Highlight Note

Pipeline Steps

An Azure Machine Learning pipeline consists of one or more *steps* that perform tasks. There are many kinds of step supported by Azure Machine Learning pipelines, each with its own specialized purpose and configuration options.

Types of Step

Common kinds of step in an Azure Machine Learning pipeline include:

- **PythonScriptStep**: Runs a specified Python script.
- **DataTransferStep**: Uses Azure Data Factory to copy data between data stores.
- **DdatabricksStep**: Runs a notebook, script, or compiled JAR on a databricks cluster.
- **AdlaStep**: Runs a U-SQL job in Azure Data Lake Analytics.
- **ParallelRunStep** - Runs a Python script as a distributed task on multiple compute nodes.

Note: For a full list of supported step types, see **azure.pipeline.steps package documentation**.

Defining Steps in a Pipeline

To create a pipeline, you must first define each step and then create a pipeline that includes the steps. The specific configuration of each step depends on the step type. For example the following code defines two **PythonScriptStep** steps to prepare data, and then train a model.

```
from azureml.pipeline.steps import PythonScriptStep

# Step to run a Python script
step1 = PythonScriptStep(name = 'prepare data',
                         source_directory = 'scripts',
                         script_name = 'data_prep.py',
                         compute_target = 'aml-cluster')

# Step to train a model
step2 = PythonScriptStep(name = 'train model',
                         source_directory = 'scripts',
                         script_name = 'train_model.py',
                         compute_target = 'aml-cluster')
```

After defining the steps, you can assign them to a pipeline, and run it as an experiment:

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

# Construct the pipeline
train_pipeline = Pipeline(workspace = ws, steps = [step1,step2])

# Create an experiment and run the pipeline
```

```
experiment = Experiment(workspace = ws, name = 'training-pipeline')
pipeline_run = experiment.submit(train_pipeline)
```