Labs Azure Machine Learning

Getting Started with Azure Machine Learning

# Context

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.

Start our predictive model by creating an Azure ML experiment.

1. Sign into Azure ML Studio: <https://studio.azureml.net/?selectAccess=true&o=1>
2. Click **NEW** and select the **Blank Experiment**.
3. Change title to ‘*Bike Rentals*’
4. Let’s start with a dataset we want to analyze. Find the dataset **Bike Rental UCI dataset** (**Saved Dataset 🡪 Samples**) and drag it to the canvas.

Now we have the dataset, we want to explore our data. 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.

1. Select the Bike Rental UCI dataset on the canvas, and note that it has a single output port. Right-click this output port and click Visualize to see the data that the dataset contains.
2. Click the column **cnt** and check the histogram on the right.
3. Select any of the other columns (for example temp, atemp, or hum), and note the statistics and histogram that is displayed. After exploring, close the popup.

# Transformations

We know what the data looks like. In an analytical process we do this kind of steps multiple times (iterative process). After each step the data can look different which can change the distribution, relationships etc. At this point we want to transform the data. Therefor you can use modules with built-in transformations or modules for executing scripts. We will look at both.

We want to predict the number of bike rentals. It doesn’t matter if they are registered or casual clients. In this dataset the column cnt is simply the columns registered and casual 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. Removing these columns is what you want.

1. Find the module **Project Columns** (tip: use the search box) and place it to the canvas.
2. Click the output port of the **Bike Rental UCI dataset** and drag a line to the input port of **Project Columns**.
3. On the right you’ll notice a properties pane. With **Project Columns** selected, click the **Launch column selector** in this pane.
4. Make sure that the folowing columns are excluded. Then close the popup.
   1. **Instant**
   2. **Registered**
   3. **Casual**
   4. **Dteday**
   5. **Yr**
   6. **Weathersit**
   7. **Casual**
5. To see how the output of **Project Columns** looks like, we have to run the experiment first.
6. On the toolbar at the bottom of the page, click **SAVE** to save the experiment. Then click **RUN** to run the experiment.
7. 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.
8. Visualize the dataset from the output port of the **Project Columns** module and verify that the columns **registered** and **casual** have been removed. Then close the popup.

An important step in preparing your data for analysis is dealing with NULL values. In Azure ML you can use the module **Clean Missing Data**.

1. Add the **Clean Missing Data** module to the canvas.
2. In the properties pane on the right you see Cleaning mode with a dropdown. You can decide what you will do when your dataset contains a NULL value. Select **Remove entire row**.
3. In this pane on the top, you also see that all columns are selected. For now that’s ok, but you can apply different action (Cleaning mode) on missing values for different columns.
4. Drag a line from the output port of **Project Columns** to the input port of **Clean Missing Data**.

Azure ML contains many kind of transformations, models and evaluations. But not everything. It’s common to use script languages in data science experiments. It enables you to include custom logic in an Azure ML. You should use the language you prefer like SQL, R or Pyhton. Because of some trial limitations in Azure ML we can’t add R or Python scripts so we ‘ll add a SQL script to do a transformation.

1. Add the **Apply SQL Transformation** module to the canvas. Copy-paste the code below to the script editor in the properties pane.

SELECT season

,mnth

,hr

,holiday

,weekday

,workingday

,temp

,atemp

,hum

,square(hum) as hmsqrd

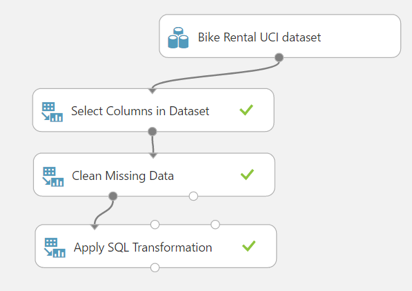
,windspeed

,cnt

FROM t1

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

1. Drag a line from the first output port of **Clean Missing Data** to the first input port of **Apply SQL Transformation**.
2. Save and Run the experiment. The experiment will look like the picture below.



1. Click on the outpu port of **Apply SQL Transformation** and then Visualize. You’ll notice that the script transformations has add the column humsqrd.

# Model

Now we have prepared the data for analysis, we can train a predictive model. In this exercise we want to predict the number of bike rentals on a given day and hour. The first thing we want to do is split the data in a training and test set. So we can train a model and test how well the model performs.

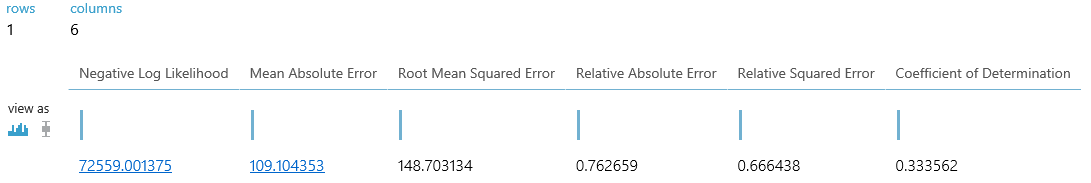
1. Add the **Split Data** module to the canvas.
2. Drag a line from the output port of **Apply SQL Transformation** to the input port of **Split Data**.
3. In the properties pane we can apply a splitting mode. Verify that **Split Rows** is selected as Splitting mode (randomly split of rows).
4. We need to decide which portion of rows we want to use for training and testing. It’s common to use between 60-80 percent for training and the rest for testing. Set the value for this at **0.7** which indicates that 70% of the rows will be in the first dataset (first output port) which we will use for training the model.
5. Find the module **Bayesian Linear Regression** and add this to the canvas. We use this algorithm to train our model.

Bayesian linear regression allows a fairly natural mechanism to survive insufficient data, or poor distributed data. It allows you to put a prior on the coefficients and on the noise so that in the absence of data, the priors can take over. More importantly, you can ask Bayesian linear regression which parts (if any) of its fit to the data is it confident about, and which parts are very uncertain (perhaps based entirely on the priors).

1. 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.
2. Add the **Train Model** module to the canvas.
3. In the properties pane, click the **Launch column selector**. Here we specify the column value we want to predict for new data. In this case it’s the number of bike rental which are stored in the column **cnt**. Add only this column and close the popup.
4. Drag a line from the output port of **Bayesian Linear Regression** to the first input port of **Train Model**.
5. Drag a line from the first output port of **Split Data** to the second input port of **Train Model**.

Now we have trained a model, we want to test and evaluate the model. This way we can - statistically – prove how good our model really is. We use the modules Score Model and Evaluate Model for this purpose.

1. Add the **Score Model** module to the canvas. This module knows how to score the data based on the algorithm you used to train the model.
2. Drag a line from the output port of **Train Model** to the first input port of **Score Model**. And drag a line from the second output port of **Split Data** to the second input port of **Score Model**. With the last line we ensure that we test the model on data we haven’t used for creating the model. This is essential for a good evaluation of your model.
3. Save and Run the experiment.
4. Click on the output port of **Score Model** and click Visualize.
5. Click the column **Scored Label Mean**. You will notice that some predictions are reasonably close and other are significantly wrong.
6. Further explore the columns and visualizations. Then close the popup.
7. Add the **Evaluate Model** module to the canvas. This module knows how to evaluate the data based on the algorithm you used to train the model.
8. Drag a line from the output port of **Score Model** to the first input port of **Evaluate Model**. You can evaluate up to two models within one Evaluate Model module. We only have one to evaluate.
9. Save and Run the experiment. The experiment will look like the picture below.
10. Click on the output port of **Evaluate Model** and click Visualize. The popup will look similar like this:



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. In this case it’s a bad model. A Relative Squared Error higher than 0.2 is in most cases not good.

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.

Webinars / sources / exercises / samples

* <https://github.com/Azure-Readiness/hol-azure-machine-learning>
* https://gallery.cortanaintelligence.com
* <https://github.com/Azure/azure-content-nlnl/blob/master/articles/machine-learning/machine-learning-data-science-for-beginners-the-5-questions-data-science-answers.md>
* <https://mva.microsoft.com/en-US/training-courses/cortana-intelligence-suite-endtoend-16972?l=W7jo0cDKD_7306218965>
* https://www.Kaggle.com