Execute Python Script

Updated: April 16, 2015

Executes a Python script from an Azure Machine Learning experiment

Category: Python Language Modules (https://msdn.microsoft.com/en-us/library/azure/dn927167.aspx)

Module Overview

You can use the **Execute Python Script** module to run Python code from experiments in Azure Machine Learning.

By integrating your IPython notebooks with Azure Machine Learning, you can perform custom tasks, create visualizations, and share your models with the world. For example:

- Visualize data using matplotlib
- Use Python client libraries to enumerate datasets and models in your Workspace
- Read, load, and manipulate data

The **Execute Python Script** module contains sample Python code that you can use as a starting point when developing new code.

How to Use Execute Python Script

To configure the **Execute Python Script** module, you provide a set of inputs, and type the Python code to execute in the **Python script** text box.

Inputs. The module has these inputs:

- **Dataset1**. An optional dataset from your Machine Learning Studio workspace, containing input data or values.
- **Dataset2**. A second dataset, also optional.
- **Script bundle**. A zipped file containing custom resources

  Any file that is placed in your ZIP file will be available for use during execution time. If there is a directory structure present it will be preserved, with the only change being that you must prepend a directory called 'src' to the path.

Before you can connect the Script Bundle to the **Execute Python Script** module, the ZIP file must be already present in the Studio workspace. To upload a ZIP file to your workspace, click **New**, click **Dataset**, and then select **From local file** and the **Zip file** option.

Outputs. The module returns these outputs:

- **Results Dataset**. A dataset with the results of any computations performed by the embedded Python code.

  The dataset uses the internal dataset format. For more information, see Data Table (https://msdn.microsoft.com/en-us/library/azure/dn905851.aspx).

- **Python Device**. This output supports both console output and display of PNG graphics using the Python interpreter.

Providing Python Script

In addition to providing custom code or other resources via the module's input ports, you can use the modules **Properties** pane to type Python script.

Python script

Type Python script into the **Python script** text box.

The **Python script** text box is pre-populated with the following sample code, which you can edit or replace.

```
# The script MUST contain a function named azureml_main
# which is the entry point for this module.
#
# The entry point function can contain up to two input
arguments:
#   Param<dataframe1>: a pandas.DataFrame
#   Param<dataframe2>: a pandas.DataFrame
def azureml_main(dataframe1 = None, dataframe2 = None):

    # Execution logic goes here
    print('Input pandas.DataFrame
#1:\r\n\r\n{0}'.format(dataframe1))

    # If a zip file is connected to the third input port is
connected,
    # it is unzipped under ".\Script Bundle". This directory
is added
    # to sys.path. Therefore, if your zip file contains a
Python file
    # mymodule.py you can import it using:
    # import mymodule

    # Return value must be of a sequence of pandas.DataFrame
    return dataframe1,
```

Examples

For examples of how to integrate Python script with Azure Machine Learning experiments, see these resources:

- Execute Python scripts in Azure Machine Learning (http://azure.microsoft.com/documentation/articles/machine-learning-execute-python-scripts/)

- IPython Notebook on Azure (http://azure.microsoft.com/documentation/articles/virtual-machines-python-ipython-notebook/)

- Access datasets with Python using the Azure Machine Learning Python client library (http://azure.microsoft.com/documentation/articles/machine-learning-python-data-access/)

The following experiments in the Model Gallery (http://gallery.azureml.net/) also use Python:

- The Execute Python Script (http://go.microsoft.com/fwlink/p/?LinkId=525942) sample demonstrates how you can perform complex processing – in this case, text tokenization, stemming, and other processing – inside the **Execute Python Script** module.

See Also

Python Language Modules (https://msdn.microsoft.com/en-us/library/azure/dn927167.aspx)
R Language Modules (https://msdn.microsoft.com/en-us/library/azure/dn905920.aspx)