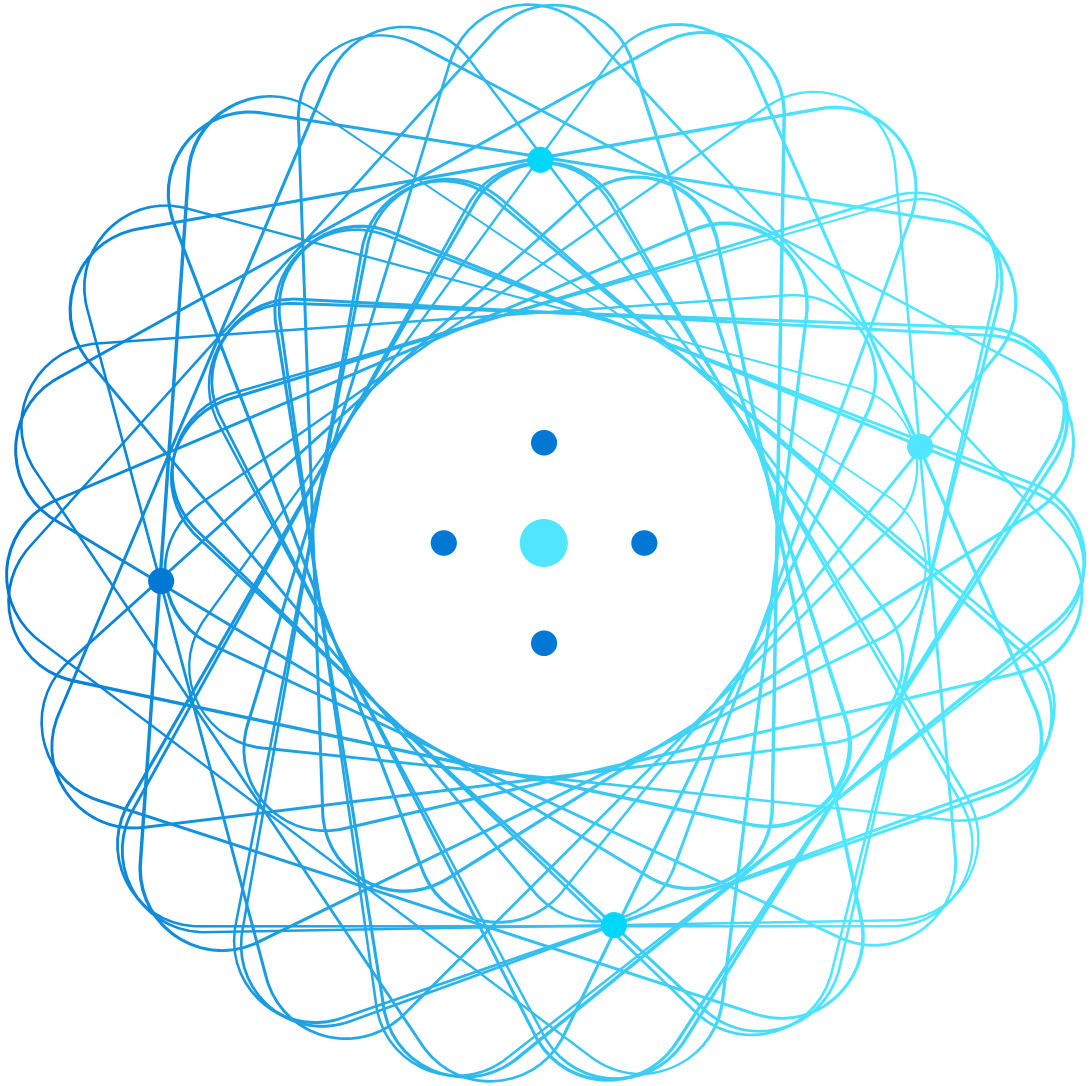
Microsoft Azure

Module 6: Orchestrating Machine Learning Workflows

Mohammed Arif

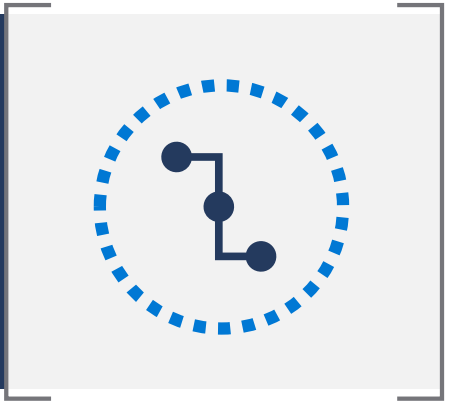


Agenda

Introduction to Pipelines

Publishing and Running Pipelines

Introduction to Pipelines



What is a Pipeline?

A workflow of machine learning tasks

- Each task is a step
- Steps may be arranged sequentially or in parallel
- Steps can be allocated to specific compute targets

An executable process

- Can be run as an experiment
- Can be published as a REST-based service

The foundation for automating ML operationalization tasks

- Automate data preparation, model training, and deployment
- Trigger based on events or schedules

Pipeline Steps

Common Step Types:

Step Class	Description
PythonScriptStep	Run a Python script
DataTransferStep	Copy data between data stores
DatabricksStep	Run a Databricks notebook, script, or JAR
AdlaStep	Run an Azure Data Lake Analytics U-SQL script
ParallelRunStep	Run a Python script as a distributed task on multiple compute nodes

```
step1 = PythonScriptStep(name='prepare_data', ...)
step2 = PythonScriptStep(name='train_model', ...)
training_pipeline = Pipeline(workspace=ws, steps=[step1,step2])
pipeline_experiment = Experiment(workspace=ws, name='training-pipeline')
pipeline_run = experiment.submit(pipeline_experiment)
```

Passing Data Between Steps

Use a OutputFileDatasetConfig object:

- Defines a data reference for an intermediary data store
- Pass as script argument *and* step input/output
- Creates flow dependency between steps

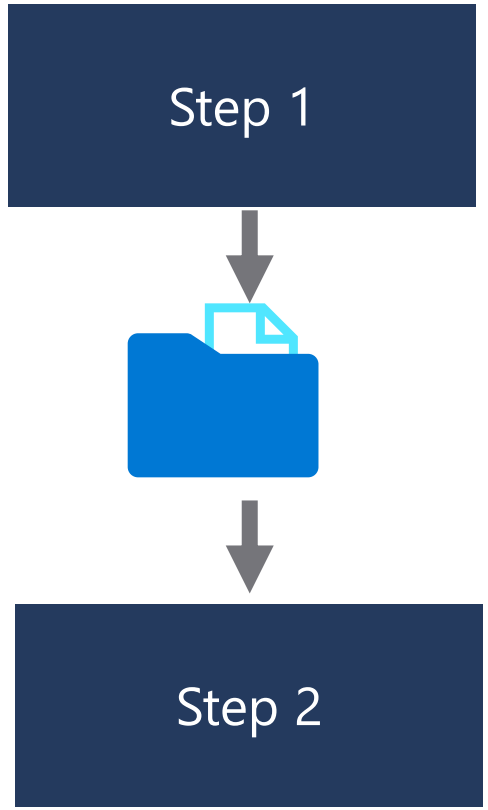
```
data_store = ws.get_default_datastore()
prepped = OutputFileDatasetConfig ('prepped_data')

step1 = PythonScriptStep(name='prepare data',
           arguments=['--output-data', prepped],
           ...)

step2 = PythonScriptStep(name='train model',
           arguments=['--input-data', prepped.as_input()],
           ...)
```

OutputFileDatasetConfig output ...

OutputFileDatasetConfig input



Pipeline Step Reuse

Reuse output without re-running the step

Control this behavior with the **allow_reuse** parameter

```
step1 = PythonScriptStep(name='prepare data', arguments = ['--folder', prepped],
                         outputs=[prepped], allow_reuse=True, ...)
```

Reuse cached step output if unchanged

Force all steps to re-run:

Use the **regenerate_outputs** parameter when submitting the experiment

```
pipeline_run = experiment.submit(pipeline_experiment, regenerate_outputs=True)
```

Override step reuse

Publishing and Running Pipelines

Pipeline Endpoints

Publish a pipeline to create a REST endpoint

```
published_pipeline = pipeline_run.publish(name='training_pipeline',
                                          description='Model training pipeline',
                                          version='1.0')
```

Post a JSON request to initiate a pipeline

- Requires an authorization header
- Returns a run ID

```
import requests
response = requests.post(rest_endpoint,
                         headers=auth_header,
                         json={"ExperimentName": "run training pipeline"})
run_id = response.json()["Id"]
```

Pipeline Parameters

Parameterize a pipeline before publishing

Increases flexibility by allowing variable input

```
reg_param = PipelineParameter(name='reg_rate', default_value=0.01)
...
step2 = PythonScriptStep(name='train model',
                         estimator_entry_script_arguments=['--reg', reg_param], ...)
...
published_pipeline = pipeline_run.publish(name='model training pipeline',
                                          description='trains a model with reg parameter',
                                          version='2.0')
```

Pass parameters in the JSON request

```
response = requests.post(rest_endpoint,
                         headers=auth_header,
                         json={"ExperimentName": "run training pipeline",
                               "ParameterAssignments": {"reg_rate": 0.1}})
```

Scheduling Pipelines

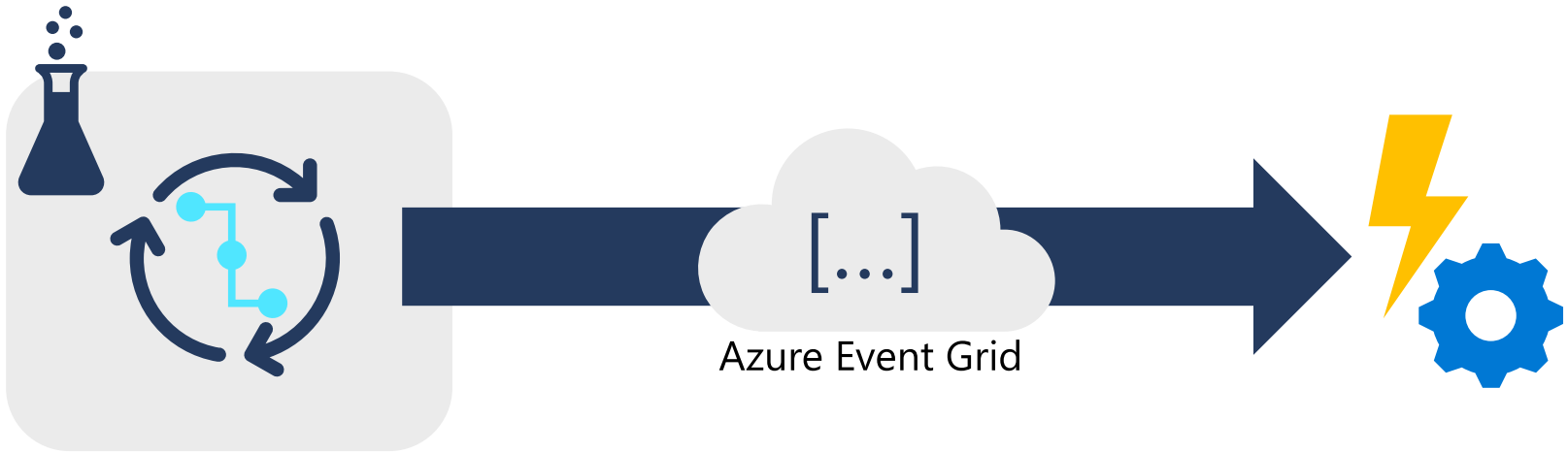
Schedule pipeline runs based on time

```
daily = ScheduleRecurrence(frequency='Day', interval=1)
pipeline_schedule = Schedule.create(ws, name='Daily Training',
                                    description='trains model every day',
                                    pipeline_id=published_pipeline_id,
                                    experiment_name='Training-Pipeline',
                                    recurrence=daily)
```

Trigger pipeline runs when data changes

```
training_datastore = Datastore(workspace=ws, name='blob_data')
pipeline_schedule = Schedule.create(ws, name='Reactive Training',
                                    description='trains model on data change',
                                    pipeline_id=published_pipeline_id,
                                    experiment_name='Training-Pipeline',
                                    datastore=training_datastore,
                                    path_on_datastore='data/training')
```

Event-Driven Workflows



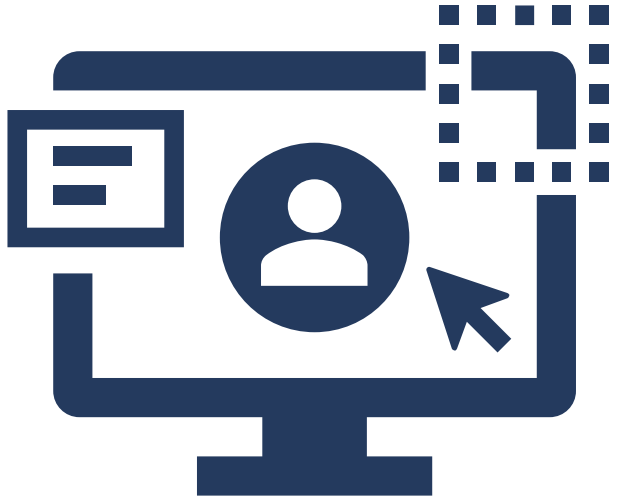
Define events for:

- Run completion
- Run failure
- Model registration
- Model deployment
- Data drift detection

Trigger automated actions:

- Azure Functions
- Azure Logic Apps
- Azure Event Hubs
- Azure Data Factory pipelines
- Generic webhooks

Lab: Create a Pipeline



1. View the lab instructions at https://aka.ms/mslearn-dp100
2. Complete the **Create a pipeline** exercise

Knowledge check

What type of object should you use to pass data between pipeline steps?

- ❑ Datastore
- ❑ Dataset
- ☑ OutputFileDatasetConfig

You plan to use the *Schedule.create* method to create a schedule for a published pipeline. What kind of object must you create first to configure how frequently the pipeline runs?

- ☑ ScheduleRecurrance
- ❑ Datastore
- ❑ PipelineParameter

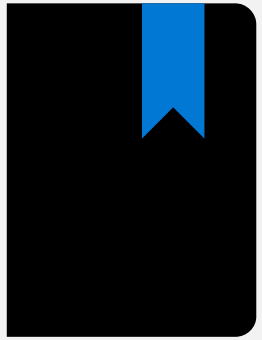
References

Microsoft Learn: Orchestrate machine learning with pipelines
https://docs.microsoft.com/learn/modules/create-pipelines-in-aml/

Azure Machine Learning pipelines documentation
https://docs.microsoft.com/azure/machine-learning/how-to-create-your-first-pipeline

Azure Machine Learning ML Ops documentation
https://docs.microsoft.com/azure/machine-learning/concept-model-management-and-deployment

Azure Machine Learning events documentation
https://docs.microsoft.com/azure/machine-learning/how-to-use-event-grid



Microsoft Azure