Real World Machine Learning at Enterprise Scale

Azure Data Conference

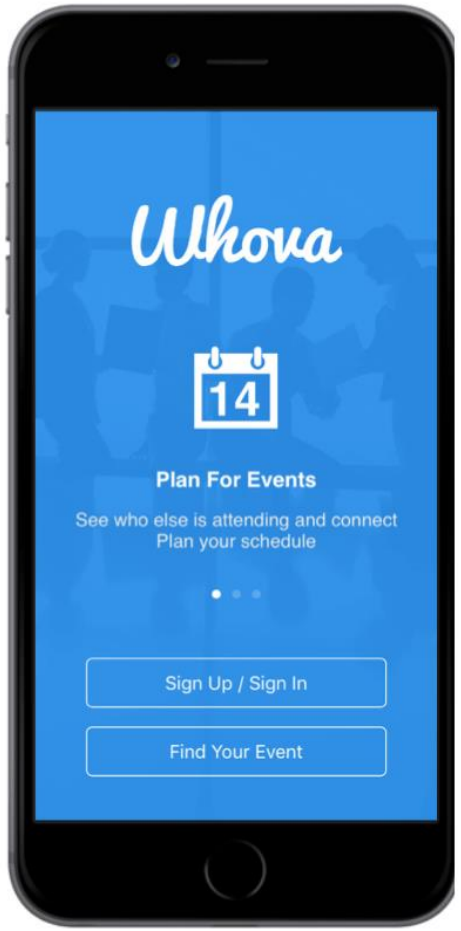
POWERED BY

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

Available on the App Store | ANDROID APP ON Google play

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

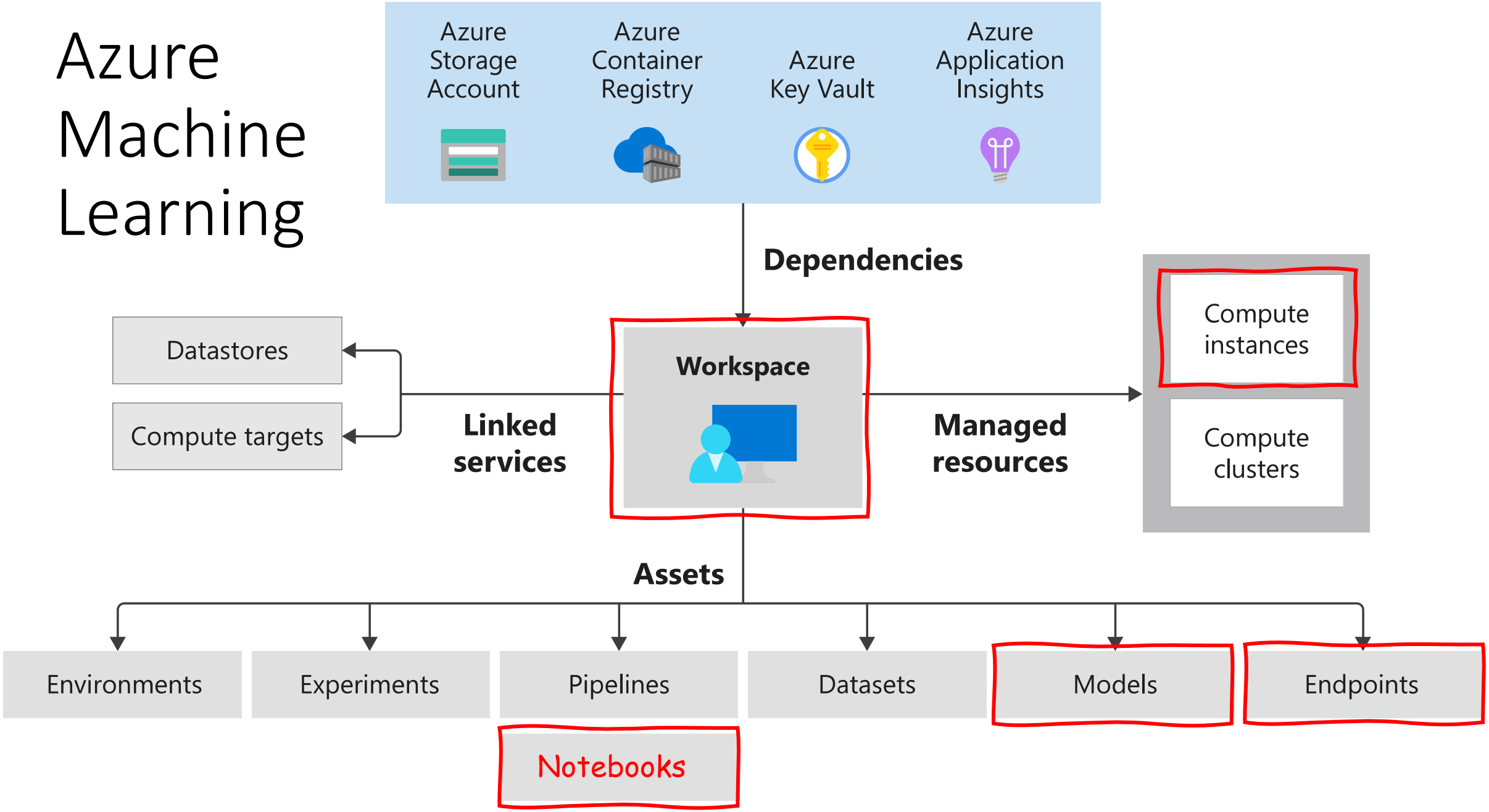

CPU


GPU

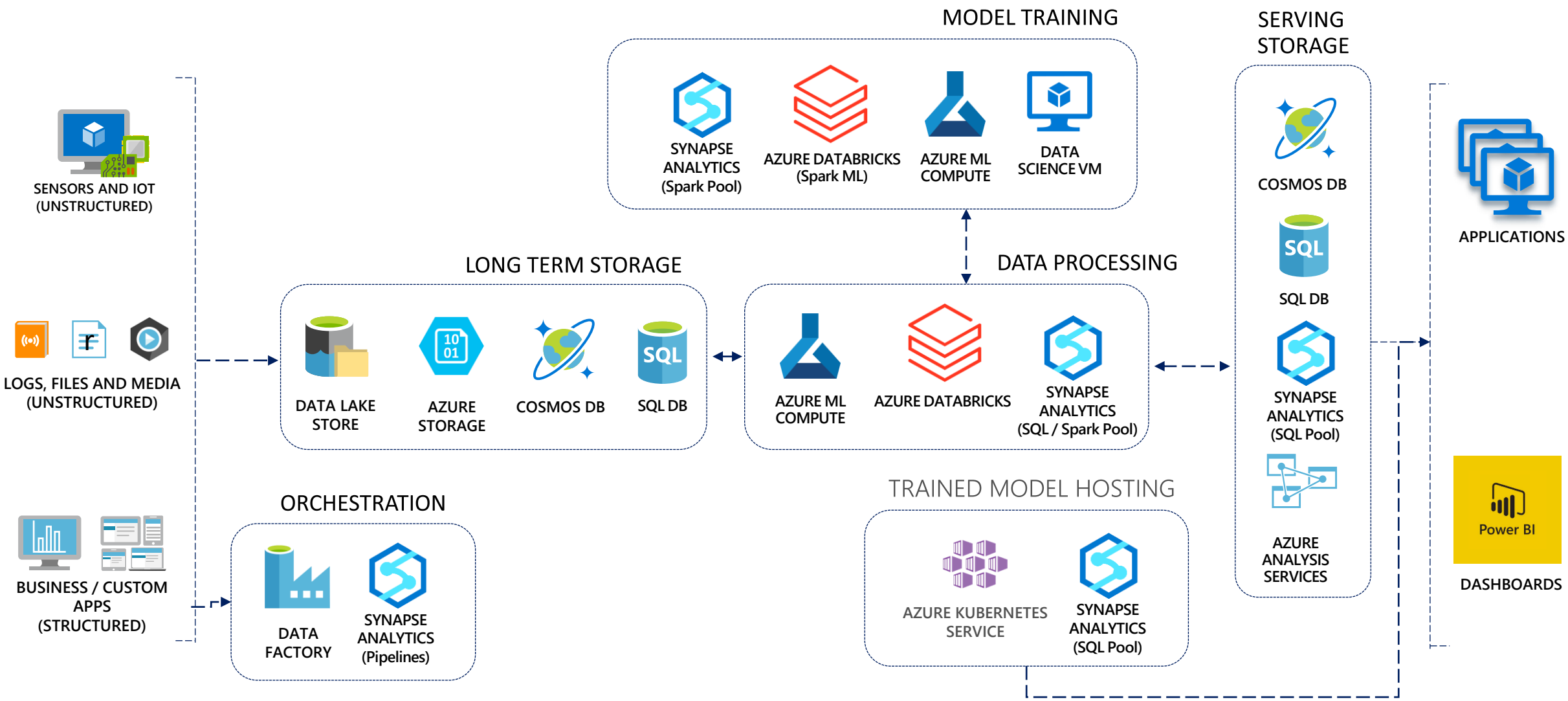

FPGA

From the Intelligent Cloud to the Intelligent Edge

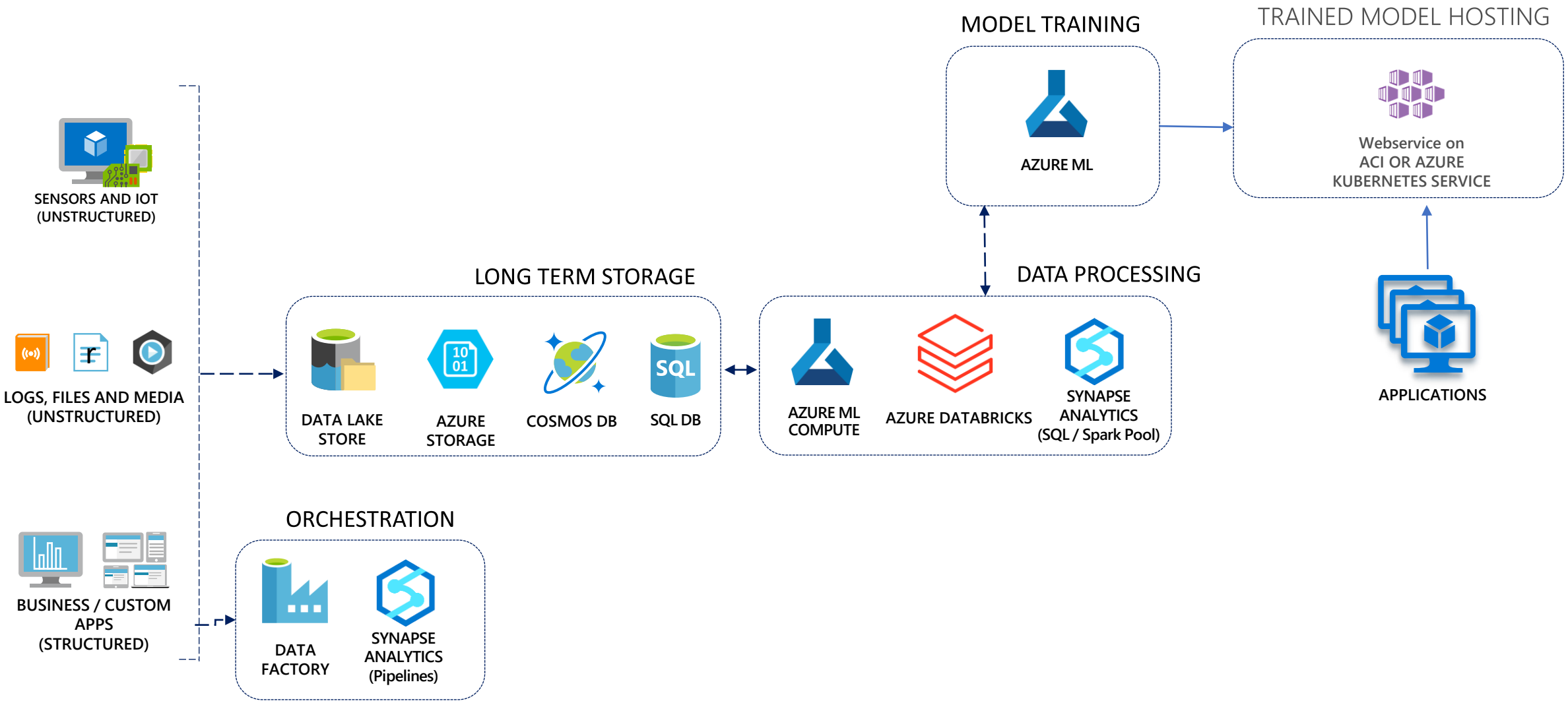
Azure Machine Learning



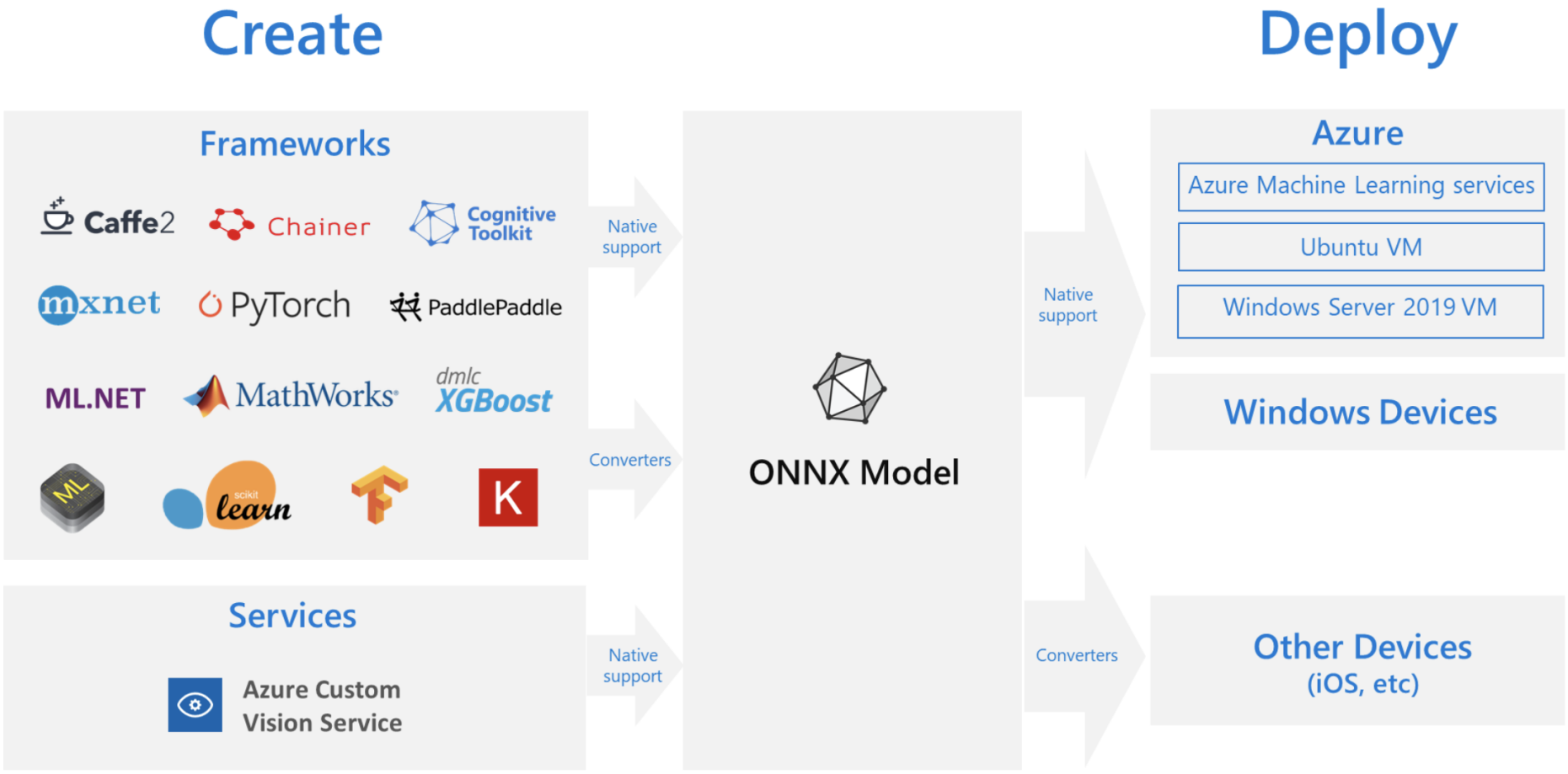
Advanced analytics pattern in Azure



Model deployment patterns in Azure



What is ONNX?



ONNX flow diagram showing training, converters, and deployment

Tools for Machine Learning

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

Azure Machine Learning CLI v2

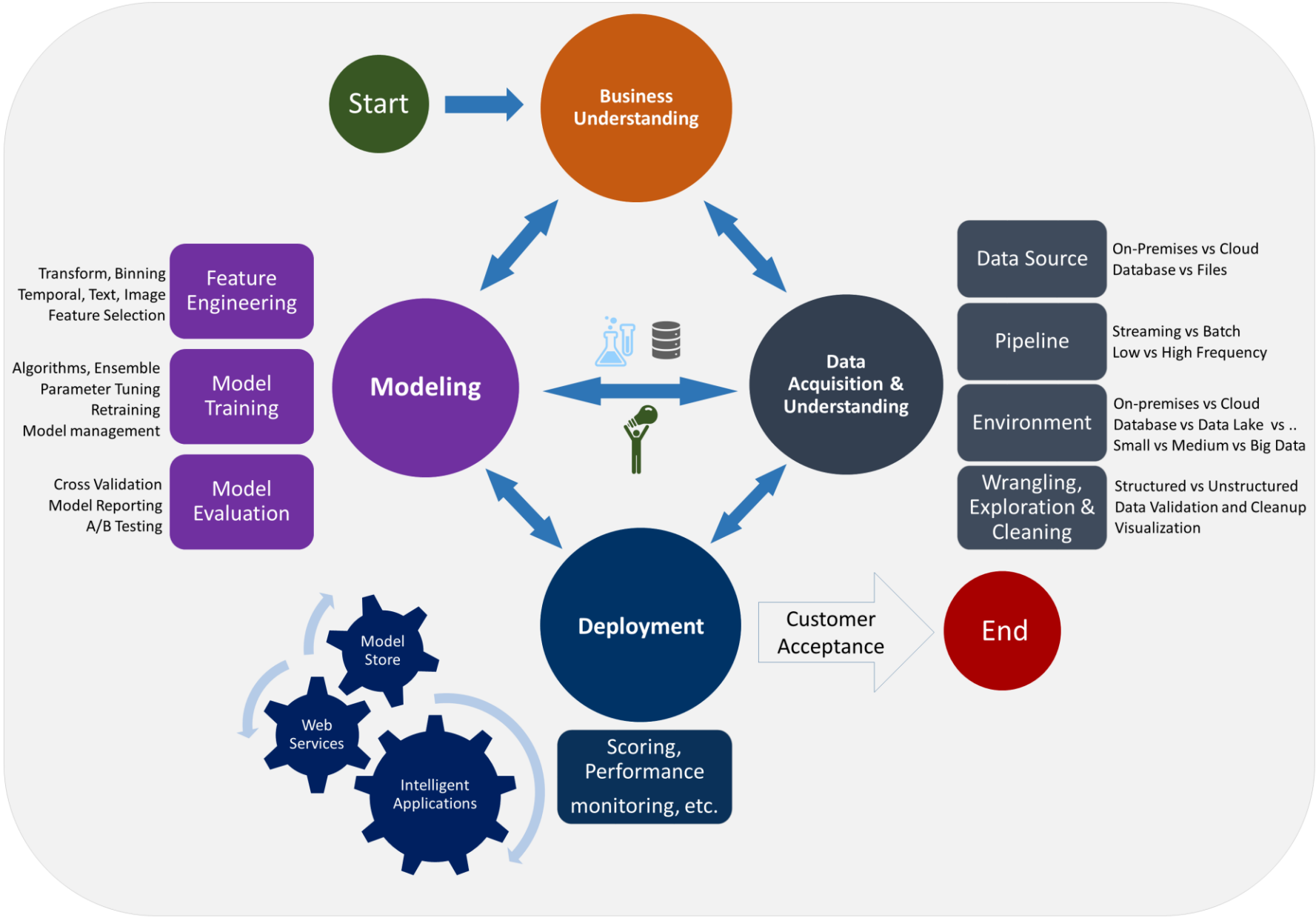
- The CLI v2 is useful in the following scenarios:
  - On board to Azure ML without the need to learn a **specific programming language**
  - Ease of deployment and automation
  - Azure ML offers endpoints to streamline model deployments for both real-time and batch inference deployments. This functionality 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

Data Science Lifecycle



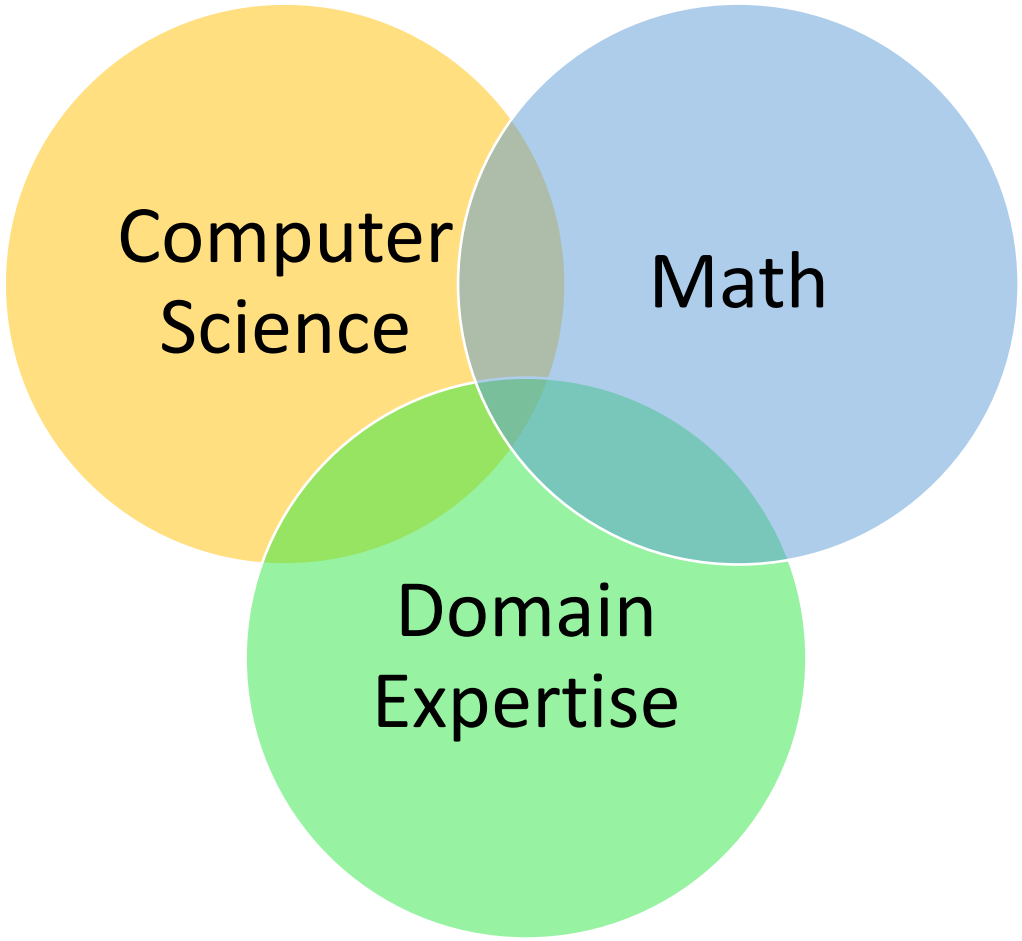
As a developer is deep learning for you?

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.

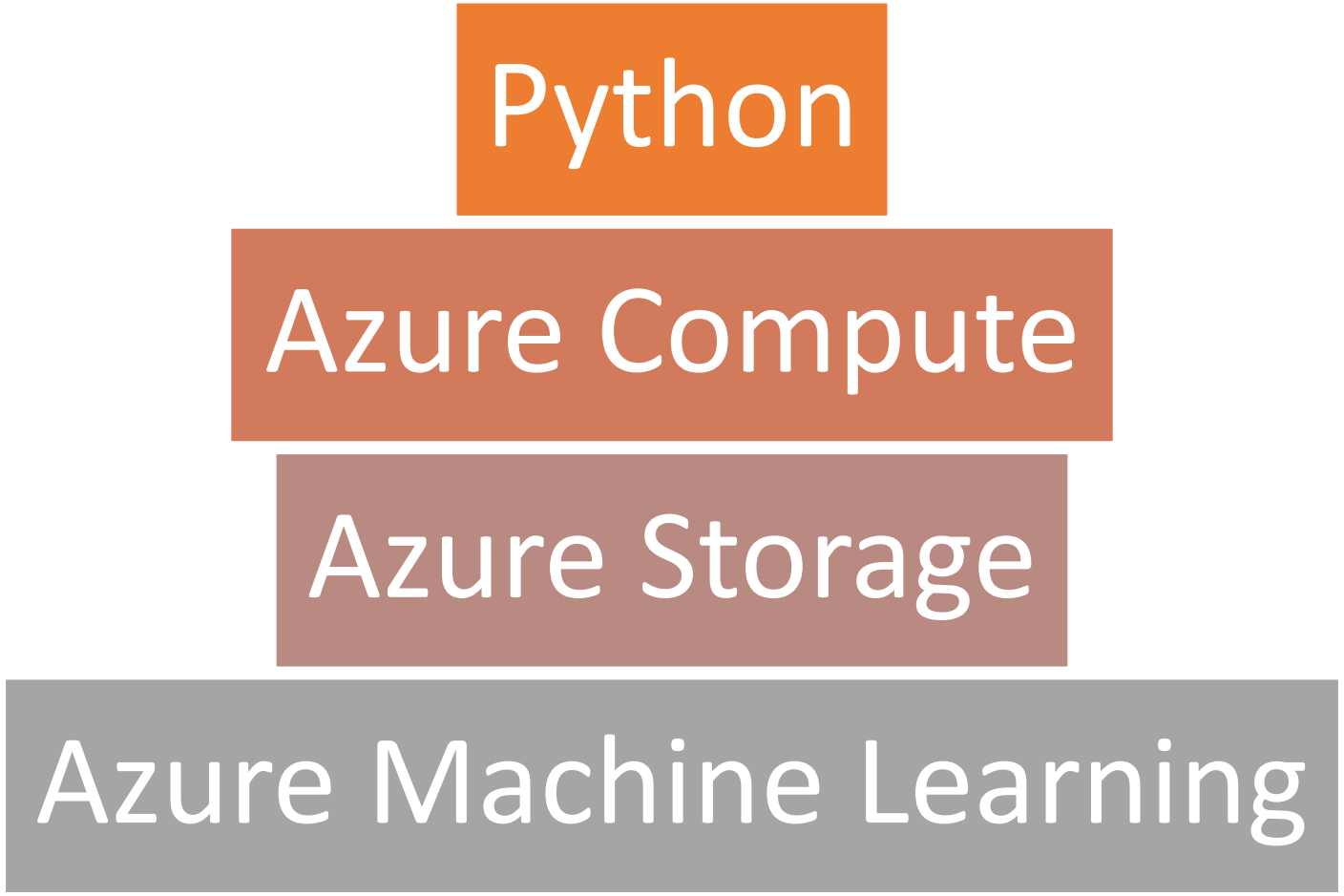
It's a hands-on discipline in which ideas are proven empirically more often than theoretically.

- Francois Chollet, Creator of Keras

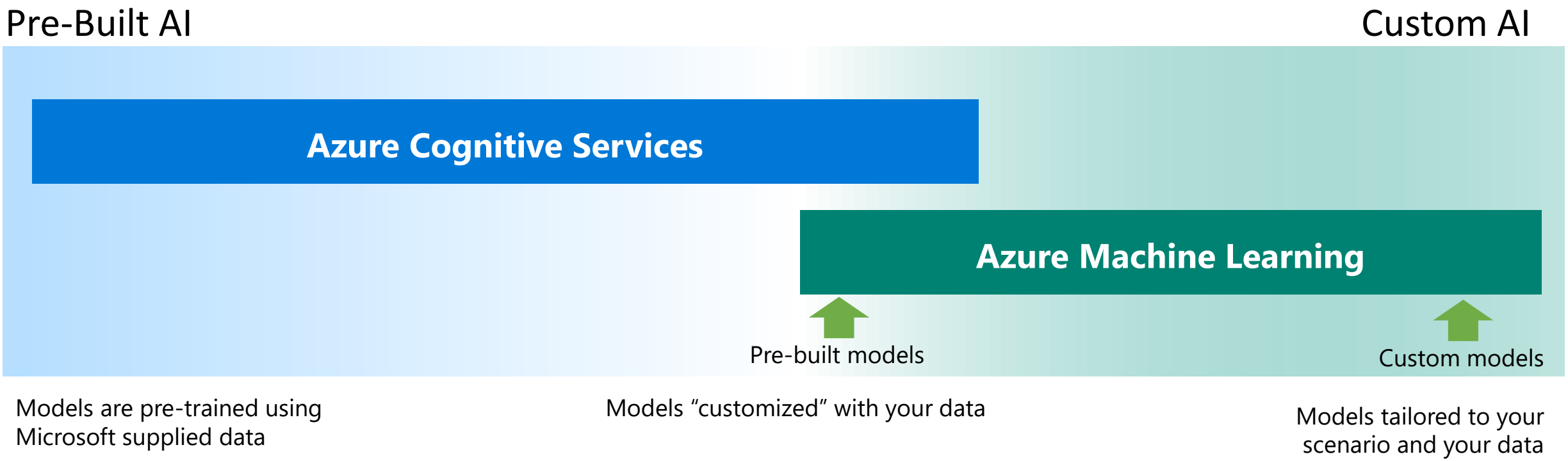
How much math did we write in this code?

```
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)
```

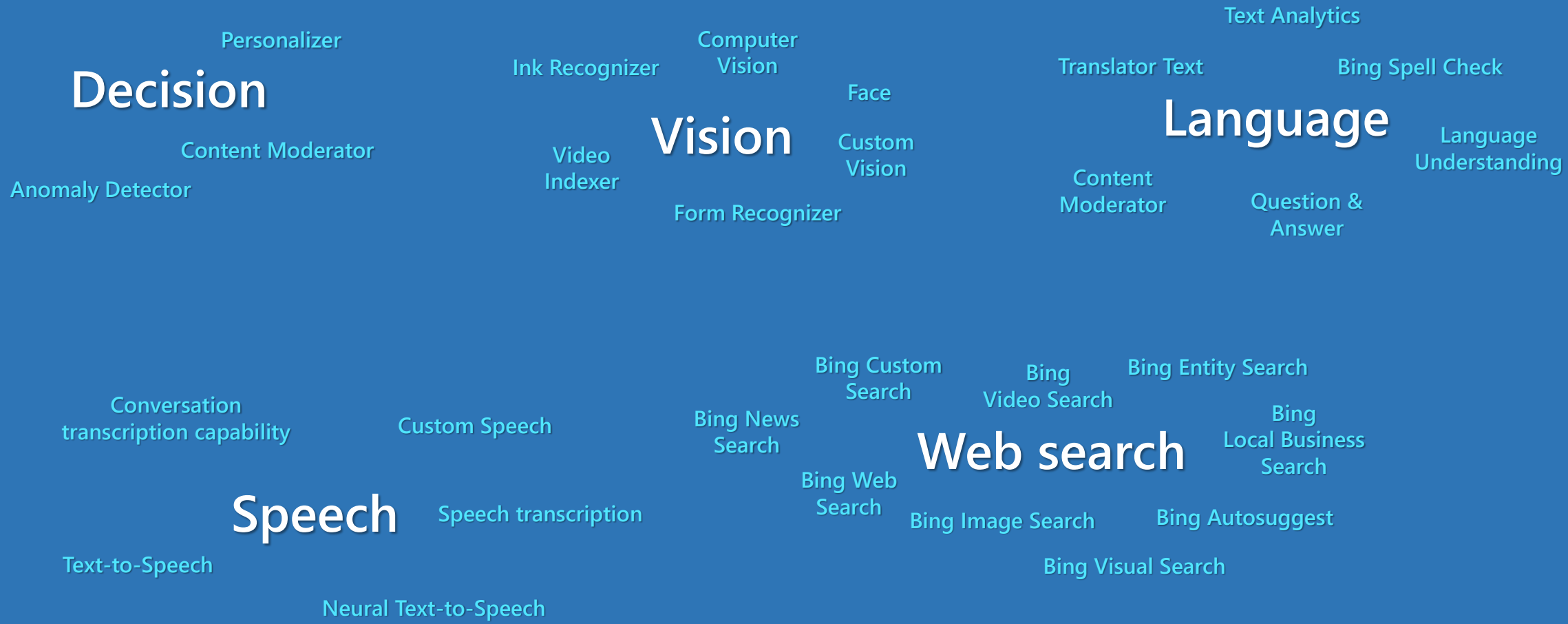
As a developer, what technologies do you need to learn?



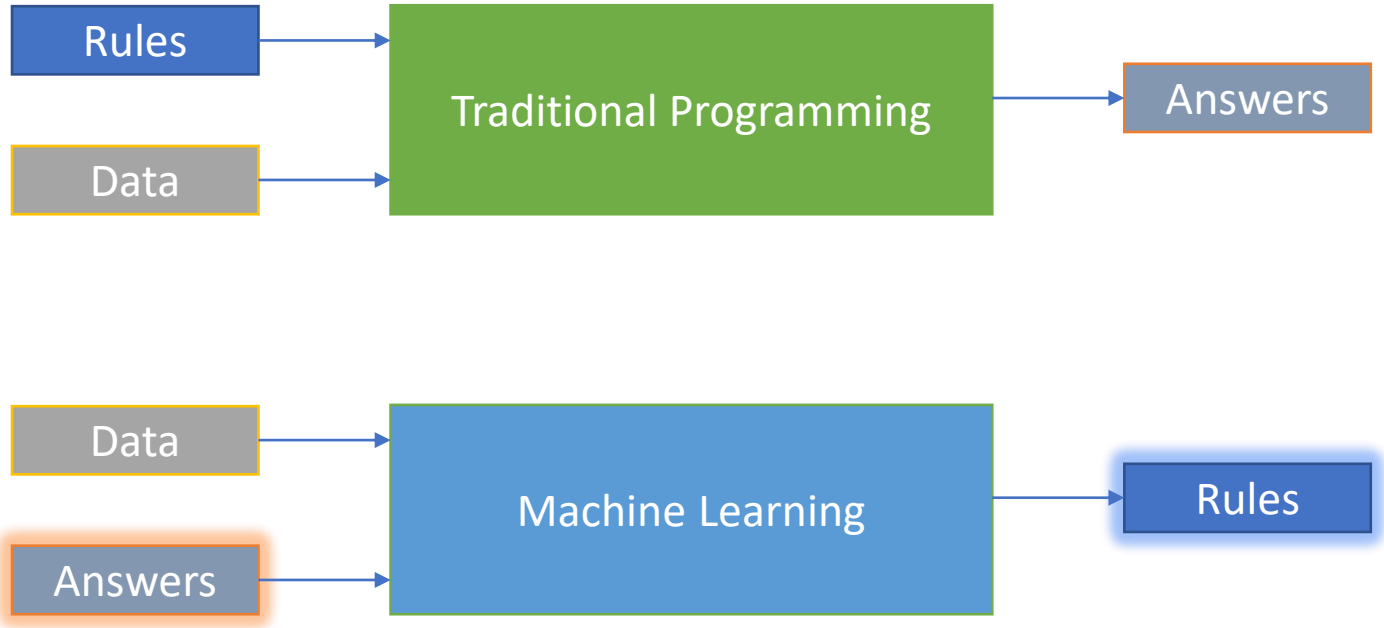
The Pre-Built and Custom AI Spectrum



Azure Cognitive Services

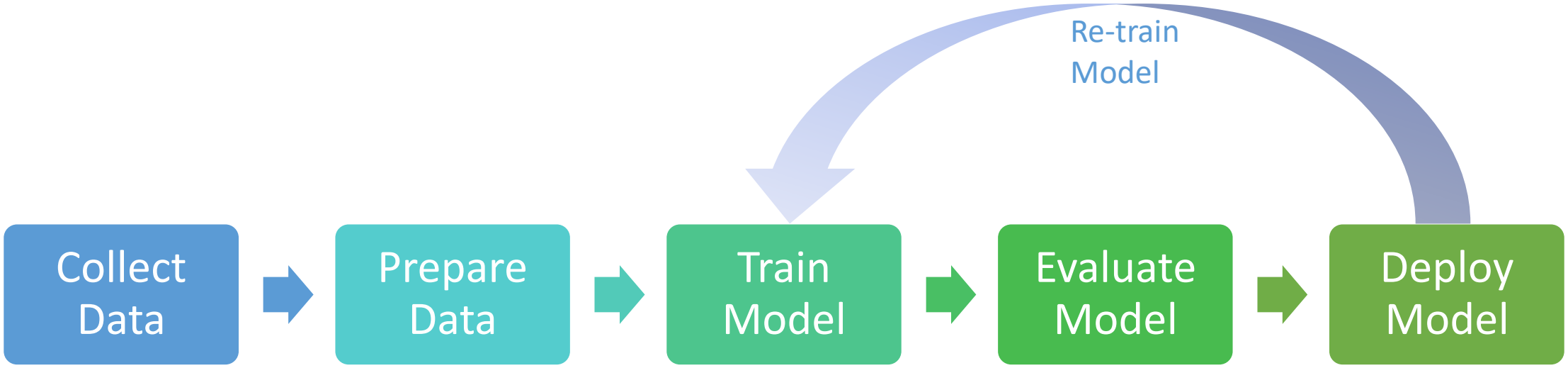


Machine Learning as the new programming paradigm

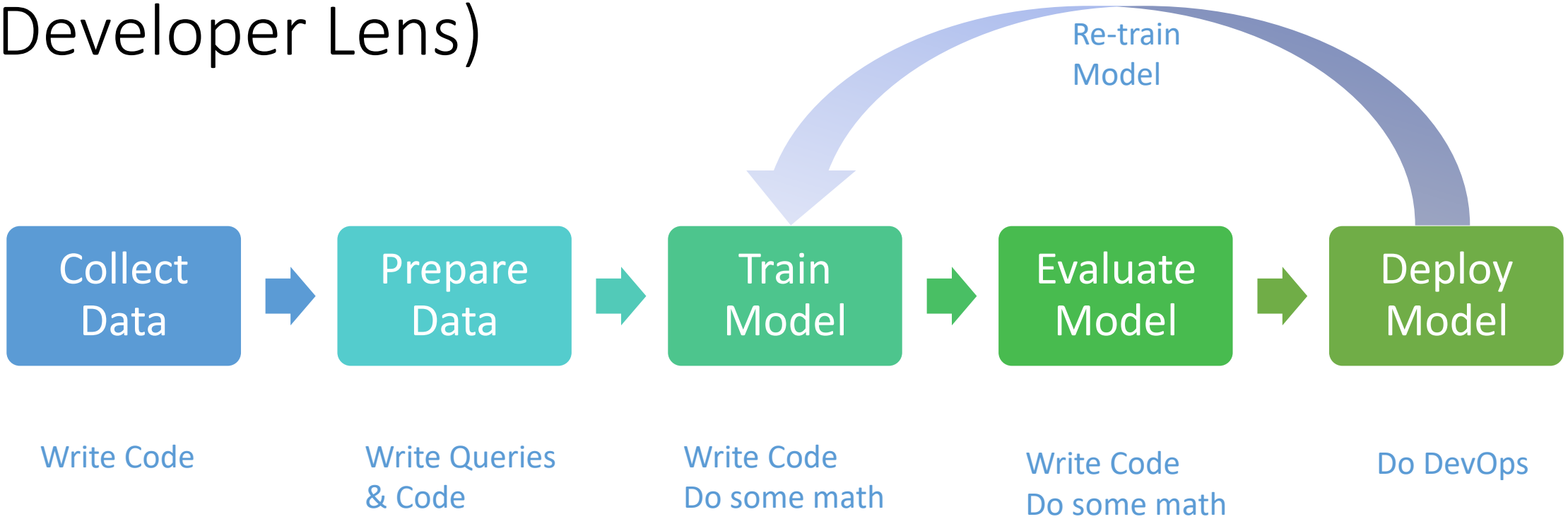


Let the machine figure out the rules based on history!

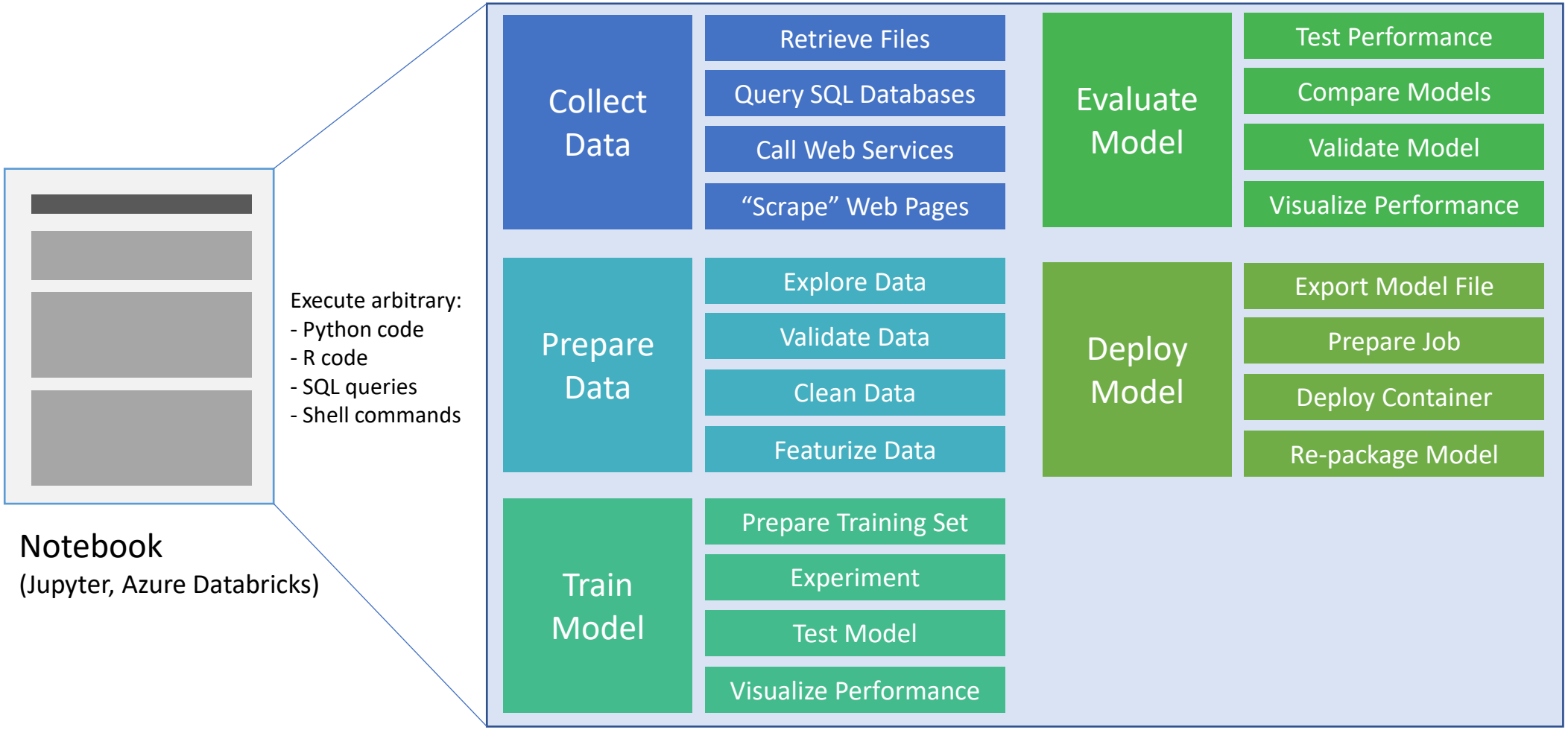
Behind Machine Learning is the the Data Science Process



The Data Science Process (Developer Lens)



The Notebook Paradigm



Examples of Tools for Data Science in the Wild

Core frameworks and tools

Machine Learning & Deep Learning

Visualization

Natural Language Processing

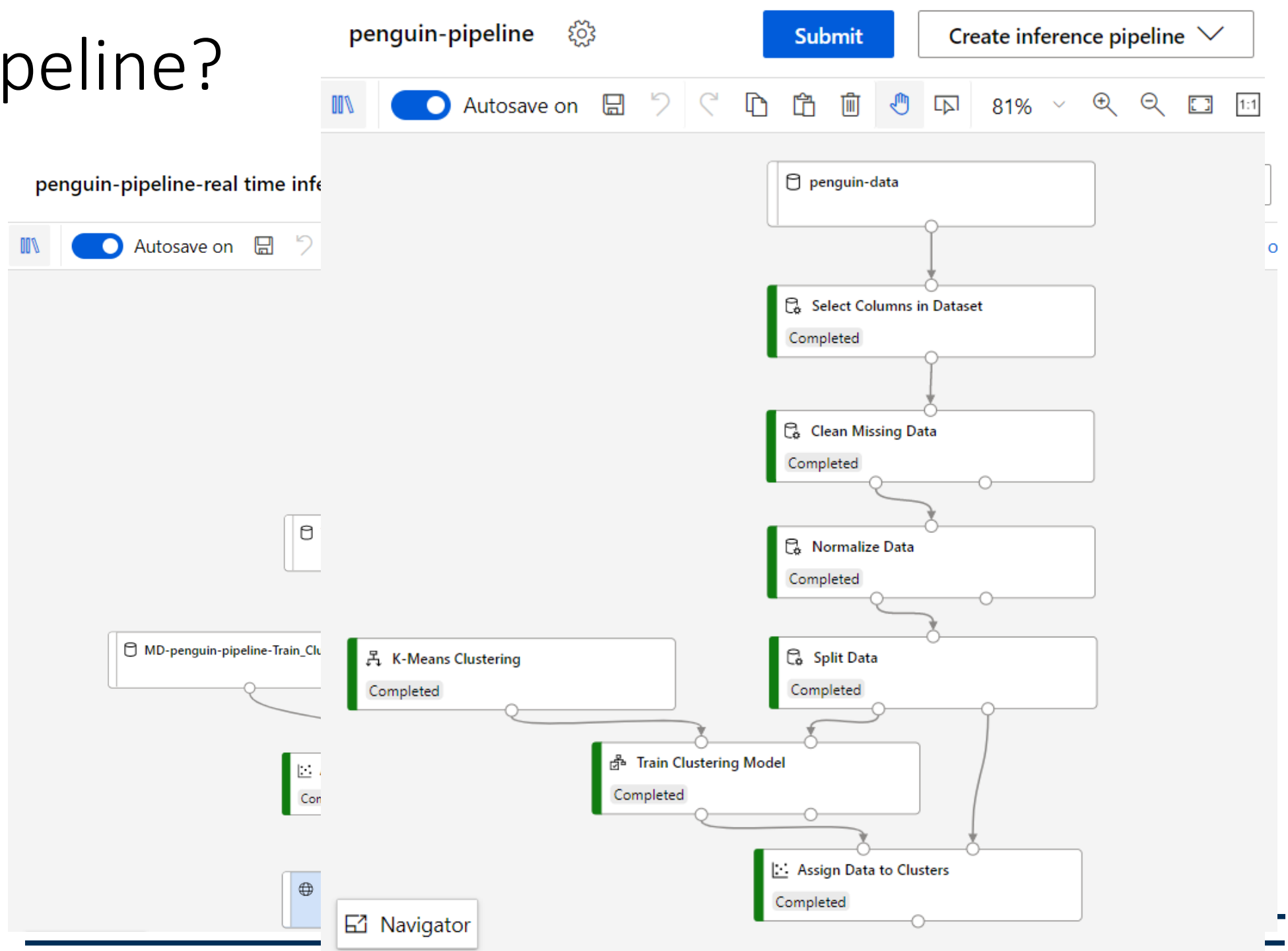
What is a Pipeline?

The working metaphor for Azure Machine Learning

Training pipelines to train models

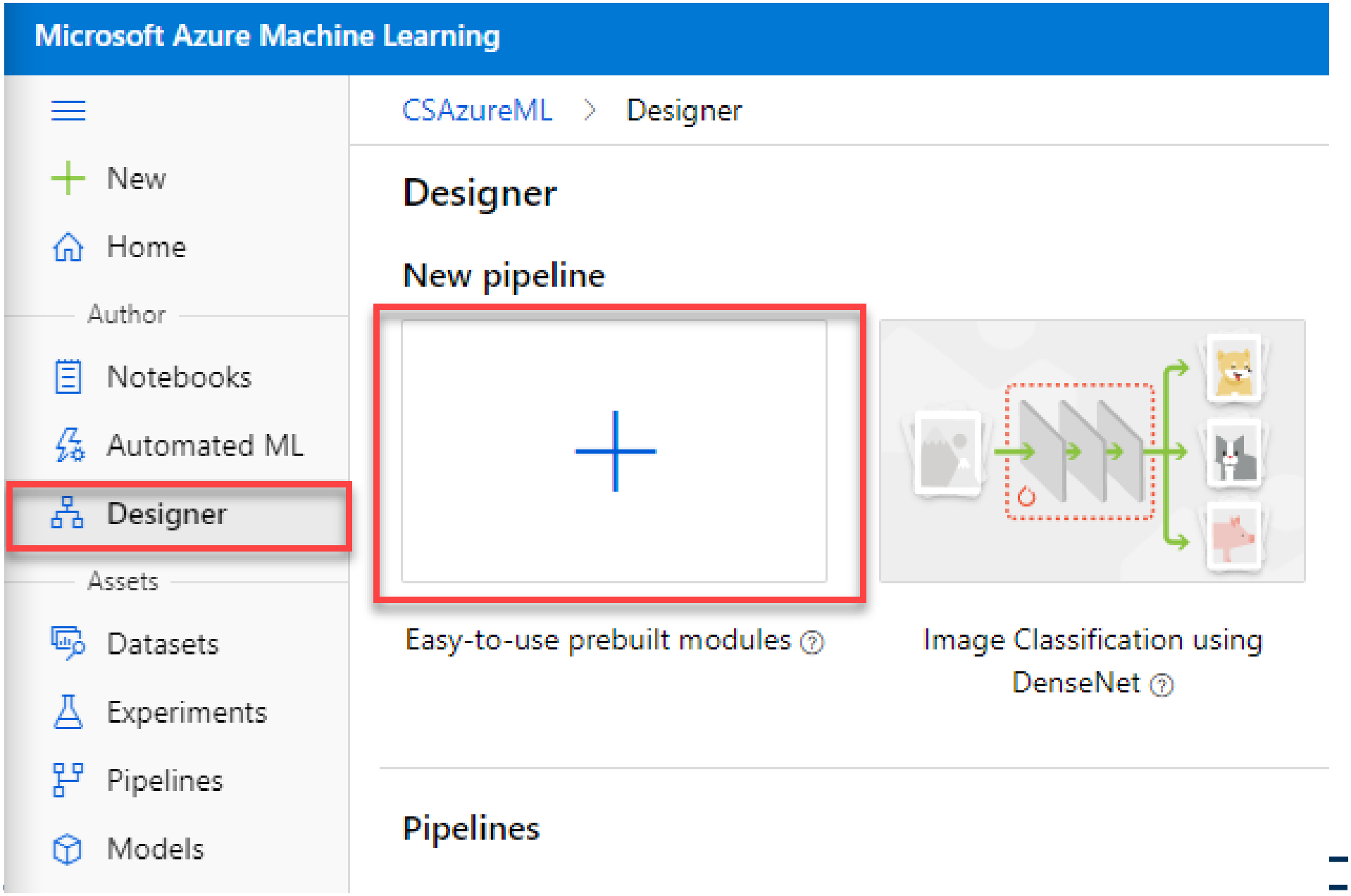
Inference pipelines to generate predictions

- Real-time inference
- Batch inference



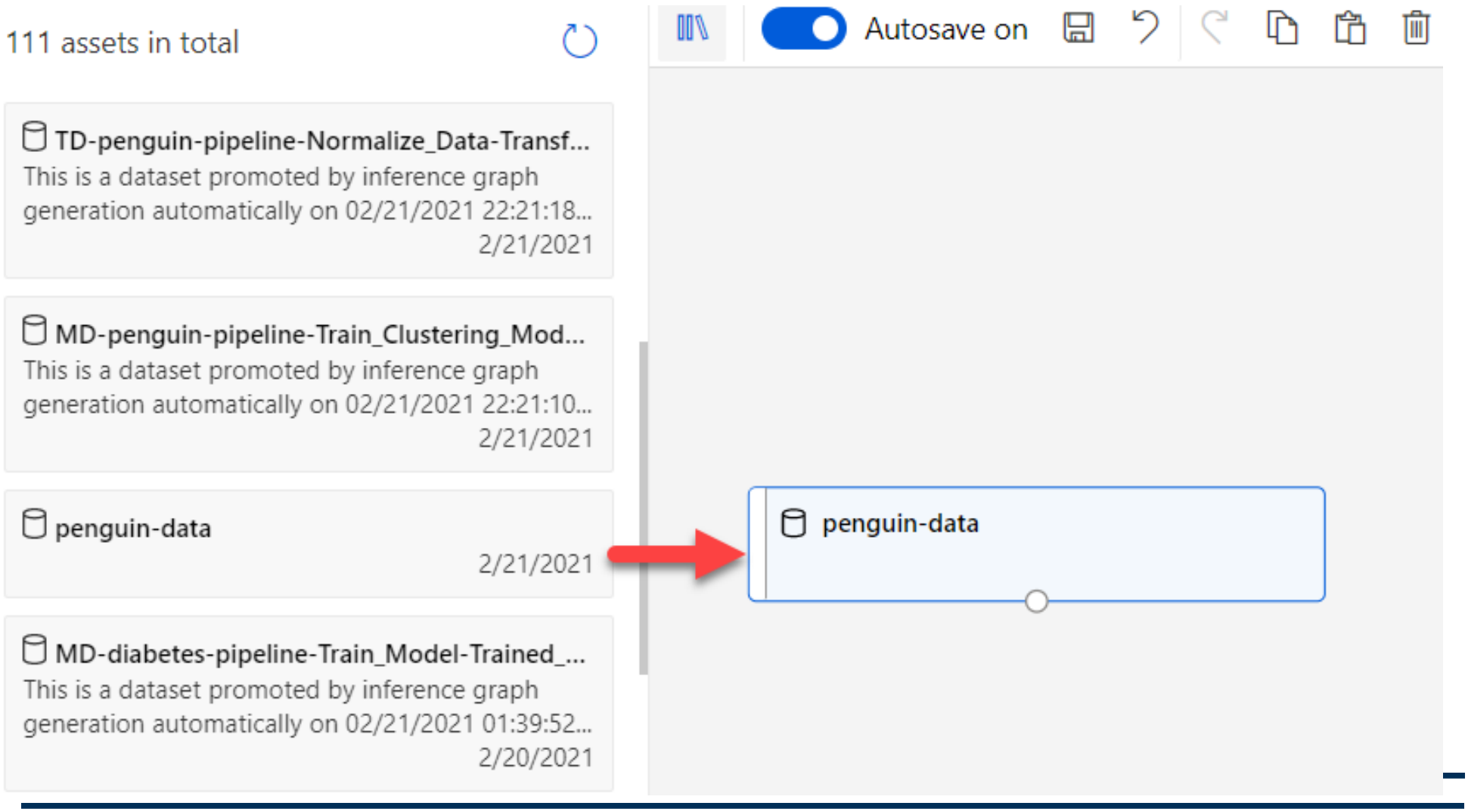
Creating a New Pipeline

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



Modifying a Pipeline

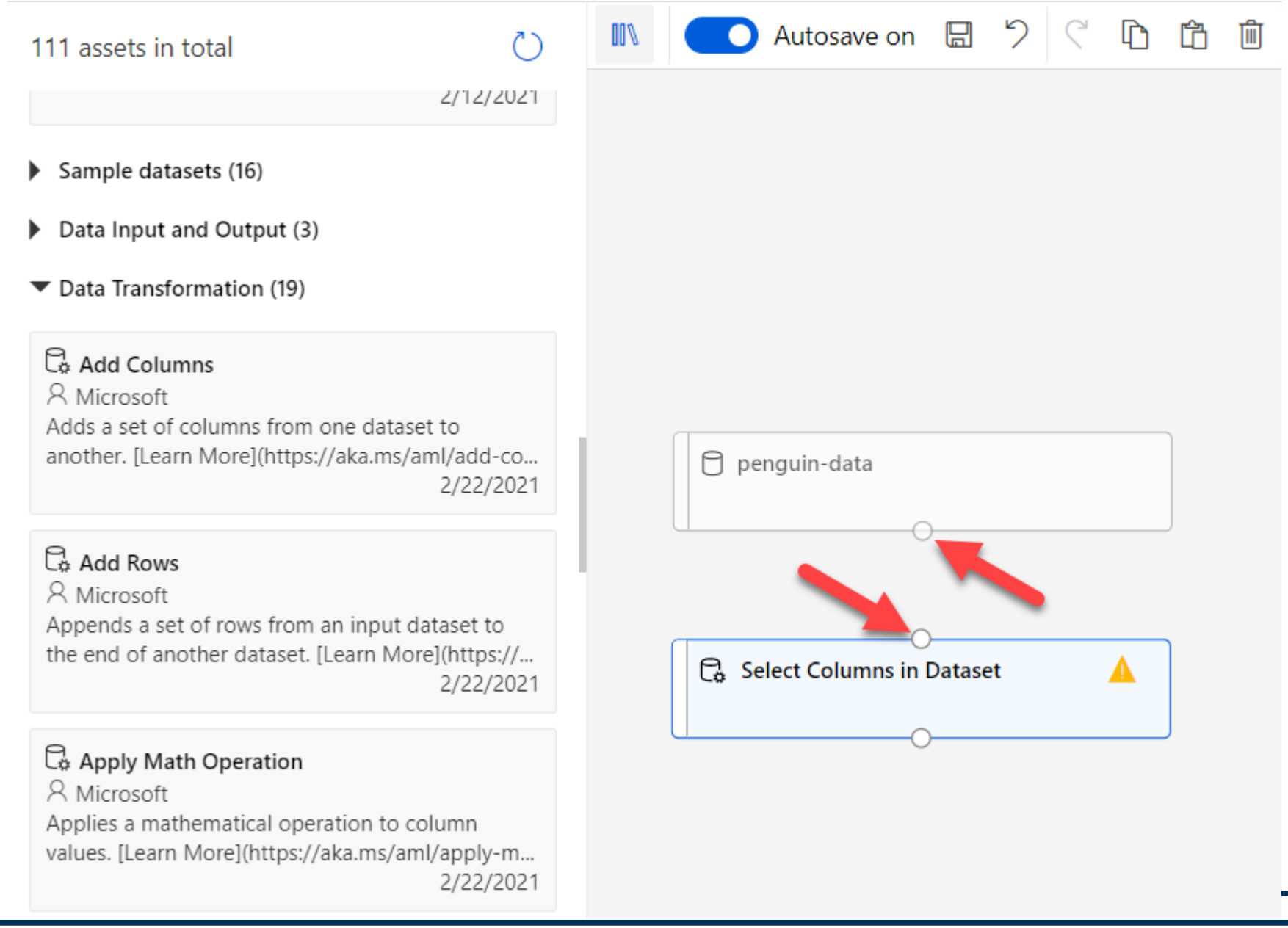
The Azure Machine Learning Studio includes a drag-and-drop interface. Bring pipeline components onto the canvas.



Modifying a Pipeline

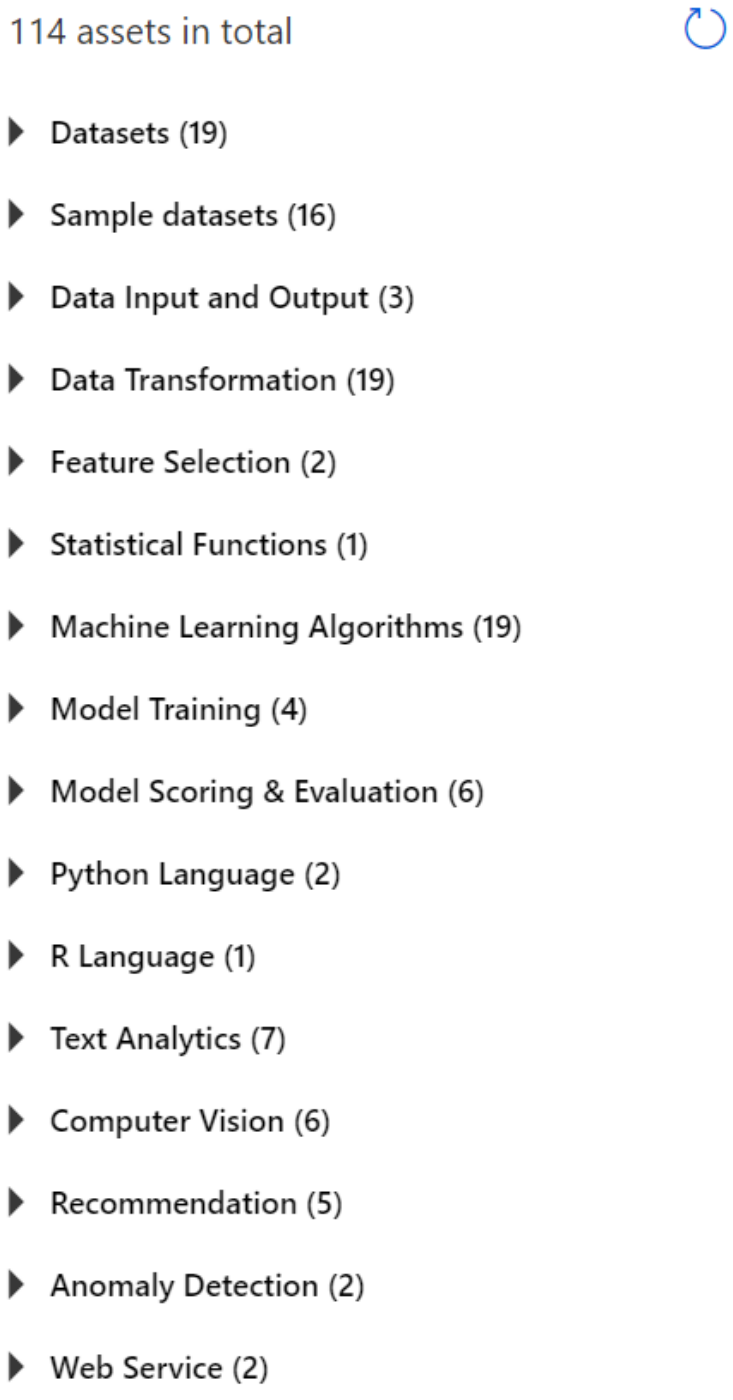
Then, connect them together.

Each connector is of a specific class, such as **DataFrameDirectory** or **TransformationDirectory**. Pipeline components can only connect if the classes match.



Built-In Components

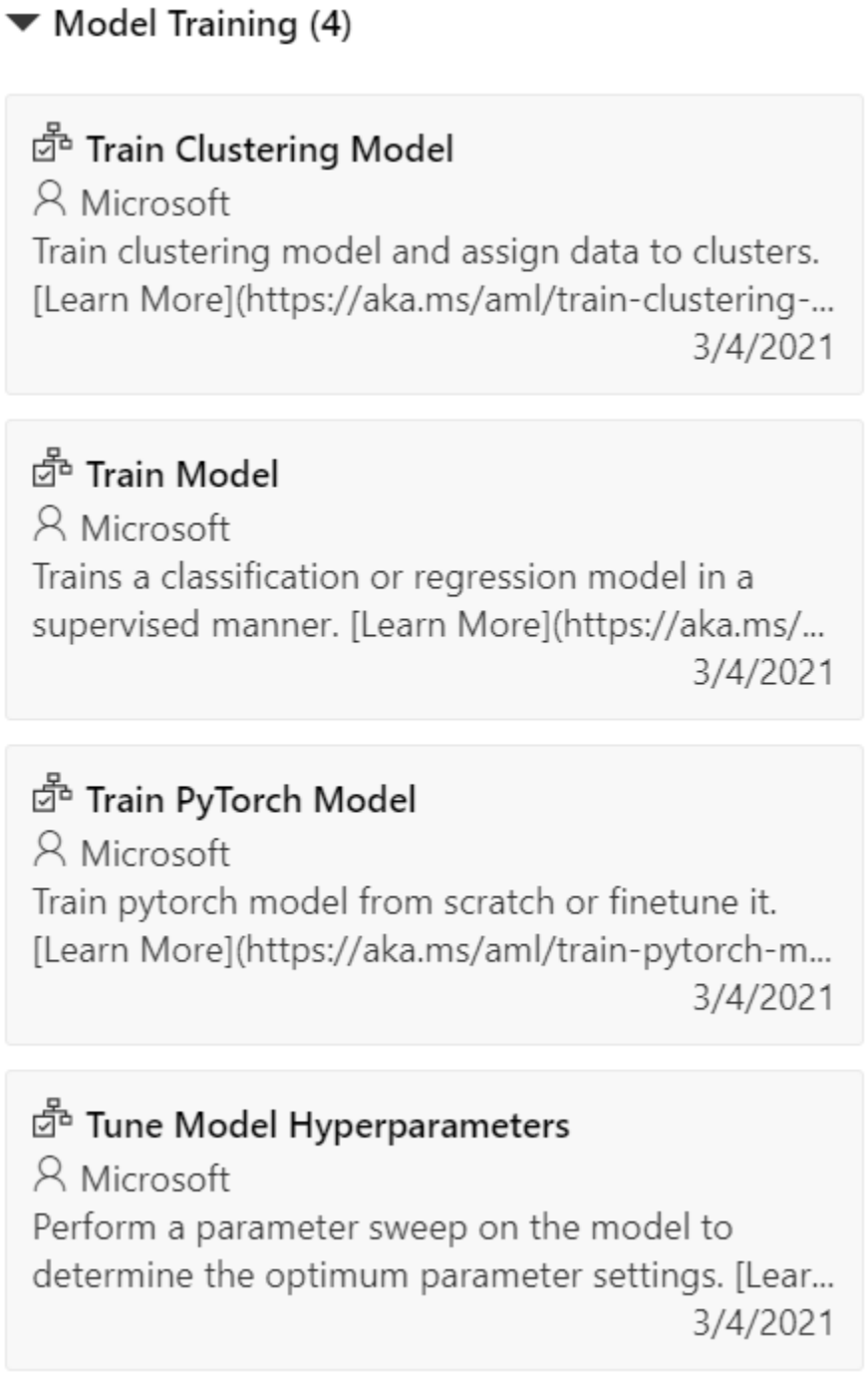
Azure Machine Learning features a variety of built-in components.



Built-In Components

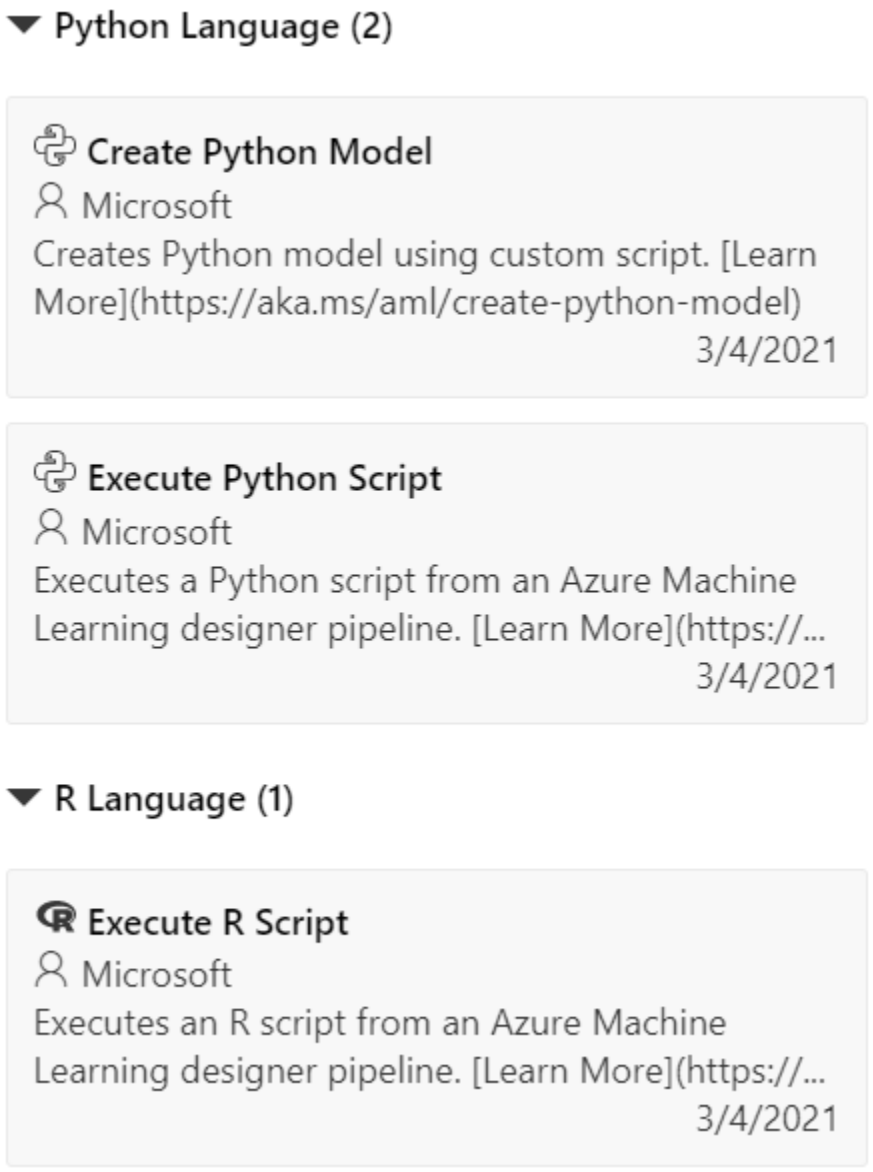
Model training includes training the following sorts of models:

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



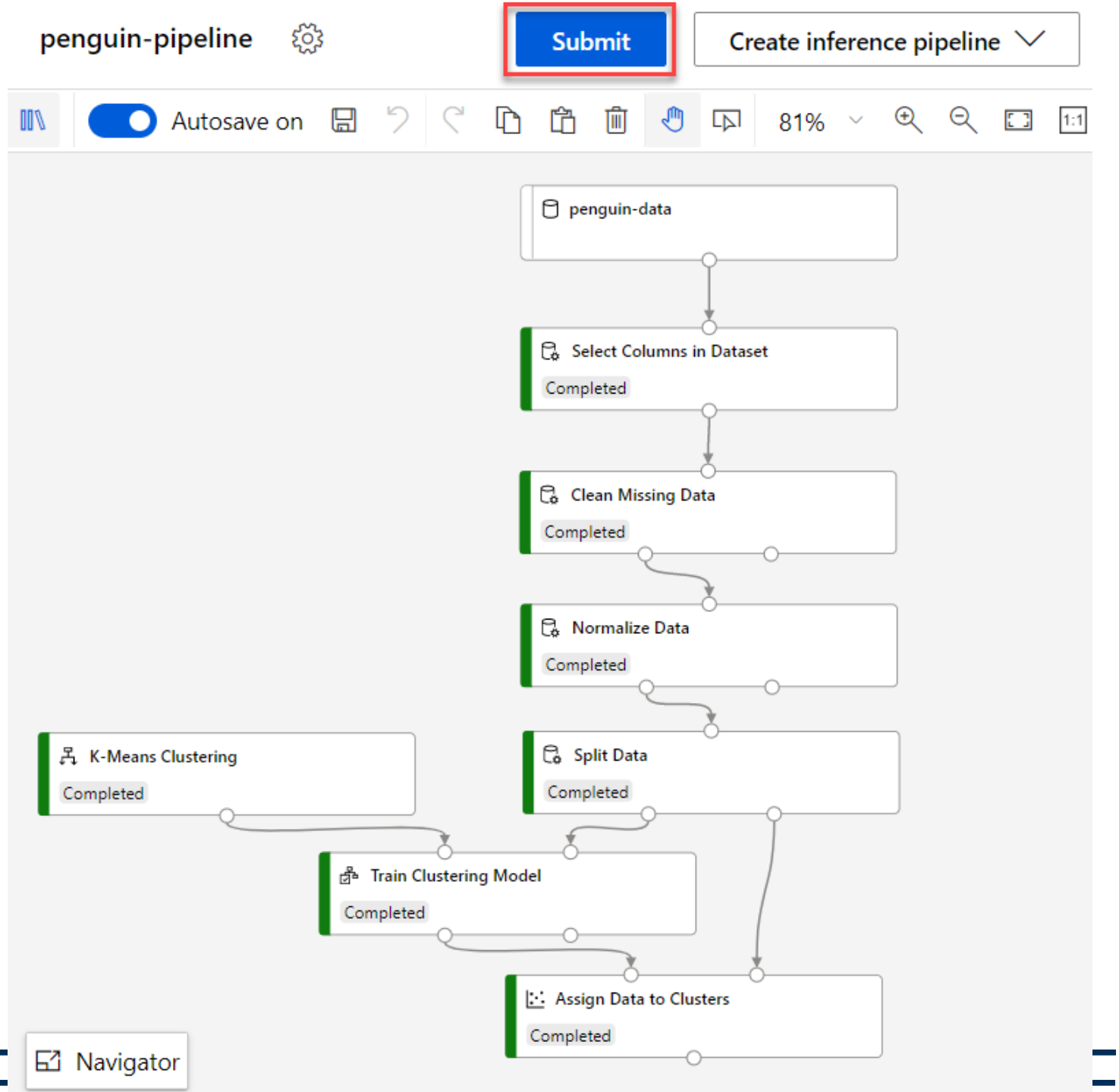
Custom Components

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



Training a Model

Train models at the click of a button.



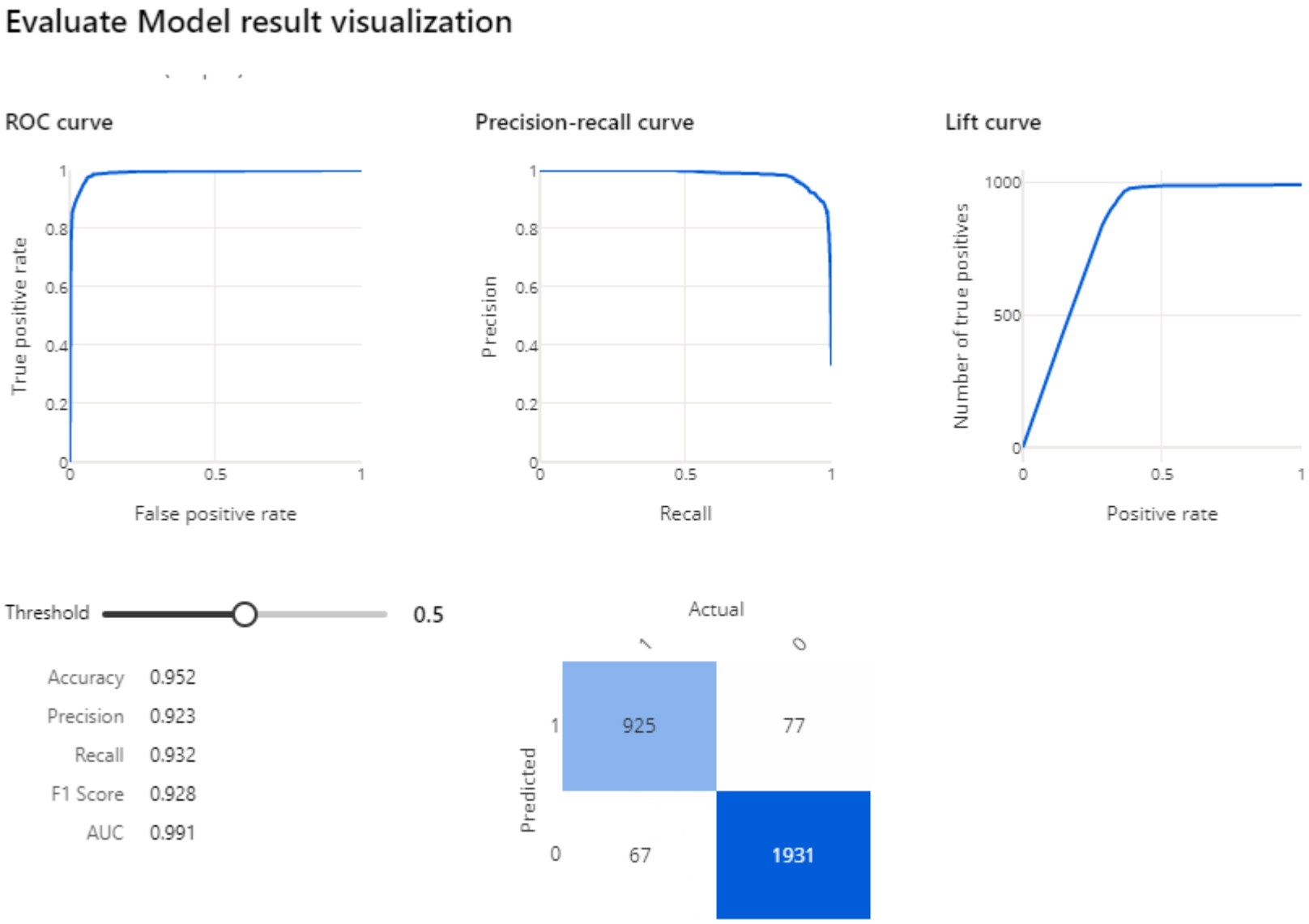
Evaluating a Model

Regression and classification models include UI components for evaluation.

Regression includes statistics such as R^2, 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.



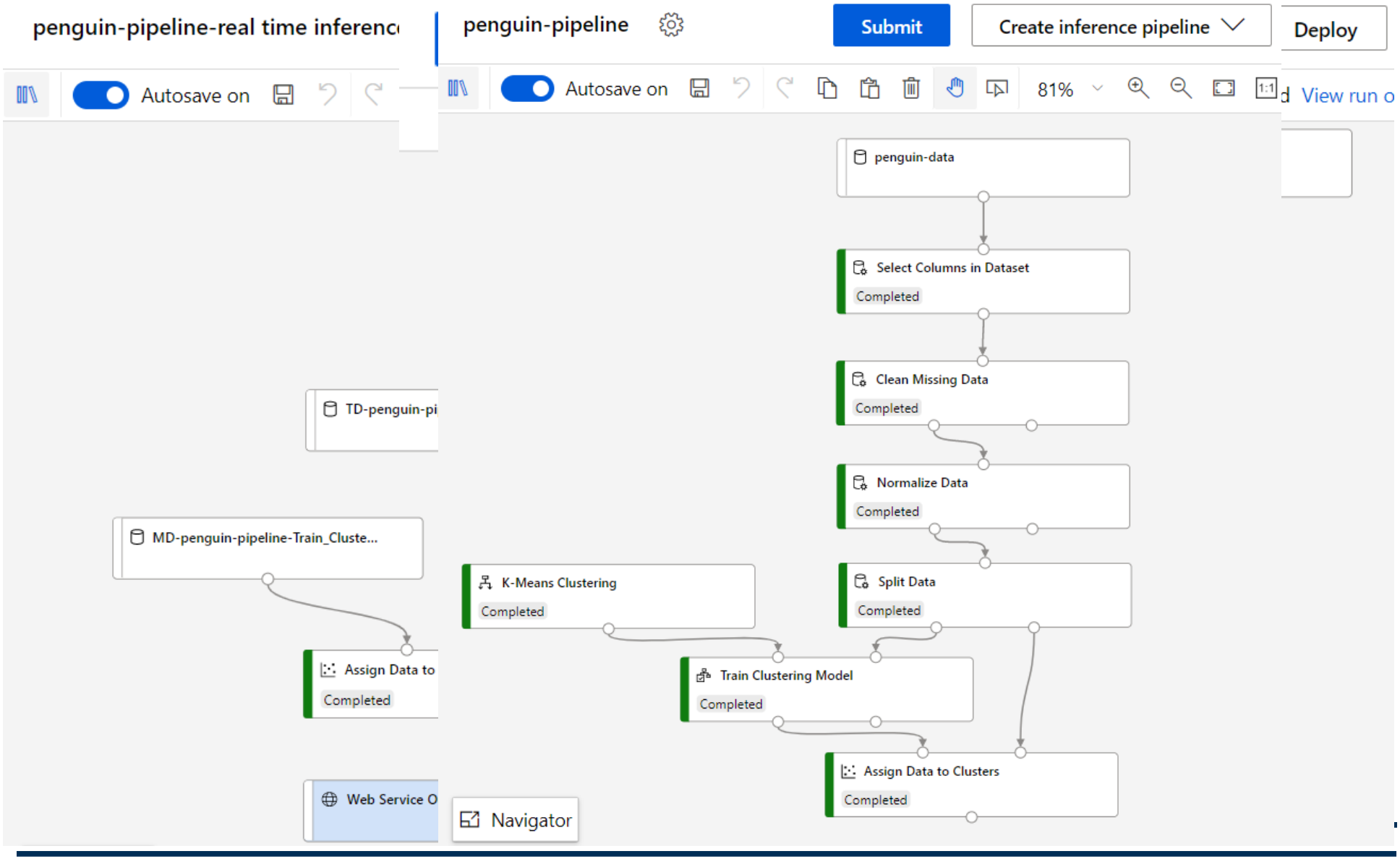
Creating an Inference Pipeline

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.

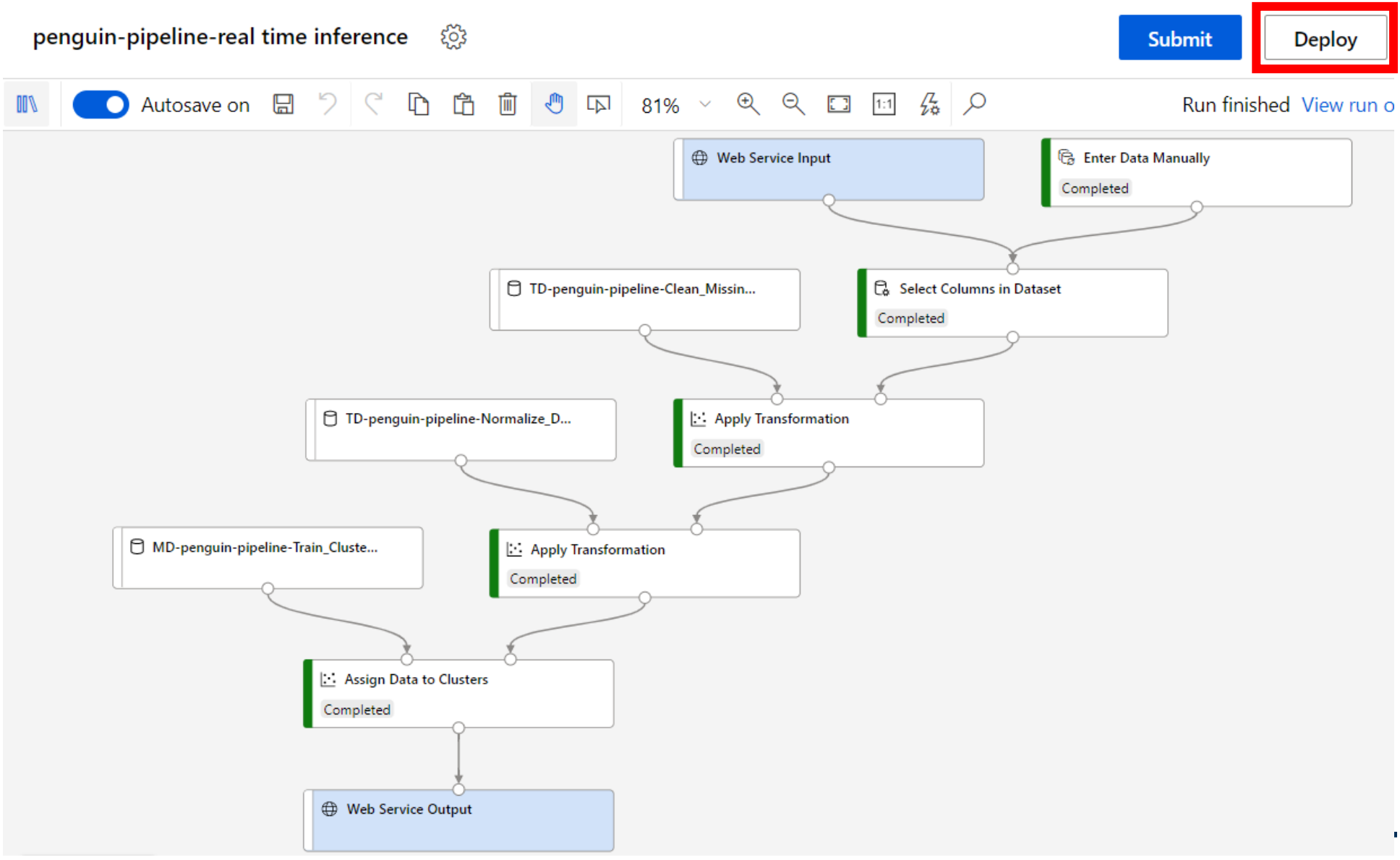
Batch inference pipelines typically write results to files or databases and operate asynchronously.



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.

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.

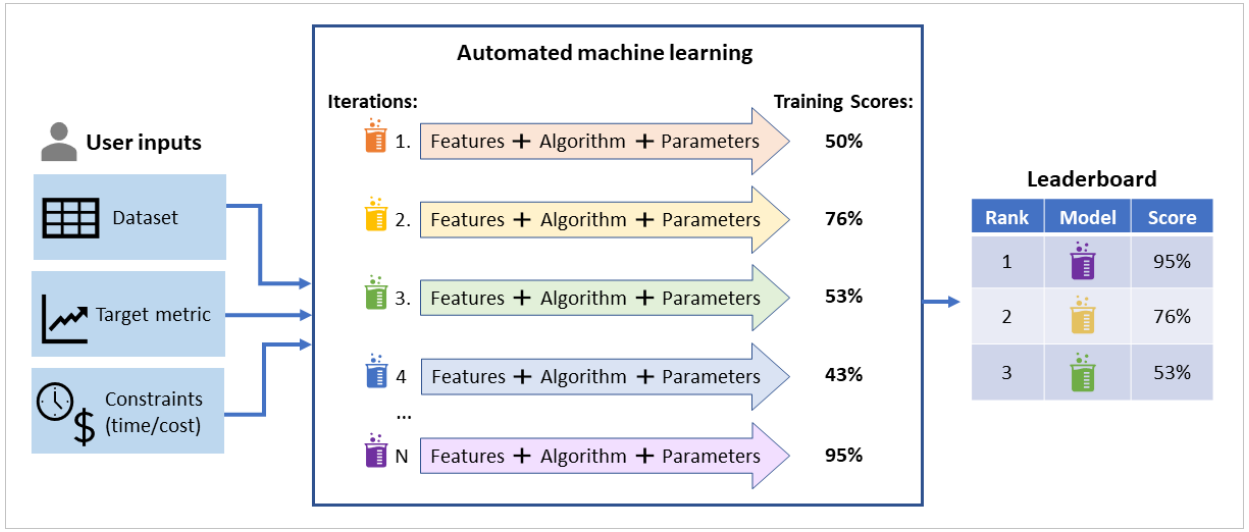
Key Scenarios:

- Classification
- Regression
- Forecasting

Not suitable outside of these scenarios.

Data Shape Required:

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



Best model

Deployment Options:

- Notebook
- Web Service
- ONNX

Not all algorithms support ONNX export.

Feature Engineering:

- Scaling/Normalization
- Dimensionality Reduction
- Encoding

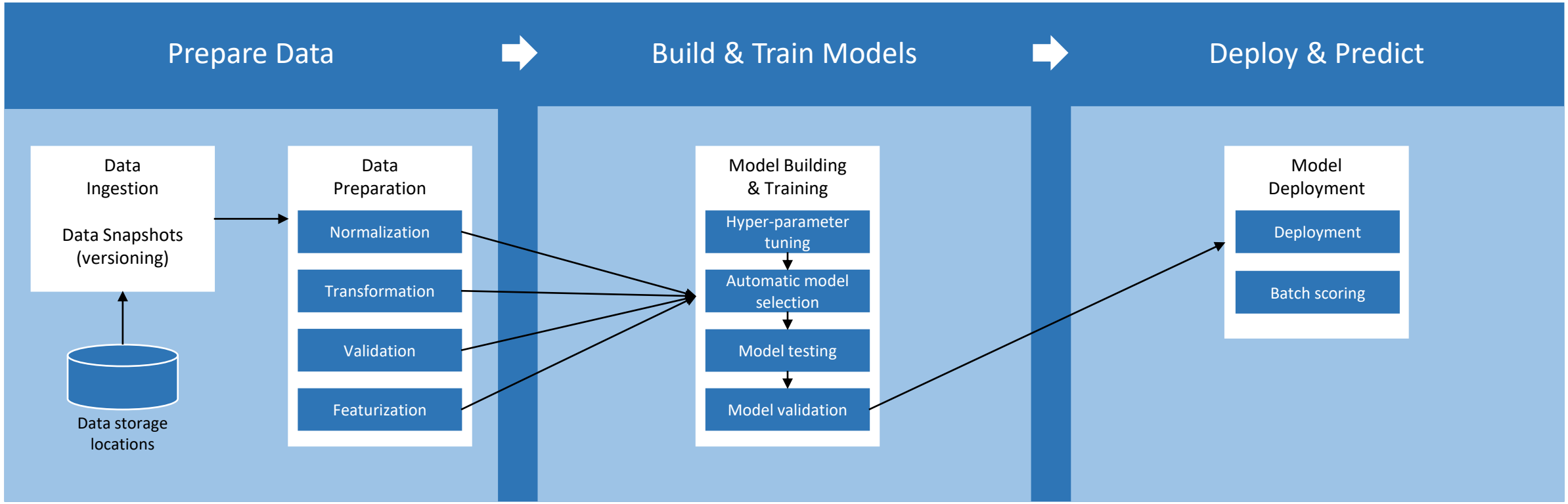
Algorithms (count):

- 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'
])
```

Us
co

Set container details

Specify packages to install

If True, use your own installed packages instead of Conda

Walkthrough: Creating a Pipeline

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

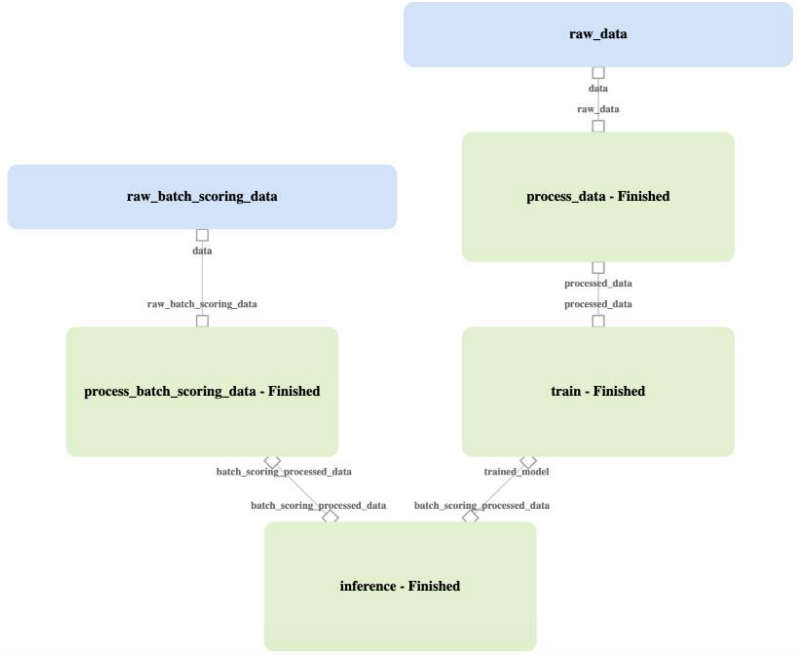
Run Properties

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

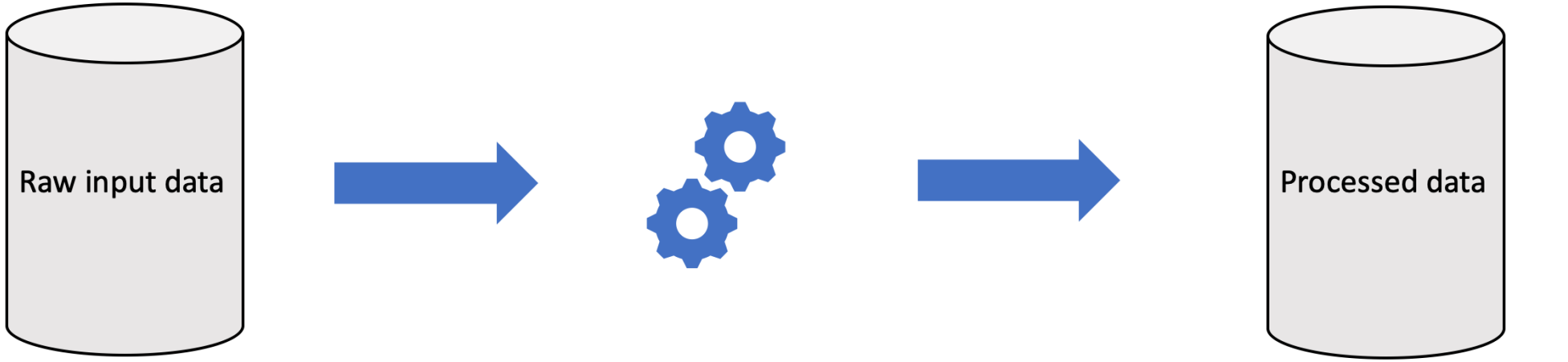
Output Logs logs/azureml/executionlogs.txt ☑ Auto-switch

```
[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	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



Create Data Prep Step



Azure Blob Storage

Data Prep Pipeline Step

- Normalization
- Transformation
- Validation
- Featurization

Azure Blob Storage

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_amlcompute
```

Pat

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

Specify the input and output objects

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

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

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

Build a pipeline, tying together the prior steps

Simple validation: check for unconnected inputs, etc.

Create an experiment and execute the pipeline

Show th
the pipeline

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

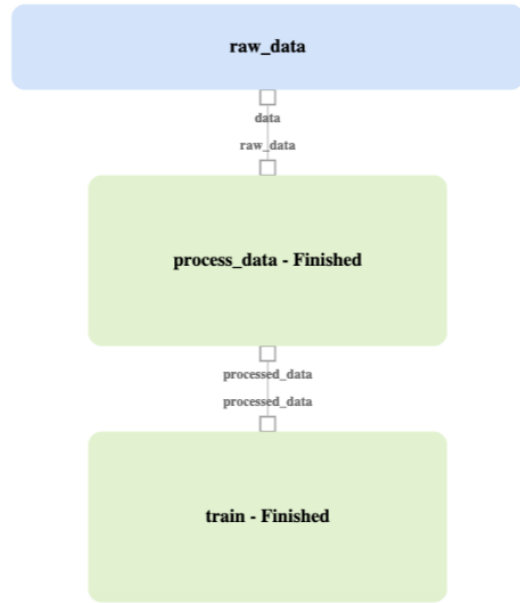
Run Properties

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

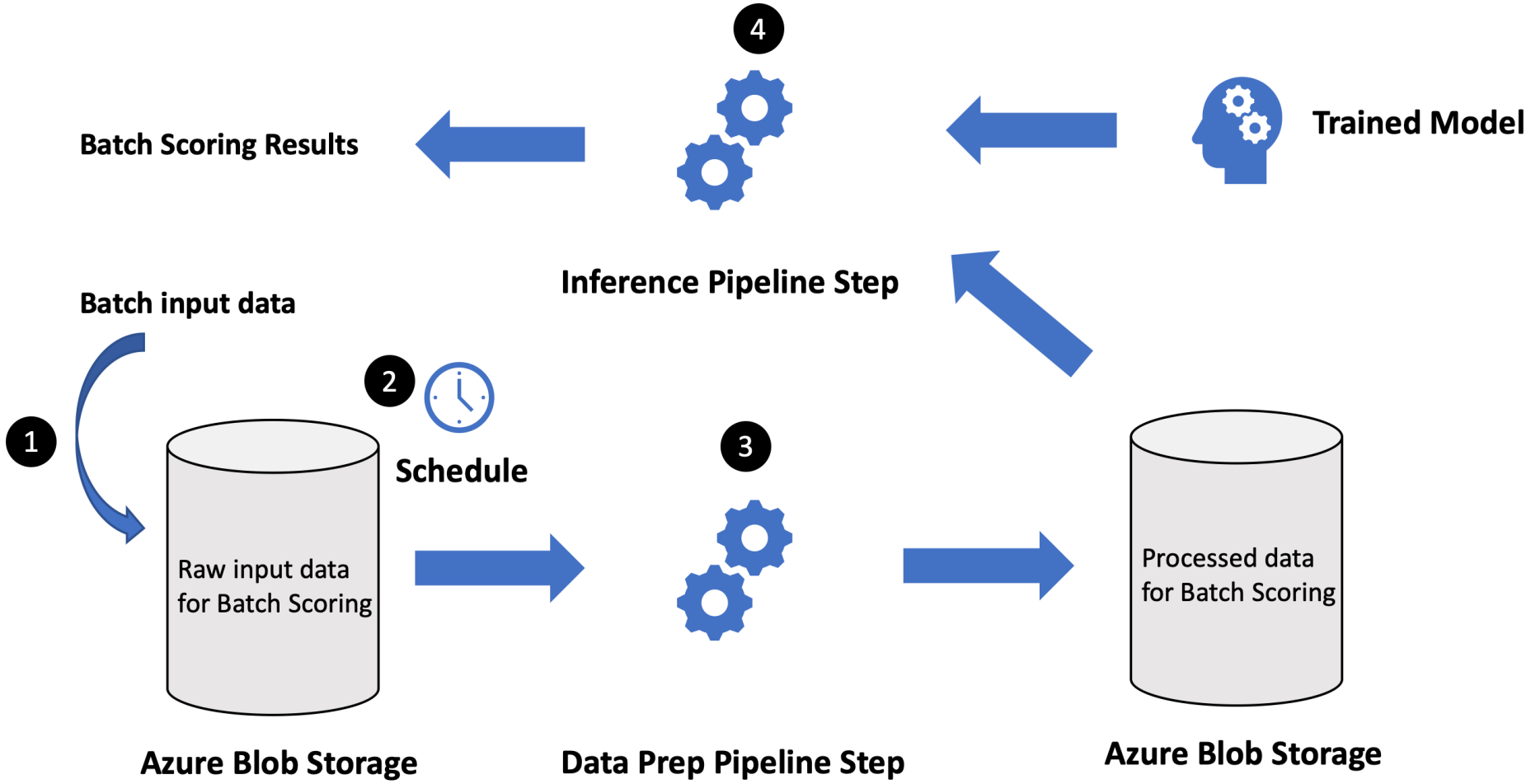
Output Logs logs/azureml/executionlogs.txt ☑ Auto-switch

```
[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



- Normalization
- Transformation
- Validation
- Featurization

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_
```

Three input arguments: train/inference, input

Specify the input and output objects

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_amlcompute)
```

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 inter
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

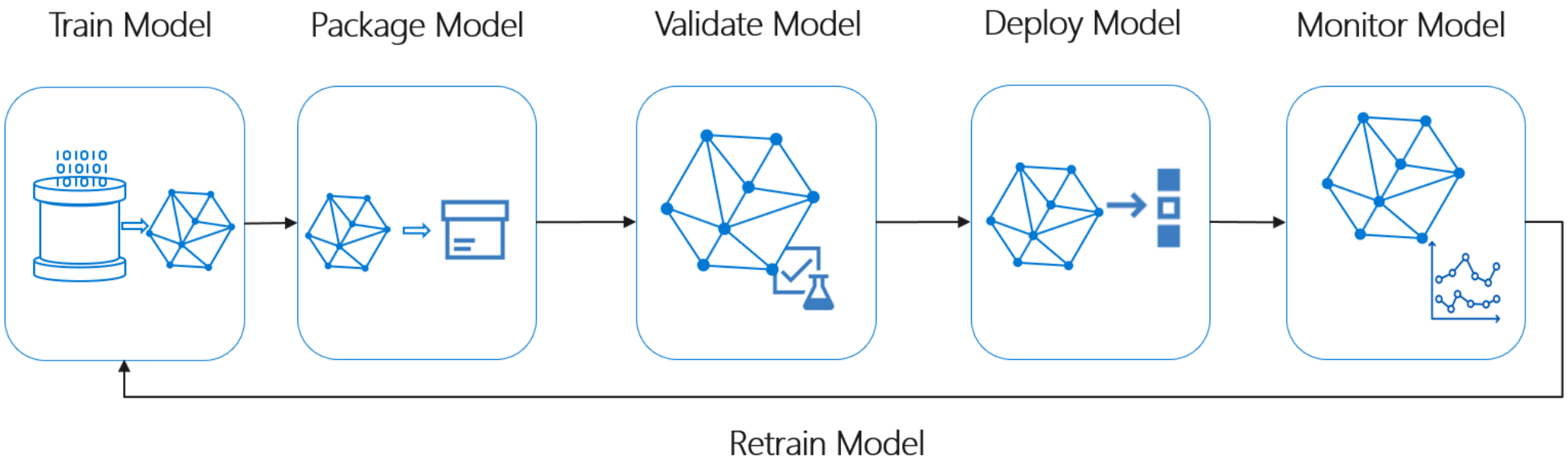
Responsible AI and Machine Learning

Azure Data Conference

POWERED BY

Azure Data Community & Azure Data Conf

The Machine Learning Lifecyle



Responsible AI

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”

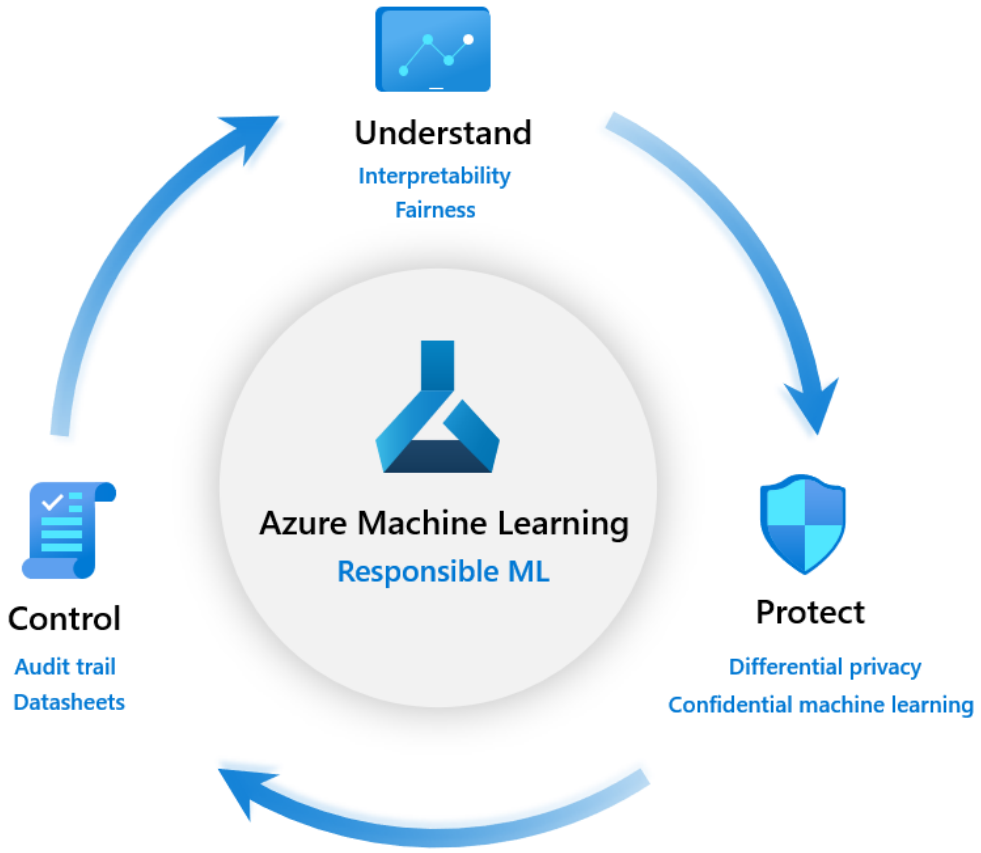
Transparency

“AI systems should be understandable”

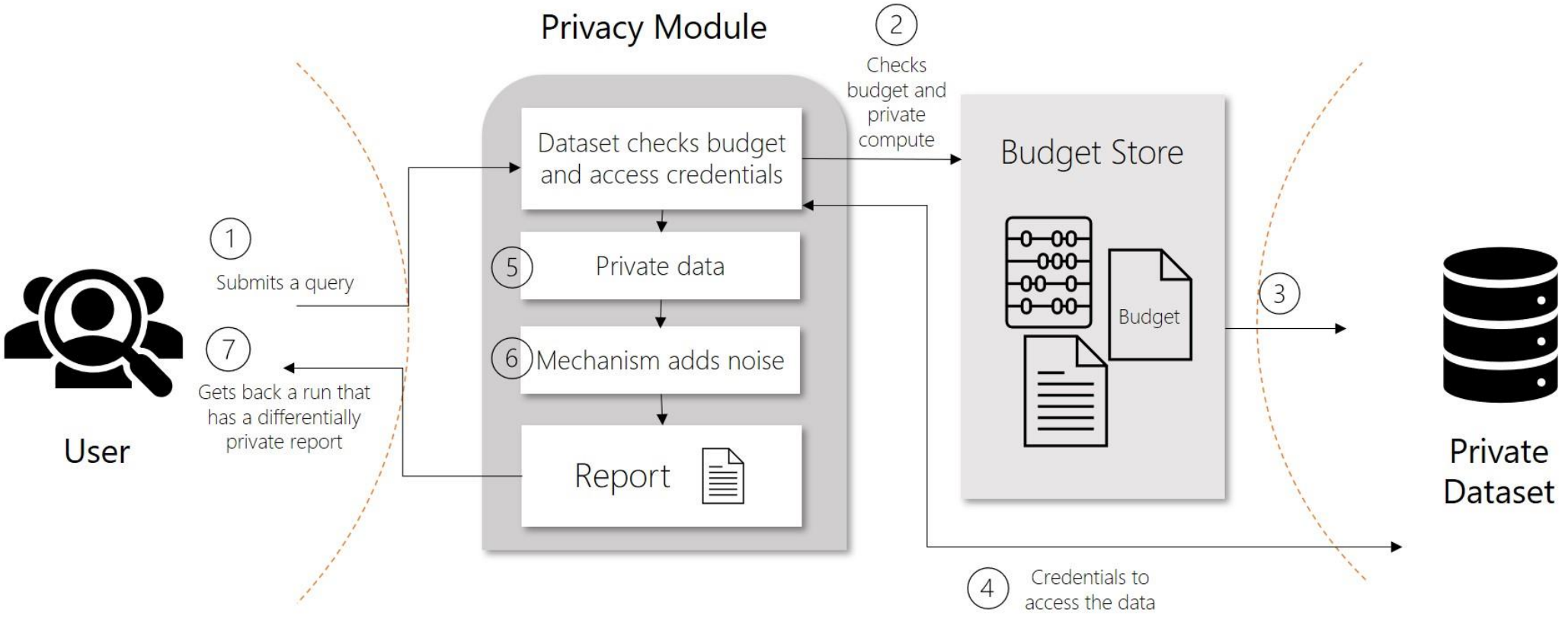
Accountability

“People should be accountable for AI systems”

Responsible Machine Learning

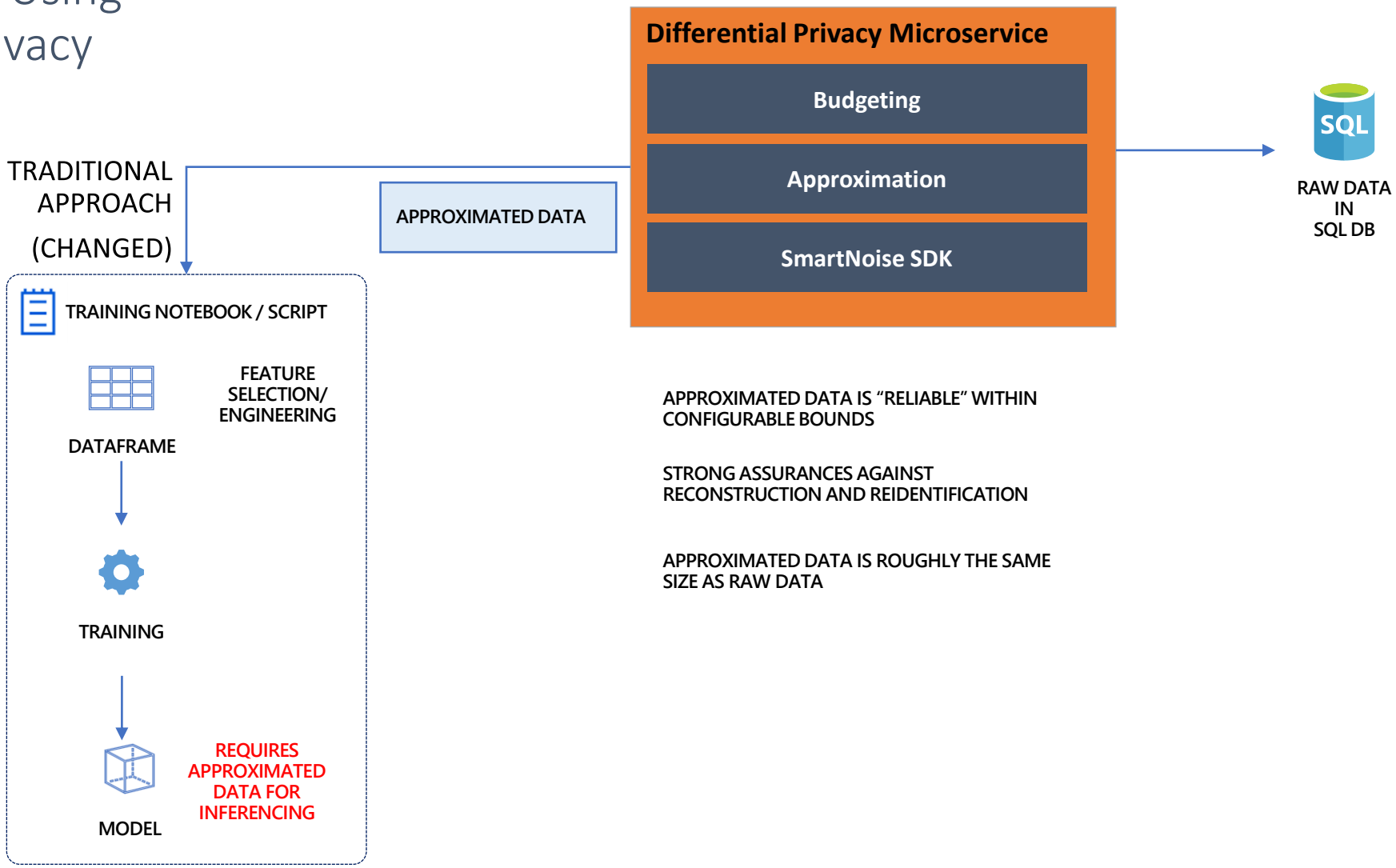


Differential Privacy

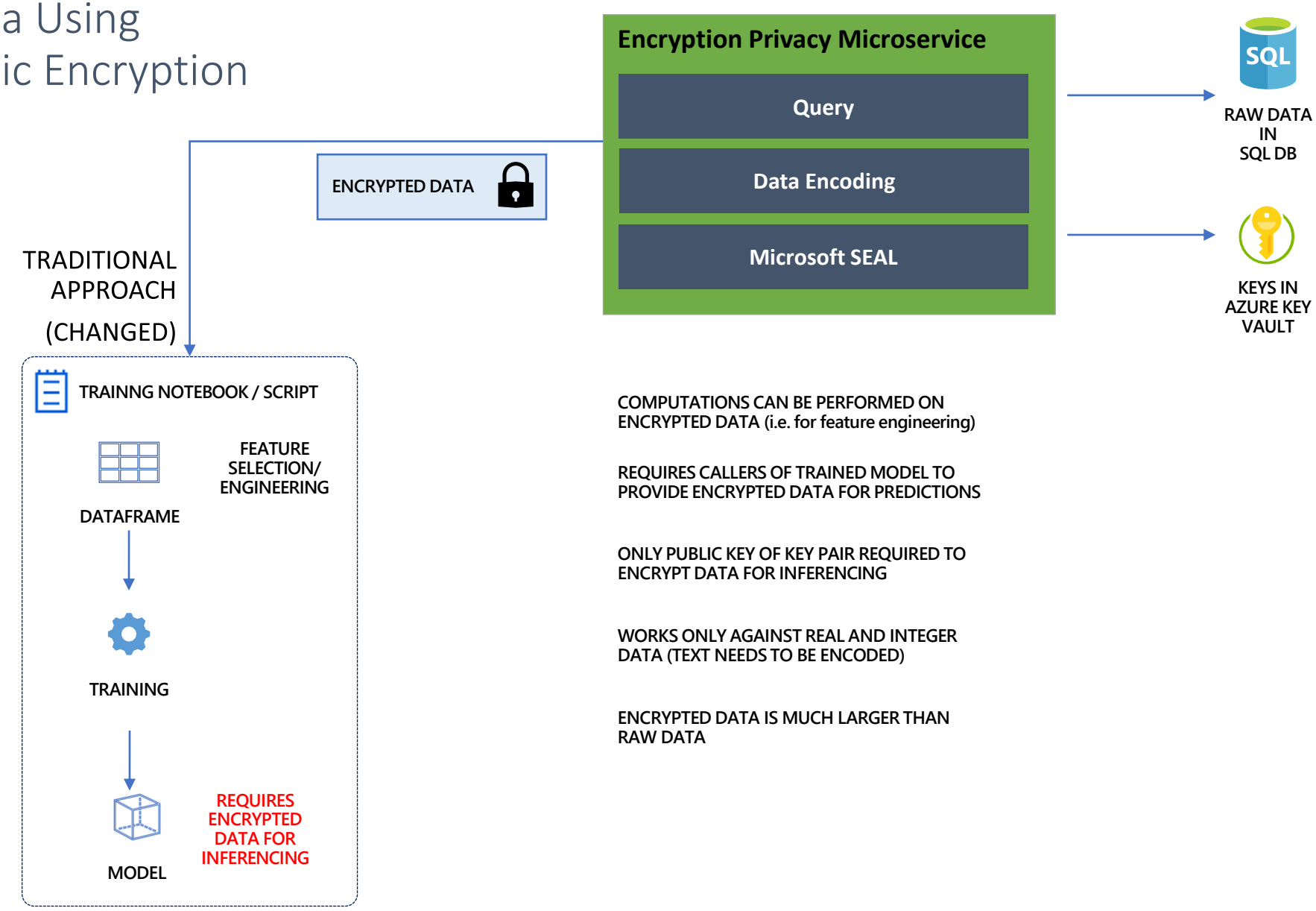


SmartNoise SDK

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

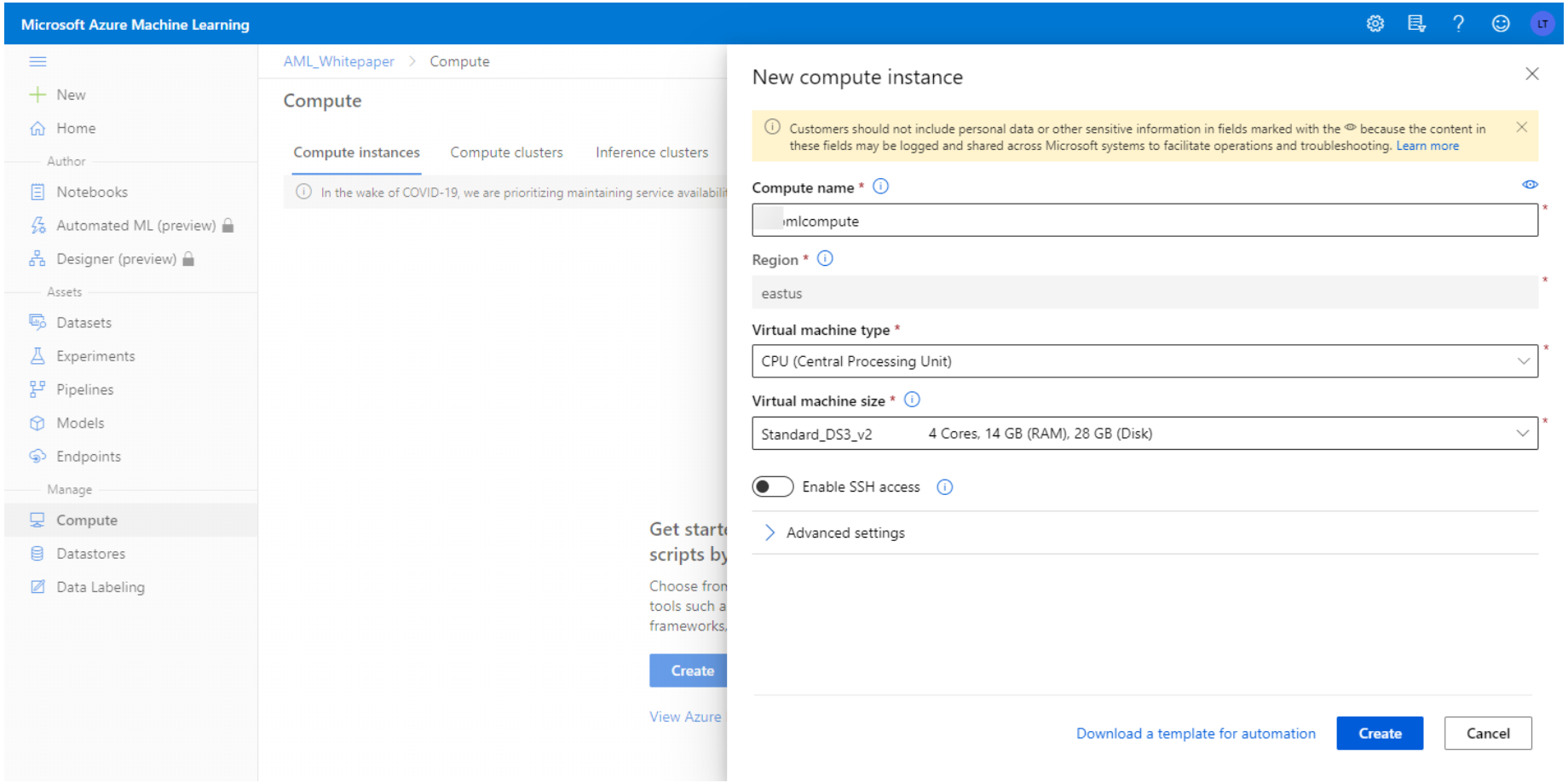
POWERED BY

Azure Data Community & Azure Data Conf

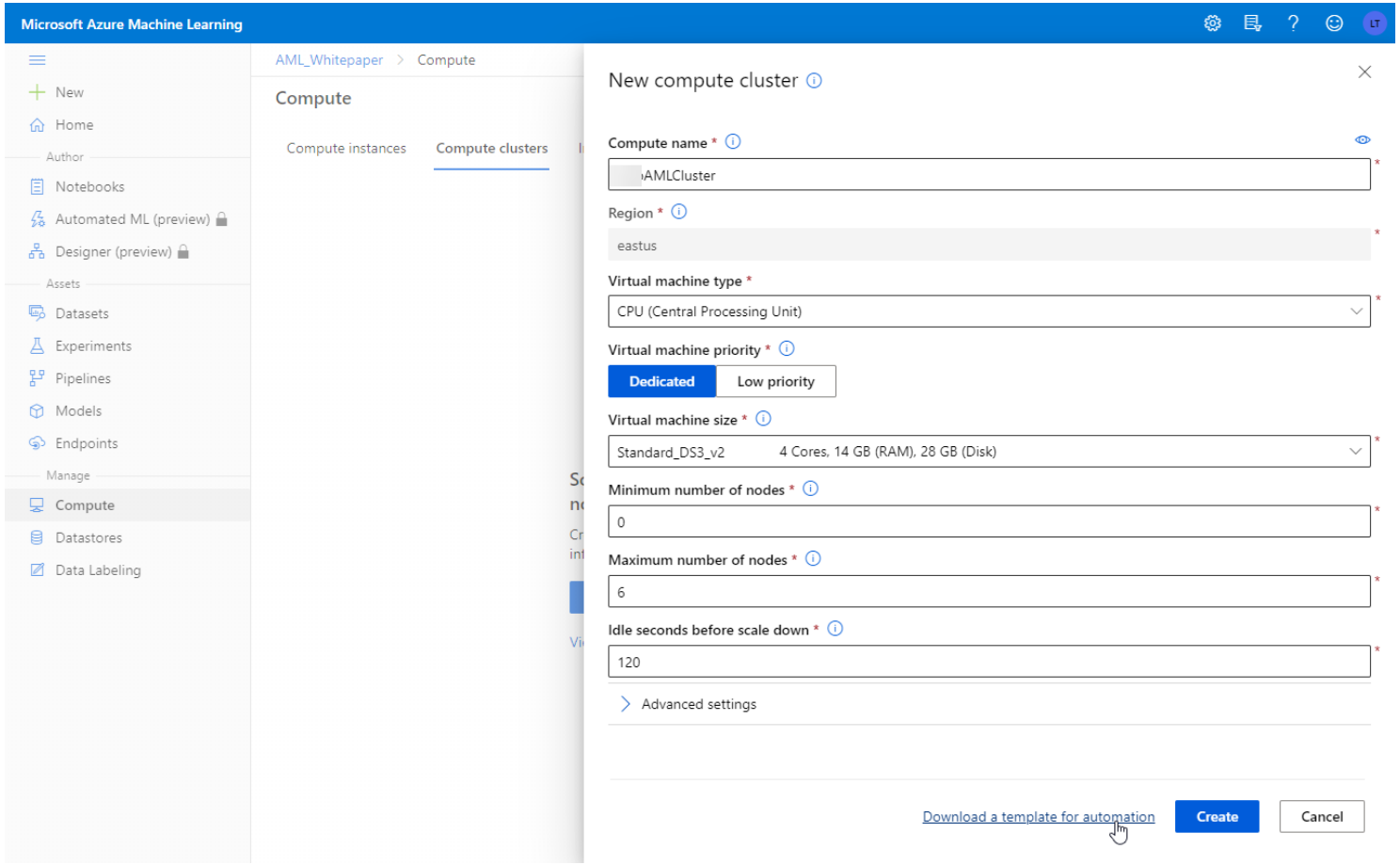
Agenda

Compute Instance
Compute Clusters
Inference Clusters
Managed Compute
Environments
Pipelines
GPU and FPGA Compute
Serverless/Azure Functions/App Service Integration
Support for NVIDIA Triton

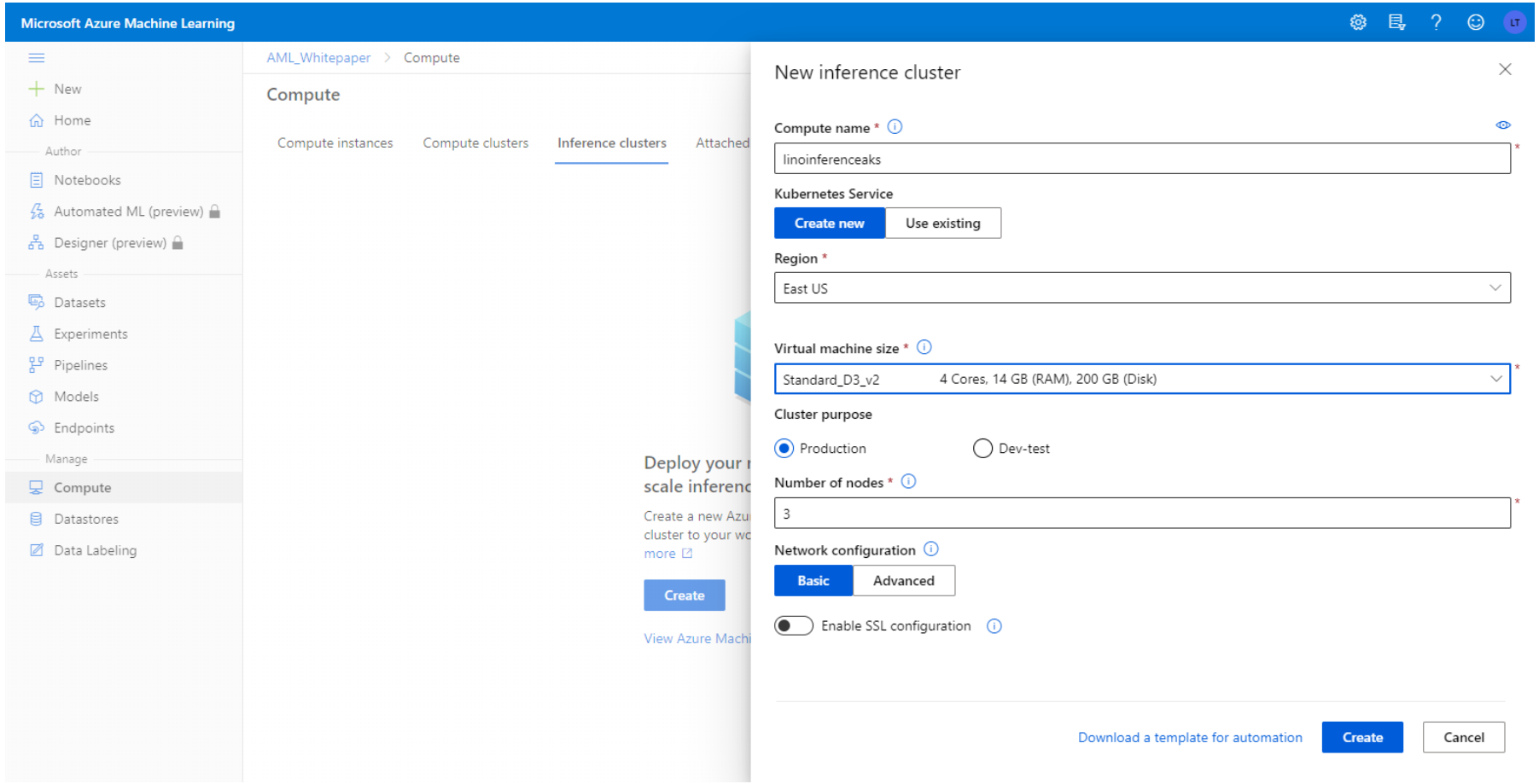
Compute Instances



Compute Clusters

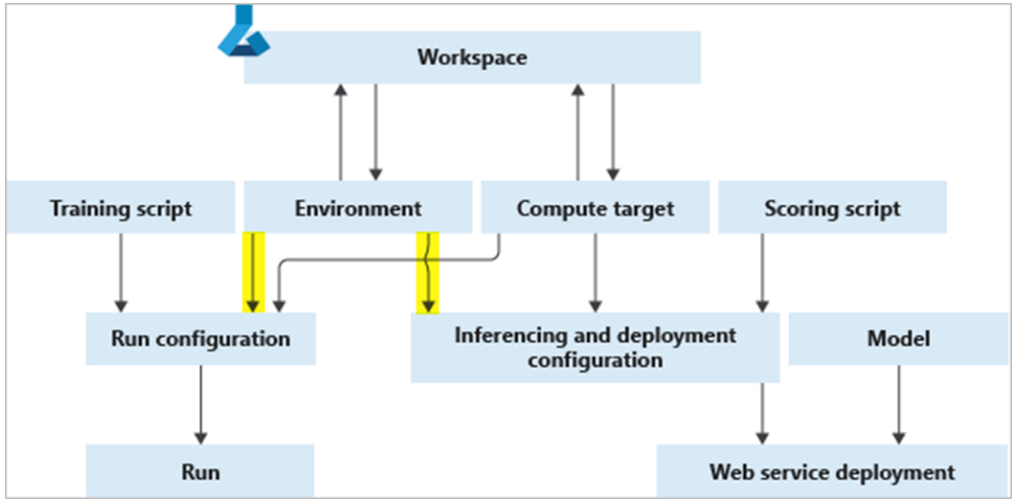


Inference Clusters



Azure Machine Learning environments

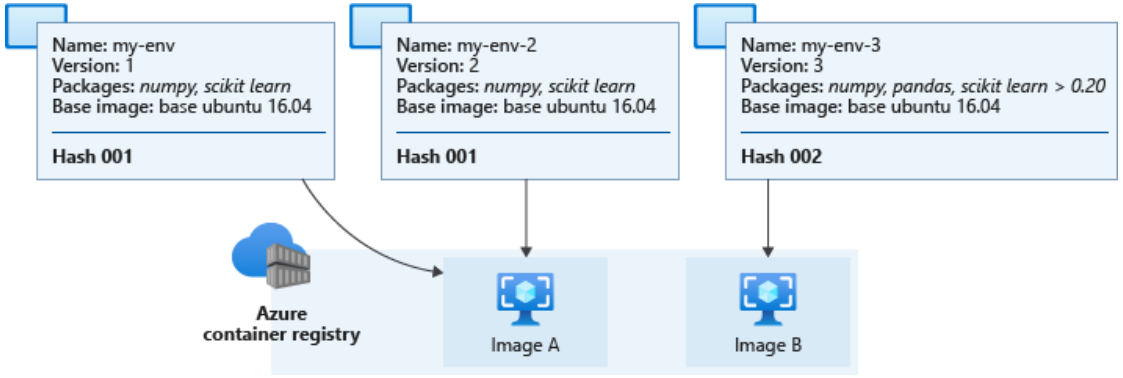
Reusable across training and scoring



Three types:

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

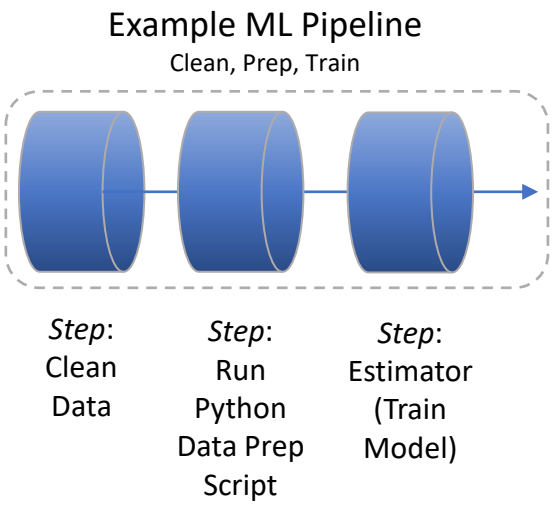
Materialized as Docker images in the workspace ACR



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

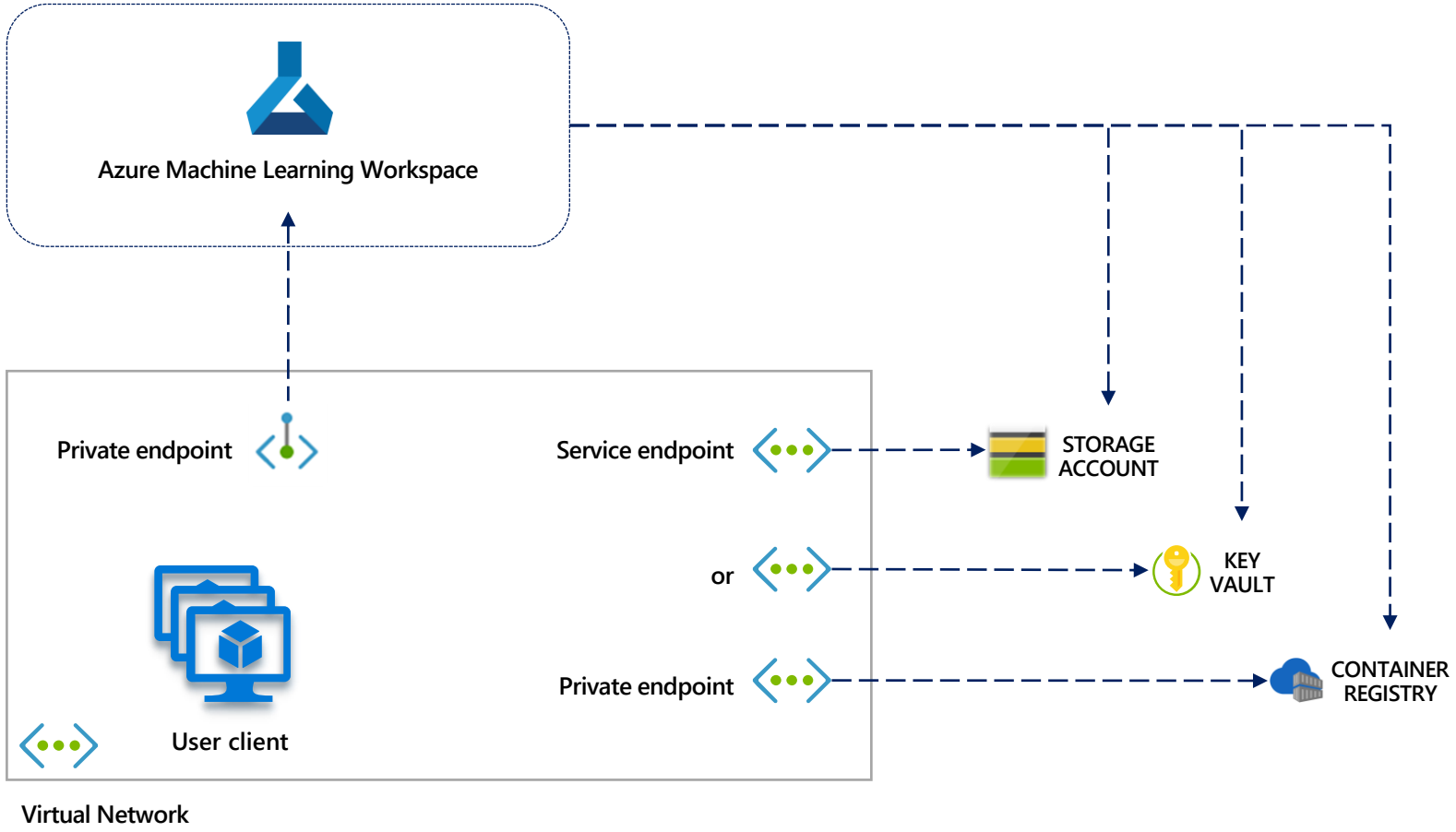
Security checklist

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 is 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.

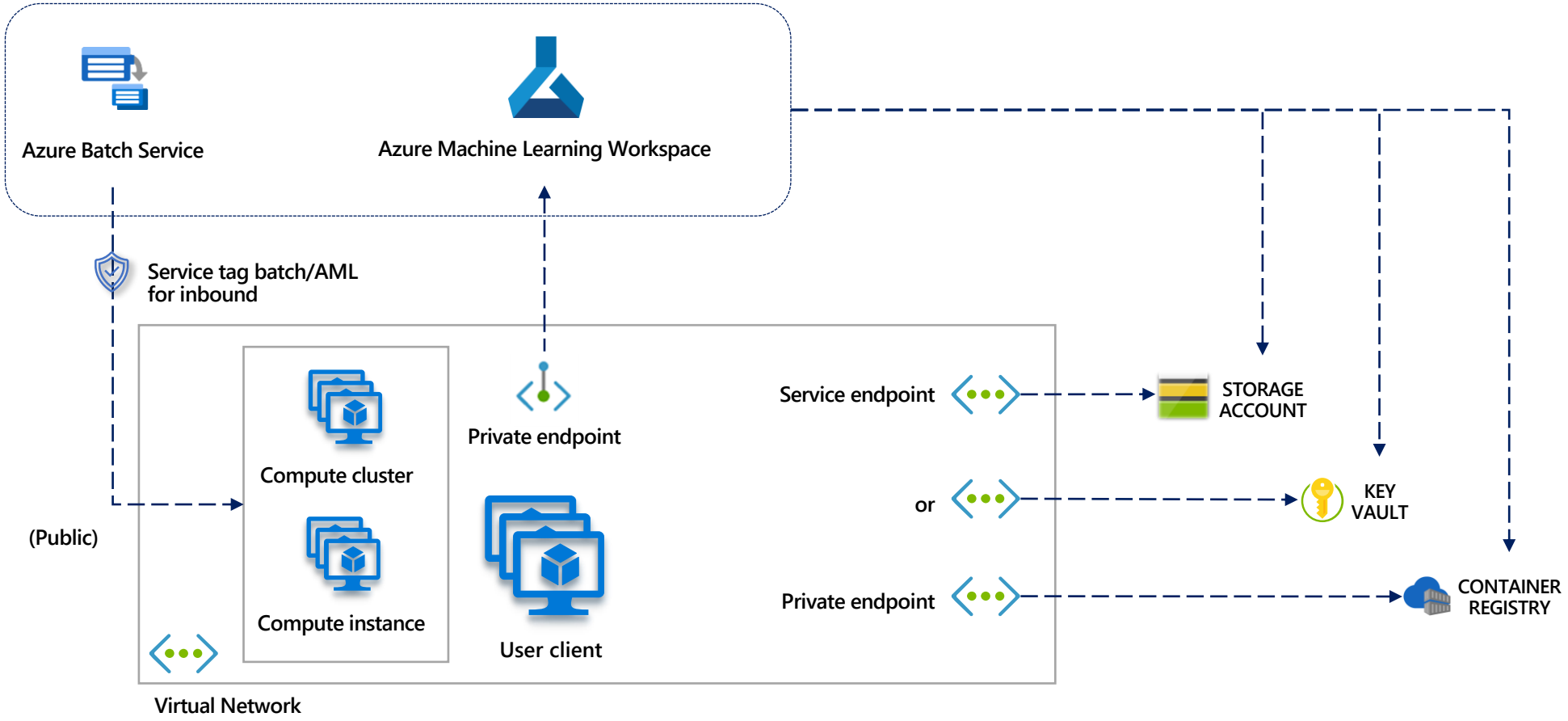
Security checklist - continued

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



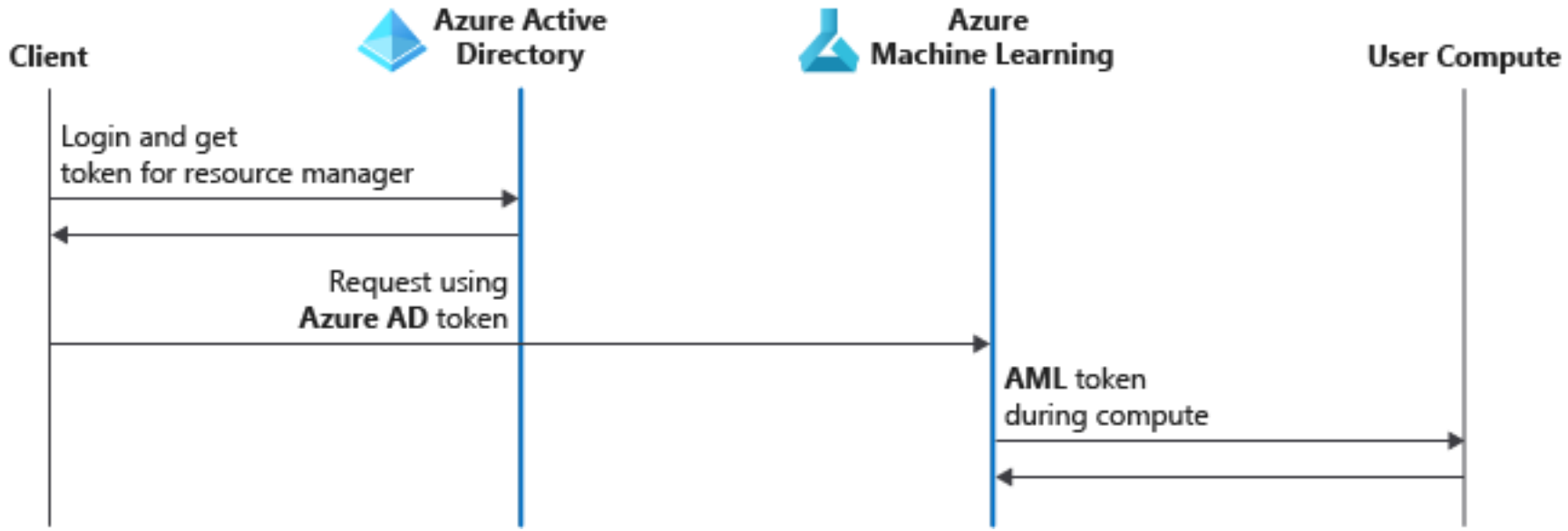
Network security - training and job submission environment



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

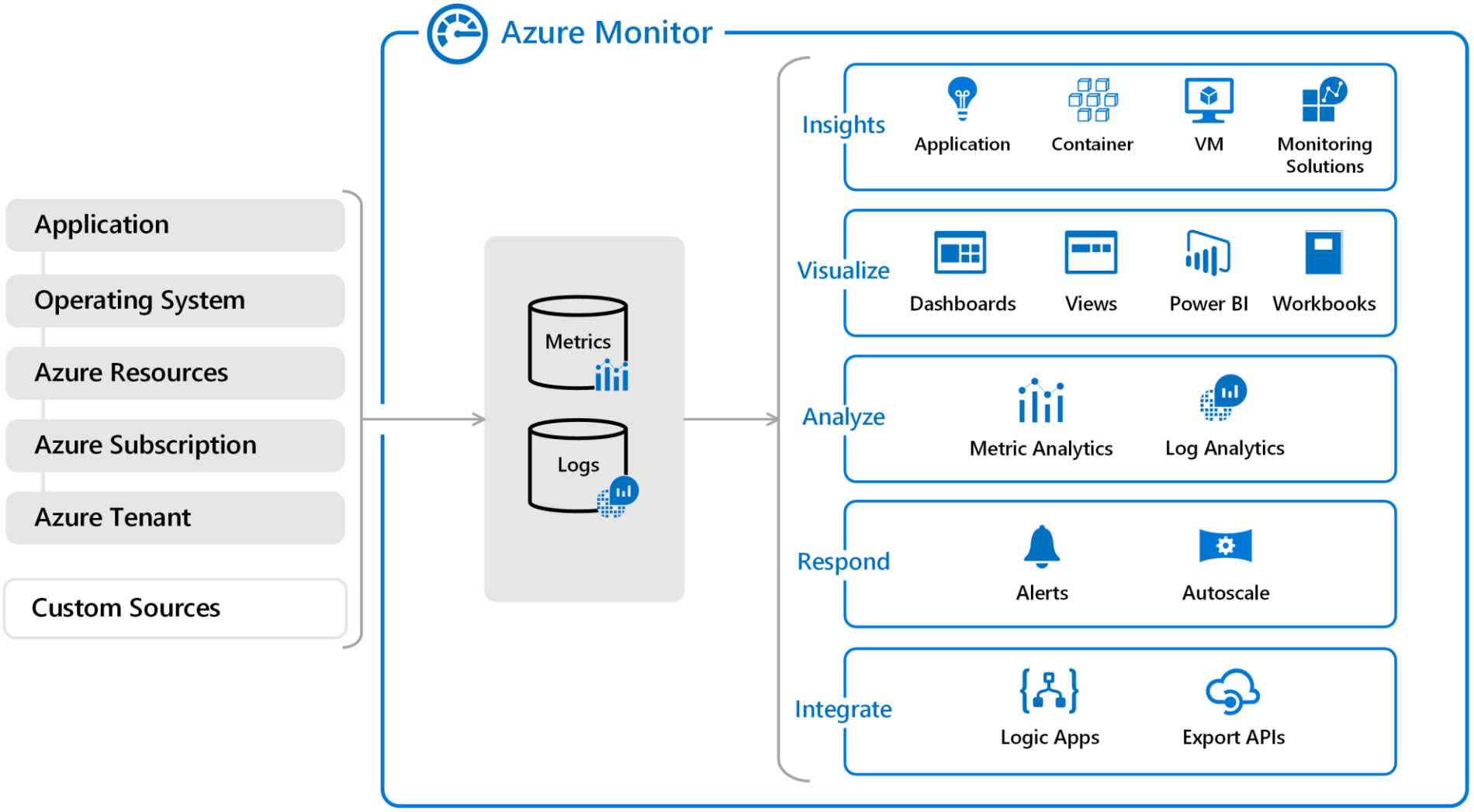
```
from azureml.core.authentication import ServicePrincipalAuthentication

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.

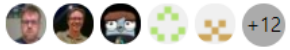
Audit - Monitoring



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

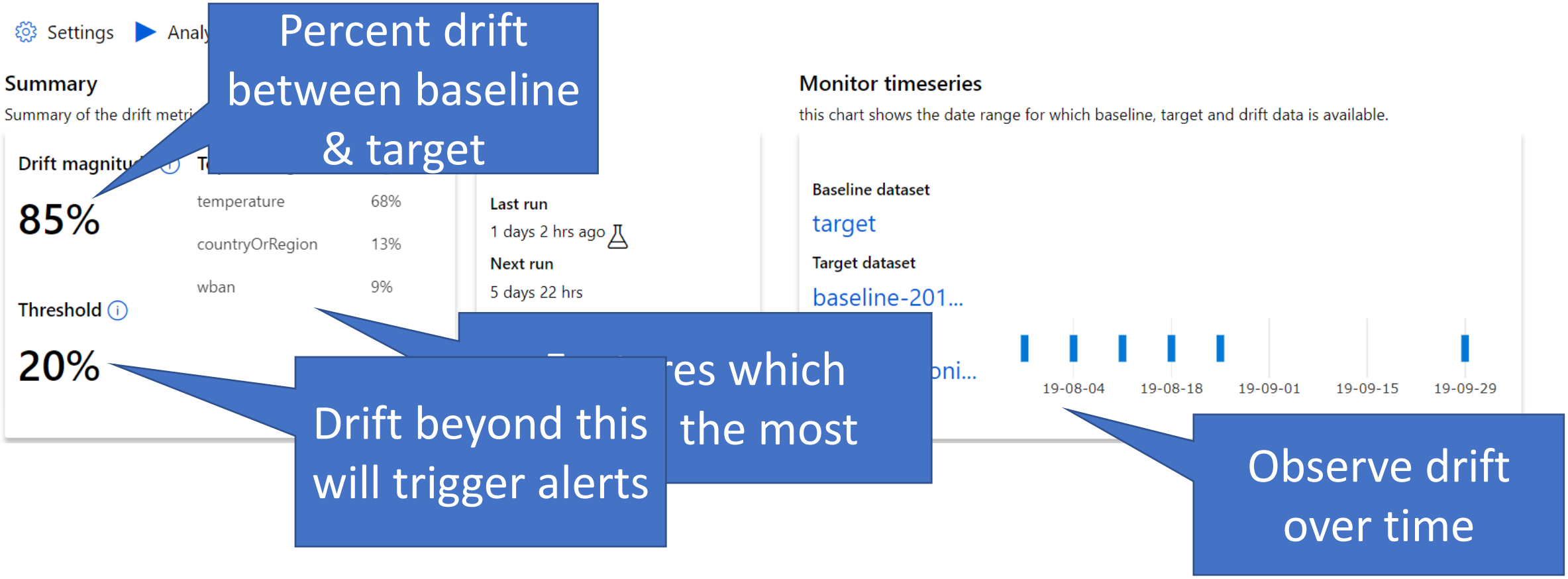
For more information, see the Azure Security Benchmark: Network Security.

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

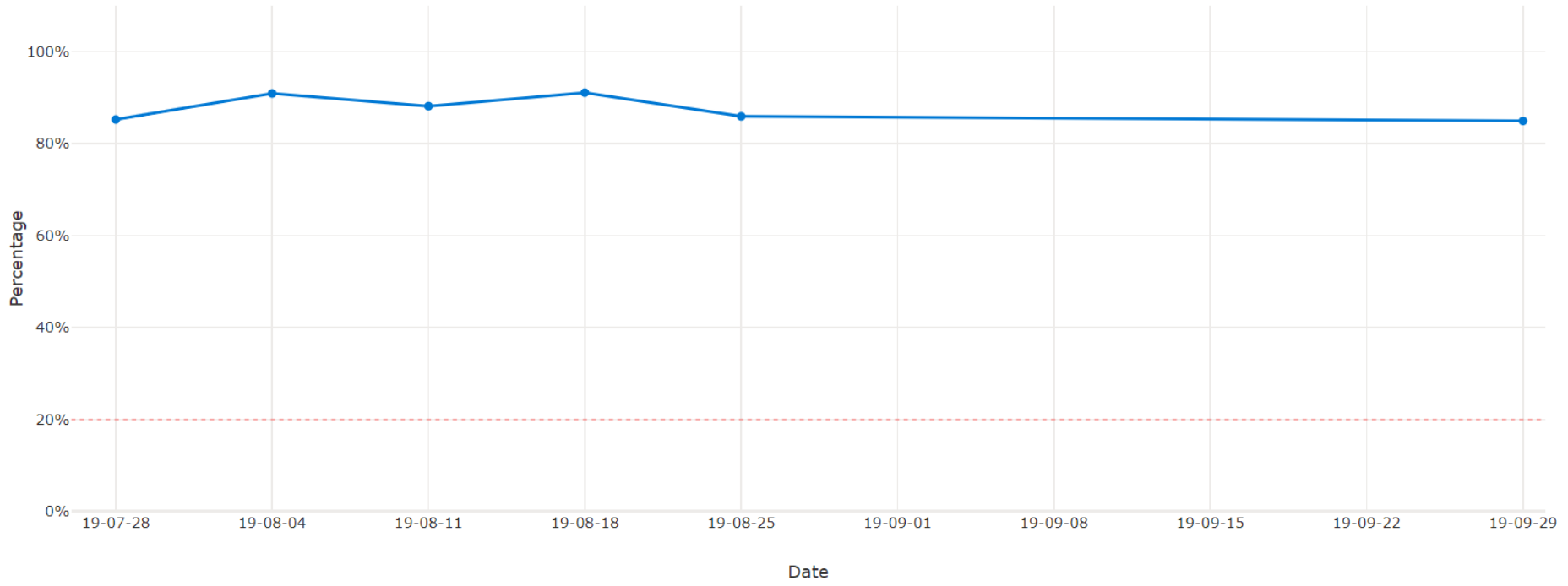
Review results in AML Studio

Start date: 7/27/2019

End date: 9/29/2019

Drift magnitude trend

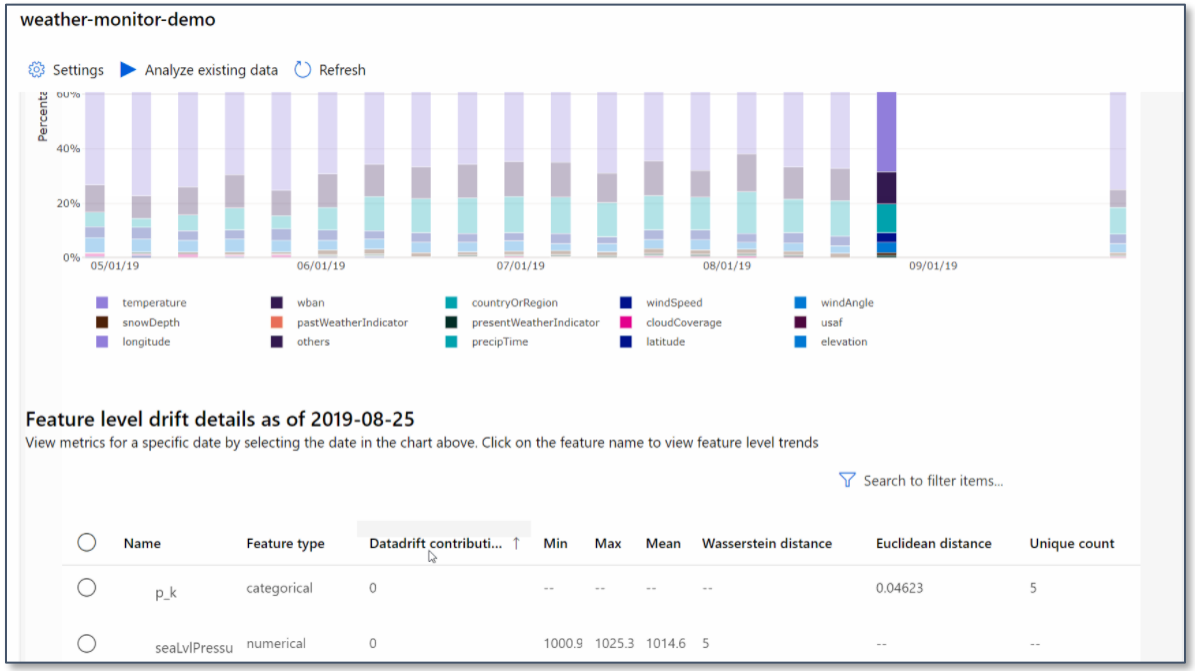
Drift magnitude metric measures the difference between target dataset for the time interval and baseline dataset. 0% implies that the target data is identical to the baseline data and 100% implies the target data is completely different from the baseline data.



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 tr

- 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/
dset = Dataset.Tabular.from_parquet_files(path
partition_format=partition_format)
dset = dset.with_timestamp_columns('da
dset = dset.register(ws, 'target')
```

Define a variable

Define a variable named “date” based on folder names

Load data using the defined partition format

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
datetime.today())
monitor = monitor.disable_schedule()
monitor = monitor.enable_schedule()
```

Backfill from January to May, 2019

Backfill from May, ow

Stop performing drift detection

Start per drift detection

Azure Machine Learning integration with Azure Synapse Analytics

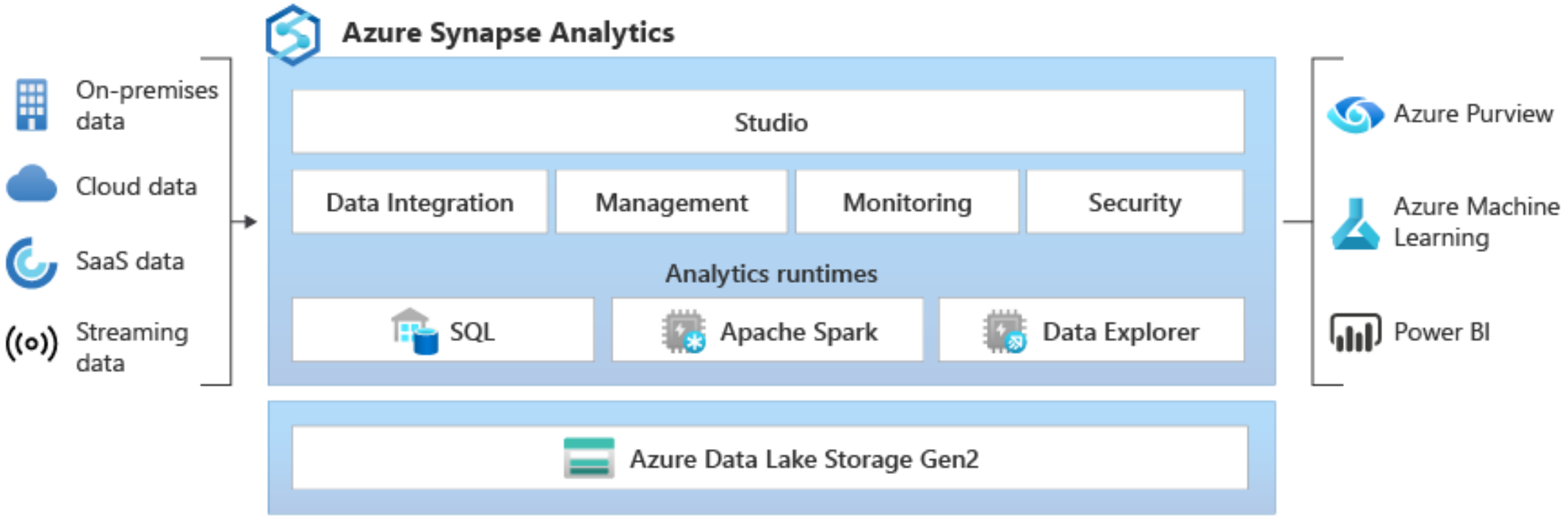
Azure Data Conference

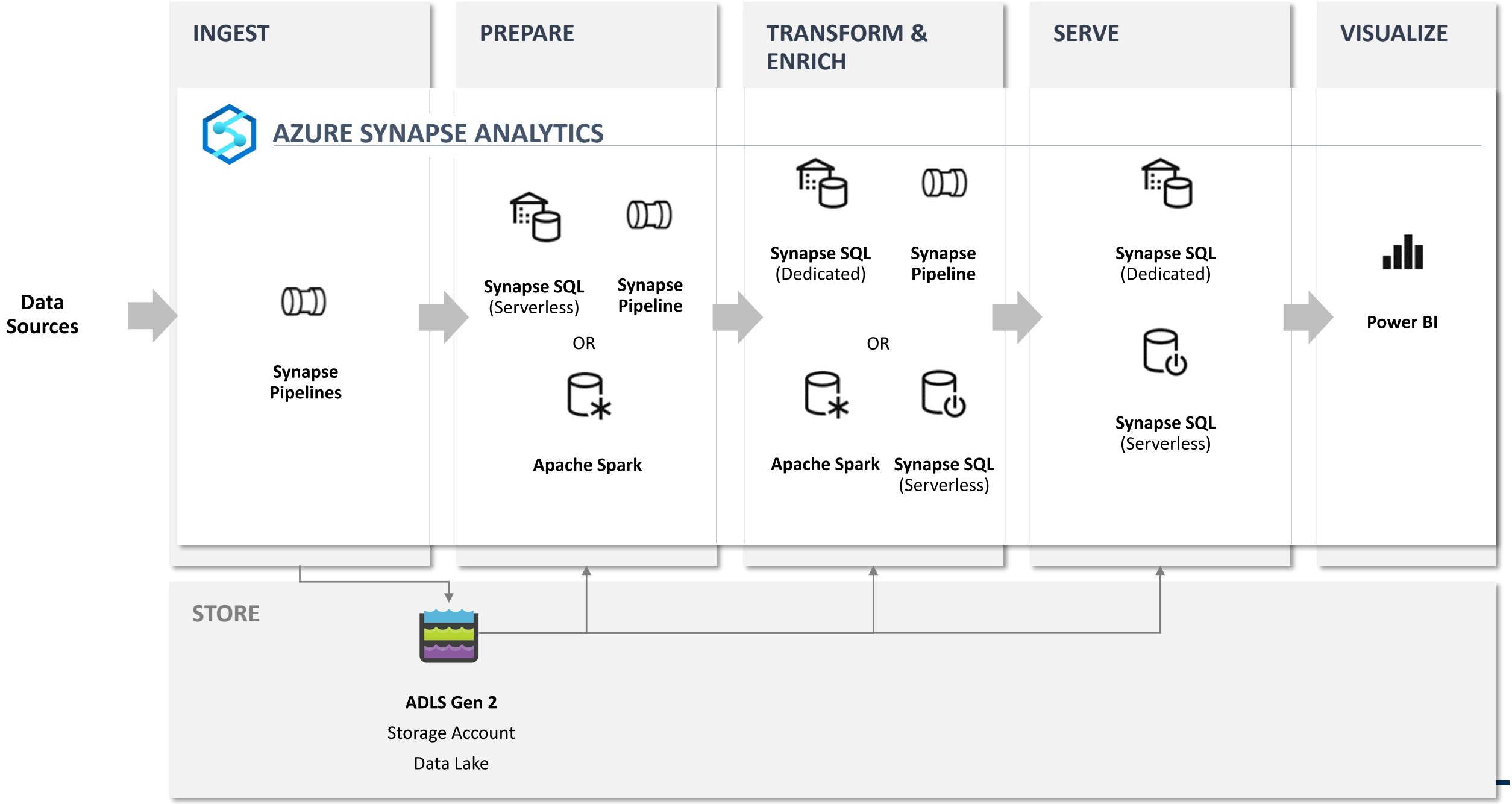
POWERED BY

Azure Data Community & Azure Data Conf

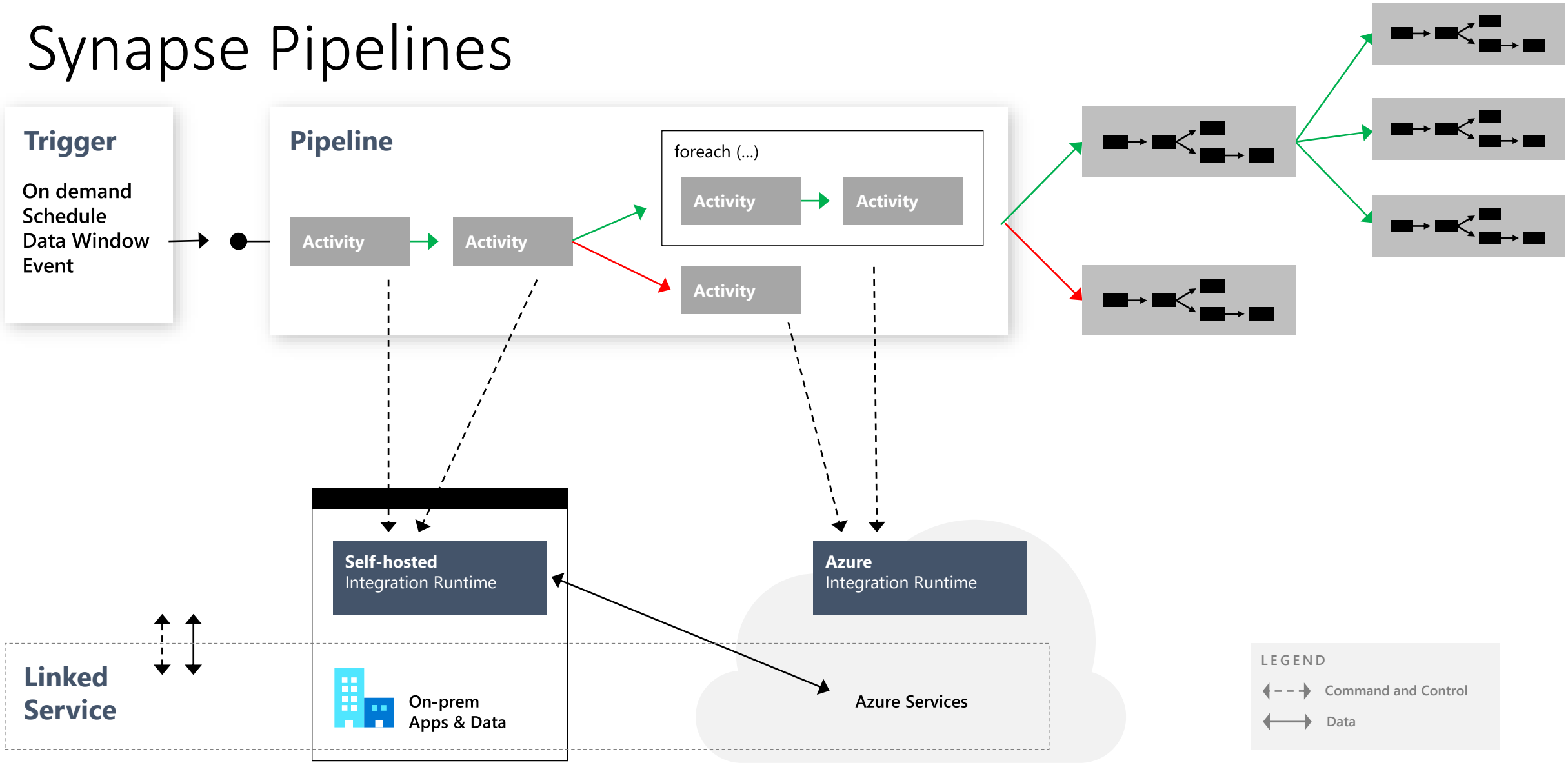
Azure Synapse Analytics

Limitless analytics service with unmatched time to insight




INGEST
PREPARE
TRANSFORM & ENRICH
SERVE
VISUALIZE
AZURE SYNAPSE ANALYTICS
Data Sources
Synapse Pipelines
Synapse SQL (Serverless)
Synapse Pipeline
OR
Apache Spark
Synapse SQL (Dedicated)
Synapse Pipeline
OR
Apache Spark
Synapse SQL (Serverless)
Synapse SQL (Dedicated)
Synapse SQL (Serverless)
Power BI
STORE
ADLS Gen 2
Storage Account
Data Lake

Synapse Pipelines

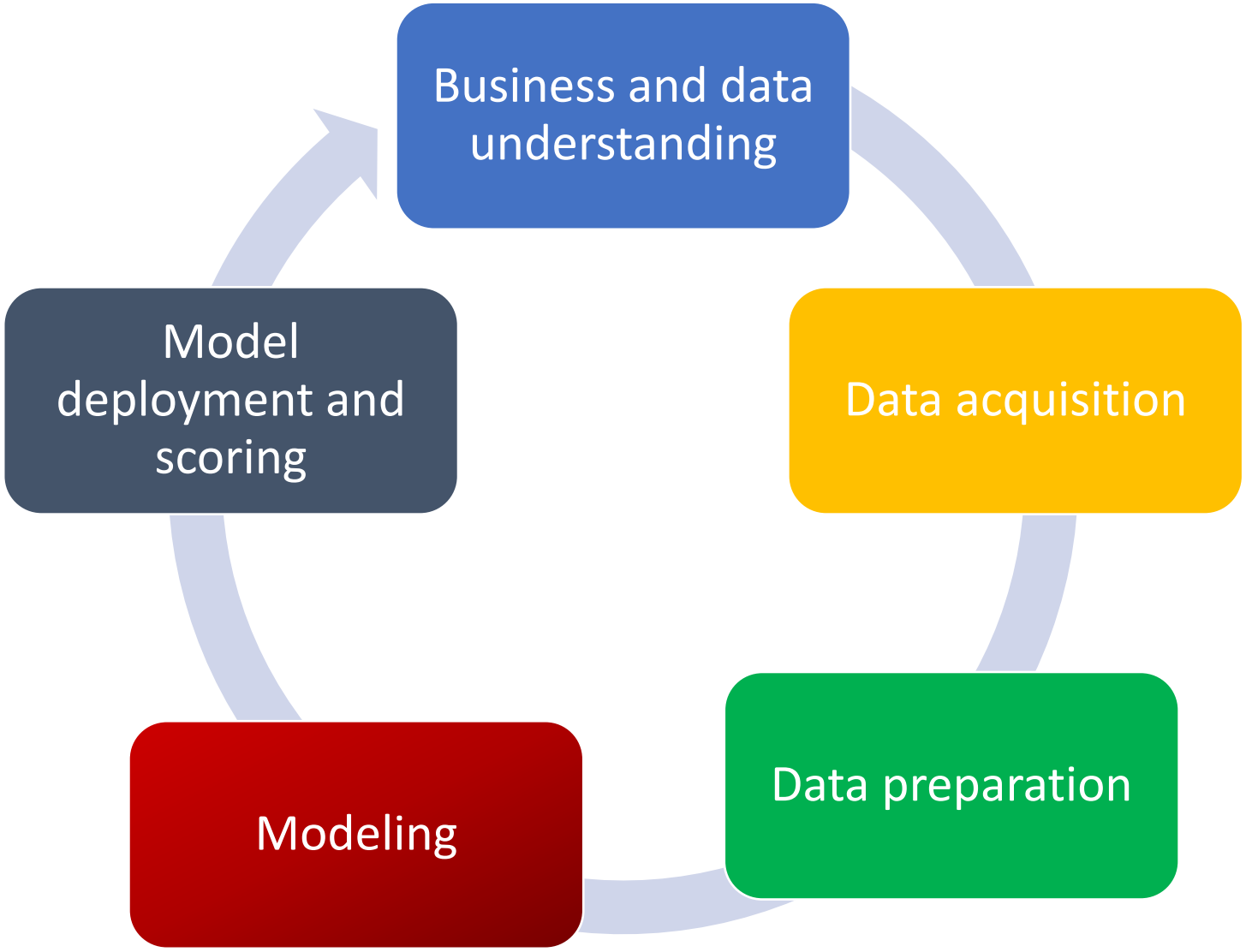


Synapse Pipelines shares codebase with Azure Data Factory

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

Classification and Regression	• Linear Models (SVMs, logistic regression, linear regression) • Naïve Bayes • Decision Trees • Ensembles of trees (Random Forest, Gradient-Boosted Trees) • Isotonic regression
Clustering	• k-means and streaming k-means • Gaussian mixture • Power iteration clustering (PIC) • Latent Dirichlet allocation (LDA)
Collaborative Filtering	• Alternating least squares (ALS)
Dimensionality Reduction	• SVD • PCA
Frequent Pattern Mining	• FP-growth • Association rules
Basic Statistics	• Summary statistics • Correlations • Stratified sampling • Hypothesis testing • Random data generation

Microsoft Machine Learning for Apache Spark

v1.0-rc

Microsoft's Open Source Contributions to Apache Spark

Distributed Machine Learning

Fast Model Deployment

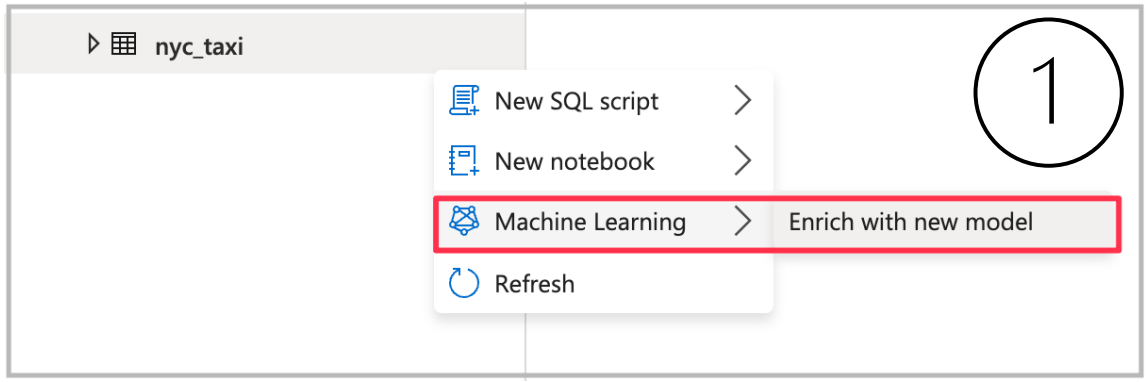
Microservice Orchestration

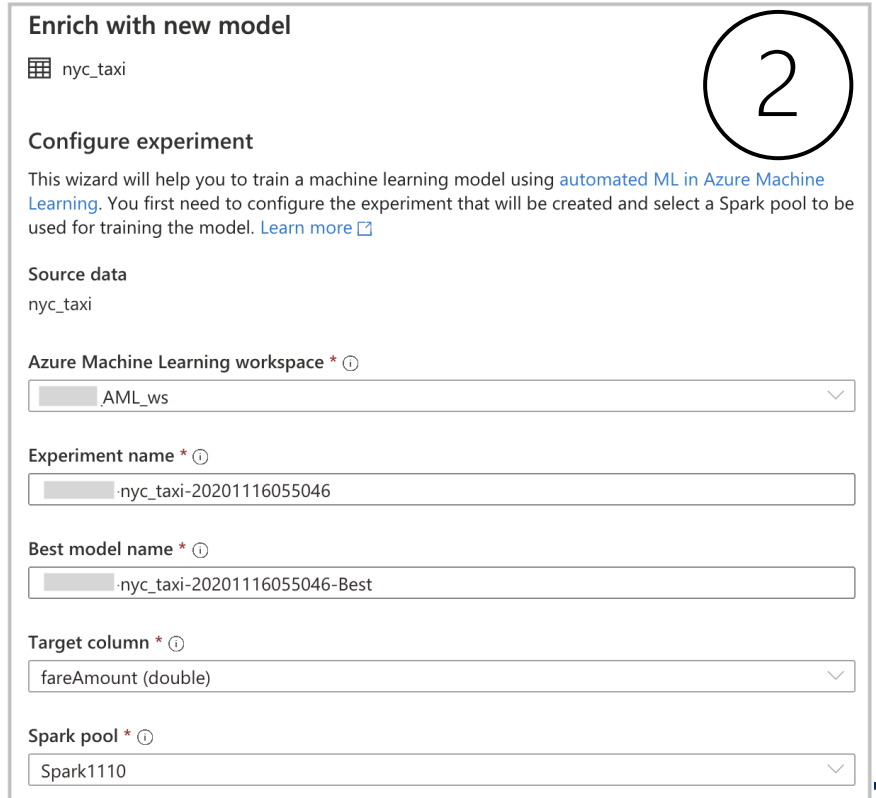
Multilingual Binding Generation

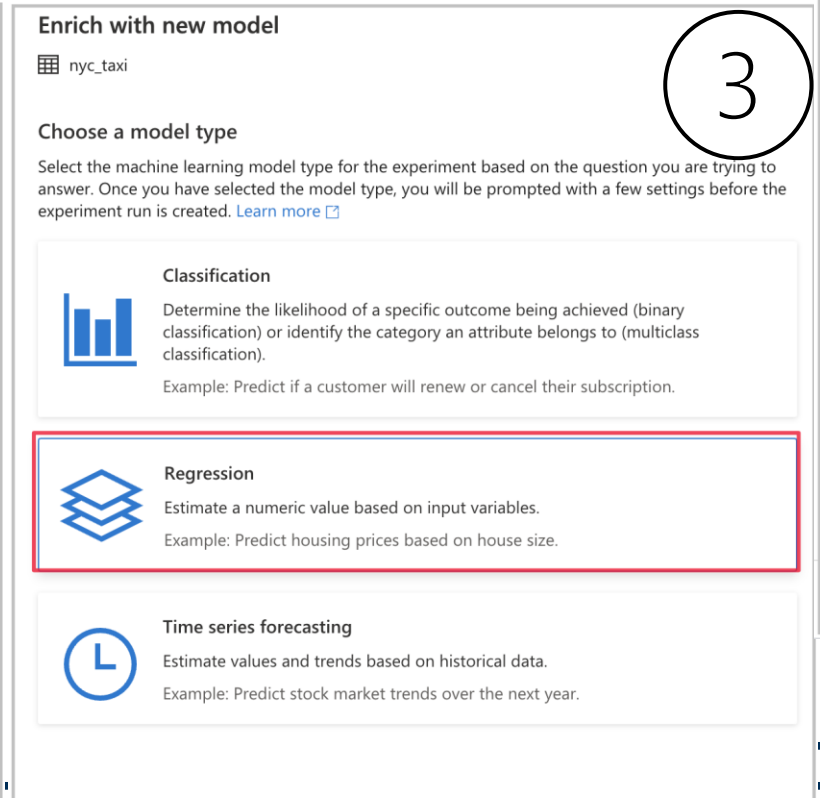
www.aka.ms/spark

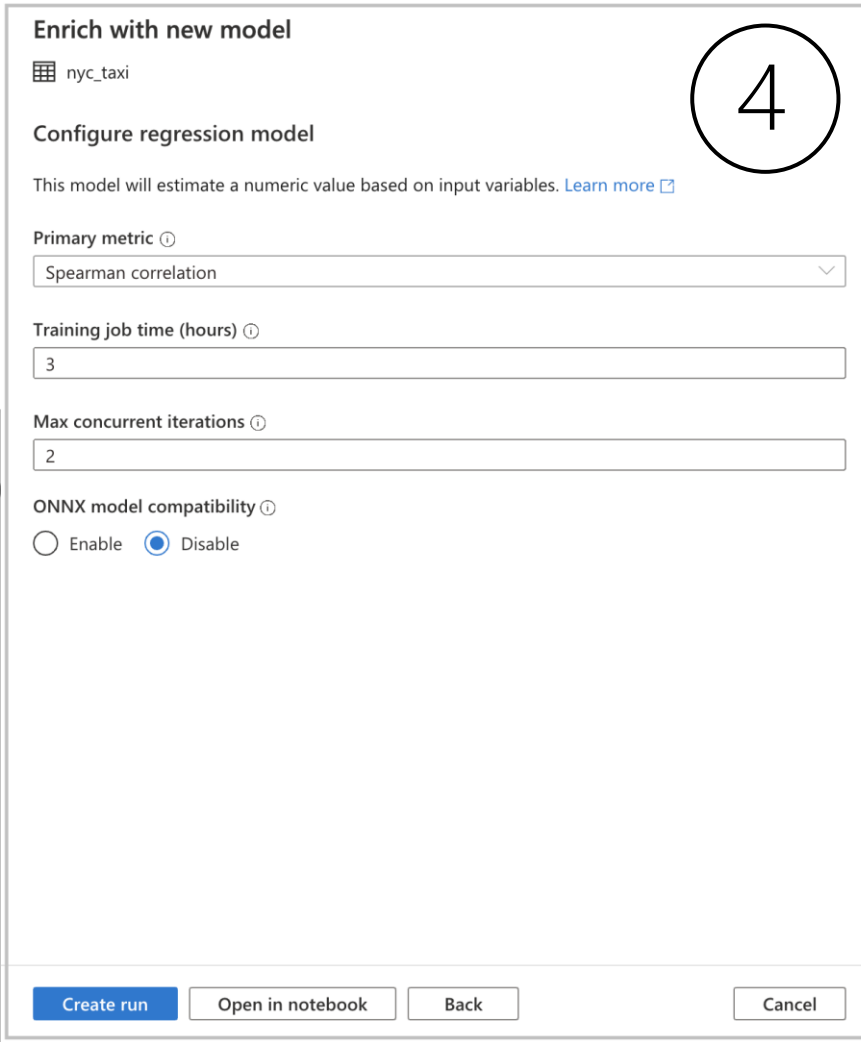
Azure/mmlspark

Modeling with Azure Machine Learning – Code Free

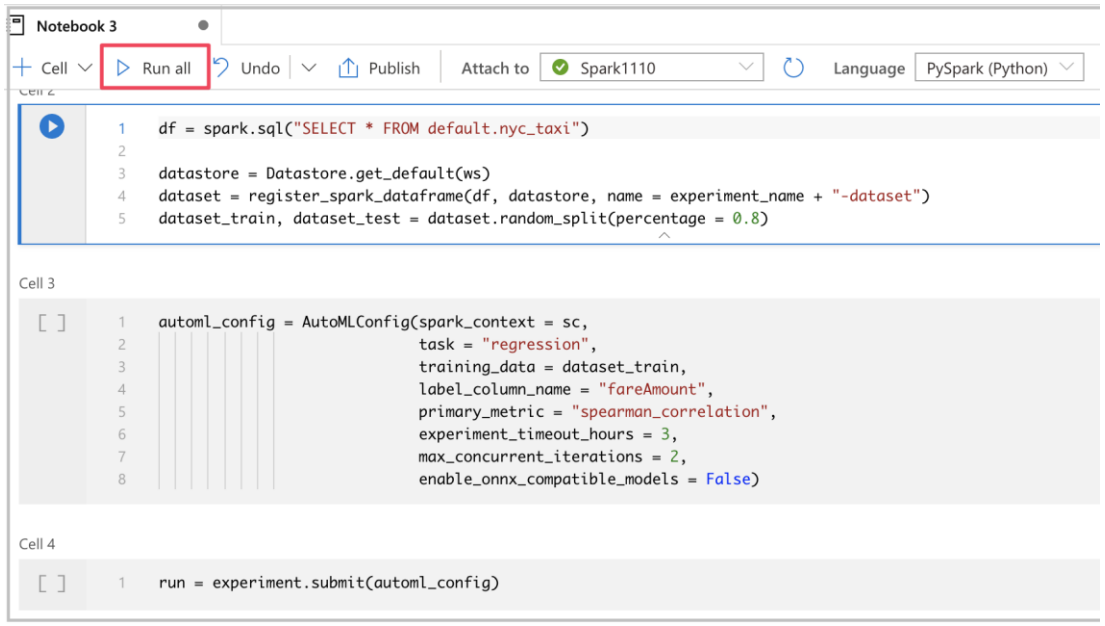


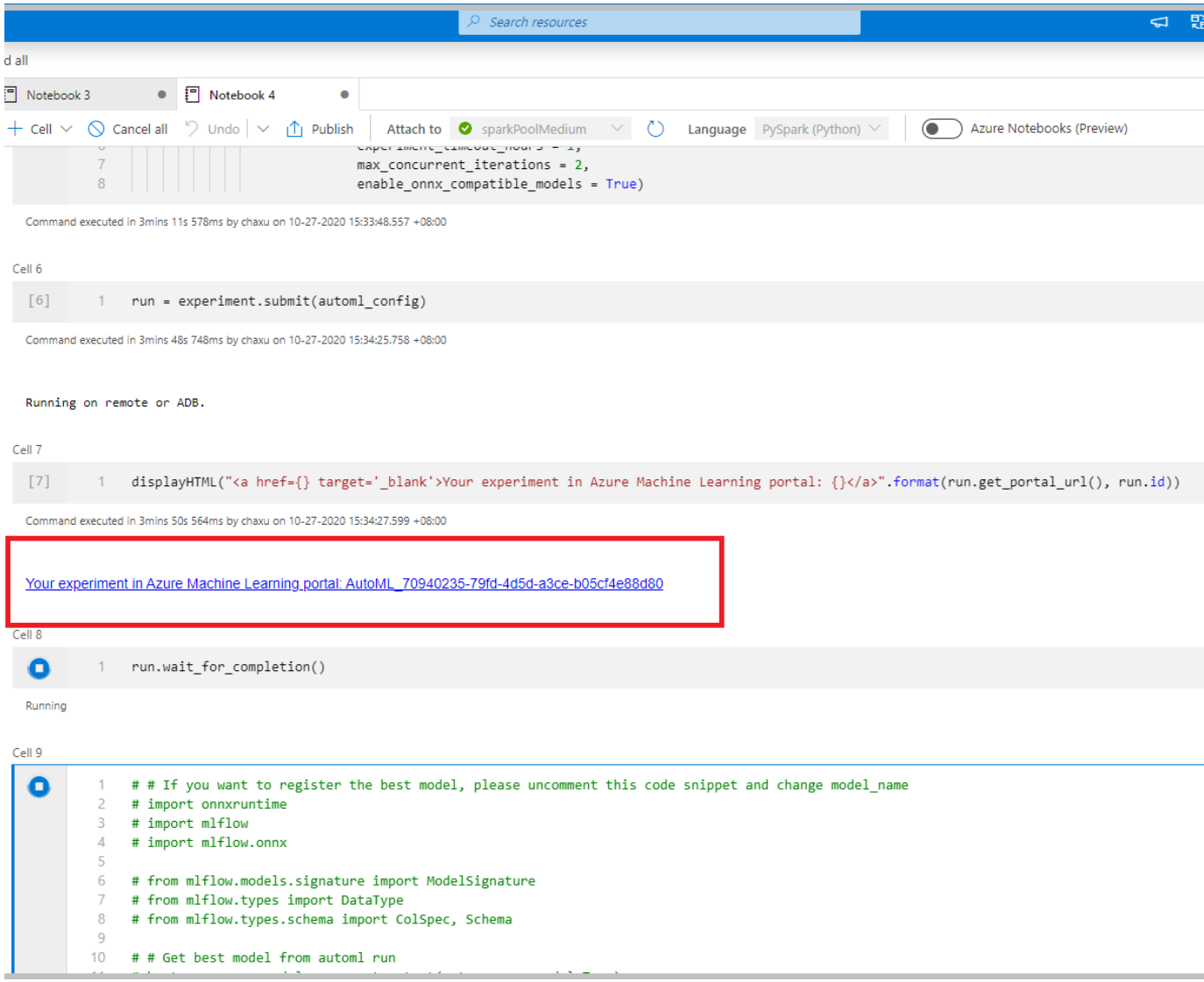




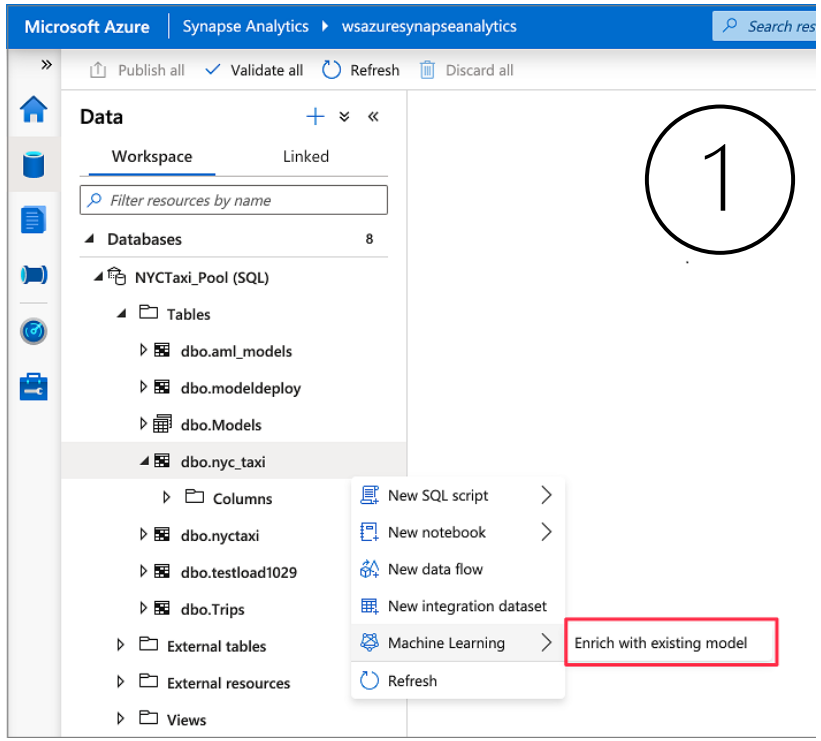


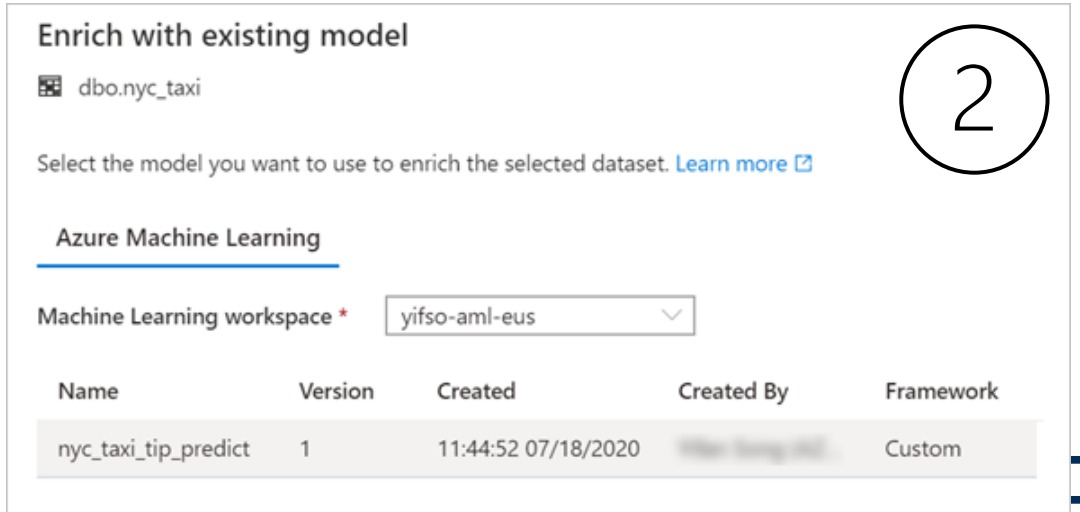
Modeling with Azure Machine Learning –

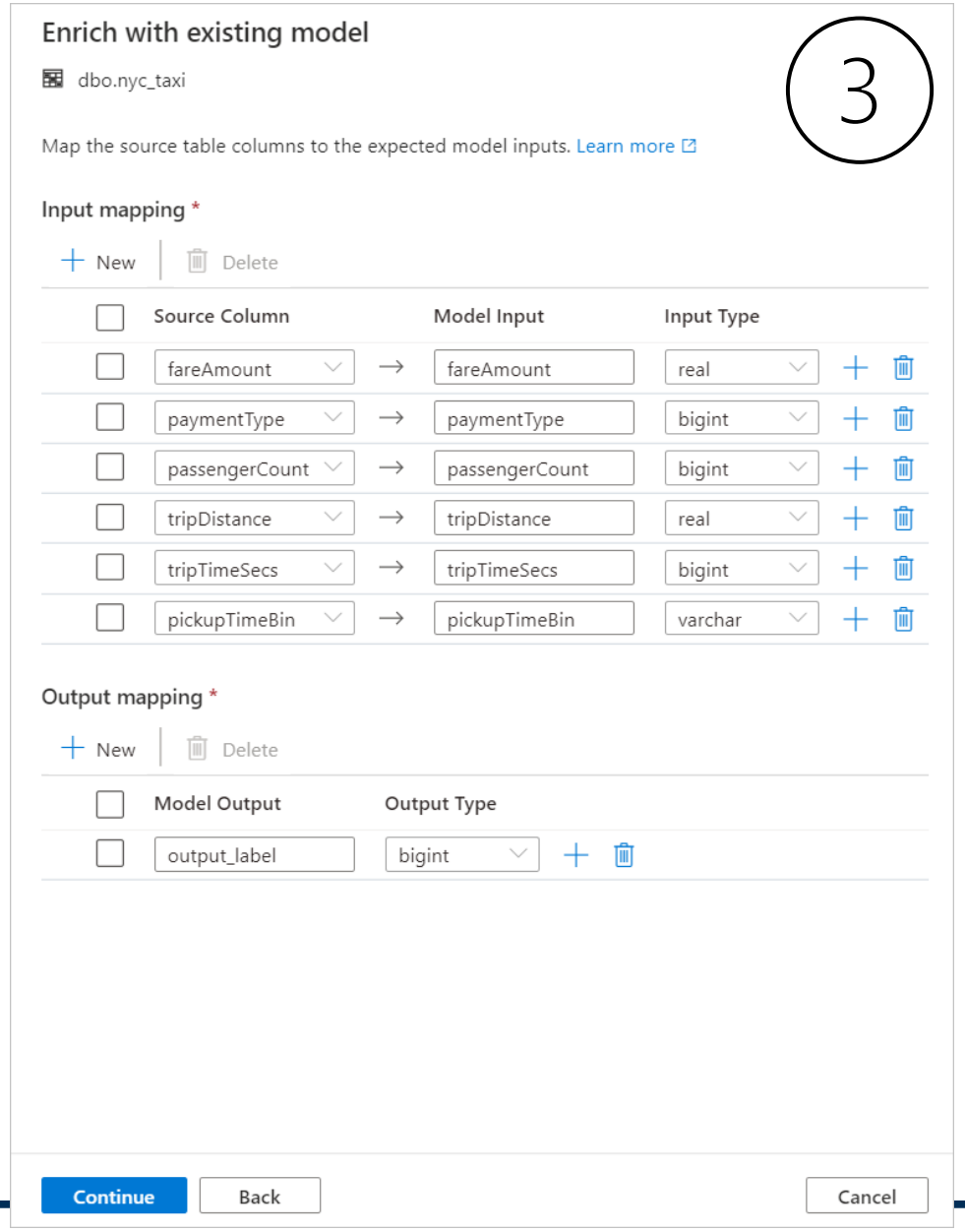




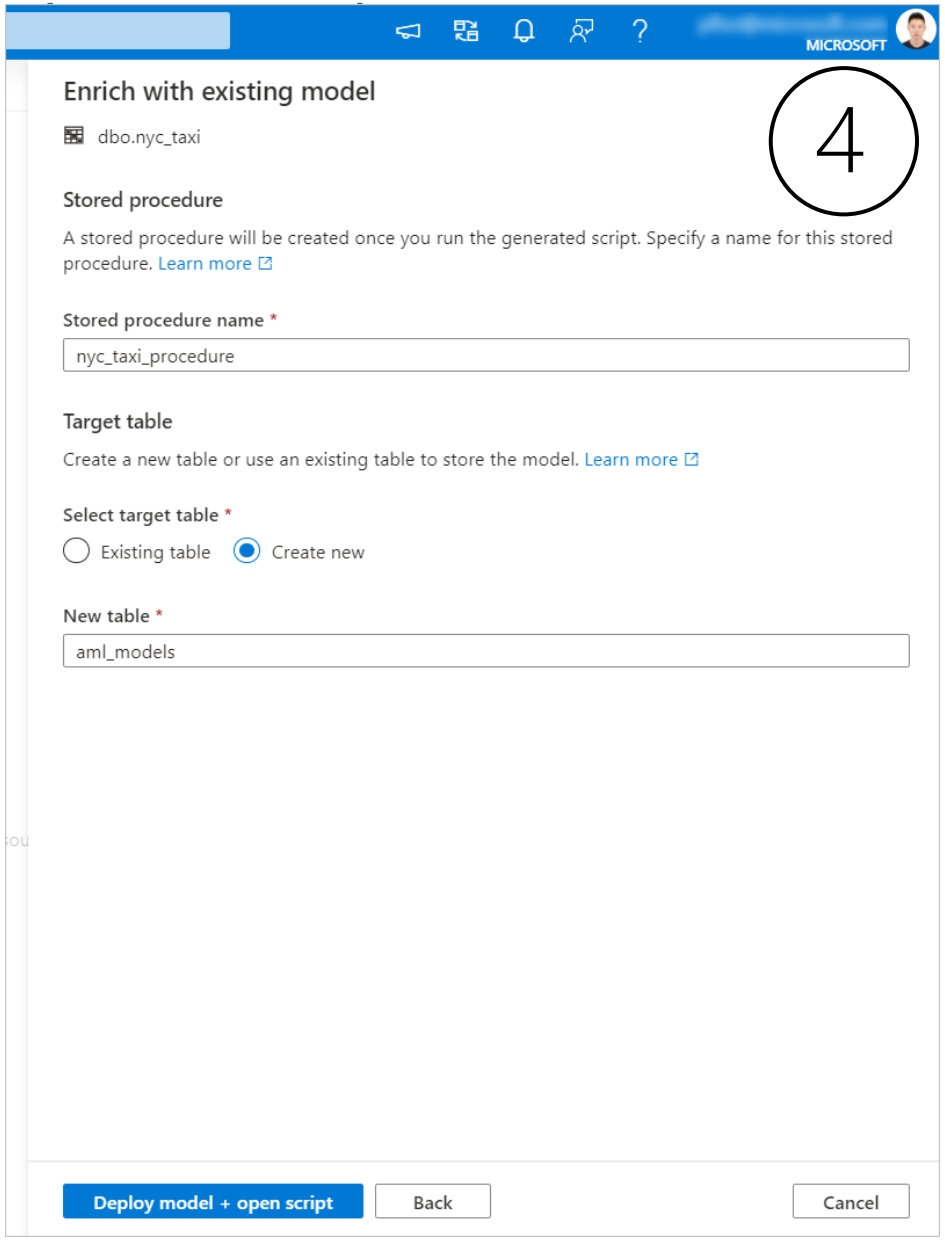
Model deployment and scoring – Code Free

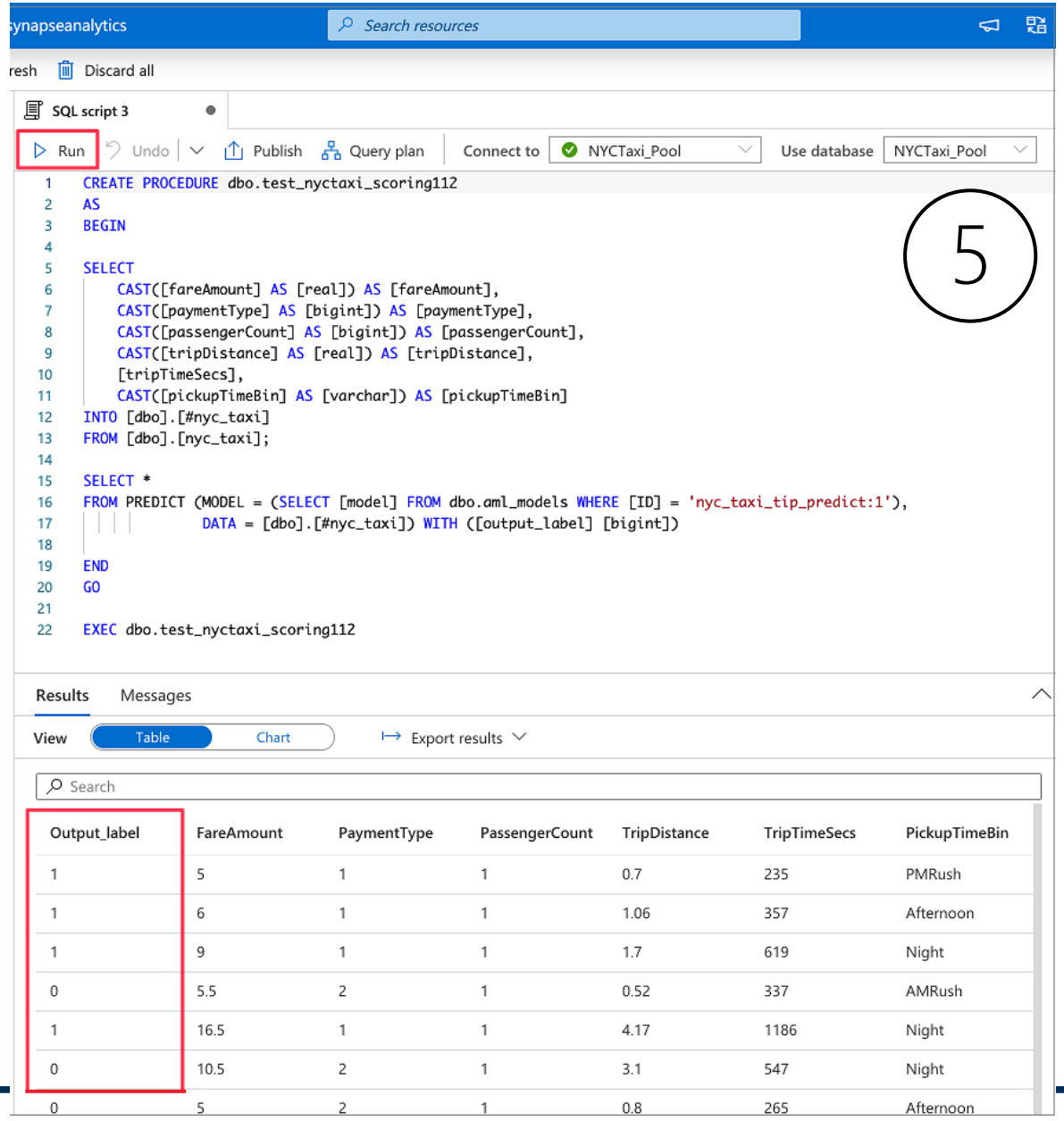






Model deployment and scoring – Code Free





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:

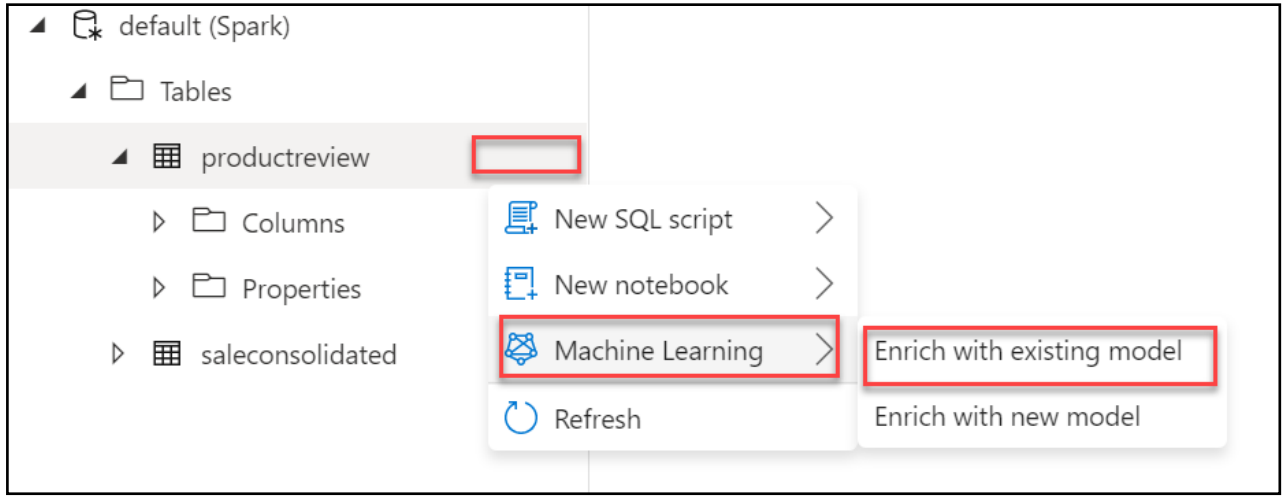
```
PREDICT
(
MODEL = @model | model_literal,
DATA = object AS <table_alias>
)
WITH ( <result_set_definition> )
<result_set_definition> ::=
{
{ column_name
data_type
}
[,...n ]
}
MODEL = @model | model_literal
```

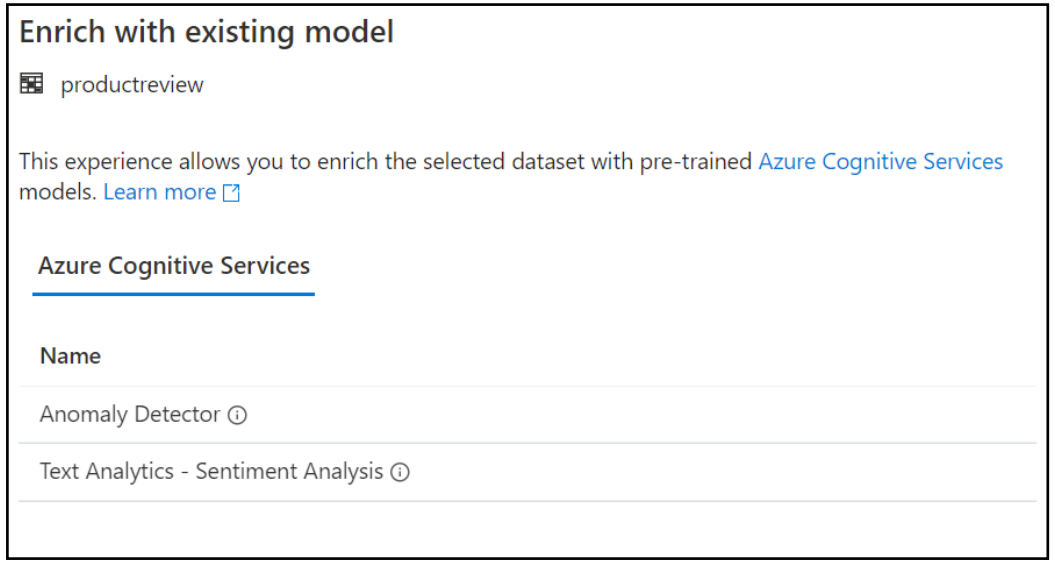
Example:

```
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

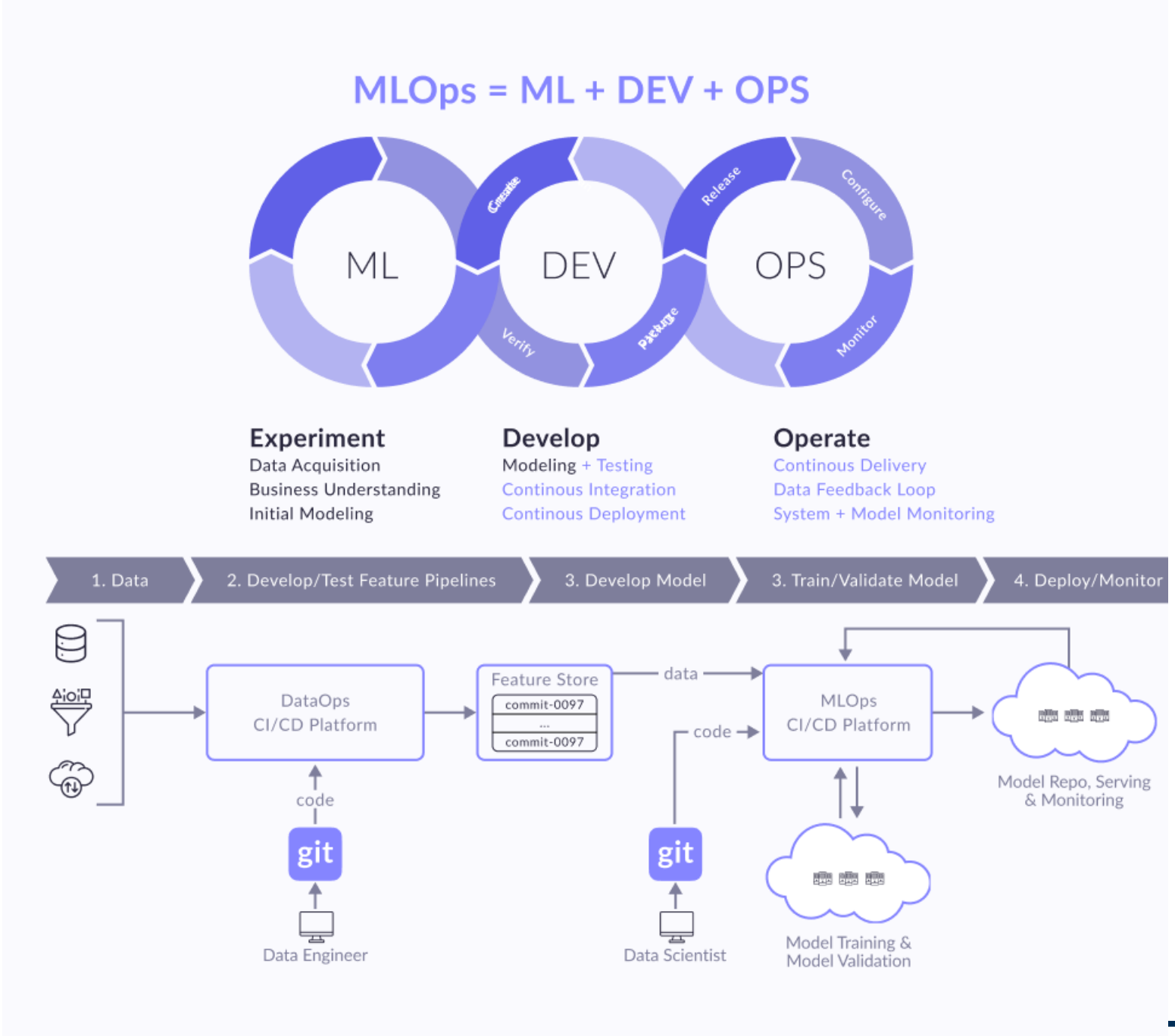
7

19

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

Definition

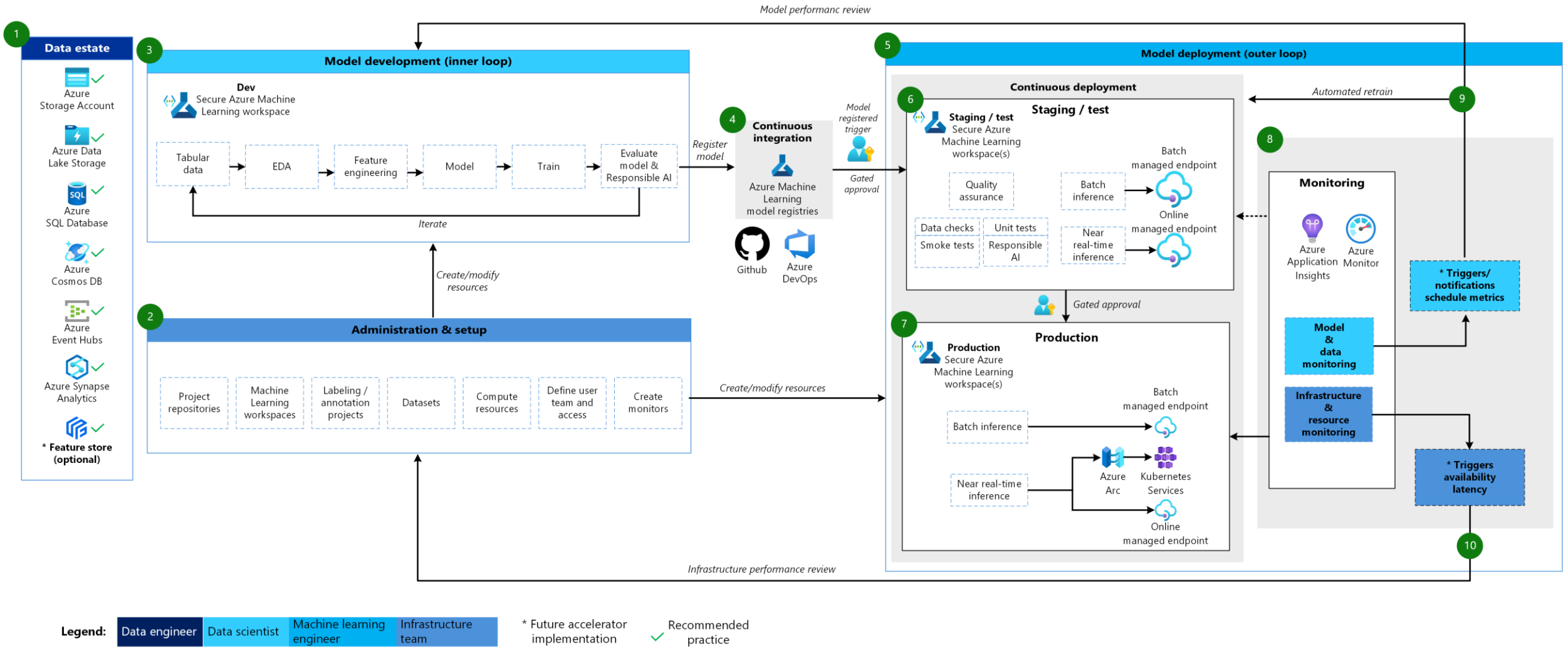
“MLOps is an approach to managing the entire machine learning lifecycle, from the development of a model to deployment and ongoing monitoring.”



What makes for good MLOps ?

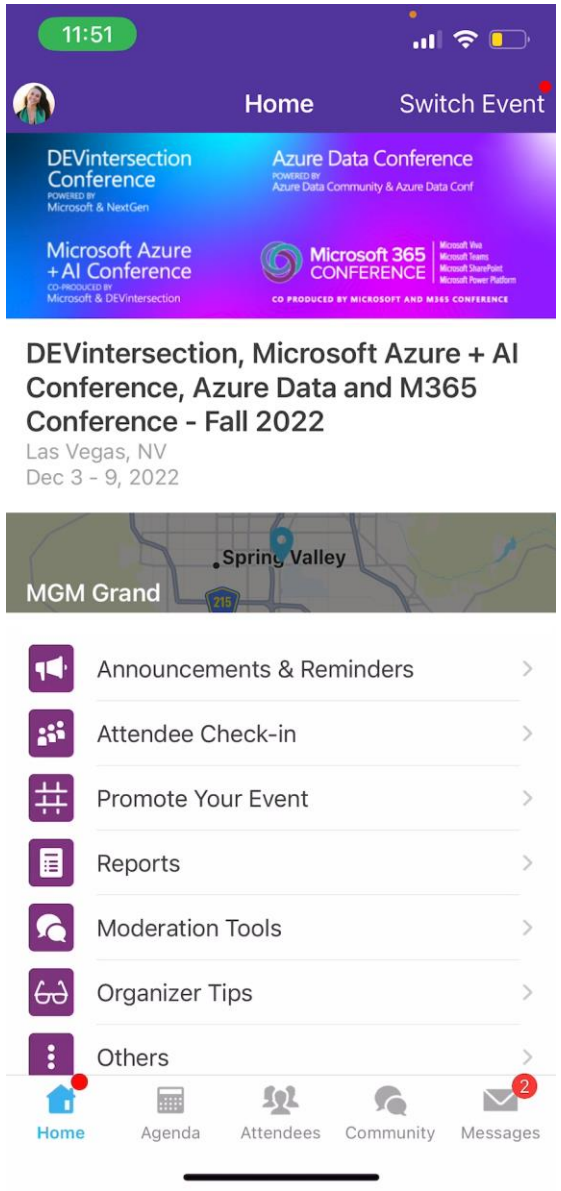
- Create reproducible machine learning pipelines
- Create reusable software environments
- Capture the governance data for the end-to-end machine learning lifecycle
- Notify and alert on events in the machine learning lifecycle
- Monitor machine learning applications for operational and machine learning-related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

MLOps in Azure:Classical machine learning architecture



Microsoft Azure

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

Microsoft Azure + AI Conference

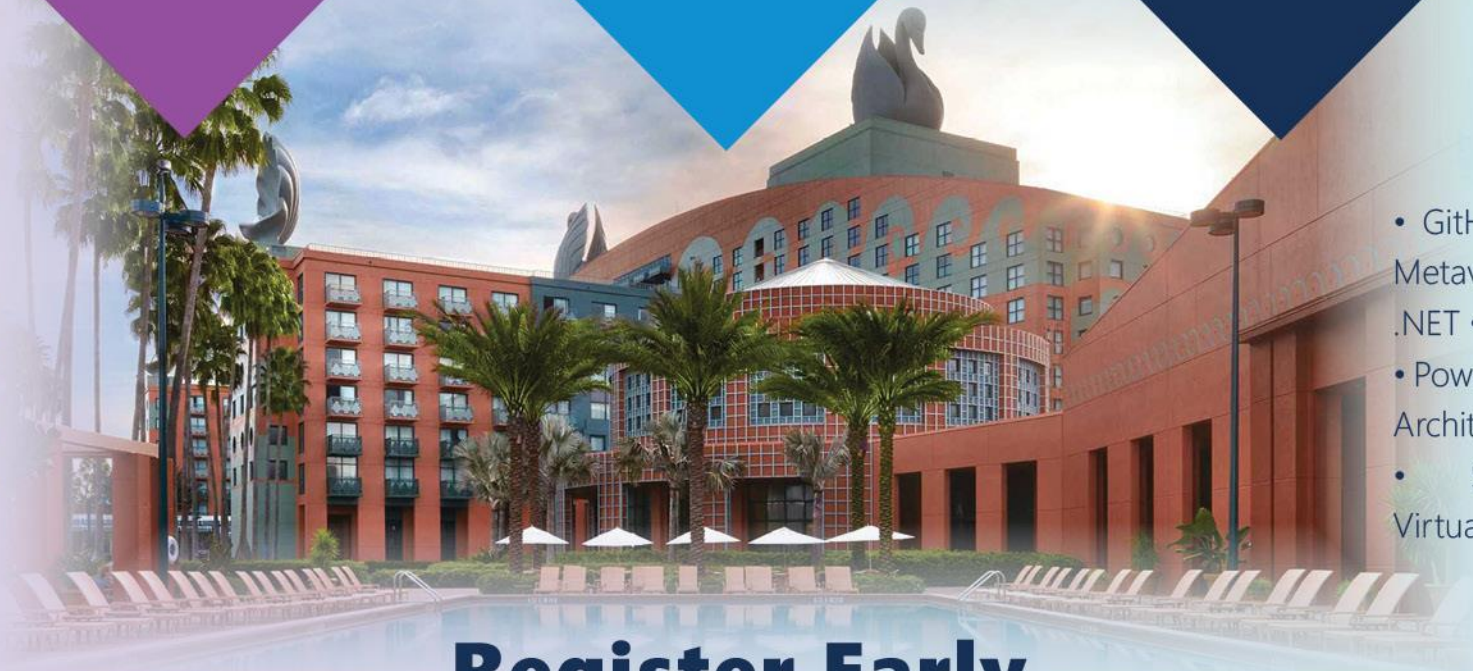
AzureAIConf.com

Azure Data Conference

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