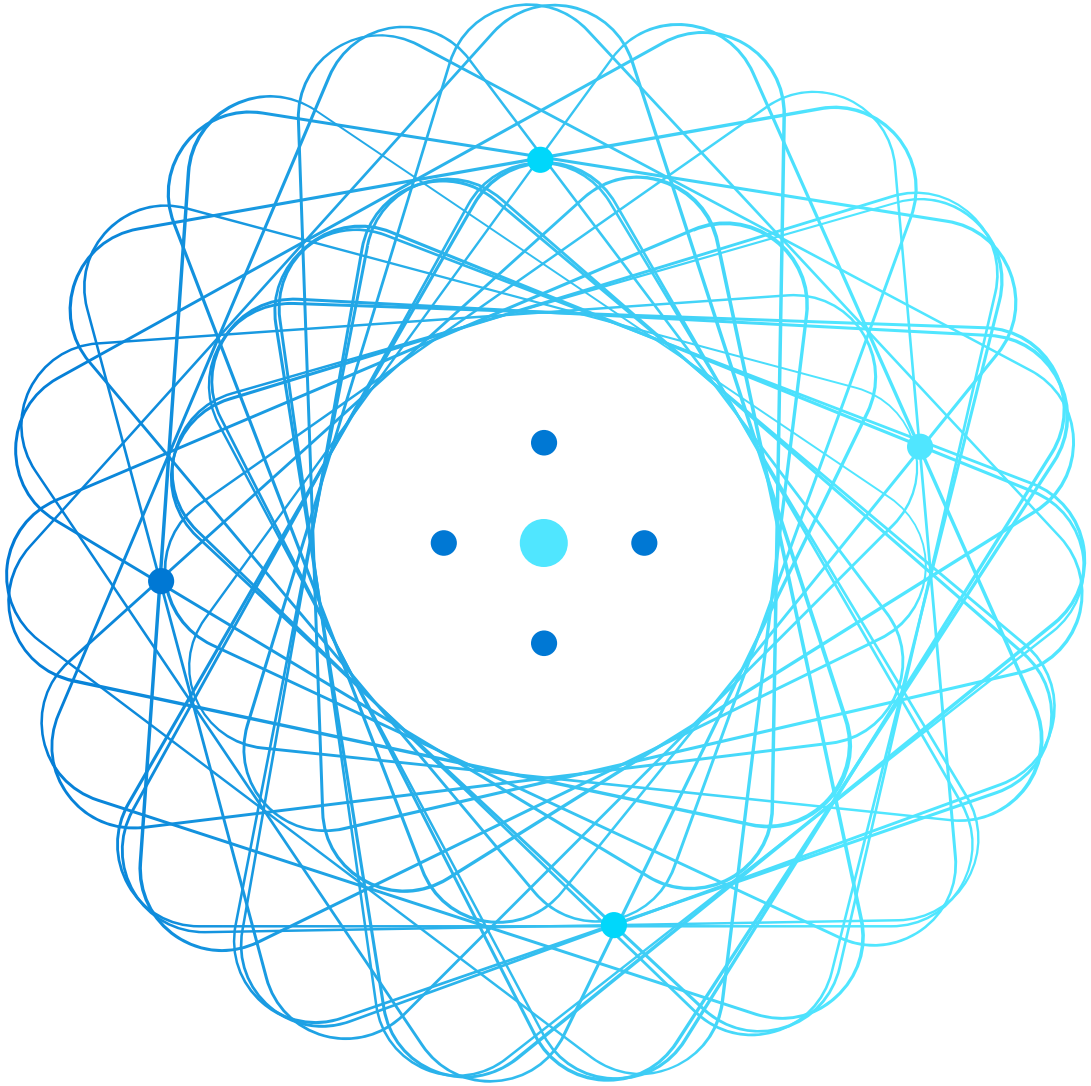
Microsoft Azure

# Optimize model training in Azure Machine Learning

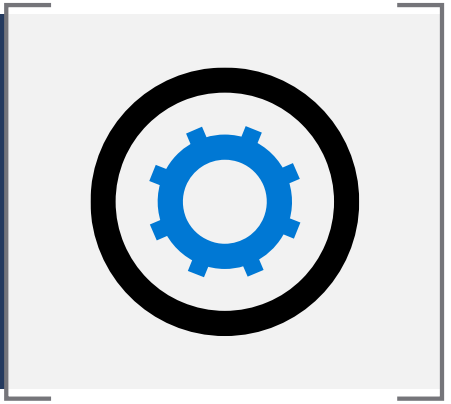


# Module Agenda

Run pipelines in Azure Machine Learning

Perform hyperparameter tuning with Azure Machine Learning

# Run pipelines in Azure Machine Learning



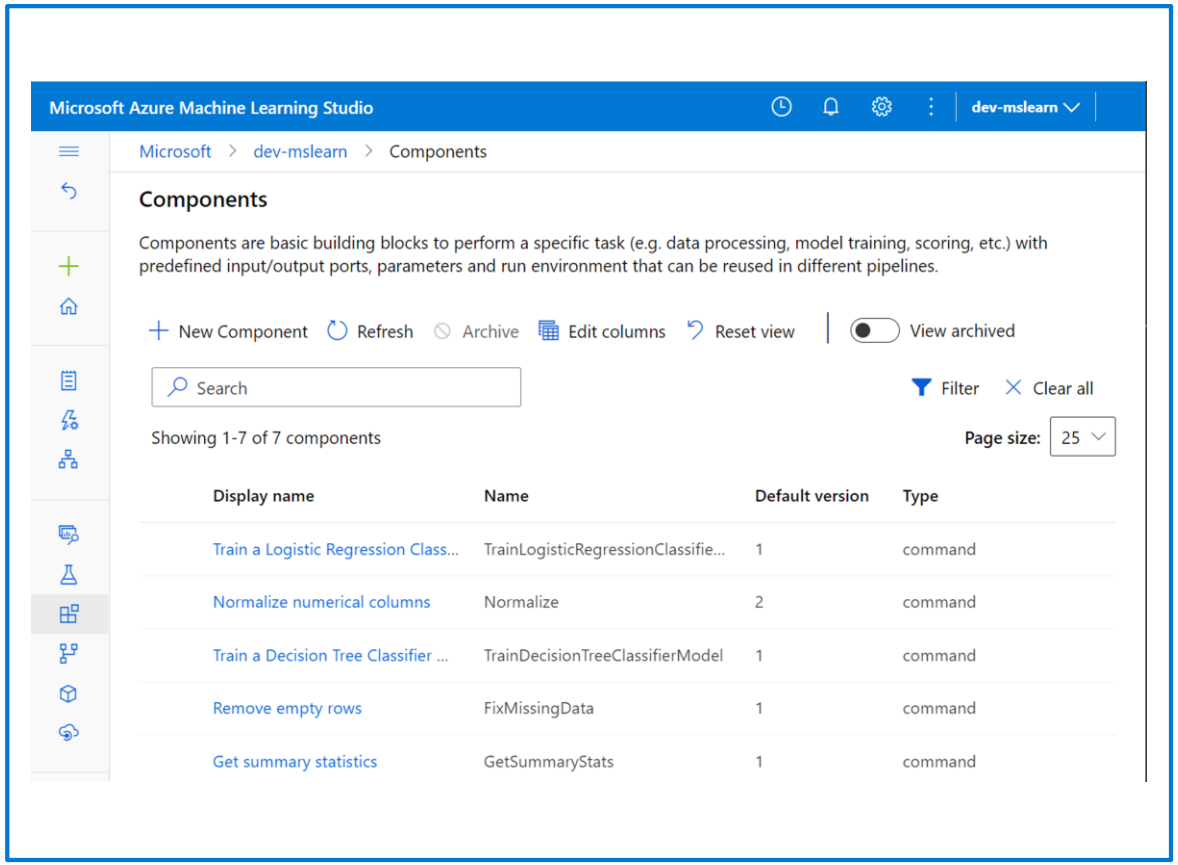
# Create components

**Components** allow you to create reusable scripts that can easily be shared across users within the same Azure Machine Learning workspace.

You can also use components to build an Azure Machine Learning pipeline.

## Use a component:

- To build a pipeline
- To share ready-to-go code



# Create a component

## A component consists of three parts:

### Metadata

Includes the component's name, version, etc.

### Interface

Includes the expected input parameters (like a dataset or hyperparameter) and expected output (like metrics and artifacts)

### Command, code and environment

Specifies how to run the code

To create a component, you need two files:

- A script that contains the workflow you want to execute
- A YAML file to define the metadata, interface, and command, code, and environment of the component

# Register a component

- To use components in a pipeline, you'll need the script and the YAML file.
- To make the components accessible to other users in the workspace, you can also register components to the Azure Machine Learning workspace.

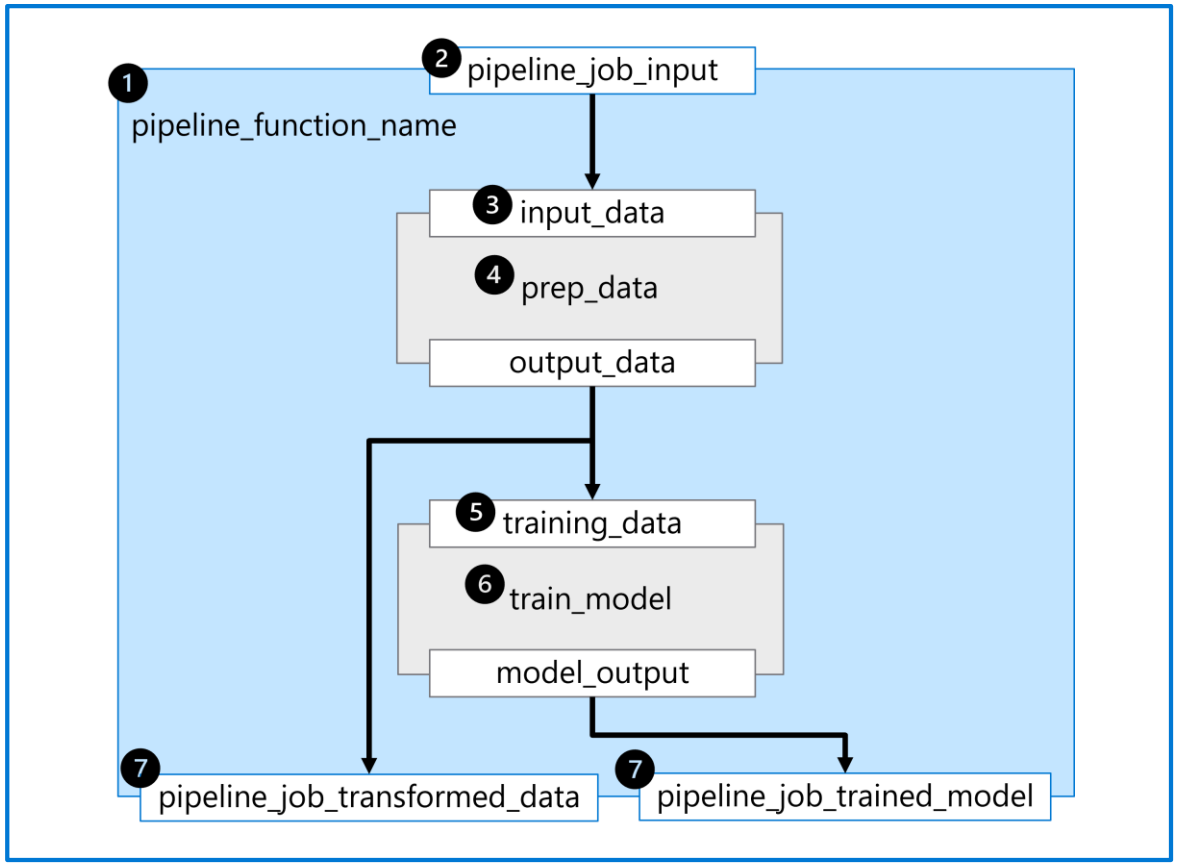
You can register a component with the following code:

Python

```
prep = ml_client.components.create_or_update(prepare_data_component)
```

# Create a pipeline

In Azure Machine Learning, a **pipeline** is a workflow of machine learning tasks in which each task is defined as a **component**.

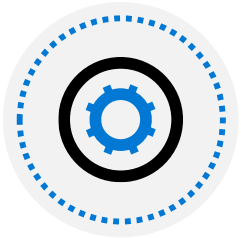



## Build a pipeline

An Azure Machine Learning pipeline is defined in a YAML file. The YAML file includes the pipeline job name, inputs, outputs, and settings.

# Run a pipeline job

When you've built a component-based pipeline in Azure Machine Learning, you can run the workflow as a **pipeline job**.



### Configure a pipeline job

A pipeline is defined in a YAML file, which you can also create using the `@pipeline()` function

### Run a pipeline job

When you've configured the pipeline, you're ready to run the workflow as a pipeline job

### Schedule a pipeline job

To schedule a pipeline job, you'll use the `JobSchedule` class to associate a schedule to a pipeline job

# Exercise - Run a pipeline job

## Task 1: Create components

## Task 2: Build a pipeline

## Task 3: Run a pipeline

## Task 4: Schedule a pipeline

# Knowledge check

You're creating a pipeline that includes two steps. Step 1 prepares some data, and step 2 uses the preprocessed data to train a model. What option should you use as input to the second step to train the model?

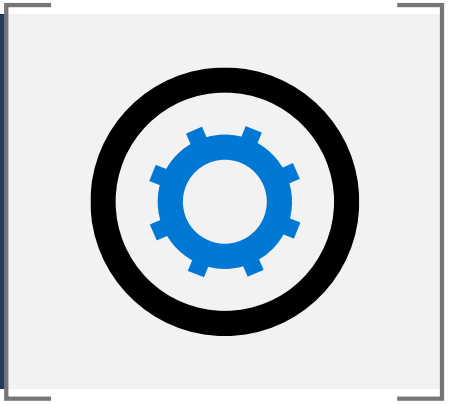
- ❑ pipeline_job_input
- ☑ prep_data.outputs.output_data
- ❑ train_model.outputs.model_output

---

You've built a pipeline that you want to run every week. You want to take a simple approach to creating a schedule. What class can you use to create the schedule that runs once per week?

- ❑ RecurrencePattern
- ❑ JobSchedule
- ☑ RecurrenceTrigger

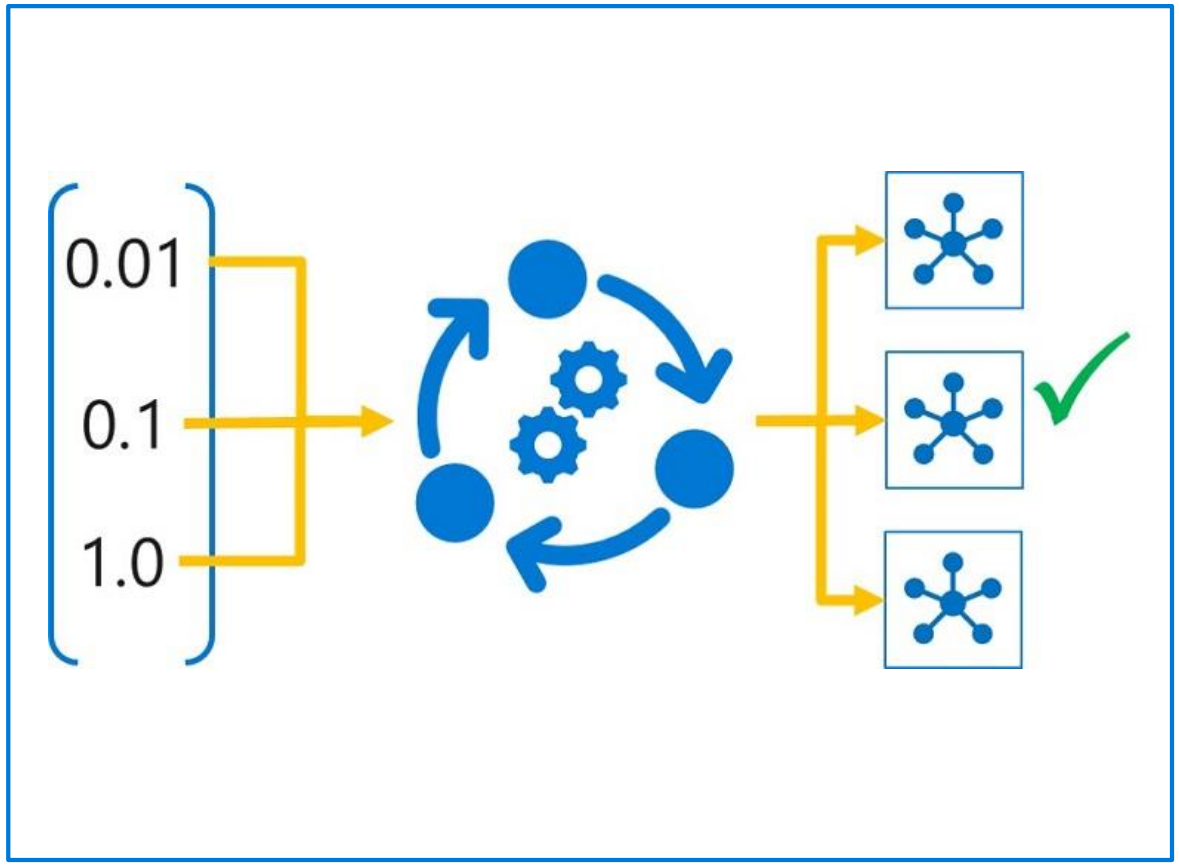
# Perform hyperparameter tuning with Azure Machine Learning



# Understand hyperparameter tuning

In machine learning, models are trained to predict unknown labels for new data based on correlations between known labels and features found in the training data.

**Hyperparameter tuning** is accomplished by training the multiple models, using the same algorithm and training data but different hyperparameter values.



# Define a search space

The set of hyperparameter values tried during hyperparameter tuning is known as the **search space**. The definition of the range of possible values that can be chosen depends on the type of hyperparameter.

**Discrete hyperparameters:** Some hyperparameters require discrete values - in other words, you must select the value from a particular *finite* set of possibilities.

**Continuous hyperparameters:** Some hyperparameters are *continuous* - in other words you can use any value along a scale, resulting in an *infinite* number of possibilities

**Defining a search space:** To define a search space for hyperparameter tuning, create a dictionary with the appropriate parameter expression for each named hyperparameter

# Configure a sampling method

The specific values used in a hyperparameter tuning run, or **sweep job**, depend on the type of **sampling** used.

There are three main sampling methods available in Azure Machine Learning:

**Grid sampling:** Tries every possible combination

**Random sampling:** Randomly chooses values from the search space

- **Sobol:** Adds a seed to random sampling to make the results reproducible

**Bayesian sampling:** Chooses new values based on previous results

# Configure early termination

- When you configure a sweep job in Azure Machine Learning, you can set a maximum number of trials.
- A more sophisticated approach may be to stop a sweep job when newer models don't produce significantly better results.
- To stop a sweep job based on the performance of the models, you can use an **early termination policy**.

## When to use an early termination policy

Whether you want to use an early termination policy may depend on the search space and sampling method you're working with.

An early termination policy can be especially beneficial when working with continuous hyperparameters in your search space.

# Configure an early termination policy

There are two main parameters when you choose to use an early termination policy:

**`evaluation_interval:`**

Specifies at which interval you want the policy to be evaluated

**`delay_evaluation:`**

Specifies when to start evaluating the policy

New models may continue to perform only slightly better than previous models. To determine the extent to which a model should perform better than previous trials, there are three options for early termination:

- **Bandit policy**
- **Median stopping policy**
- **Truncation selection policy**

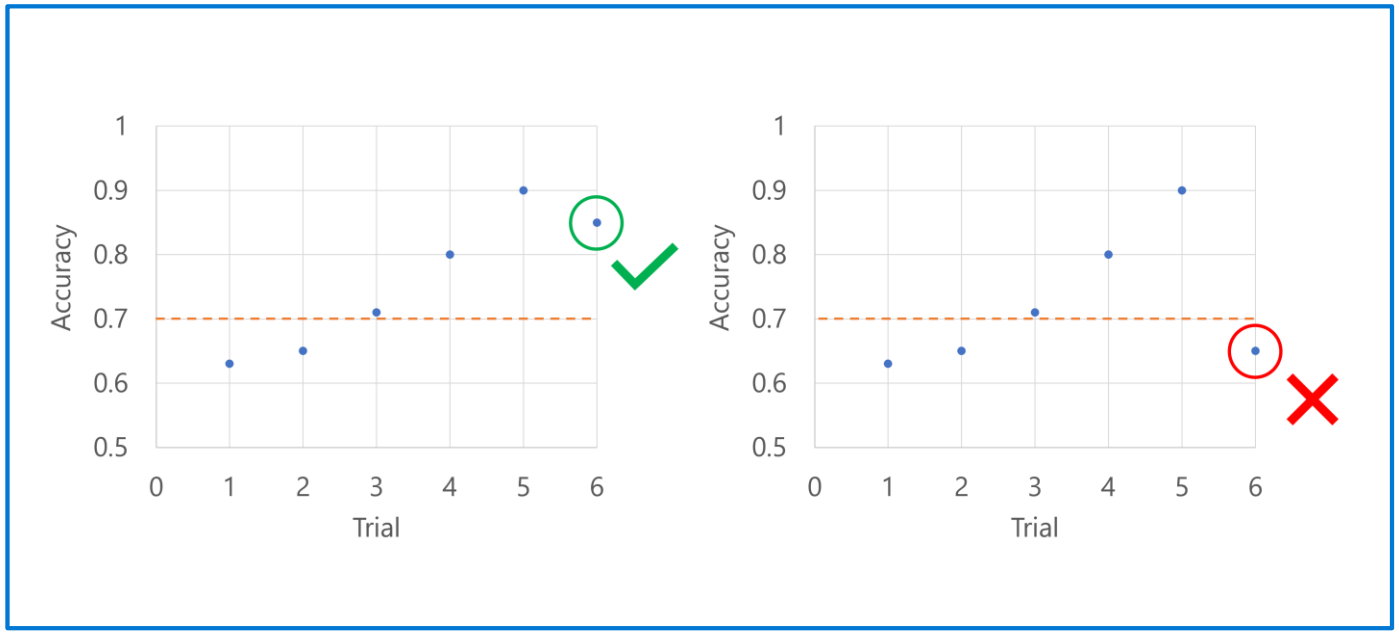
# Bandit policy

You can use a bandit policy to stop a trial if the target performance metric underperforms the best trial so far by a specified margin.

Python

```
from azure.ai.ml.sweep import BanditPolicy

sweep_job.early_termination = BanditPolicy(
    slack_amount = 0.2,
    delay_evaluation = 5,
    evaluation_interval = 1
)
```



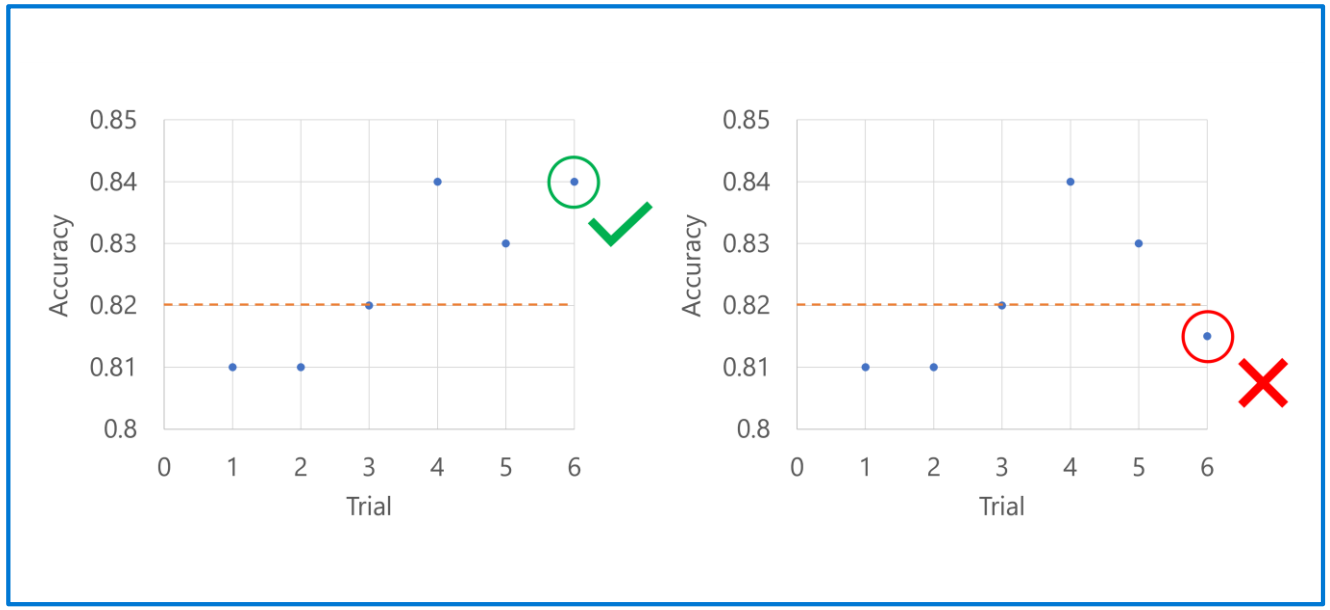
# Median stopping policy

A median stopping policy abandons trials where the target performance metric is worse than the median of the running averages for all trials.

Python

```
from azure.ai.ml.sweep import MedianStoppingPolicy

sweep_job.early_termination = MedianStoppingPolicy(
    delay_evaluation = 5,
    evaluation_interval = 1
)
```



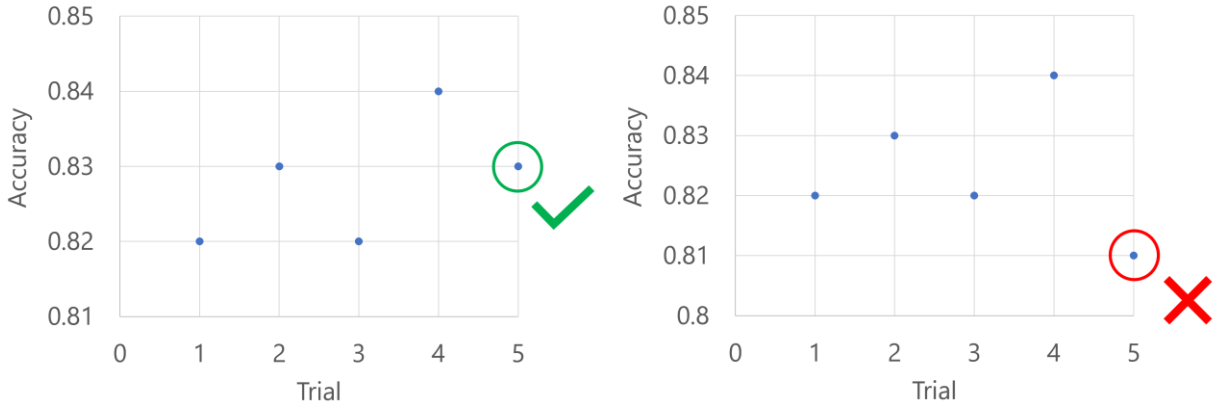
# Truncation selection policy

A truncation selection policy cancels the lowest performing X% of trials at each evaluation interval based on the *truncation_percentage* value you specify for X.

Python

```
from azure.ai.ml.sweep import TruncationSelectionPolicy

sweep_job.early_termination = TruncationSelectionPolicy(
    evaluation_interval=1,
    truncation_percentage=20,
    delay_evaluation=4
)
```



# Use a sweep job for hyperparameter tuning

## In Azure Machine Learning, you can tune hyperparameters by running a sweep job.

Create a training script for hyperparameter tuning

Configure and run a sweep job

Monitor and review sweep jobs

# Exercise - Run a sweep job

Task 1: Run a command job

Task 2: Configure a sweep job using the command job as a base

Task 3: Submit the sweep job

## Instructions

Follow these instructions to complete the exercise:

- View the exercise repo at https://microsoftlearning.github.io/mslearn-azure-ml/.
- Complete the **Perform hyperparameter tuning with a sweep job** exercise.

# Knowledge check

You plan to use hyperparameter tuning to find optimal discrete values for a set of hyperparameters. You want to try every possible combination of a set of specified discrete values. What kind of sampling should you use?

- ❑ Random sampling
- ☑ Grid sampling
- ❑ Bayesian sampling

---

You're using hyperparameter tuning to train an optimal model based on a target metric named "AUC". What should you do in your training script?

- ❑ Import the logging package and use a logging.info() statement to log the AUC.
- ❑ Include a print() statement to write the AUC value to the standard output stream.
- ☑ Use a mlflow.log_metric() statement to log the AUC value.

Microsoft Azure