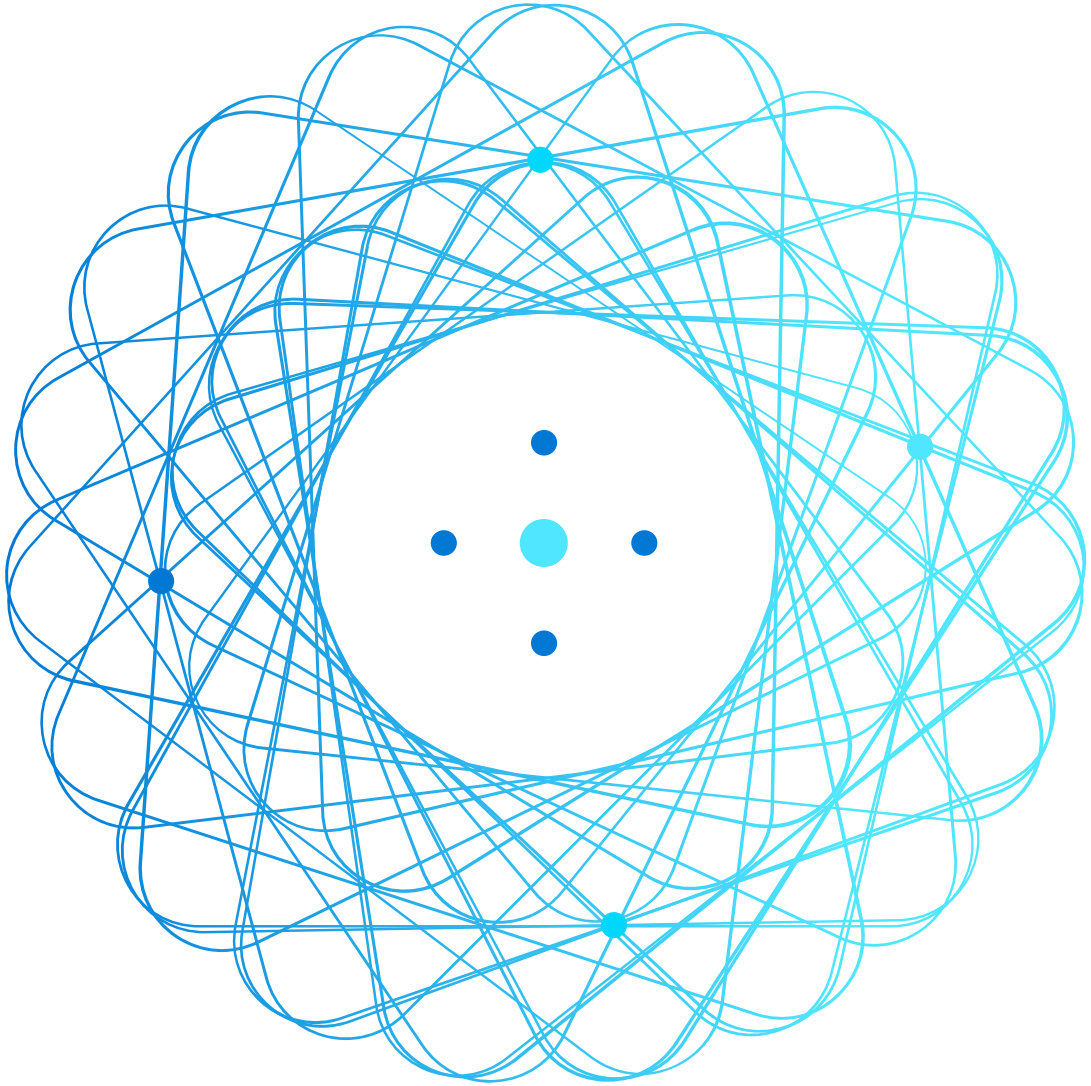
Microsoft Azure

Work with compute in Azure Machine Learning

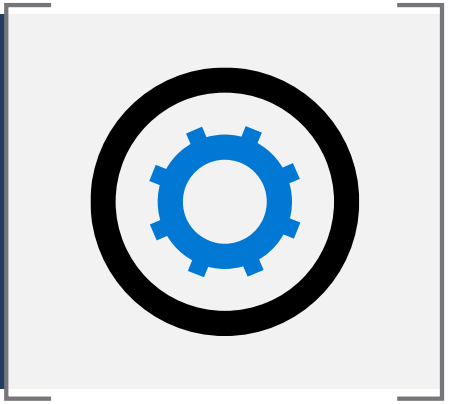


Agenda

Work with compute targets in Azure Machine Learning

Work with environments in Azure Machine Learning

Lesson 1: Work with compute targets in Azure Machine Learning



Choose the appropriate compute target

Compute targets are physical or virtual machines on which jobs are run.

Azure Machine Learning supports multiple types of compute for experimentation, training, and deployment.:

- **Compute instance**: Virtual machine designed to run notebooks. Ideal for *experimentation*.
- **Compute clusters**: Multi-node clusters of virtual machines that automatically scale up or down to meet demand. Ideal for *model training*.
- **Kubernetes clusters**: Create or attach Azure Kubernetes cluster for *real-time deployment*.
- **Attached compute**: Attach existing Azure compute resource. For example, Azure Virtual Machine.

Create a compute instance

To create a compute instance with the Python SDK, you can use the following code:

Python

```
from azure.ai.ml.entities import ComputeInstance

ci_basic_name = "basic-ci-12345"
ci_basic = ComputeInstance(
        name=ci_basic_name,
        size="STANDARD_DS3_v2"
)

ml_client.begin_create_or_update(ci_basic).result()
```

Use a setup script

You can use a setup script to configure the compute instance on creation.

Bash

```
#!/bin/bash


# clone repository
git clone https://github.com/MicrosoftLearning/mslearn-azure-ml.git azure-ml-labs
```

Configure the compute instance

Assign a compute instance to a user

- A compute instance can only be assigned to *one* user, as the compute instance can't handle parallel workloads.
- When you create a new compute instance, you can assign it to someone else if you have the appropriate permissions.

Create a schedule

- Manually start and stop a compute instance whenever you need it.
- Schedule your compute instance to start or stop at set times to avoid unnecessary costs.

Create a compute cluster

To create a compute instance with the Python SDK, you can use the following code:

Python

```
from azure.ai.ml.entities import AmlCompute

cluster_basic = AmlCompute(
        name="cpu-cluster",
        type="amlcompute",
        size="STANDARD_DS3_v2",
        location="westus",
        min_instances=0,
        max_instances=2,
        idle_time_before_scale_down=120,
        tier="low_priority",)
ml_client.begin_create_or_update(cluster_basic).result()
```

Use a compute cluster (1/2)

A compute cluster is ideal for three main scenarios:

- Running a **pipeline job**.
- Running an **Automated Machine Learning job**.
- Running a script as a **command job**.

In each of these scenarios, a compute cluster is ideal as a compute cluster will automatically scale up when a job is submitted, and automatically shut down when a job is completed.

Use a compute cluster (2/2)

To run a script as a command job, you can set the compute target to your compute cluster with the following code:

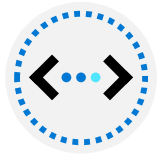
Python

```
from azure.ai.ml import command

job = command(
        code="./src",
        command="python diabetes-training.py",
        environment="AzureML-sklearn-0.24-ubuntu18.04-py37-cpu@latest",
        compute="cpu-cluster",
        display_name="train-with-cluster",
        experiment_name="diabetes-training")

returned_job = ml_client.create_or_update(job)
```

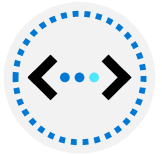
Exercise – Work with compute resources

In this exercise, you will:

Task 1: Create the compute setup script



Task 2: Create and use a compute instance



Task 3: Create and use a compute cluster

Instructions

Follow these instructions to complete the exercise:

1. View the exercise repo at https://microsoftlearning.github.io/mslearn-azure-ml/.
2. Complete the **Work with compute resources in Azure Machine Learning** exercise.

Knowledge check

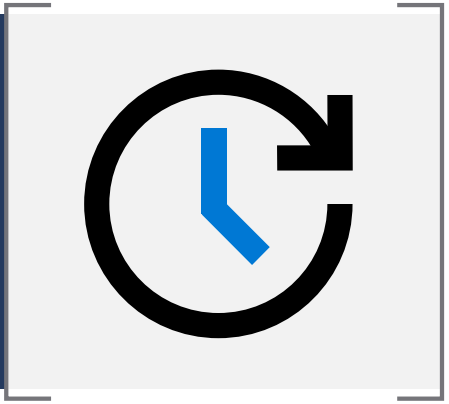
You're working in Visual Studio Code. You cloned a GitHub repository to Visual Studio Code and you're editing code in a Jupyter notebook. To test the code, you want to run a cell within the notebook. Which compute should you use?

- [x] Compute instance
- [ ] Compute cluster
- [ ] Azure Databricks cluster

You're experimenting with component-based pipelines in the Designer. You want to quickly iterate and experiment, as you're trying out varying configurations of a pipeline. You're using a compute cluster. To minimize the start-up time each time you submit a pipeline, which parameter should you change?

- [ ] Compute size
- [x] Idle time before scale down
- [ ] Maximum number of instances

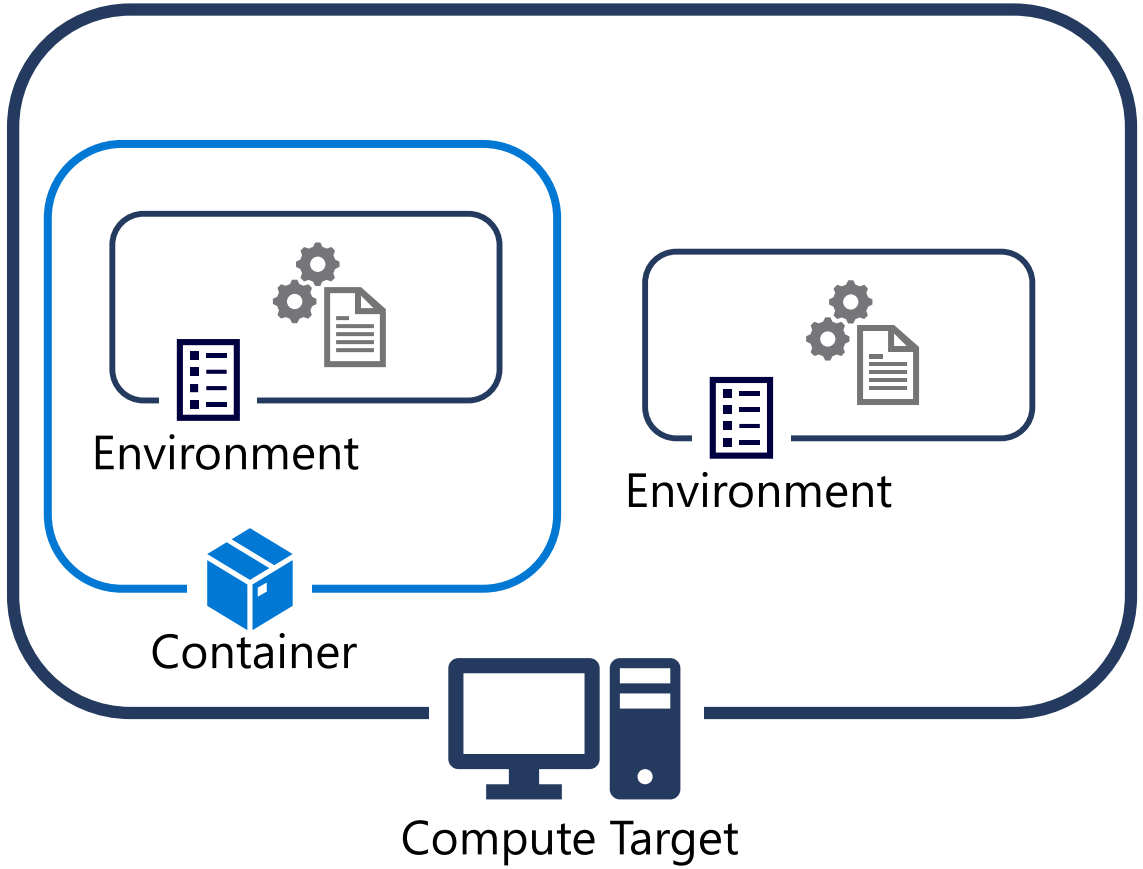
Lesson 2: Work with environments in Azure Machine Learning



Understand environments in Azure Machine Learning

Environments specify the Python packages, environment variables, and software settings required to execute your scripts.

By creating and using an environment, you ensure you can run your script on any compute target.



Use curated environments

Curated environments are prebuilt environments for the most common machine learning workloads, available in your workspace by default.

You can list all environments in the workspace:

Python

```
envs = ml_client.environments.list()
for env in envs:
    print(env.name)
```

Curated environments use the prefix **AzureML-**

Create and use a custom environment (1/2)

Use **custom environments** when you need to include specific Python packages or other dependencies to run your script.

You can create an environment from a Docker image:

Python

```
from azure.ai.ml.entities import Environment

env_docker_image = Environment(
    image="mcr.microsoft.com/azureml/openmpi3.1.2-ubuntu18.04",
    name="docker-image-example",
    description="Environment created from a Docker image.")
ml_client.environments.create_or_update(env_docker_image)
```

Create and use a custom environment (2/2)

If the base Docker image doesn't suffice, you can add a conda specification file listing the additional necessary packages and dependencies.

To create an environment using a Docker image and a conda specification file:

Python

```
from azure.ai.ml.entities import Environment

env_docker_conda = Environment(
    image="mcr.microsoft.com/azureml/openmpi3.1.2-ubuntu18.04",
    conda_file="./src/conda-env.yml",
    name="docker-image-plus-conda-example",
    description="Environment from a Docker image plus Conda environment.")
ml_client.environments.create_or_update(env_docker_conda)
```

Use environments in jobs

To use an environment, list the name of the environment and the version in the job configuration:

Python

```
from azure.ai.ml import command

job = command(
        code="./src",
        command="python diabetes-training.py",
        environment="docker-image-plus-conda-example:1",
        compute="cpu-cluster",
        display_name="train-with-custom-env",
        experiment_name="diabetes-training")

returned_job = ml_client.create_or_update(job)
```

Exercise – Work with environments

In this exercise, you will:

Task 1: Explore environments	Task 2: Run a job with a curated environment	Task 3: Create and use a curated environment
		

Instructions

Follow these instructions to complete the exercise:

1. View the exercise repo at https://microsoftlearning.github.io/mslearn-azure-ml/.
2. Complete the **Work with environments in Azure Machine Learning** exercise.

Knowledge check

You created a script that trains a machine learning model using the open-source library scikit-learn. You want to quickly test whether the script can run on your compute cluster, what type of environment should you use?

- ❑ Default
- ☑ Curated
- ❑ Custom

You use a curated environment and get an error that a module is not found. Which file should you create when you need to list the specific additional Python package you need to have installed on your compute cluster before running your script?

- ❑ Training script
- ❑ Docker image
- ☑ Conda specification

Microsoft Azure