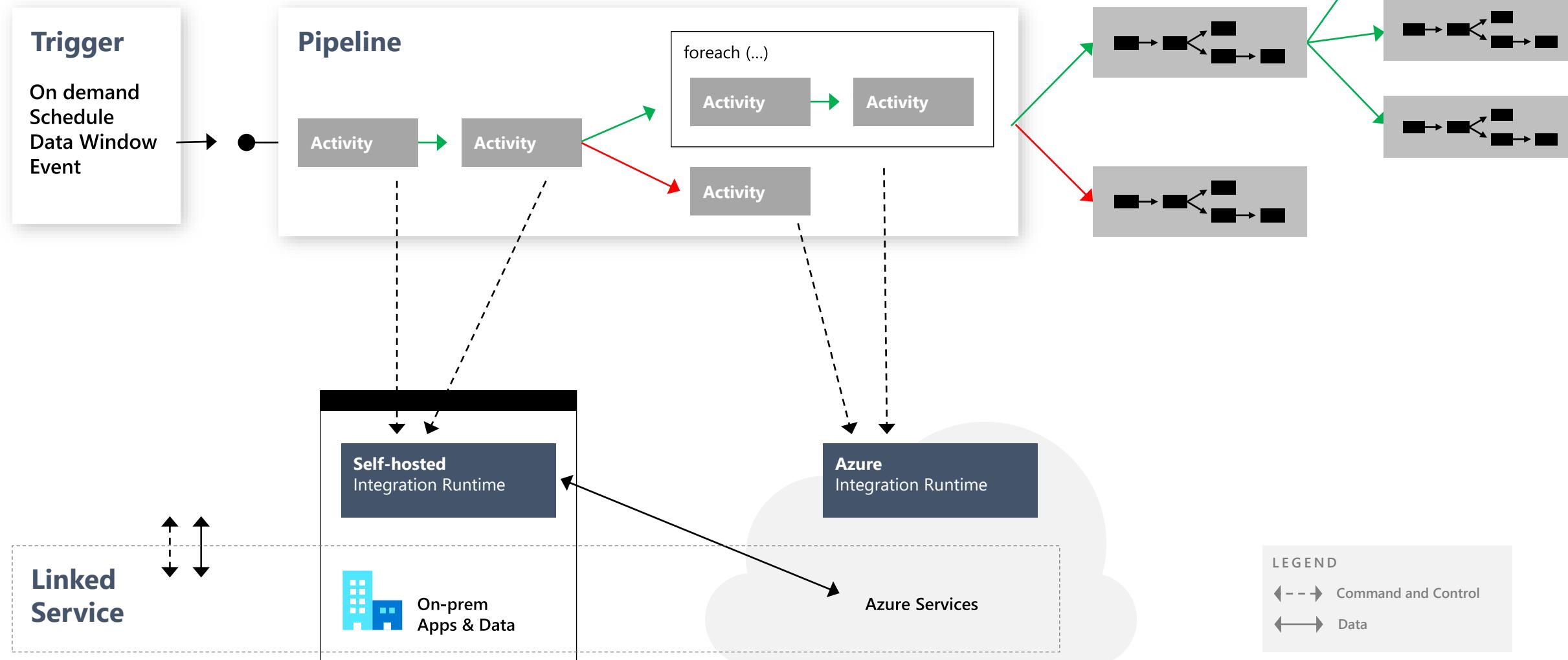
# Azure Synapse Analytics

Limitless analytics service with unmatched time to insight





# Synapse Pipelines

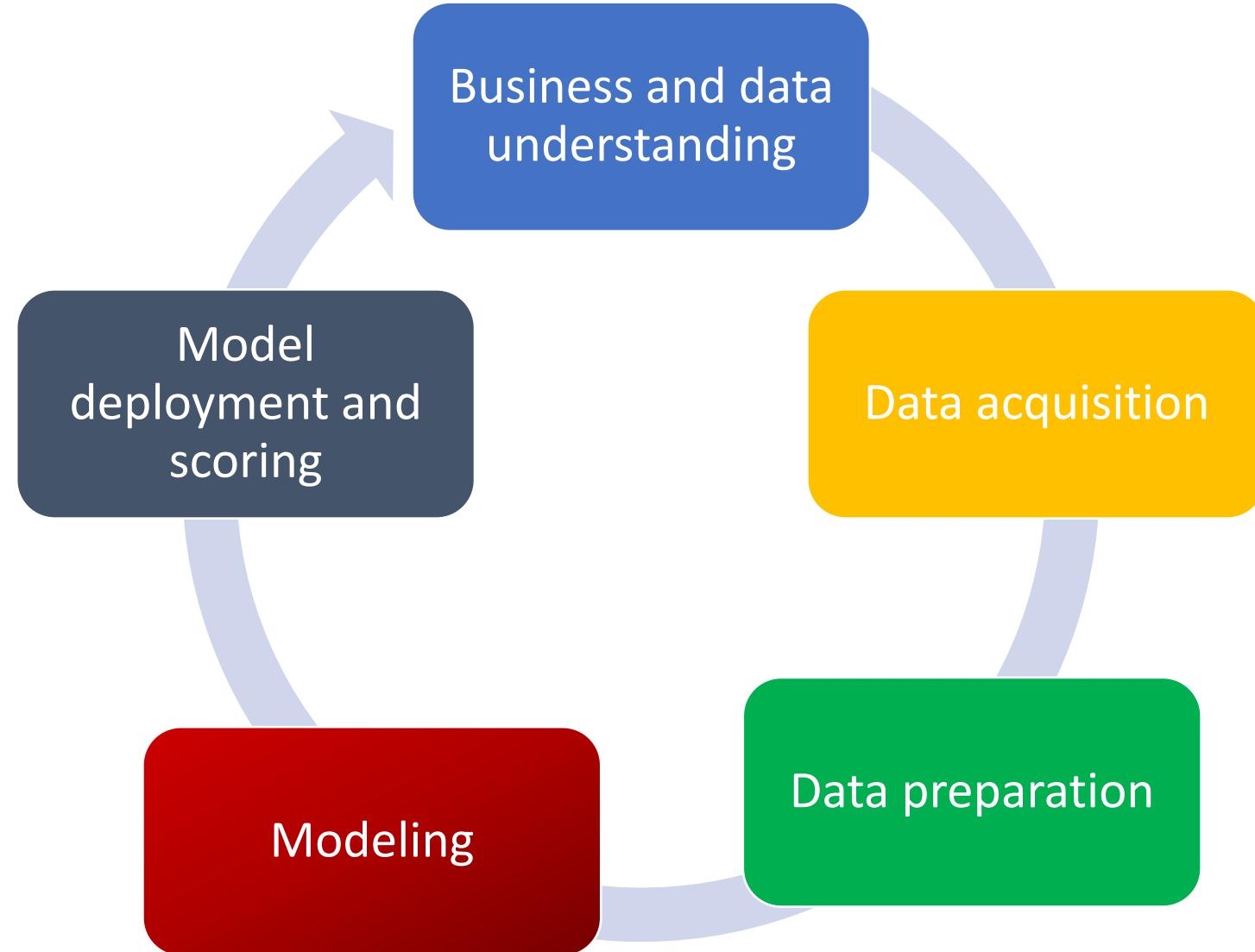


Synapse Pipelines shares codebase with Azure Data Factory

# Understanding Synapse Pipelines and ML Pipelines

	Synapse Pipeline	ML Pipeline
Used for:	Data movement and transformation	Model training
Used by:	Data engineers	Data scientists

# High level machine learning process



# Modeling with Spark ML Algorithms

## Spark ML Algorithms

Classification and Regression	<ul style="list-style-type: none"><li>• Linear Models (SVMs, logistic regression, linear regression)</li><li>• Naïve Bayes</li><li>• Decision Trees</li><li>• Ensembles of trees (Random Forest, Gradient-Boosted Trees)</li><li>• Isotonic regression</li></ul>
Clustering	<ul style="list-style-type: none"><li>• k-means and streaming k-means</li><li>• Gaussian mixture</li><li>• Power iteration clustering (PIC)</li><li>• Latent Dirichlet allocation (LDA)</li></ul>
Collaborative Filtering	<ul style="list-style-type: none"><li>• Alternating least squares (ALS)</li></ul>
Dimensionality Reduction	<ul style="list-style-type: none"><li>• SVD</li><li>• PCA</li></ul>
Frequent Pattern Mining	<ul style="list-style-type: none"><li>• FP-growth</li><li>• Association rules</li></ul>
Basic Statistics	<ul style="list-style-type: none"><li>• Summary statistics</li><li>• Correlations</li><li>• Stratified sampling</li><li>• Hypothesis testing</li><li>• Random data generation</li></ul>

# Microsoft Machine Learning for Apache Spark

v1.0-rc

Microsoft's Open Source  
Contributions to Apache Spark



Distributed  
Machine Learning



Fast Model  
Deployment



Microservice  
Orchestration

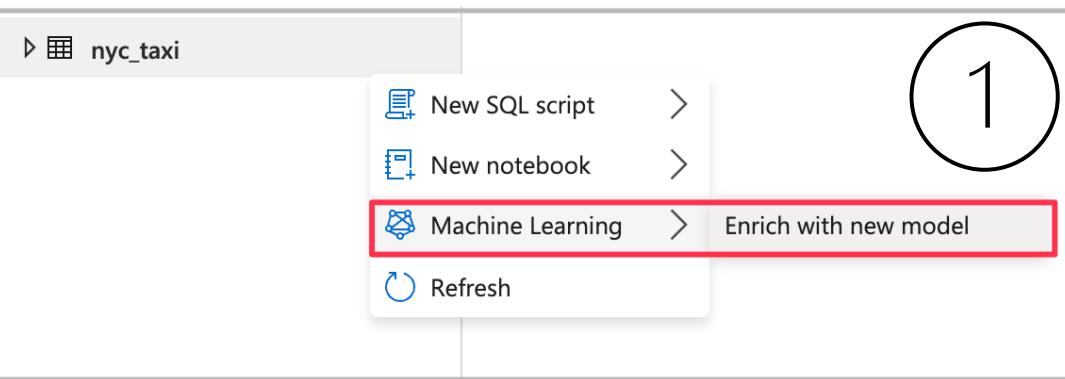


Multilingual Binding  
Generation

[www.aka.ms/spark](http://www.aka.ms/spark)

 [Azure/mmlspark](https://github.com/Azure/mmlspark)

# Modeling with Azure Machine Learning – Code Free



This is the second step of the 'Enrich with new model' configuration wizard. It shows the 'Configure experiment' section. The 'Source data' is set to 'nyc\_taxi'. Under 'Azure Machine Learning workspace', a dropdown shows 'AML\_ws'. The 'Experiment name' field contains 'nyc\_taxi-20201116055046'. The 'Best model name' field contains 'nyc\_taxi-20201116055046-Best'. The 'Target column' is set to 'fareAmount (double)'. The 'Spark pool' is set to 'Spark1110'.

This is the third step of the 'Enrich with new model' configuration wizard, titled 'Choose a model type'. It provides instructions for selecting a machine learning model type based on the question being answered. It lists three options: 'Classification' (with a bar chart icon), 'Regression' (with a stack of bars icon), and 'Time series forecasting' (with a line graph icon). The 'Regression' option is highlighted with a red box and has a circular number '3' to its right. Below each option is a brief description and an example.

This is the fourth step of the 'Enrich with new model' configuration wizard, titled 'Configure regression model'. It includes fields for 'Primary metric' (set to 'Spearman correlation'), 'Training job time (hours)' (set to '3'), 'Max concurrent iterations' (set to '2'), and 'ONNX model compatibility' (set to 'Disable'). At the bottom are buttons for 'Create run', 'Open in notebook', 'Back', and 'Cancel'.

# Modeling with Azure Machine Learning –

Notebook 3

+ Cell ▾ Run all Undo Publish Attach to Spark1110 Language PySpark (Python)

```
1 df = spark.sql("SELECT * FROM default.nyc_taxi")
2
3 datastore = Datastore.get_default(ws)
4 dataset = register_spark_dataframe(df, datastore, name = experiment_name + "-dataset")
5 dataset_train, dataset_test = dataset.random_split(percentage = 0.8)
```

Cell 3

```
[ ] 1 automl_config = AutoMLConfig(spark_context = sc,
2                                task = "regression",
3                                training_data = dataset_train,
4                                label_column_name = "fareAmount",
5                                primary_metric = "spearman_correlation",
6                                experiment_timeout_hours = 3,
7                                max_concurrent_iterations = 2,
8                                enable_onnx_compatible_models = False)
```

Cell 4

```
[ ] 1 run = experiment.submit(automl_config)
```

Search resources

all

Notebook 3 Notebook 4

+ Cell ▾ Cancel all Undo Publish Attach to sparkPoolMedium Language PySpark (Python) Azure Notebooks (Preview)

```
1 experiment_timeout_hours = 3,
2 max_concurrent_iterations = 2,
3 enable_onnx_compatible_models = True
```

Command executed in 3mins 11s 578ms by chaxu on 10-27-2020 15:33:48.557 +08:00

Cell 6

```
[6] 1 run = experiment.submit(automl_config)
```

Command executed in 3mins 48s 748ms by chaxu on 10-27-2020 15:34:25.758 +08:00

Running on remote or ADB.

Cell 7

```
[7] 1 displayHTML("<a href={} target='_blank'>Your experiment in Azure Machine Learning portal: {}</a>".format(run.get_portal_url(), run.id))
```

Command executed in 3mins 50s 564ms by chaxu on 10-27-2020 15:34:27.599 +08:00

Your experiment in Azure Machine Learning portal: [AutoML\\_70940235-79fd-4d5d-a3ce-b05cf4e88d80](https://mlflow.org/experiments/70940235-79fd-4d5d-a3ce-b05cf4e88d80)

Cell 8

```
[8] 1 run.wait_for_completion()
```

Running

Cell 9

```
[9] 1 # # If you want to register the best model, please uncomment this code snippet and change model_name
2 # import onnxruntime
3 # import mlflow
4 # import mlflow.onnx
5
6 # from mlflow.models.signature import ModelSignature
7 # from mlflow.types import DataType
8 # from mlflow.types.schema import ColSpec, Schema
9
10 # # Get best model from automl run
```

# Model deployment and scoring – Code Free

Microsoft Azure | Synapse Analytics > wsazuresynapseanalytics

Data

Workspace Linked

Filter resources by name

Databases 8

NYCTaxi.Pool (SQL)

- Tables
  - dbo.aml\_models
  - dbo.modeldeploy
  - dbo.Models
  - dbo.nyc\_taxi
    - Columns
    - dbo.nyctaxi
    - dbo.testload1029
    - dbo.Trips
    - External tables
    - External resources
    - Views

New SQL script >

New notebook >

New data flow

New integration dataset

Machine Learning > **Enrich with existing model**

Refresh

1

### Enrich with existing model

dbo.nyc\_taxi

Select the model you want to use to enrich the selected dataset. [Learn more](#)

#### Azure Machine Learning

Machine Learning workspace \*

Name	Version	Created	Created By	Framework
nyc_taxi_tip_predict	1	11:44:52 07/18/2020	[REDACTED]	Custom

2

### Enrich with existing model

dbo.nyc\_taxi

Map the source table columns to the expected model inputs. [Learn more](#)

#### Input mapping \*

+ New | Delete

Source Column	Model Input	Input Type
fareAmount	fareAmount	real
paymentType	paymentType	bigint
passengerCount	passengerCount	bigint
tripDistance	tripDistance	real
tripTimeSecs	tripTimeSecs	bigint
pickupTimeBin	pickupTimeBin	varchar

#### Output mapping \*

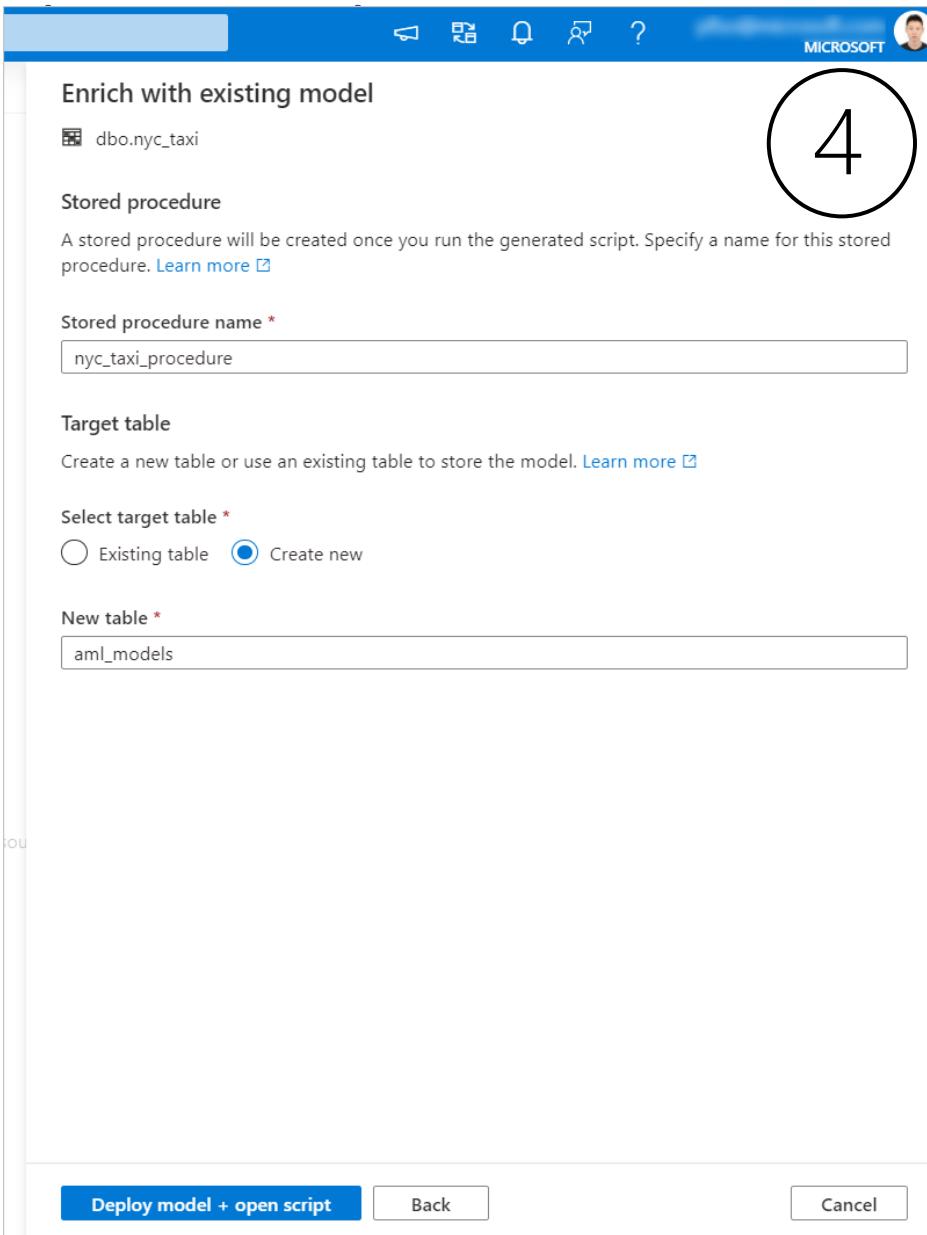
+ New | Delete

Model Output	Output Type
output_label	bigint

Continue Back Cancel

3

# Model deployment and scoring – Code Free



Enrich with existing model

dbo.nyc\_taxi

Stored procedure

A stored procedure will be created once you run the generated script. Specify a name for this stored procedure. [Learn more](#)

Stored procedure name \*

nyc\_taxi\_procedure

Target table

Create a new table or use an existing table to store the model. [Learn more](#)

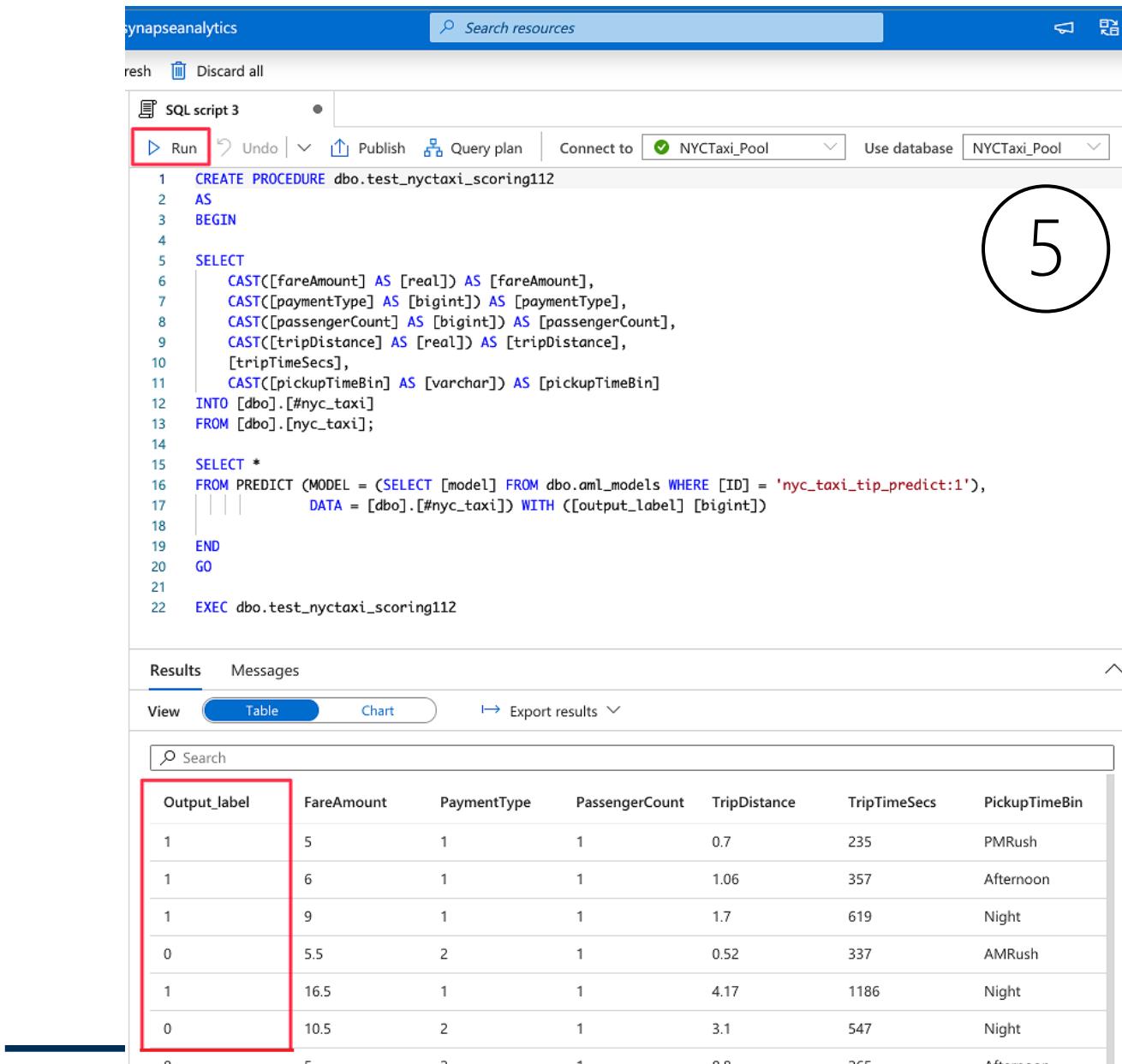
Select target table \*

Existing table  Create new

New table \*

aml\_models

Deploy model + open script Back Cancel



synapseanalytics Search resources

SQL script 3

Run Undo Publish Query plan Connect to NYCTaxi\_Pool Use database NYCTaxi\_Pool

```
1 CREATE PROCEDURE dbo.test_nyctaxi_scoring112
2 AS
3 BEGIN
4
5 SELECT
6     CAST([fareAmount] AS [real]) AS [fareAmount],
7     CAST([paymentType] AS [bigint]) AS [paymentType],
8     CAST([passengerCount] AS [bigint]) AS [passengerCount],
9     CAST([tripDistance] AS [real]) AS [tripDistance],
10    [tripTimeSecs],
11    CAST([pickupTimeBin] AS [varchar]) AS [pickupTimeBin]
12 INTO [dbo].[#nyc_taxi]
13 FROM [dbo].[nyc_taxi];
14
15 SELECT *
16 FROM PREDICT (MODEL = (SELECT [model] FROM dbo.aml_models WHERE [ID] = 'nyc_taxi_tip_predict:1'),
17 | | | DATA = [dbo].[#nyc_taxi]) WITH ([output_label] [bigint])
18
19 END
20 GO
21
22 EXEC dbo.test_nyctaxi_scoring112
```

Results Messages

View Table Chart Export results

Output_label	FareAmount	PaymentType	PassengerCount	TripDistance	TripTimeSecs	PickupTimeBin
1	5	1	1	0.7	235	PMRush
1	6	1	1	1.06	357	Afternoon
1	9	1	1	1.7	619	Night
0	5.5	2	1	0.52	337	AMRush
1	16.5	1	1	4.17	1186	Night
0	10.5	2	1	3.1	547	Night
0	5	2	1	0.8	265	Afternoon

# Predict

## Overview

It provides ability to import existing machine learning models and score them within provisioned SQL. It takes ONNX (Open Neural Network Exchange) and data as inputs and generates prediction based on model.

## Benefits

1. It empowers data engineers to successfully deploy machine learning models with the familiar T-SQL interface
2. It offers seamless collaboration with data scientists
3. It generates new columns, but the number of columns and their data types depends on the type of model that was used for prediction.

### Syntax:

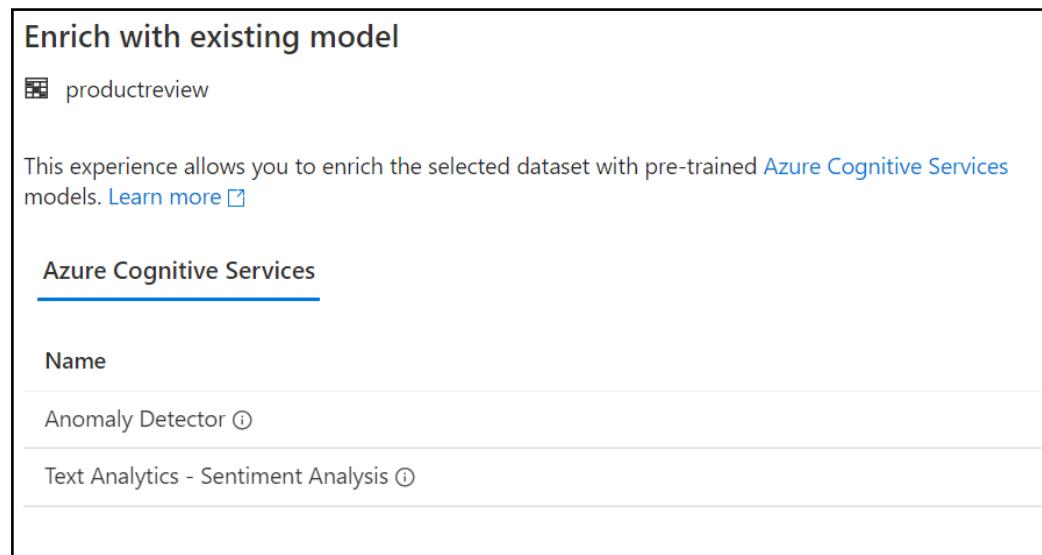
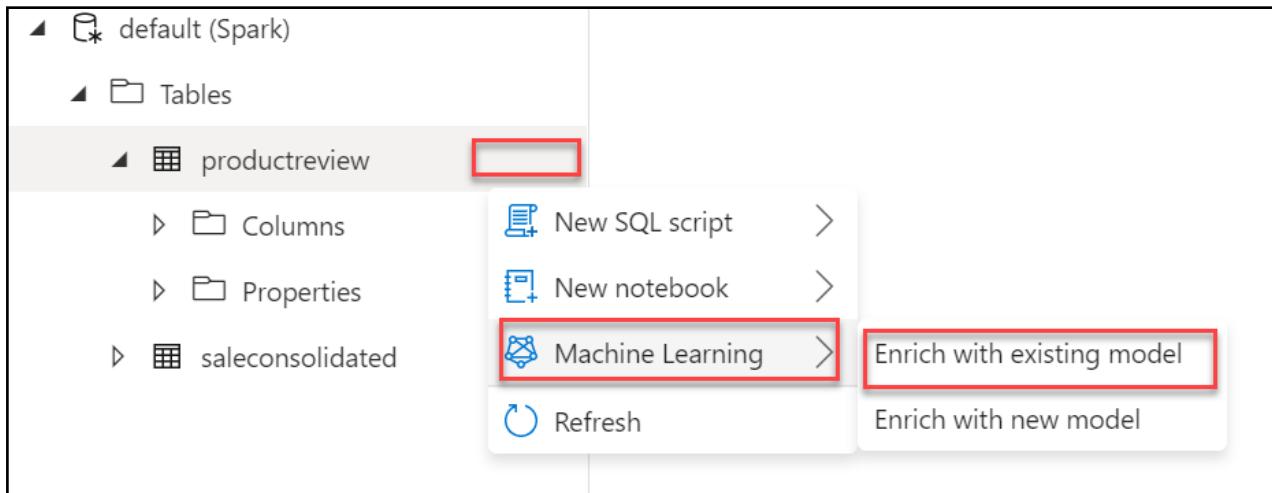
```
PREDICT
(
    MODEL = @model | model_literal,
    DATA = object AS <table_alias>
)
WITH ( <result_set_definition> )
<result_set_definition> ::= 
{
    { column_name
        data_type
    }
    [,...n]
}
MODEL = @model | model_literal
```

### Example:

```
DECLARE @model varbinary(max) = (SELECT Model FROM Models WHERE Id = <>);
SELECT d.*, p.Score
FROM PREDICT(MODEL = @model,
    DATA = dbo.mytable AS d) WITH (Score float) AS p;
```

# Leveraging Cognitive services

- Enrich with existing model
- Select from the Cognitive Services available
- Choose the appropriate Cognitive Service resource
- Cog. Svc. API Key must be stored in a Key Vault
- Set input column
- Generates a PySpark Notebook that applies the cognitive service model to the column chosen



# MLOps

Azure Data Conference

POWERED BY

Azure Data Community & Azure Data Conf

---

# Problem statement

- Simplify support for multiple environments
  - Reduce copy and paste code
  - Standardize model validation
  - Tracking and versioning (model, data)
  - Eliminate manual deployments
  - Eliminate model development in a “vacuum”
  - Need better documentation for Transparency
  - Active monitoring (model drift, data drift)
- 
-



**ginablaber**

@ginablaber

Follow



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

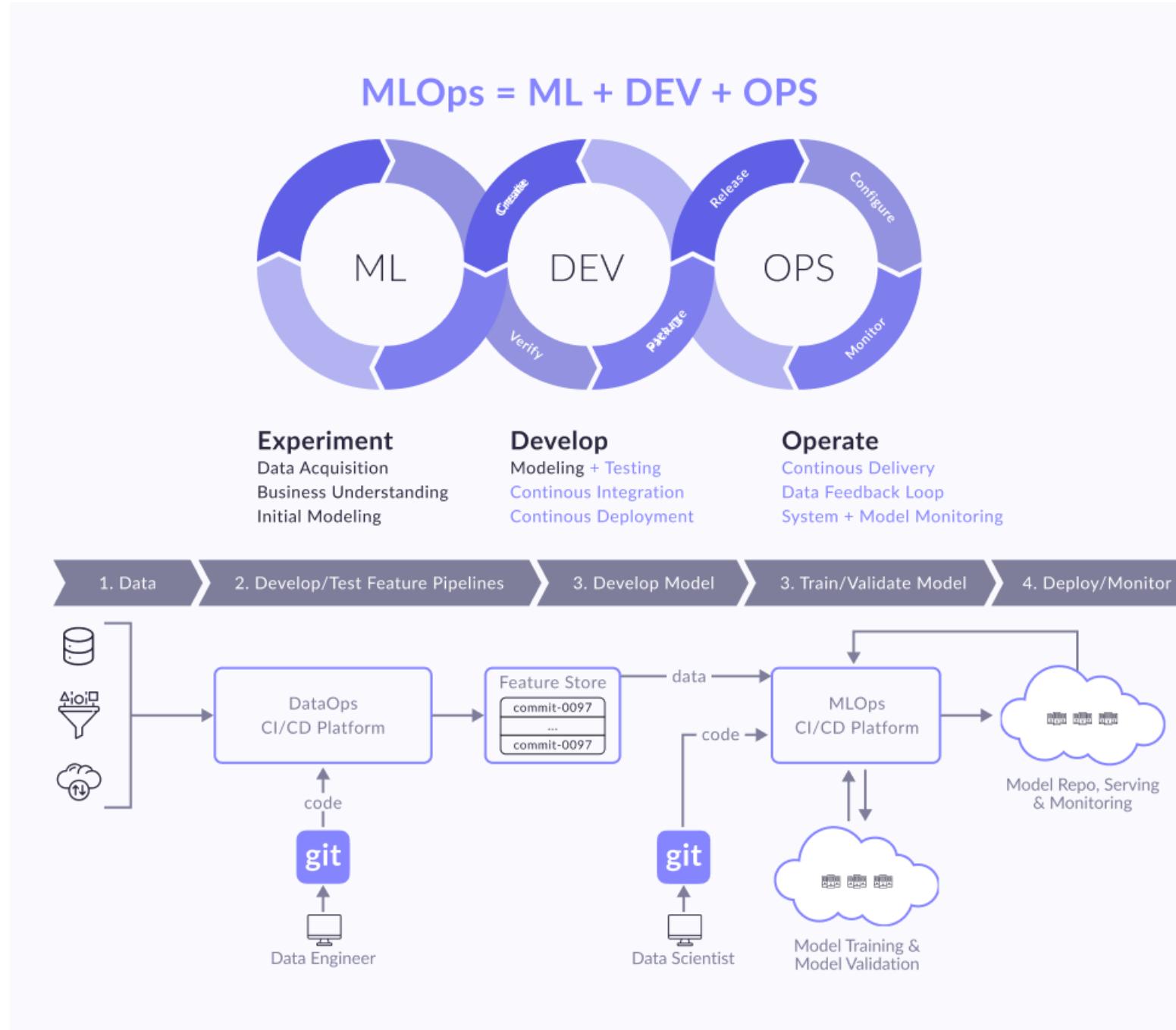




A Dall-E 2 rendering of **a data scientist in a wizard costume training a machine learning model in front of a computer.**

# Definition

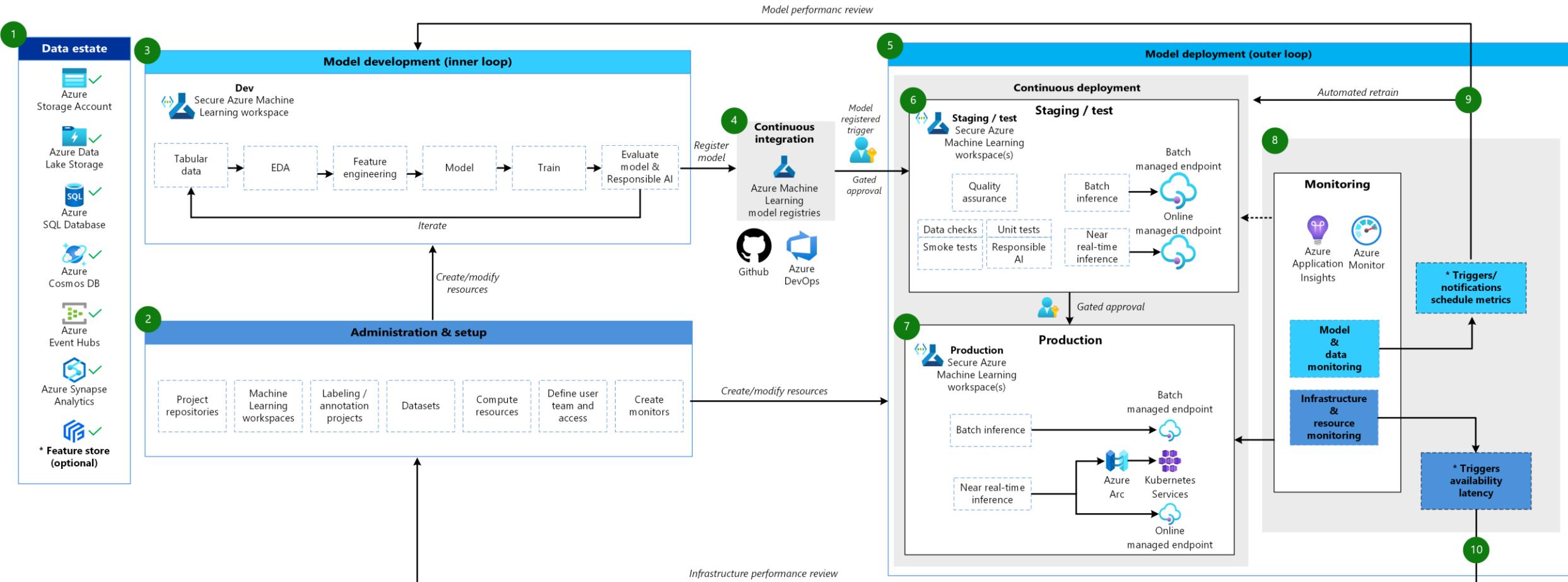
“MLOps is an approach to managing the entire machine learning lifecycle, from the development of a model to deployment and ongoing monitoring.”



# What makes for good MLOps ?

- Create reproducible machine learning pipelines
  - Create reusable software environments
  - Capture the governance data for the end-to-end machine learning lifecycle
  - Notify and alert on events in the machine learning lifecycle
  - Monitor machine learning applications for operational and machine learning-related issues
  - Automate end-to-end machine learning lifecycle
- 
-

# MLOps in Azure: Classical machine learning architecture



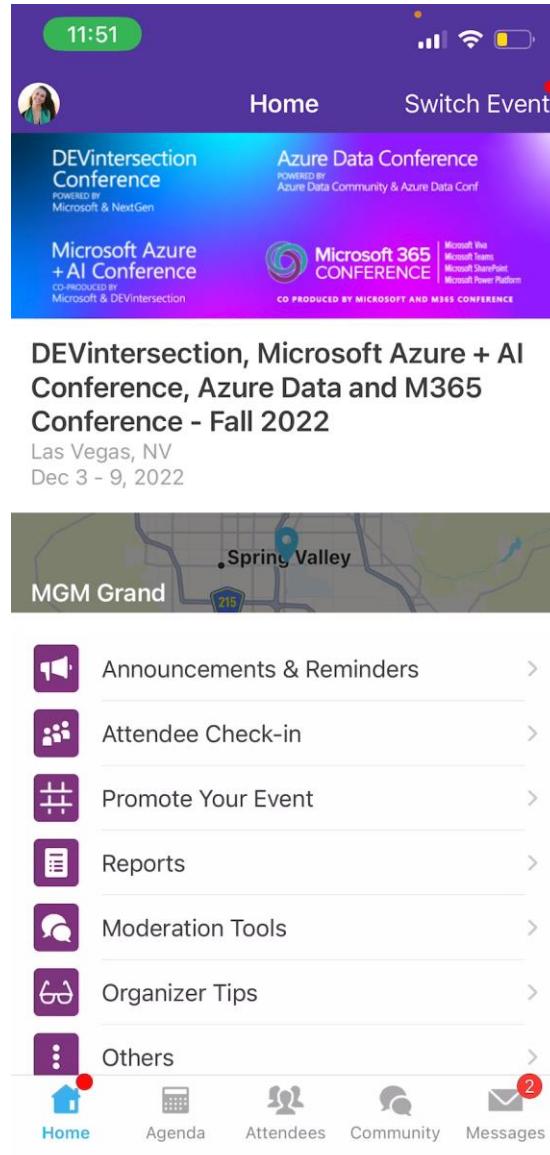
Legend: Data engineer Data scientist Machine learning engineer Infrastructure team

\* Future accelerator implementation

✓ Recommended practice

# Demo





# Session Feedback Surveys

In the pursuit of making our conferences even better, we need to hear your feedback about this session.

## Here's How -

- Simply go to the Whova App on your smartphone
- Go to the conference homepage
- Scroll down to 'Additional Resources' and click 'Surveys'.
- Click 'Session Feedback'.
- Scroll down to click on this session title.
- Complete the session feedback survey.
- Finally, click 'Submit'

**DEC 5-7  
2023**

**WORKSHOPS  
DEC 3, 4 & 8**

## DEVintersection Conference

[DEVintersection.com](https://DEVintersection.com)

## Microsoft Azure + AI Conference

[AzureAIConf.com](https://AzureAIConf.com)

## Azure Data Conference

[AzureDataConf.com](https://AzureDataConf.com)

**DISNEY  
WORLD**

**SWAN &  
DOLPHIN  
RESORT**

AI • Angular • Architecture • ASP.NET • Azure • Azure Data • Azure Databricks • Azure Functions • Azure IoT • Azure SQL • Azure Synapse C# • Blazor • Cloud Security • Codespaces • Cognitive Services • CosmosDB • Dapr • Data Science • DevOps • Docker

• GitHub Actions • Machine Learning • Metaverse • Microservices • MySQL • .NET • .NET MAUI • Node.js • PostgreSQL • Power Apps • Power BI • React • Scalable Architectures • Security & Compliance • SQL Server 2022 • TypeScript • Virtual Machines • Visual Studio



**Register Early**  
for Optional Workshops



**and Receive Your Option of Hardware**