100 XP

Running automated machine learning experiments

5 minutes

To run an automated machine learning experiment, you can either use the user interface in Azure Machine Learning studio, or submit an experiment using the SDK.

Configure an automated machine learning experiment

The user interface provides an intuitive way to select options for your automated machine learning experiment. When using the SDK, you have greater flexibility, and you can set experiment options using the **AutoMLConfig** class, as shown in the following example.

Python Copy

```
from azureml.train.automl import AutoMLConfig

automl_run_config = RunConfiguration(framework='python')
automl_config = AutoMLConfig(name='Automated ML Experiment',
                             task='classification',
                             primary_metric = 'AUC_weighted',
                             compute_target=aml_compute,
                             training_data = train_dataset,
                             validation_data = test_dataset,
                             label_column_name='Label',
                             featurization='auto',
                             iterations=12,
                             max_concurrent_iterations=4)
```

Specify data for training

Automated machine learning is designed to enable you to simply bring your data, and have Azure Machine Learning figure out how best to train a model from it.

When using the Automated Machine Learning user interface in Azure Machine Learning studio, you can create or select an Azure Machine Learning *dataset* to be used as the input for your automated machine learning experiment.

When using the SDK to run an automated machine learning experiment, you can submit the data in the following ways:

- Specify a dataset or dataframe of *training data* that includes features and the label to be predicted.
- Optionally, specify a second *validation data* dataset or dataframe that will be used to validate the trained model. if this is not provided, Azure Machine Learning will apply cross-validation using the training data.

Alternatively:

- Specify a dataset, dataframe, or numpy array of *X* values containing the training features, with a corresponding *y* array of label values.
- Optionally, specify *X_valid* and *y_valid* datasets, dataframes, or numpy arrays of *X_valid* values to be used for validation.

Specify the primary metric

One of the most important settings you must specify is the **primary_metric**. This is the target performance metric for which the optimal model will be determined. Azure Machine Learning supports a set of named metrics for each type of task. To retrieve the list of metrics available for a particular task type, you can use the **get_primary_metrics** function as shown here:

Python Copy

```
from azureml.train.automl.utilities import get_primary_metrics

get_primary_metrics('classification')
```

More Information: You can find a full list of primary metrics and their definitions in Understand automated machine learning results .

Submit an automated machine learning experiment

You can submit an automated machine learning experiment like any other SDK-based experiment.

Python Copy

```
from azureml.core.experiment import Experiment
```

```
automl_experiment = Experiment(ws, 'automl_experiment')
automl_run = automl_experiment.submit(automl_config)
```

You can monitor automated machine learning experiment runs in Azure Machine Learning studio, or in the Jupyter Notebooks **RunDetails** widget.

Retrieve the best run and its model

You can easily identify the best run in Azure Machine Learning studio, and download or deploy the model it generated. To accomplish this programmatically with the SDK, you can use code like the following example:

Python Copy

```
best_run, fitted_model = automl_run.get_output()
best_run_metrics = best_run.get_metrics()
for metric_name in best_run_metrics:
    metric = best_run_metrics[metric_name]
    print(metric_name, metric)
```

Explore preprocessing steps

Automated machine learning uses scikit-learn pipelines to encapsulate preprocessing steps with the model. You can view the steps in the fitted model you obtained from the best run using the code above like this:

Python Copy

```
for step_ in fitted_model.named_steps:
    print(step_)
```

Next unit: Exercise - Using automated machine learning

Continue >