Developing Intelligent Apps

Lab 1 – Creating a Simple Client Application

By Gerry O'Brien

Overview

In this lab you will construct a simple client application that will call an Azure ML web service that you will create. The Azure ML Web service you will create is based on a dataset that you will import into Azure ML Studio and is designed to perform an energy efficiency regression experiment.

What You'll Need

To complete this lab, you will need the following:

- An Azure ML account
- A web browser and Internet connection
- Microsoft Visual Studio

Create the Sample Experiment

In this exercise you will import a data set into Azure ML Studio and use that to create an experiment that will be used to publish as a web service. Subsequent lab steps will make use of this experiment. The experiment we create will not be complex and is only used to illustrate how to create and consume a web service.

1. Download the Building Data.csv file from the course GitHub repository at https://github.com/MicrosoftLearning/BuildingIntelligentApplications.
2. If you have not already done so, open a browser and browse to https://studio.azureml.net. Then sign in using the Microsoft account associated with your Azure ML account.
3. Click the Datasets icon on the left nav bar of Azure ML Studio.
4. Click the New button in the lower left corner to upload the dataset.
5. Choose DataSet, then From Local File.
6. Click Browse and locate the file you just downloaded and select it.
7. Click the Check Mark in the lower right corner to upload the dataset.
8. Verify that Building Data.csv is listed as a dataset.
9. Click the Experiments icon in Azure ML Studio.
10. Click New to create a new experiment and select Blank Experiment.
11. Expand My Datasets and locate Building Data.csv.

12. Drag and drop this dataset onto the designer surface.
13. Expand the Data Transformation category and then Sample and Split subcategory.
14. Drag a Split Data operation onto the designer surface.
15. Set the Fraction of rows in the first output dataset to 0.7 to split the data at 70/30.
16. Connect the dataset to the Split Data operation by dragging the dataset handle to the Split Data operation.
17. Expand the Machine Learning category, then Initialize Model, and finally Regression.
18. Locate the Decision Forest Regression entry and drag it onto the designer surface.
19. Expand the Score subcategory and locate the Score Model entry.
20. Drag the Score Model operation onto the designer.
21. Expand Train and drag a Train Model operation onto the designer.
22. Expand the Evaluate subcategory and drag an Evaluate Model operation onto the designer.
23. Your experiment window should now look like this:

Draft saved at 2:04:15 PM
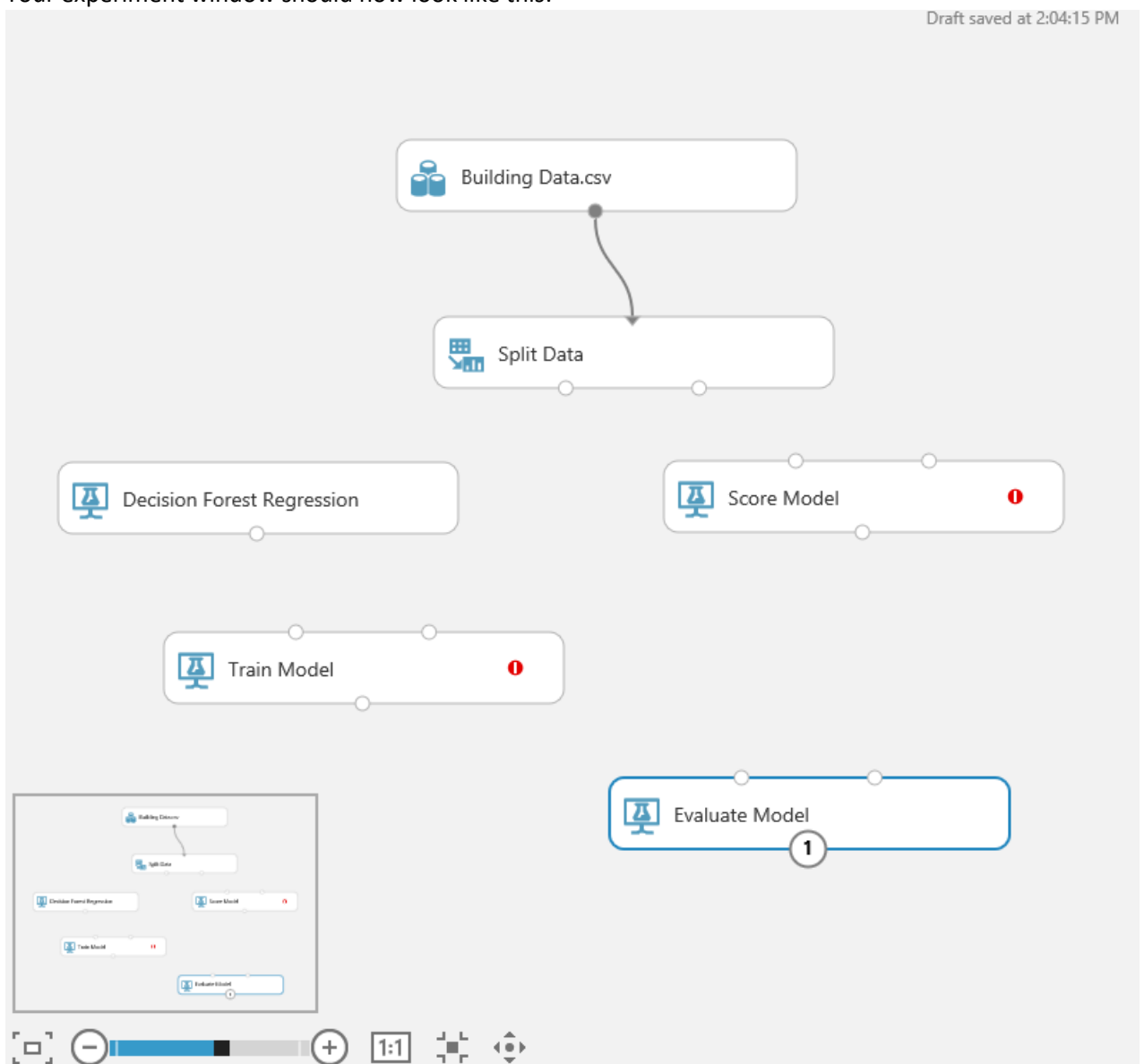
24. Connect output 1 of the Split Data operation to the Train Model operation.

25. Select the Train Model operation.
26. Click the Launch Column Selector in the right pane of ML Studio.
27. Click the With Rules entry and then click inside the empty column list and select the Wall Area column name.
28. Click the check mark to accept the selection.
29. Connect the Decision Forest Regression operation to the remaining input of the Train Model operation.
30. Connect the second output of the Split Data operation to the Score Model operation.
31. Connect the output of the Score Model operation to the left input of the Evaluate Model operation.
32. Connect the output of the Train Model operation to the remaining Score Model input.
33. Your experiment should now look like this.
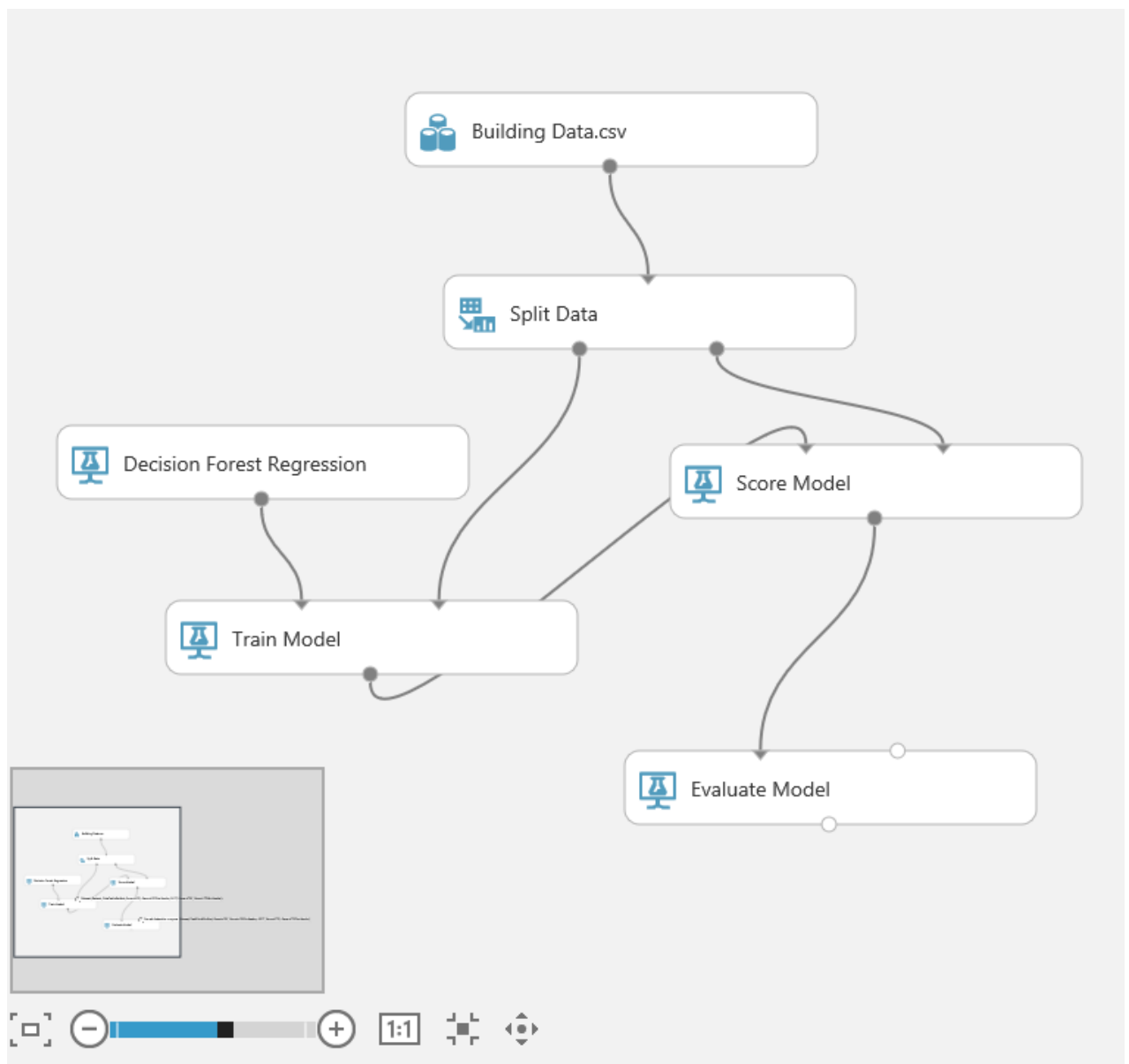
34. Run the experiment by clicking the Run button on the bottom of the designer window.

35. Ensure that you have green check marks in all the operation boxes and that the upper right corner of the designer displays Finished Running with a green check mark.
36. Save your experiment as Energy Efficiency Regression

Prepare and Deploy the Web Service

In this exercise you will prepare and publish your experiment as a web service, and then consume the web service from a client application.

Prepare the Web Service

You should have already completed the previous exercise to ensure that we have an experiment to use for publishing a web service.

1. If you have not already done so, open a browser and browse to https://studio.azureml.net. Then sign in using the Microsoft account associated with your Azure ML account.
2. Click the Experiments icon on the left nav bar of Azure ML Studio.
3. Locate your completed Energy Efficiency Regression experiment.
4. Click the experiment to open it in Azure ML Studio.
5. Depending on the state of the web service, you may not have to run it again. If it indicates Finished running in the upper right corner of the designer and you have green checks in the different stages, you are ready for the next step. If not, then click the Run button.

Set Up the Web Service

1. With the Energy Efficiency Regression experiment open in the designer and displaying green check marks showing completion, hover your mouse over the Set Up Web Service button in the button bar at the bottom of the designer.
2. Select Predictive Web Service [Recommended].
3. Verify that a new tab opens called Predictive experiment with the predictive experiment visible on this new tab. You should see a Web service input and Web service output operation on the designer.
4. Before we can deploy this web service, we need to run this new experiment. Click the Run button.
5. Verify successful completion by checking for a green check mark in the Score Model operation and Finished running in the upper right corner.
6. You are now ready for the next exercise.

Deploy the Web service

1. Click the Deploy Web Service button.
2. Azure ML Studio begins the creation of web service and opens the following screen indicating success. The API key has been removed from this image but you will have an API key present in your web service.

Test the Web Service

Now that the web service is deployed, let's test if first before we begin creation of the client application.

1. Click the Test button to open the data entry dialog.
2. Enter the following values in the data entry dialog (these values are taken from the training model data)
   - 296 for wall area
   - 110.25 for roof area
   - 7 for overall height
   - 0 for glazing area
   - 15.55 for heating load
3. Click the check mark in the bottom right corner of the data entry dialog.
4. Look at the status bar for the output. You can also click the Details button to see an example of the returned JSON.

Viewing the Sample Code

Before you create the client application, investigate the sample request and response headers and sample code that is provided by Azure ML Studio.

View the Request and Response Headers

1. Click the Request/Response link under API Help Page for the Default Endpoint
2. Azure ML Studio opens another browser tab or window with the API documentation for this web service.

3. Scroll down and view the Request Headers and Request Body to get an idea of how the request will be formed. This request is what will be sent to the web service for evaluation and is in JSON format.

4. Scroll down further and evaluate the Response Headers and Response Body sample. This shows the format of data that will be returned in the response as well as a sample of the JSON that will be returned. You can use this sample to understand how to parse the JSON for the data you want to use in your application.

5. Scrolling down further presents information about the input and output parameters which includes the names and data types expected. If you will be creating a class to represent the returned information, you can use these parameters to create the proper data types for your member variables.

Create a Simple Client Application

In this exercise, you will create a sample client application using either C# or Python, Visual Studio and the sample code that is provided on the web service API documentation page for your published web service. If you want to test this using R, paste the sample code into your R environment ensuring you replace the API Key and values.
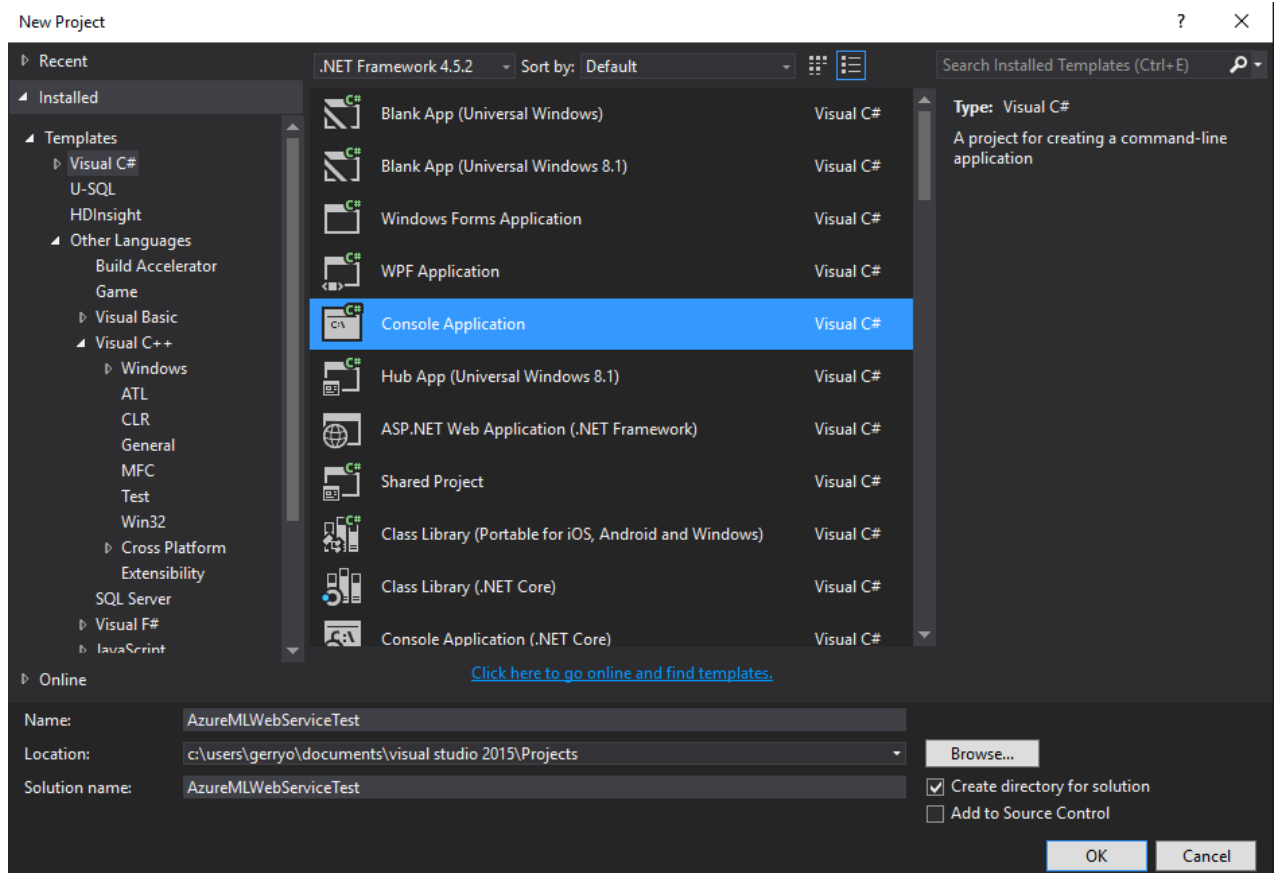
Evaluate the Sample Code

The sample code is what you will use for the client application created in this lab. Reviewing the code first, will help you understand where to place it in our client application and how it works to create the request and consume the response.

1. Scroll all the way to the bottom of the web service API documentation window. Note the Sample Code entry contains samples in C#, Python, and R.

2. Select the tab of the language you are most familiar with and scroll through the code to see what it is doing.

3. Note the location where the API key will need to be replaced. The value abc123 is used as a placeholder and you will paste your API key in this location when creating your client application.

Create a C# Console Client Application

The steps in this section assume you are using Visual Studio 2015. If you are not using Visual Studio or you are using a different version, you will need to make adjustments to some of the steps as necessary to accommodate your environment.

1. Start Visual Studio if it is not already open.

2. Click New Project or select File, New, Project.

3. Expand Templates if it is not already expanded.

4. Choose Visual C#

5. Select Console Application in the middle pane.

6. In the Name: text box, give your application a name that makes sense such as AzureMLWebServiceTest.

7. Click OK to create the application and project.

8. Visual Studio creates an application stub for you containing a Program.cs class with stub code including a Main method. You will replace this code with the sample code from the web service API documentation page.

9. Switch back to your browser and locate the C# sample source code.

10. Click the Select sample code button then copy the selected code to the clipboard

11. Switch back to Visual Studio.

12. Highlight the entire code listing in Program.cs and delete it.

13. Paste the sample code into the code editor window.

14. Scroll to the top of the code file in Program.cs.

15. Note the red squiggly line under Formatting in the using System.Net.Http.Formatting entry. This means that we are missing a library that implements the functionality necessary for this using statement.

16. Locate the last comment line in the comments at the top of the code file (// Install-Package Microsoft.AspNet.WebApi.Client).

17. Copy the command without the two back slashes (//)

18. Click the Tools menu and then move your mouse over the Nuget Package Manager, and finally select Package Manager Console.

19. The Package Manager console opens at the bottom of the Visual Studio IDE.

20. Paste the command you copied into this console window and press Enter.

21. You should see a successfully installed message in the console. If so, you can close the package manager console by clicking the small x in the right corner of the console window.

22. Verify there are no more errors in the code.

23. In order for your application to access the web service, you need to paste the API key into your source code. Go back the Web service window in Azure ML Studio where you ran the first test by clicking the Test button.

24. Copy the API key. (NOTE: the key in the below screen shot is not a working key.)
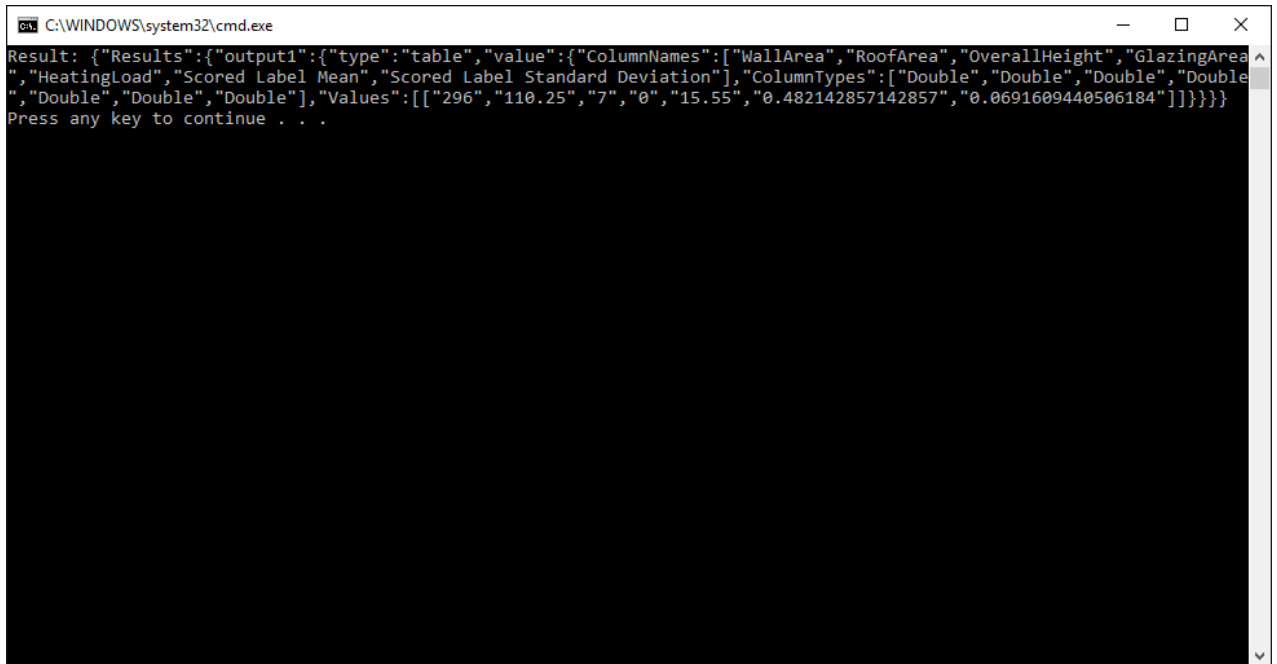
25. Switch back to Visual Studio and locate the *const string apiKey* entry and replace the abc123 with the copied API key. Be sure to leave the double quotes surrounding the key.

26. Next, locate the Values = new string[,] line of code.

27. Replace the first set of values with the same sample data that we used above for the quick test.

28. You final entry should look like this,

```
{
    ColumnNames = new string[] {"WallArea", "RoofArea", "OverallHeight", "GlazingArea", "HeatingLoad"},
    Values = new string[,] {  { "296", "110.25", "7", "0", "15.55" }  }
}
```

29. Press CTRL + F5 or select Debug, Start Without Debugging to run your client application.

30. Verify the output.

31. Press Enter to close the application and return to Visual Studio.

32. Close Visual Studio, the C# application portion is complete.

Create a Python Client Application

The steps in this section assume you are using Visual Studio 2015. If you are not using Visual Studio or you are using a different version, you will need to make adjustments to some of the steps as necessary to accommodate your environment.

NOTE: This lab assumes a Python 2.7 environment as that is what the sample code targets. If you are using Python 3+ you will need to make changes to the urllib2 entries in the code per the comments in the code file.

1. Start Visual Studio if it is not already open.
2. Click New Project or select File, New, Project.
3. Expand Templates if it is not already expanded.
4. Expand Other Languages.
5. Select Python.
6. Select Python Application.
7. In the Name: text box, give your application a name that makes sense such as AzureMLTestClientPython.

8. Click OK to create the application and project.

9. Visual Studio creates an application stub for you containing an AzureMLTestClientPython.py. You will replace this code with the sample code from the web service API documentation page.

10. Switch back to your browser and locate the Python sample source code.

11. Click the Select sample code button then copy the selected code to the clipboard

12. Switch back to Visual Studio.

13. Highlight the entire code listing in AzureMLTestClientPython.py and delete it.

14. Paste the sample code into the code editor window.

15. Scroll to the top of the code file. If the import urllib2 has an underline, if means you have an incorrectly configured Python environment in Visual Studio or you have Python 3+ installed. Install and configure Python 2.7 for this sample to work.

16. Copy and paste the API key into the api_key variable assignment.

17. Replace the first set of values with the same sample data that we used above for the quick test.

18. You final entry should look like this,

```
"ColumnNames": ["WallArea", "RoofArea", "OverallHeight", "GlazingArea", "HeatingLoad"],
"Values": [ [ "296", "110.25", "7", "0", "15.55" ] ]
```

19. Press F5 or select Debug, Start to run your client application.

20. Verify the output.

```
C:\Python27\python.exe                                          —  □  ×
{"Results":{"output1":{"type":"table","value":{"ColumnNames":["WallArea","RoofArea","OverallHeight","GlazingArea","Heati
ngLoad","Scored Label Mean","Scored Label Standard Deviation"],"ColumnTypes":["Double","Double","Double","Double","Doubl
e","Double","Double"],"Values":[["296","110.25","7","0","15.55","0.482142857142857","0.0691609440506184"]]}}}}
Press any key to continue . . .
```

21. Press Enter to close the application and return to Visual Studio.

22. Close Visual Studio, the Python application portion is complete.

Summary

In this lab you have:

- Prepared and deployed a web service using Azure ML Studio
- Created a simple client application using C# and Visual Studio
- Created a simple client application using Python and Visual Studio