Data Science and Machine Learning Essentials

Lab 1 – Getting Started with Azure Machine Learning

By Graeme Malcolm and Stephen Elston

Overview

In this lab, you will learn how to open and navigate Microsoft Azure Machine Learning (Azure ML) Studio. You will also learn how to create and run experiments and to use SQL in Azure ML modules.

What You'll Need

To complete this lab, you will need the following:

- An Azure ML account.
- A web browser and Internet connection.
- The lab files for this lab

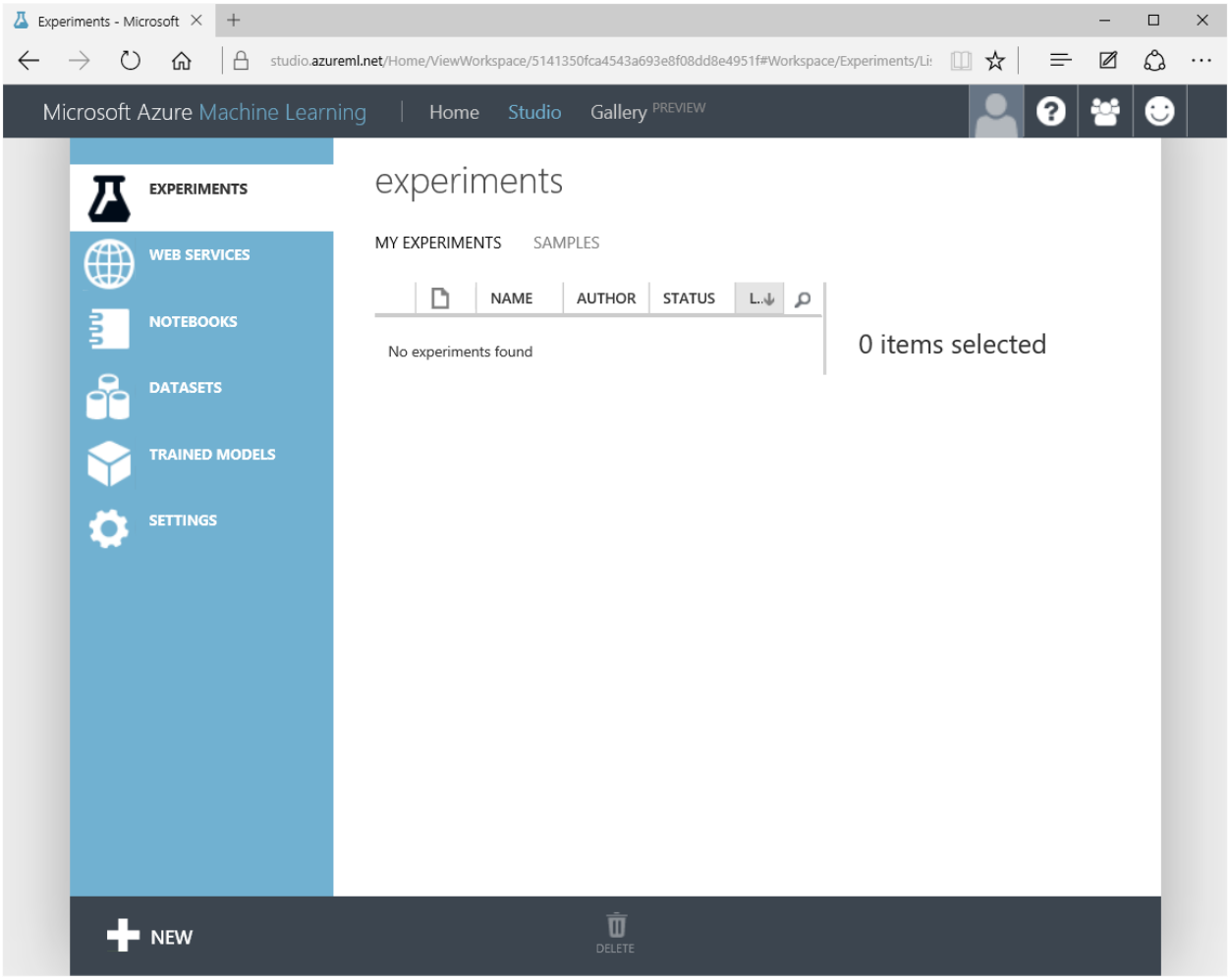
Note: To set up the required environment for the lab, follow the instructions in the **Setup Guide** for this course. Then download and extract the lab files for this lab.

Creating an Azure ML Experiment

Azure ML enables you to create experiments in which you can manipulate data, create predictive models, and visualize the results. In this exercise, you will create a simple experiment in which you will explore a sample dataset that contains details of bike rentals, from which you would like to predict the number of bike rentals on a given day based on seasonal and weather variables.

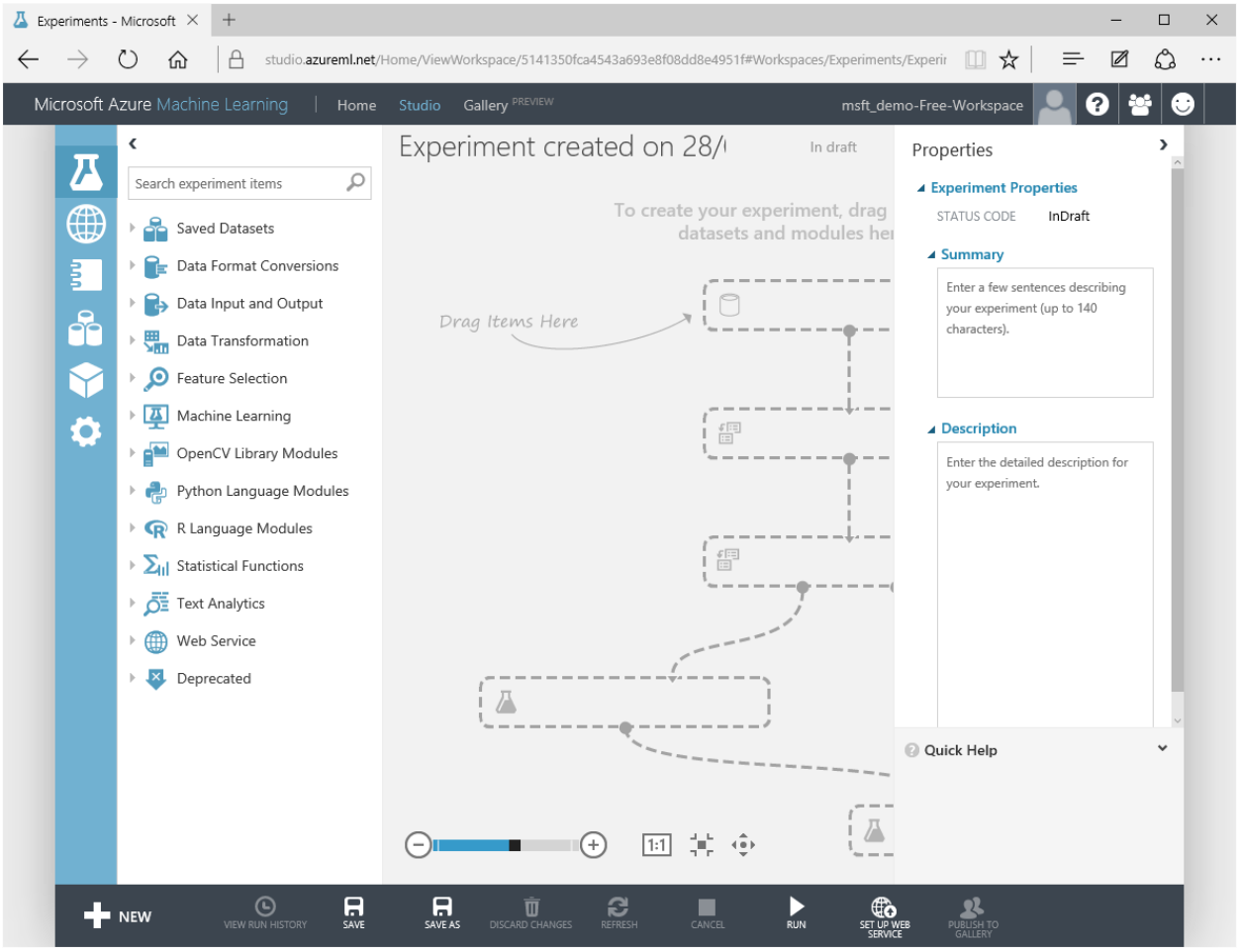
Sign into Azure ML Studio

1. Open a browser and browse to https://studio.azureml.net.
2. Click **Sign In** and sign in using the Microsoft account associated with your free Azure ML account.
3. If the **Welcome** page is displayed, close it by clicking the **OK** icon (which looks like a checkmark). Then, if the **New** page (containing a collection of Microsoft samples) is displayed, close it by clicking the **Close** icon (which looks like an X).
4. You should now be in Azure ML Studio with the **Experiments** page selected, which looks like the following image (if not, click the **Studio** tab at the top of the page).

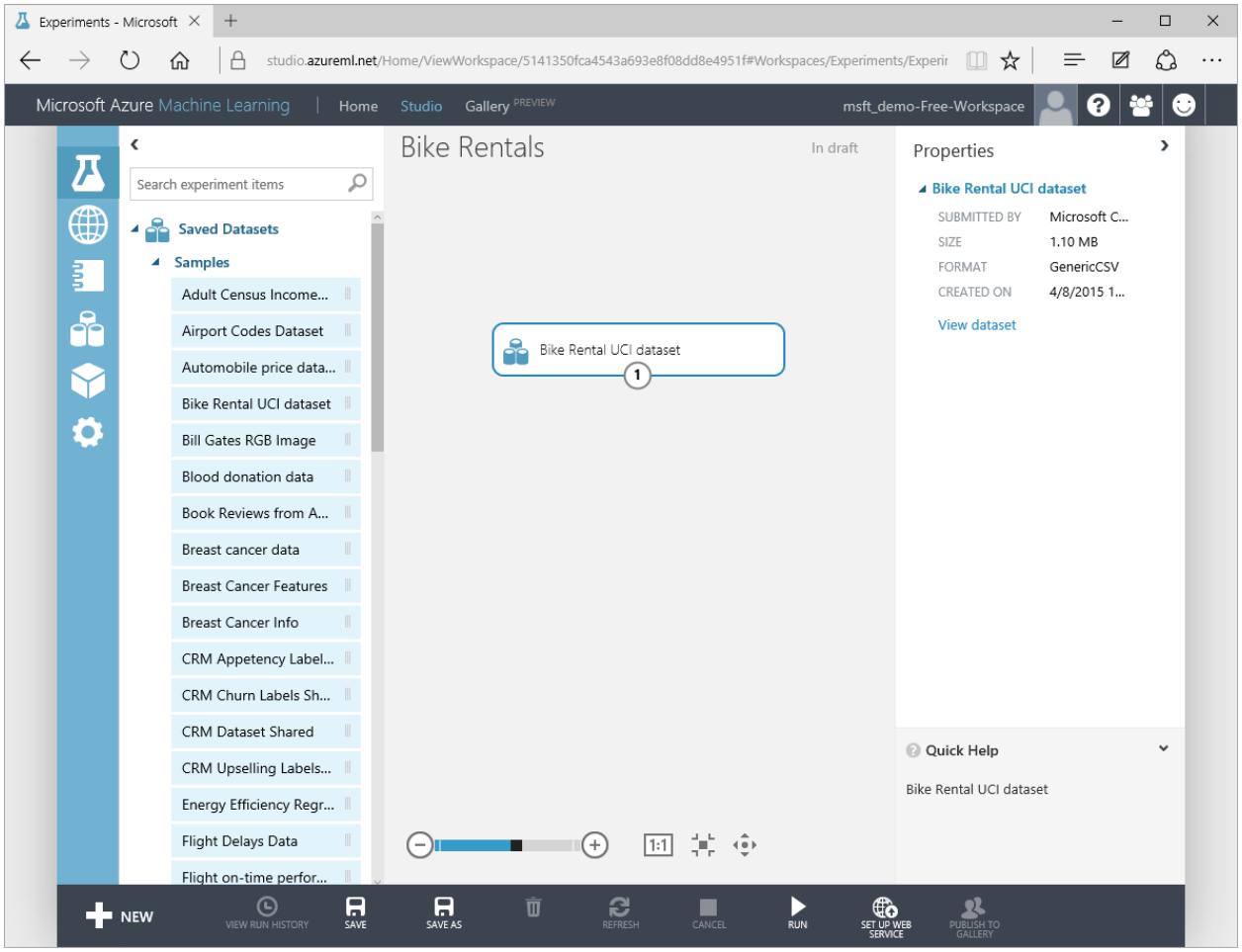


Create an Experiment and Add Modules

1. In the Studio, at the bottom left, click **NEW**. Then in the collection of Microsoft samples, select **Blank Experiment**. This creates a blank experiment, which looks similar to the following image.

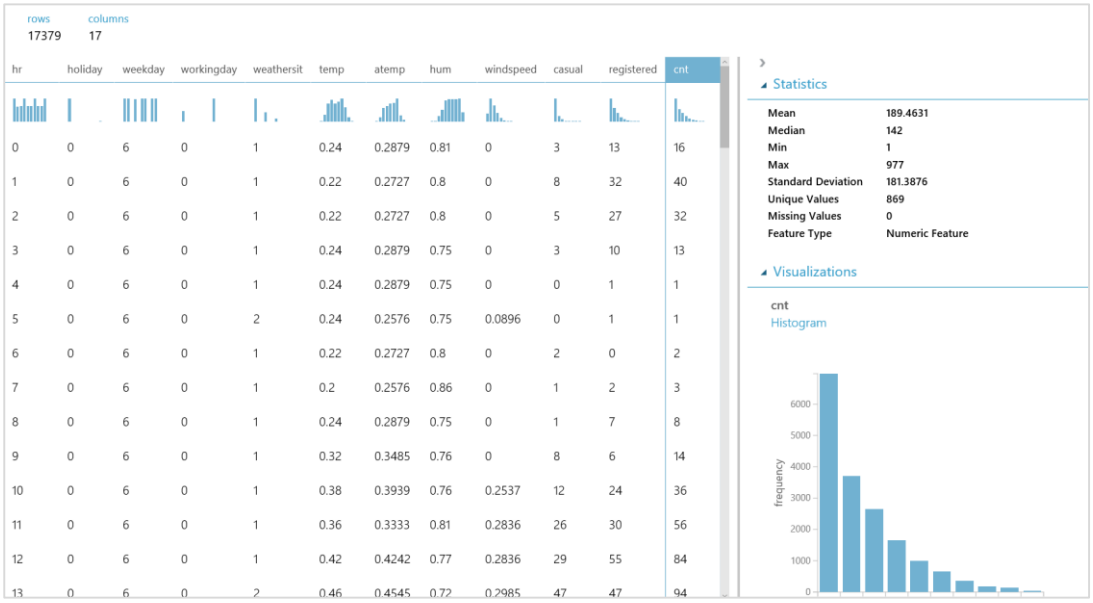


2. Change the title of your experiment from "Experiment created on *today's date*" to "Bike Rentals"
3. In the experiment items pane on the left, expand **Saved Datasets**, expand **Samples**, and drag **Bike Rental UCI dataset** to the experiment canvas in the middle of the page, as shown in the following image.



Note: The **Bike Rentals UCI** dataset is one of a number of sample datasets provided with Azure ML. In Module 2 of this course, you'll learn how to upload your own datasets and consume data from external data sources.

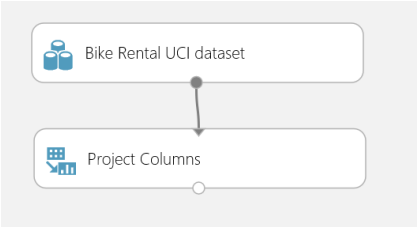
4. Select the **Bike Rental UCI** dataset on the canvas, and note that it has a single output port (indicated as a circle containing the value **1** at the bottom of the dataset icon). Right-click this output port and click **Visualize** to see the data that the dataset contains.
5. In the dataset, scroll the table pane on the left if necessary to see the **cnt** column, and then click the **cnt** column header so that a column summary and histogram for that column is displayed as shown in the following image. This column represents the count of bike rentals on a given day and hour.



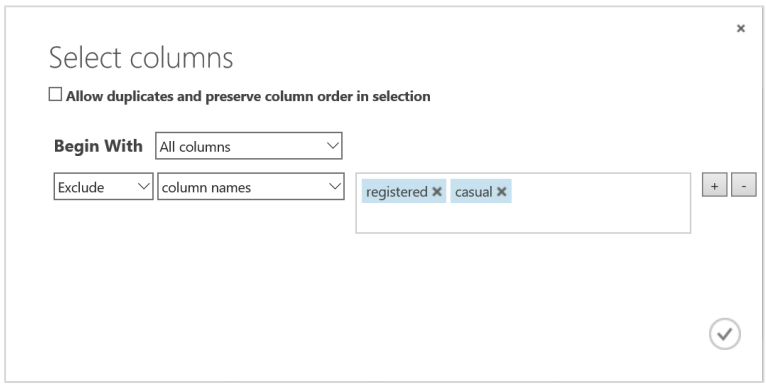
6. Select any of the other columns (for example **temp**, **atemp**, or **hum**), and note the statistics and histogram that is displayed. By visualizing statistics about the distribution of values in your data, and the relationships between values in columns, you can learn a lot about your data and refine it to build a more effective and accurate predictive model.

> **Note**: You will learn about more advanced techniques for visualizing data in Module 2 of this course.

7. Close the dataset, and in the experiment items pane, in the search box, type "Project Columns". Then, in the filtered experiment items pane, under **Data Transformation** and **Manipulation**, drag the **Project Columns** module to the canvas and place it under the **Bike Rental UCI dataset**.
8. Click the output port of the **Bike Rentals UCI dataset**, and drag it to the input port at the top of the **Project Columns** module to connect the items. Your experiment should now look like the following image.

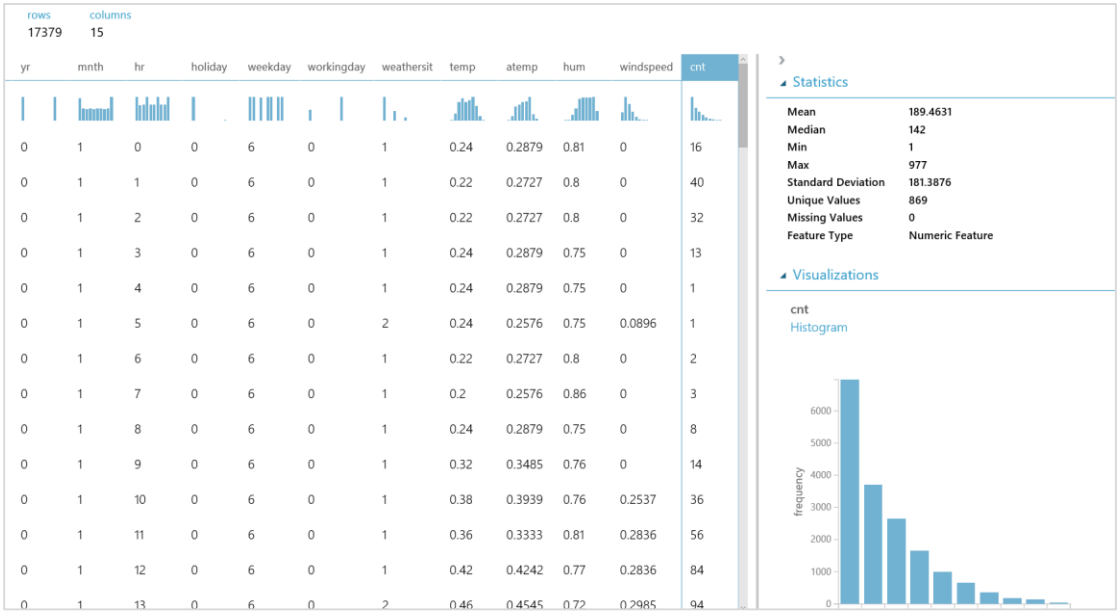


9. Select the **Project Columns** module, and in the **Properties** pane on the right, click **Launch column selector**. The column selector is a common user interface element in Azure ML modules, and enables you to select the columns you want to use in the module. In this case, the **Project Columns** module is used to filter out columns you don't need, so that only the columns you want to use are passed (or *projected*) into the data flow for the next module.
10. In the **Select columns** dialog box, select option to begin with **all columns**, and **exclude** the **registered** and **casual** column names as shown in the image below. Then click the **OK** icon to close the column selector.



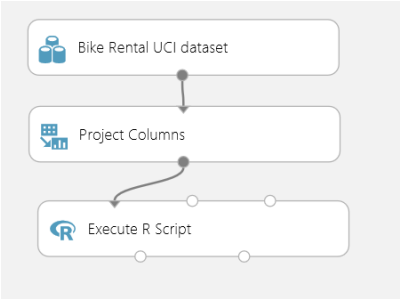
The **cnt** column is simply the **registered** and **casual** columns added together, so while including these columns in the model would make predicting the total number of rentals quite easy, the model might not work well when the number of casual and registered bike rentals are unknown!

11. On the toolbar at the bottom of the page, click **SAVE** to save the experiment. Then click **RUN** to run the experiment.
12. When the experiment has finished running, note the status displayed at the top-right of the experiment canvas and the green checkmark that indicates that the **Project Columns** module completed successfully.
13. Visualize the **Result Dataset** output port of the **Project Columns** module, and verify that the **registered** and **casual** columns have been removed, as shown here. Then close the results dataset.



Add Script Modules to an Experiment

1. In the experiment items pane, search for "Execute R Script". Then drag the **Execute R Script** module to the experiment canvas, under the **Project Columns** module; and connect the output port from the **Project Columns** module to the first (left-most) input port of the **Execute R Script** module so that your experiment looks like this:



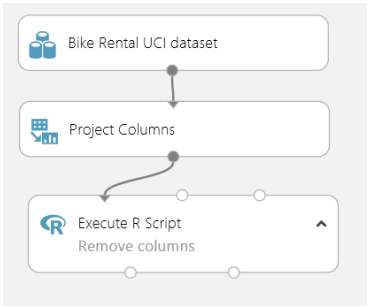
Note: R is a commonly used scripting language in data science experiments, and it enables you to include custom logic in an Azure ML experiment. You'll learn more about using R in data science experiments in Module 2, but for now, you'll use a simple R script to remove some more columns from the dataset.

2. Select the **Execute R Script** module, and in the **Properties** pane, replace the default R script with the following code. You can copy and paste this code from the **Bike Rentals Code.txt** file in the folder where you extracted the lab files for this lab:

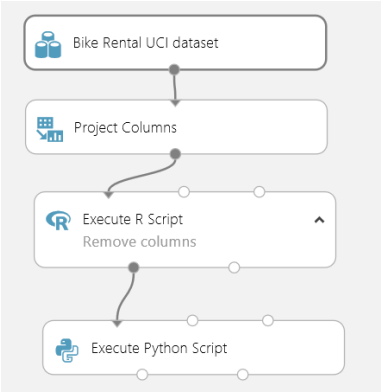
```
frame1 <- maml.mapInputPort(1)
## Delete instant and dteday columns
frame1$instant <- NULL
frame1$dteday <- NULL
maml.mapOutputPort('frame1')
```

   This code creates an R data frame from the data that is passed to the first input port of the script module, and then removes the **instant** and **dteday** columns, before passing the modified data frame to the output port of the script module. Because scripts are so flexible, it can be useful to document what the script does as a comment on the module in the Azure ML experiment.

3. Right-click the **Execute R Script** module and click **Edit Comment**. Then type "Remove columns" in the comment box, and click a blank area of the canvas to finish editing the comment, and click the **v** icon in the **Execute R Script** module to expand the comment. Your experiment should now look like this:



4. In the experiment items pane, search for "Execute Python Script". Then drag the **Execute Python Script** module to the experiment canvas, under the **Execute R Script** module, and connect the **Result Dataset** output port (the left-most output) from the **Execute R Script** module to the first (left-most) input port of the **Execute Python Script** module so that your experiment looks like this:



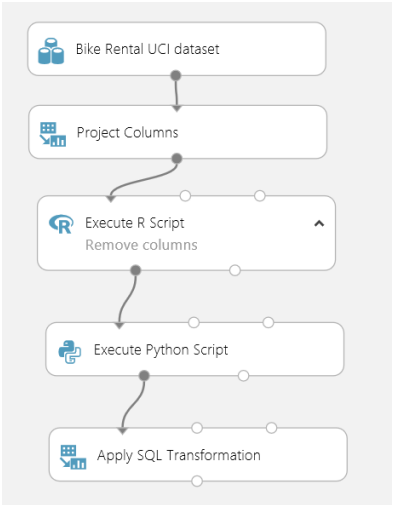
Note: Python is another commonly used scripting language in data science experiments; and like R, it enables you to include custom logic in an Azure ML experiment. You'll learn more about using Python in data science experiments in Module 2, and throughout the rest of this course you'll have the opportunity to choose either R or Python for scripting tasks.

5. Select the **Execute Python Script** module, and in the **Properties** pane, replace the default Python script with the following code. You can copy and paste this code from the **Bike Rentals Code.txt** file in the folder where you extracted the lab files for this lab – make sure you indent the code under the comment **## delete yr and weathersit columns** (the code uses four spaces to indent lines):

```
def azureml_main(dataframe1 = None, dataframe2 = None):
## delete yr and weathersit columns
    dataframe1 = dataframe1.drop('yr', 1)
    dataframe1 = dataframe1.drop('weathersit', 1)
    return dataframe1
```

   This code creates a Python pandas data frame from the data that is passed to the first input port of the script module, and then removes the **yr** and **weathersit** columns, before passing the modified data frame to the output port of the script module.

6. In the experiment items pane, search for "Apply SQL Transformation". Then drag the **Apply SQL Transformation** module to the experiment canvas, under the **Execute Python Script** module, and connect the **Result Dataset** output port (the left-most output) from the **Execute Python Script** module to the first (left-most) input port of the **Apply SQL Transformation** module so that your experiment looks like this:



Note: The **Apply SQL Transformation** module enables you to write custom log in SQLite, a variant of the ANSI SQL language. If you are familiar with Transact-SQL in Microsoft databases such as SQL Server and Azure SQL Database, you can apply your SQL knowledge to working with data in an Azure ML experiment.

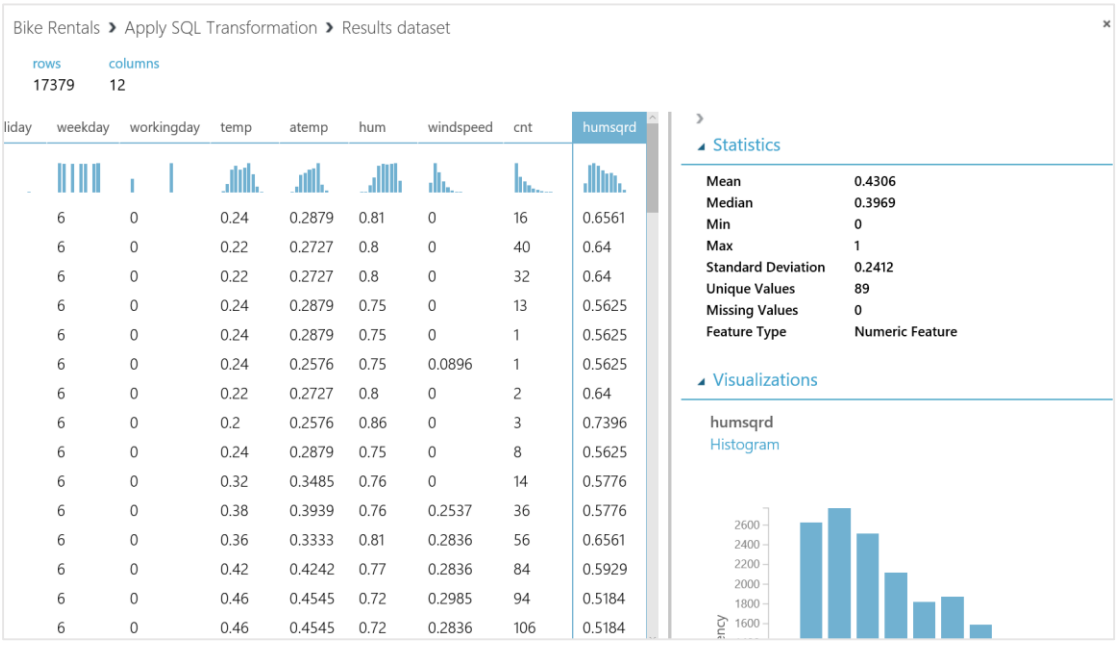
7. Select the **Apply SQL Transformation** module, and in the **Properties** pane, replace the default SQL script with the following code. You can copy and paste this code from the **Bike Rentals Code.txt** file in the folder where you extracted the lab files for this lab:

```
SELECT *, hum*hum AS humsqrd FROM t1
WHERE mnth BETWEEN 1 AND 12
AND hr BETWEEN 0 AND 23;
```

   This code returns all columns for the rows passed to the input port in which the **mnth** column value is between 1 and 12, and the **hr** column value is between 0 and 23. This removes any rows that contain an invalid month or hour value. Additionally, it generates a new column named **humsqrd** that contains the humidity value squared. Sometimes you can improve the performance of a predictive model by generating polynomial columns such as this, in a technique called *feature engineering*.

   Note: You will learn about more options for filtering and cleansing data in Module 2. Feature engineering and column selection for model training are discussed in more depth in Module 3.

8. Save and run the experiment. Then, when the experiment has finished, visualize the **Results dataset** output of the **Apply SQL Transformation** module, and view the filtered data, as shown here. Then close the results dataset.

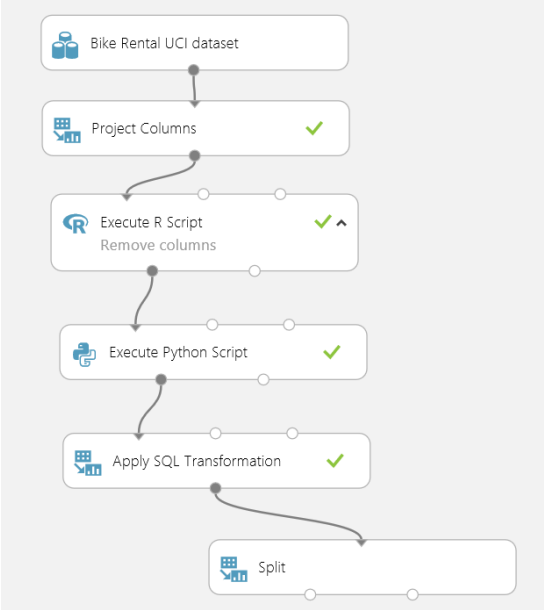


Creating a Model

Now that you have created a simple experiment that processes data, you can use the data to train a predictive model. In this exercise, you will use the data to create a model that tries to predict the number of bike rentals on a given day and hour based on the features in your dataset.

Split the Data

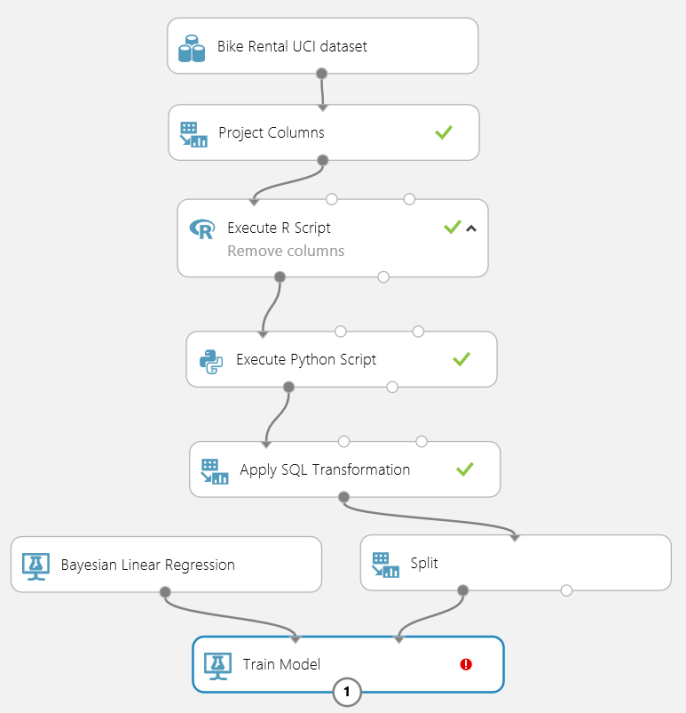
1. In the experiment items pane, search for "Split". Then drag the **Split** module that is found in the **Data Transformation** category to the canvas and place it under the **Apply SQL Transformation** module, and connect it to the end of the workflow as shown here:



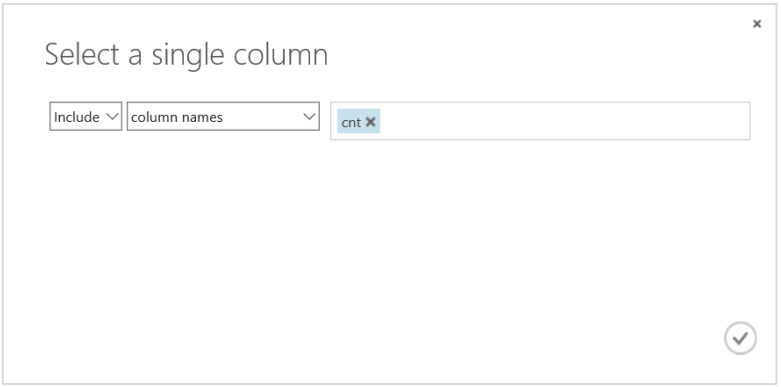
2. Select the **Split** module, and in the **Properties** pane, view the default split settings; which split the data randomly into two equal datasets.

Add and Train a Model

1. In the experiment items pane, search for "Regression", and view the range of regression-based models that are supported in Azure ML. Then, drag the **Bayesian Linear Regression** module that is found in the **Machine Learning > Initialize Model > Regression** category to the canvas, and place it next to the **Split** module without connecting it to anything.
2. Select the **Bayesian Linear Regression** module, and in the **Properties** pane, note that you can configure a **Regularization weight** property that determines how the regularization function in the model is calculated to reduce over-fitting – leave this at its default value of 1.
3. In the experiment items pane, search for "Train Model", and drag the **Train Model** module to the canvas below the **Bayesian Linear Regression** and **Split** modules. Then connect the output from the **Bayesian Linear Regression** module to the first (left-most) input of the **Train Model** module; and connect the first (left-most) output port of the **Split** module to the second (right-most) input of the **Train Model** module, as shown here:



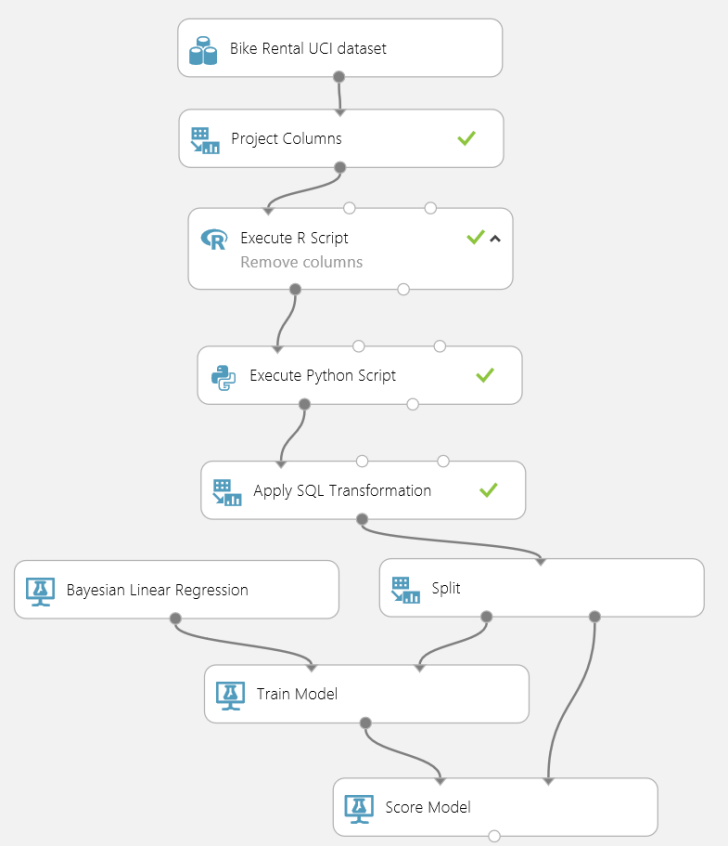
4. Select the **Train Model** module, and in the properties pane, launch the column selector and select the **cnt** column, as shown here:



The experiment is now configured to train a Bayesian Linear Regression model using a training dataset that consist of half of your original data. When training the model, Azure ML will attempt to determine a suitable function that can be used to predict the **cnt** label value based on the other feature columns in the dataset.

Note: Model training and evaluation are discussed in more depth in Module 3 of this course.

5. In the experiment items pane, search for "Score Model", and drag the **Score Model** module to the canvas below the **Train Model** module. Then connect the output from the **Train Model** module to the first (left-most) input of the **Score Model** module; and connect the second (right-most) output port of the **Split** module to the second (right-most) input of the **Score Model** module, as shown here:

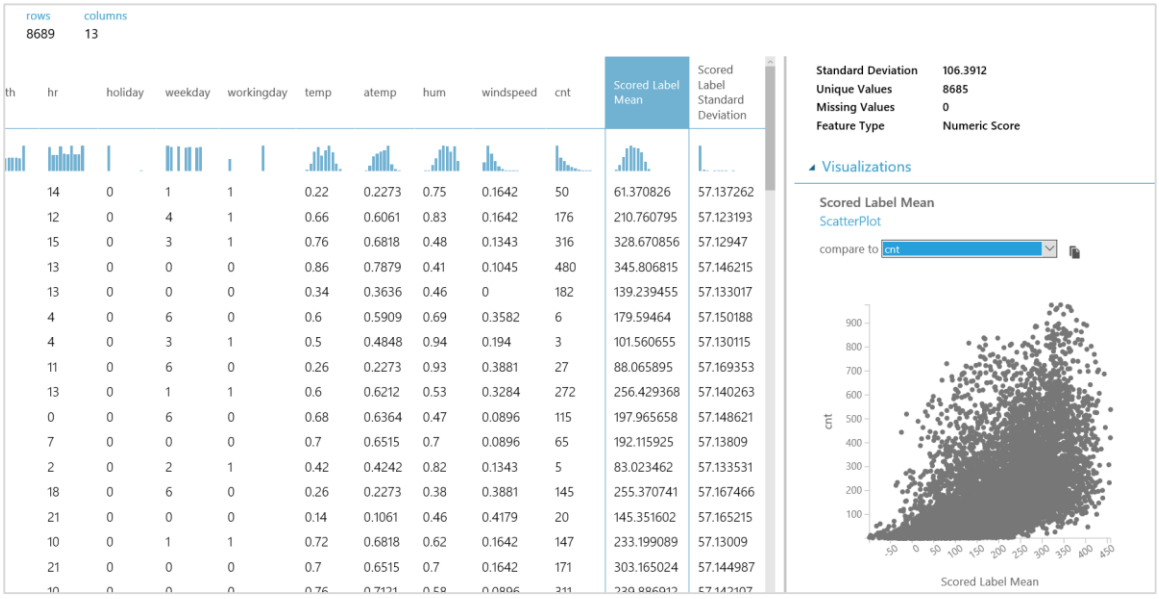


The experiment will score the trained model by comparing the predicted and actual values for the **cnt** value in the test dataset, which consists of the second half of data from the **Split** module.

6. Save and run the experiment.

View the Scored results

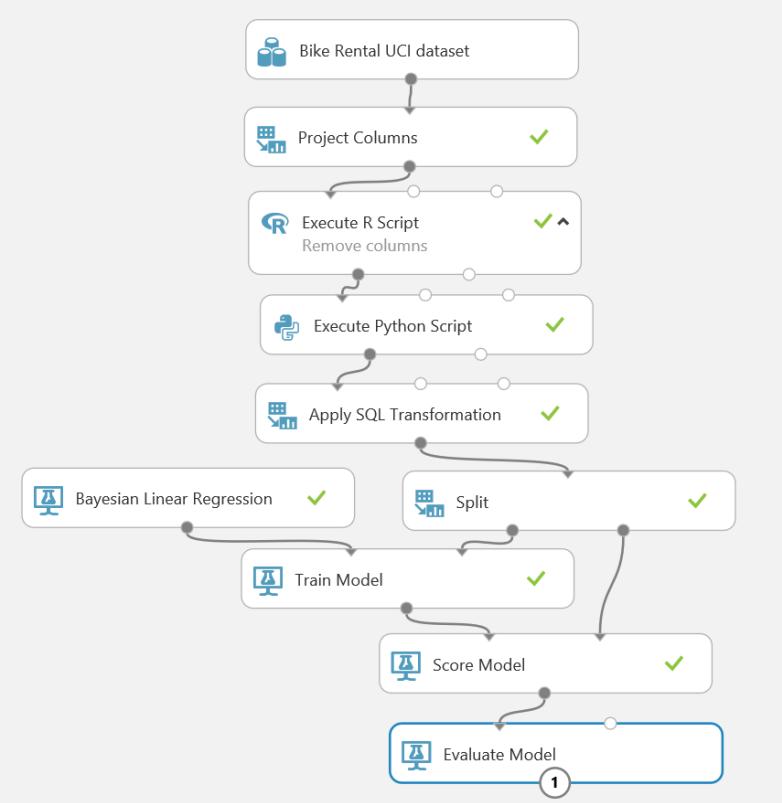
1. When the experiment is finished, visualize the output from the **Score Model** module.
2. Select the **Scored Label Mean** column. This represents the predicted values for the **cnt** label.
3. Compare the first ten or so values in the **Scored Label Mean** and **cnt** columns, noting that some predictions are reasonably close, but others are significantly wrong.
4. With the **Scored Label Mean** column selected, in the **Visualizations** area, in the **compare to** list, select **cnt**; and view the resulting scatter plot chart as shown here:



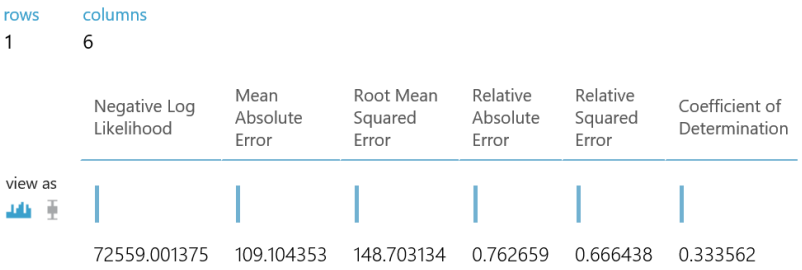
5. Note that the scatter plot shows a loose correlation between the predicted and actual values, but the margin of error is quite large. The plots show a slight diagonal trend – a perfect model would show a single diagonal line in which the predicted value always matches the actual value.
6. Close the scored dataset.

Evaluate the Model

1. In the experiment items pane, search for "Evaluate Model", and drag the **Evaluate Model** module to the canvas below the **Score Model** module. Then connect the output from the **Score Model** module to the first (left-most) input of the **Evaluate Model** module, as shown here:



2. Save and run the experiment. Then, when the experiment has finished, visualize the **Evaluation Results** dataset output of the **Evaluate Model** module, which should look similar to this:



3. The values shown are measurements of the accuracy of the model when comparing the label values that it predicts to the known values in the test dataset. For example, in the case of this regression model, the **Relative Squared Error** value indicates how well the model explains variance in the predicted label value and the known label value, with a lower number indicating a better predictive result. The specific measurements vary by model type, and you can connect two scored models to the **Evaluate Model** module in order to compare their effectiveness.

   Note: You will learn more about evaluating models in Module 3 of this course.

4. Close the evaluation results dataset.

Save and Close the Experiment

1. Click a blank are of the experiment canvas to select the experiment. Then in the **Properties** pane, enter the **Summary** *Bike rental prediction experiment* and the **Description** *A simple experiment to predict bike rentals*. Then save the experiment.
2. In the toolbar at the bottom of the page, note (but do not click) the **Set up Web Service** button. This enables you to create a web service for your experiment, so that it can be used by client applications that need to predict bike rentals from new seasonal and meteorological data.

   Note: You will learn about considerations for publishing experiments as web services in Module 4 of this course.

3. In the Azure ML Studio page, on the left side, click the **Experiments** icon and note that your experiment is listed. You can return to it at any time from here.

Summary

This lab was designed to help you become familiar with the Azure ML Studio environment and the basic process of creating and scoring a model. The model you produced is not particularly effective at predicting an accurate value for the number of bike rentals, so clearly some iterative work would be required to further cleanse the data, identify the most meaningful features to include in the model, and compare the results when using a range of different algorithms.

In the rest of this course, you will learn how to employ a range of techniques to prepare data for modeling, build effective models, and evaluate model performance to create a suitable accurate predictive solution.