**Engineer and select features**

Lab Overview

This lab demonstrates the feature engineering process for building a regression model using bike rental demand prediction as an example. In machine learning predictions, effective feature engineering will lead to a more accurate model. We will use the Bike Rental UCI dataset as the input raw data for this experiment. This dataset is based on real data from the Capital Bikeshare company, which operates a bike rental network in Washington DC in the United States. The dataset contains 17,379 rows and 17 columns, each row representing the number of bike rentals within a specific hour of a day in the years 2011 or 2012. Weather conditions (such as temperature, humidity, and wind speed) were included in this raw feature set, and the dates were categorized as holiday vs. weekday etc.

The field to predict is cnt which contains a count value ranging from 1 to 977, representing the number of bike rentals within a specific hour. Our main goal is to construct effective features in the training data, so we build two models using the same algorithm, but with two different datasets. Using the Split Data module in the visual designer, we split the input data in such a way that the training data contains records for the year 2011, and the testing data, records for 2012. Both datasets have the same raw data at the origin, but we added different additional features to each training set:

* Set A = weather + holiday + weekday + weekend features for the predicted day
* Set B = number of bikes that were rented in each of the previous 12 hours

We are building two training datasets by combining the feature set as follows:

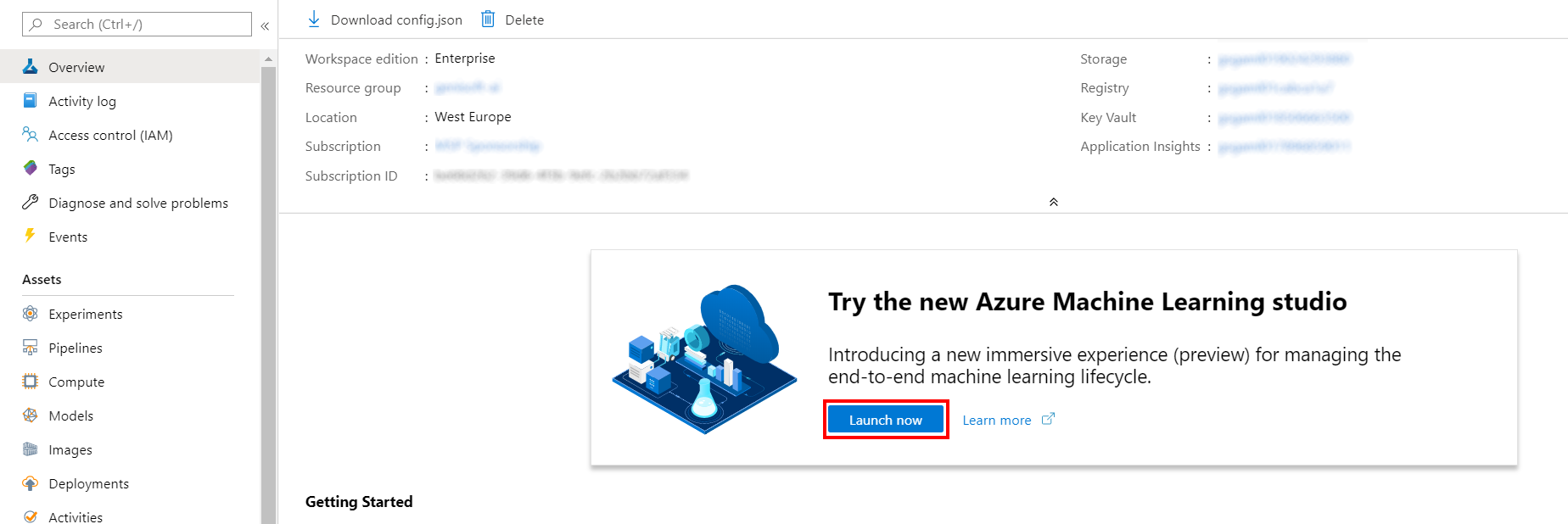
* Training set 1: feature set A only
* Training set 2: feature sets A+B

For the model, we are using regression because the number of rentals (the label column) contains continuos real numbers. As the algorithm for the experiment, we will be using the Boosted Decision Tree Regression.

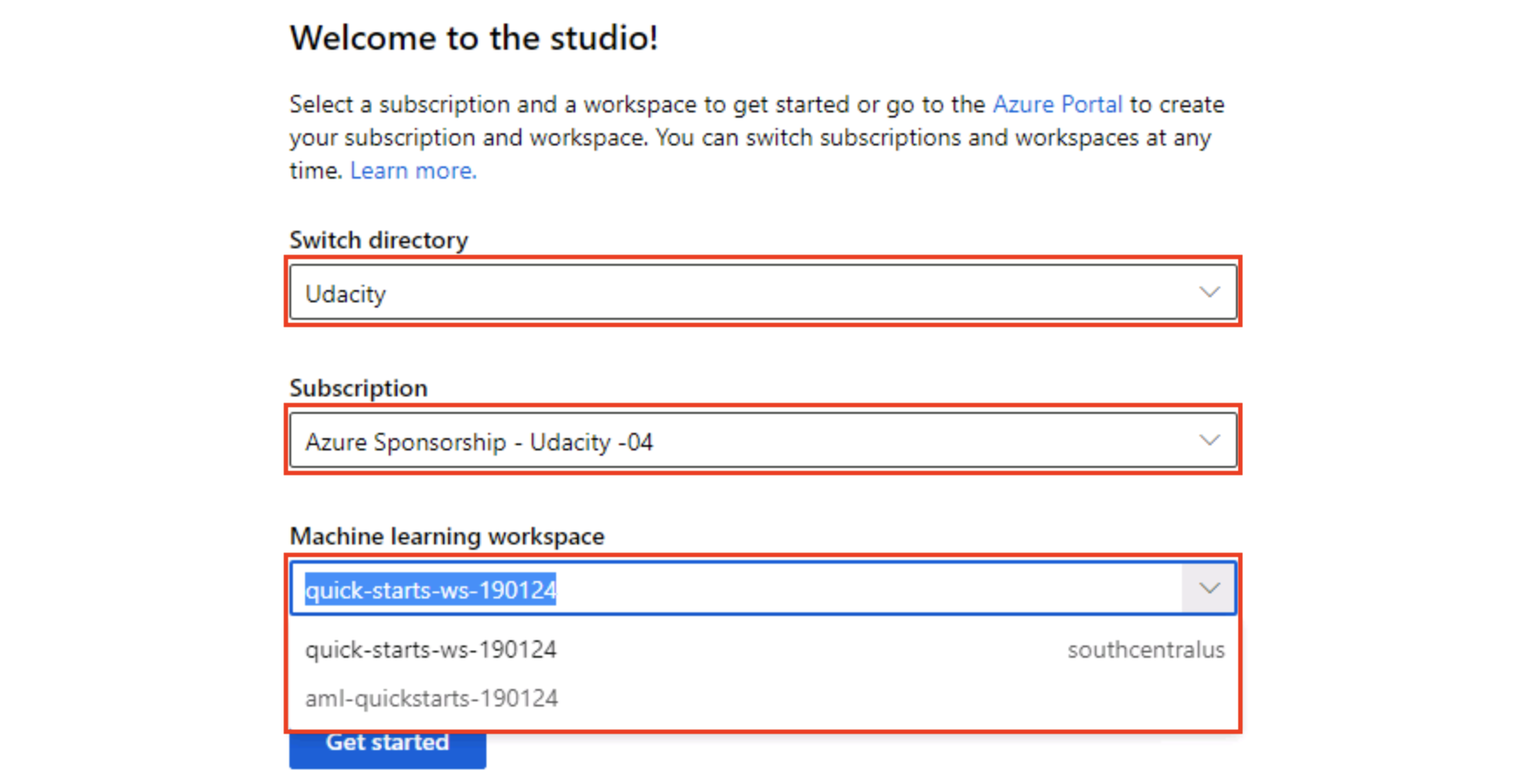
**Exercise 1: Data pre-processing using the Pipeline Authoring Editor**

Task 1: Upload Dataset

1. In [Azure portal](https://portal.azure.com/), open the available machine learning workspace.
2. Select **Launch now** under the **Try the new Azure Machine Learning studio** message.

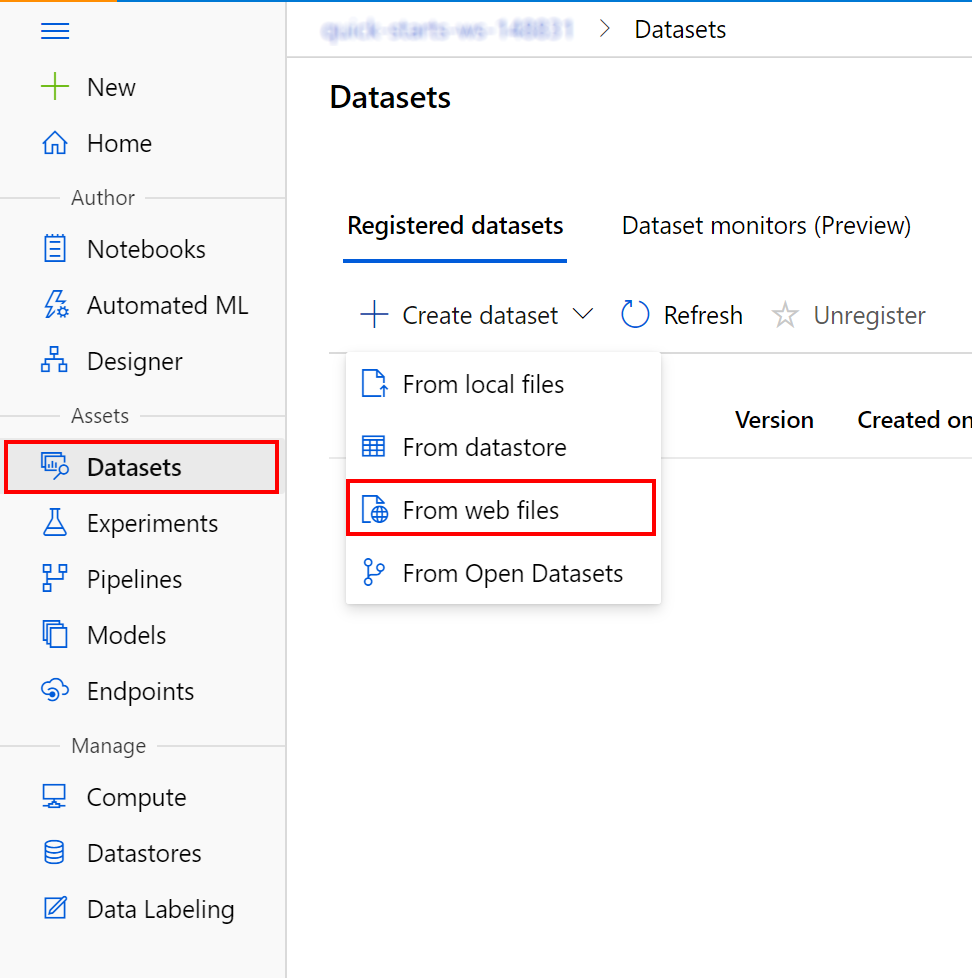


1. When you first launch the studio, you may need to set the directory and subscription. If so, you will see this screen:

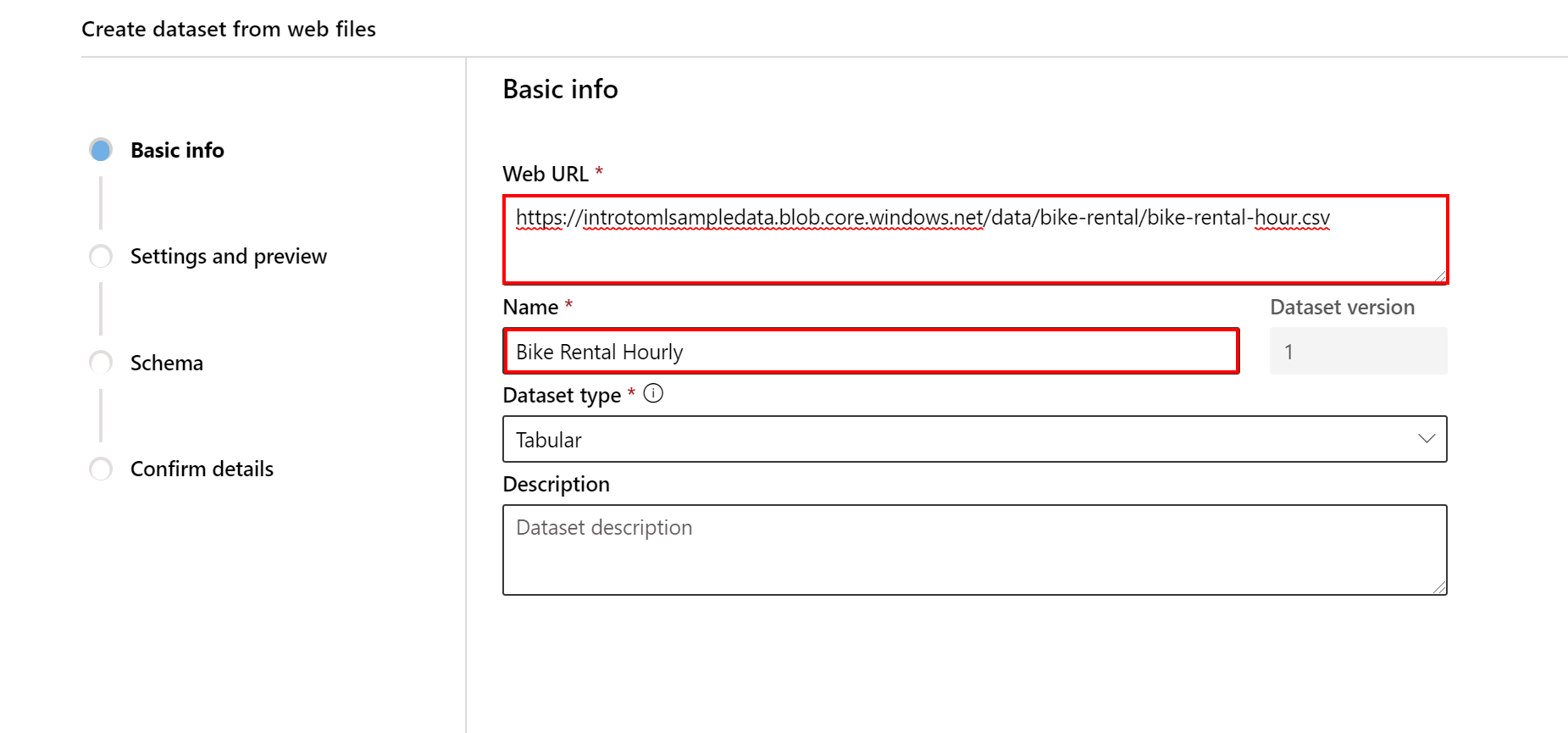


For the directory, select **Udacity** and for the subscription, select **Azure Sponsorship**. For the machine learning workspace, you may see multiple options listed. **Select any of these** (it doesn’t matter which) and then click **Get started**.

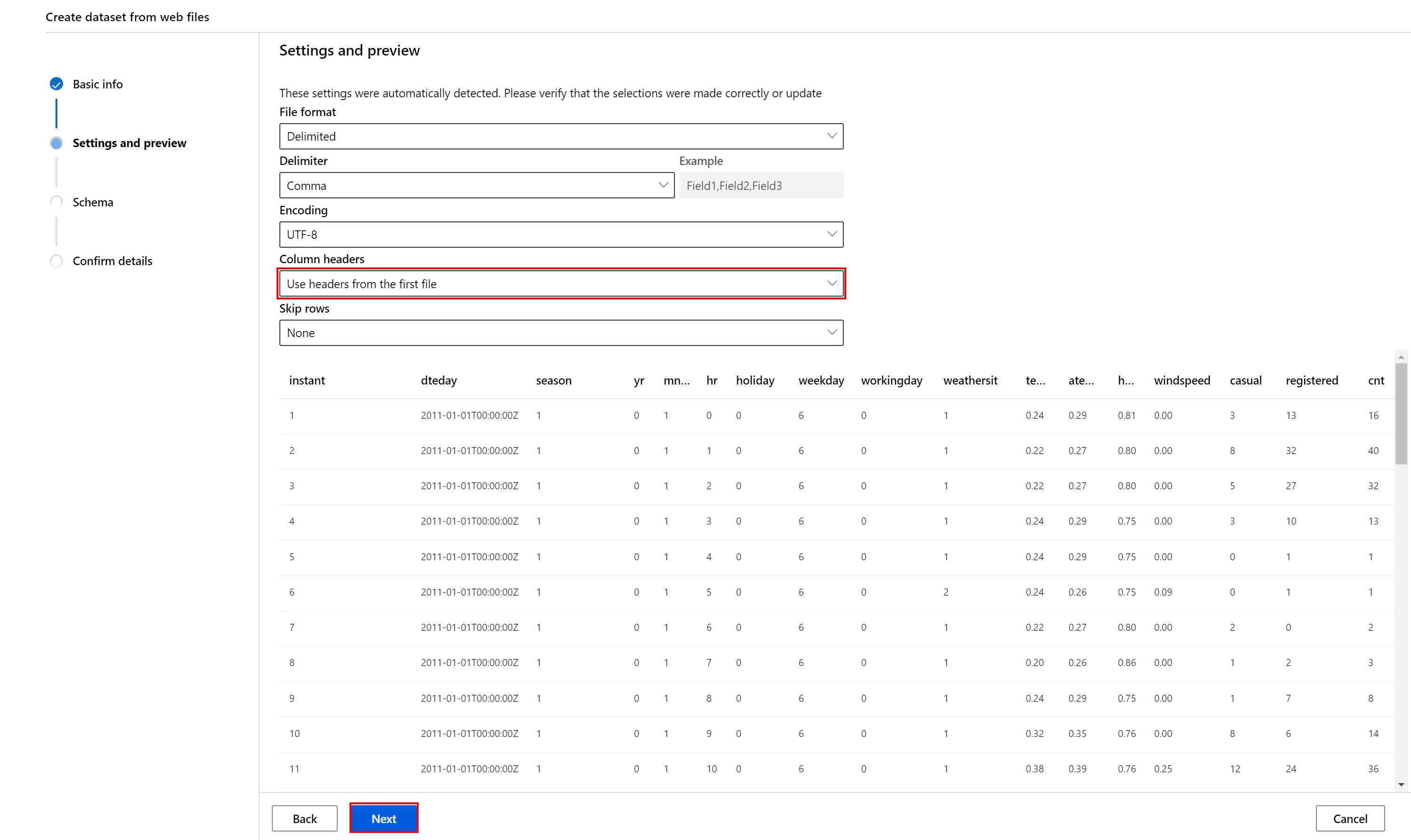
1. From the studio, select **Datasets, + Create dataset, From web files**. This will open the Create dataset from web files dialog on the right.



1. In the Web URL field provide the following URL for the training data file: https://introtomlsampledata.blob.core.windows.net/data/bike-rental/bike-rental-hour.csv
2. Provide Bike Rental Hourly as the Name, leave the remaining values at their defaults and select **Next**.

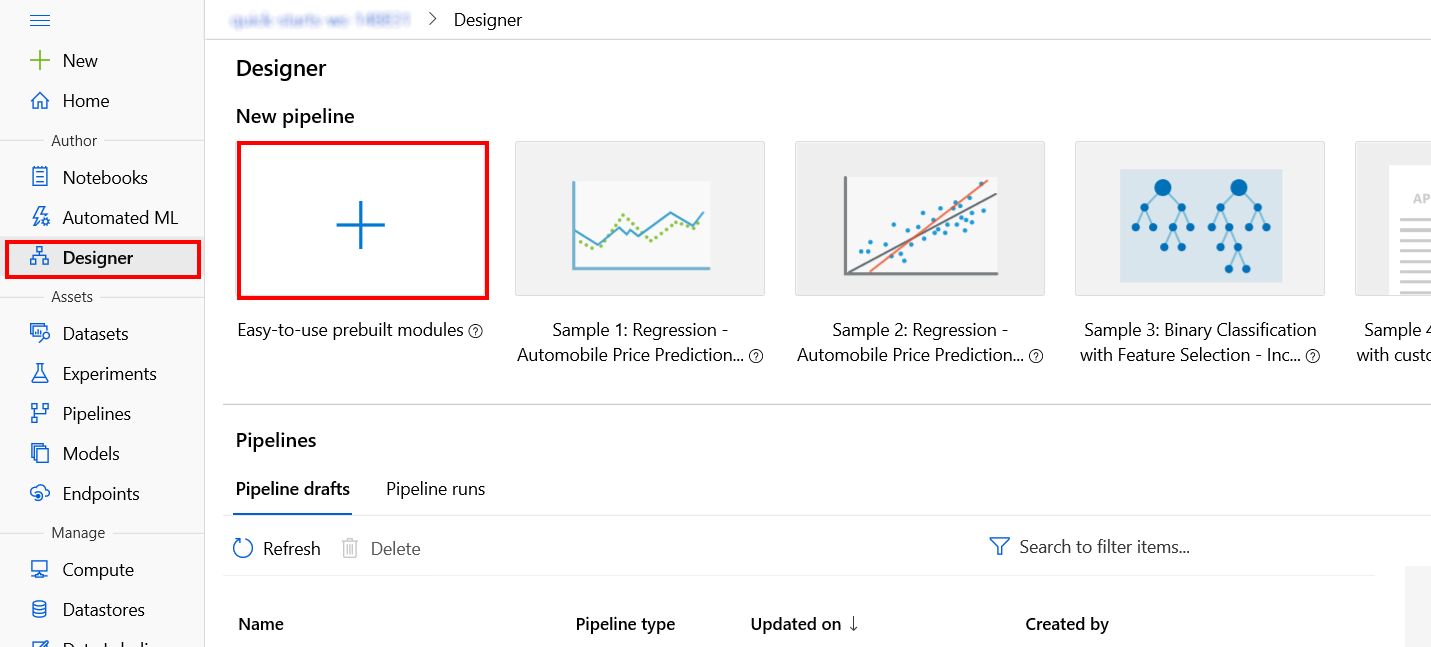


1. Select the option to Use headers from the first file in the **Settings and preview** dialog and then select **Next**, **Next** and **Create** to confirm all details in registering the dataset.



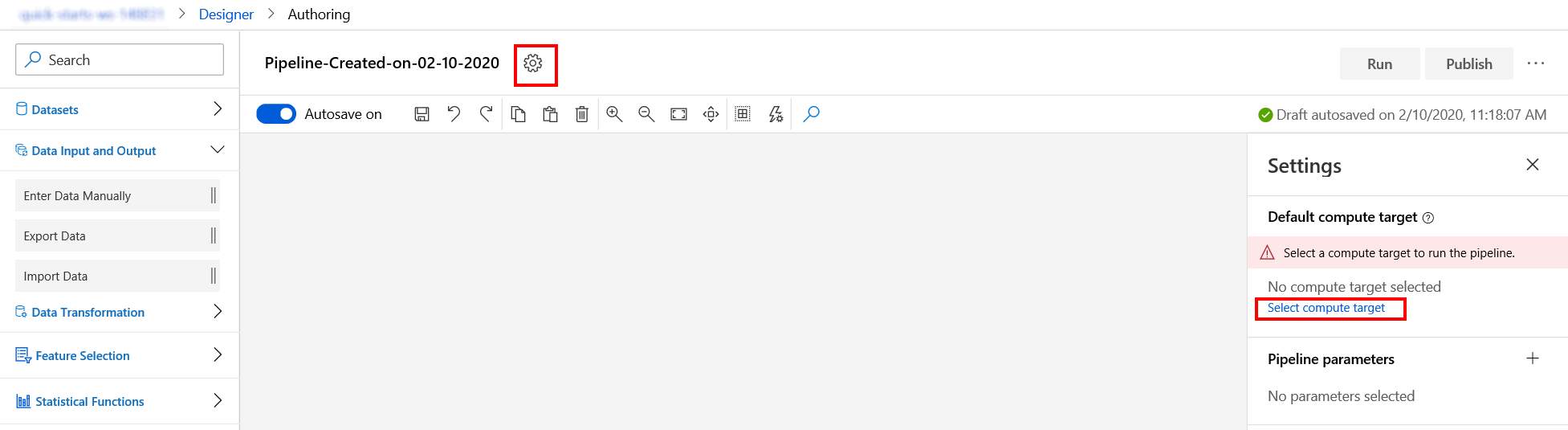
Task 2: Open Pipeline Authoring Editor

1. From the left navigation, select **Designer, +**. This will open a visual pipeline authoring editor.



Task 3: Setup Compute Target

1. In the settings panel on the right, select **Select compute target**.

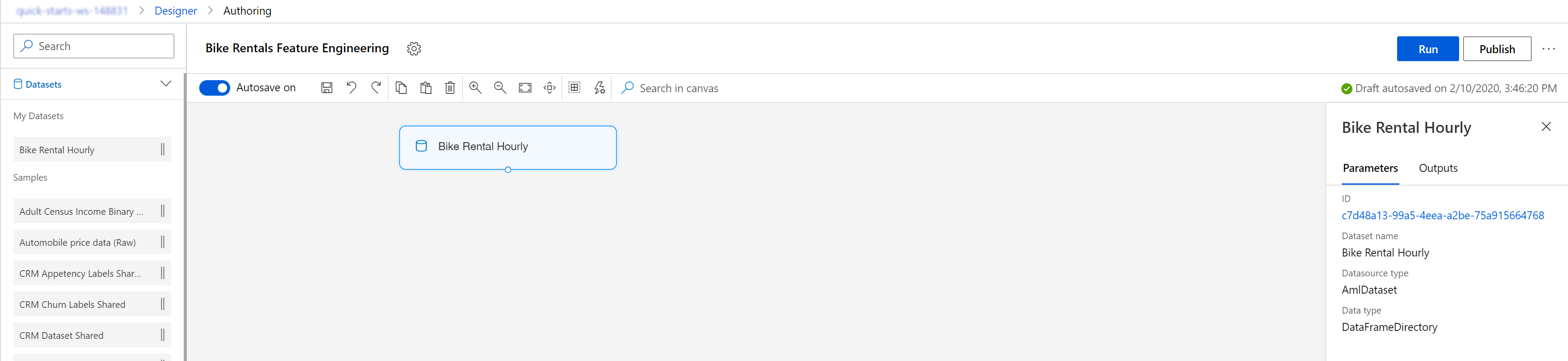


1. In the Set up compute target editor, select the existing compute target, choose a name for the pipeline draft: Bike Rental Feature Engineering and then select **Save**.

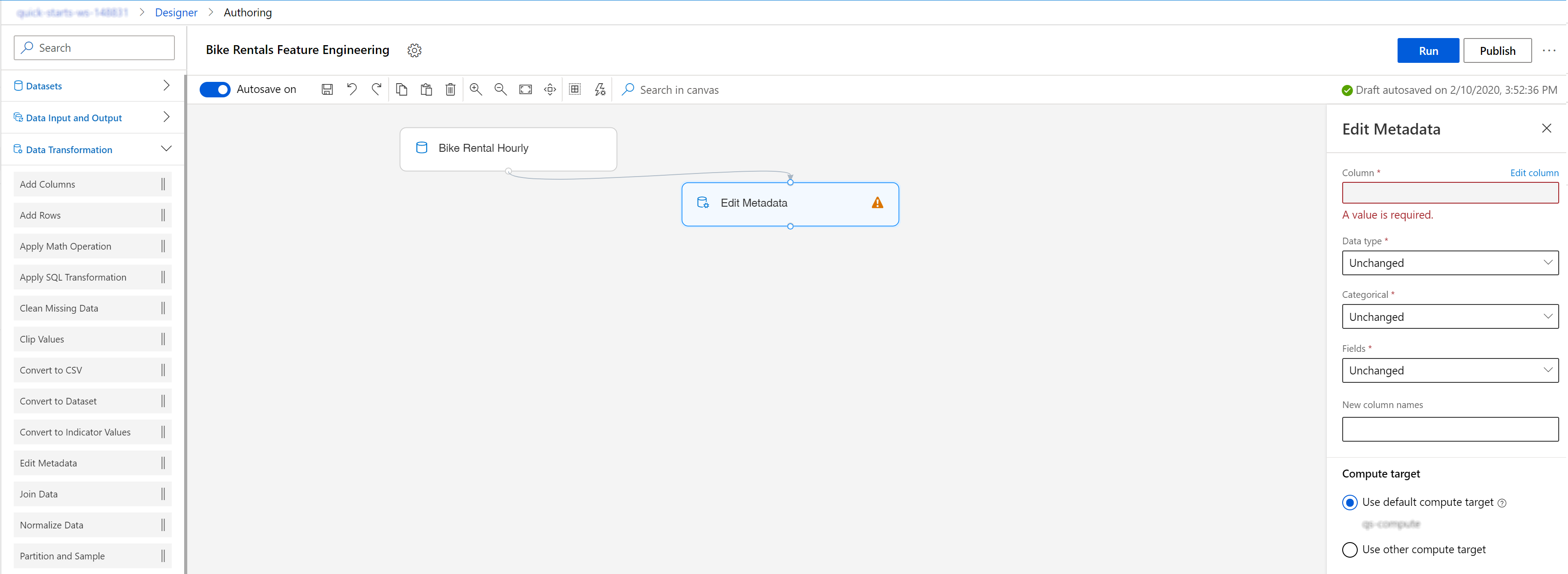
Note: If you are facing difficulties in accessing pop-up windows or buttons in the user interface, please refer to the Help section in the lab environment.

Task 4: Select columns in the dataset

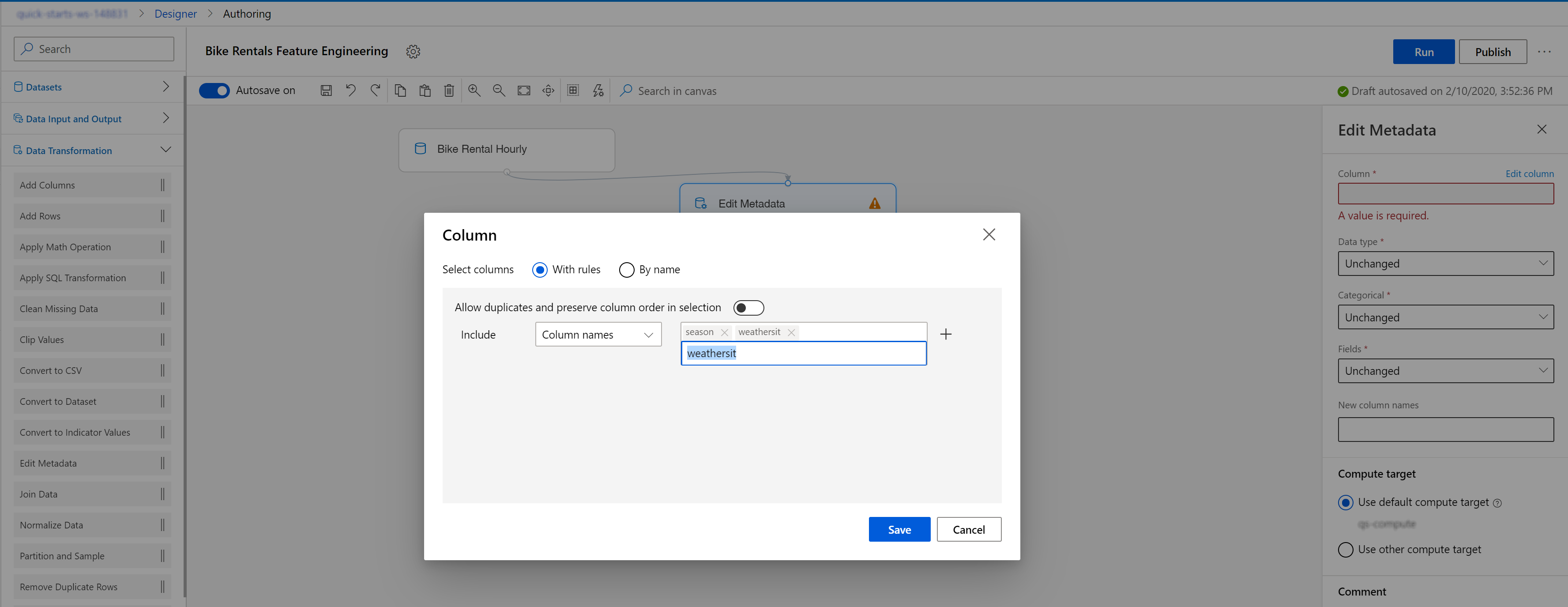
1. Drag and drop on the canvas, the available Bike Rental Hourly dataset under the **Datasets** category on the left navigation.



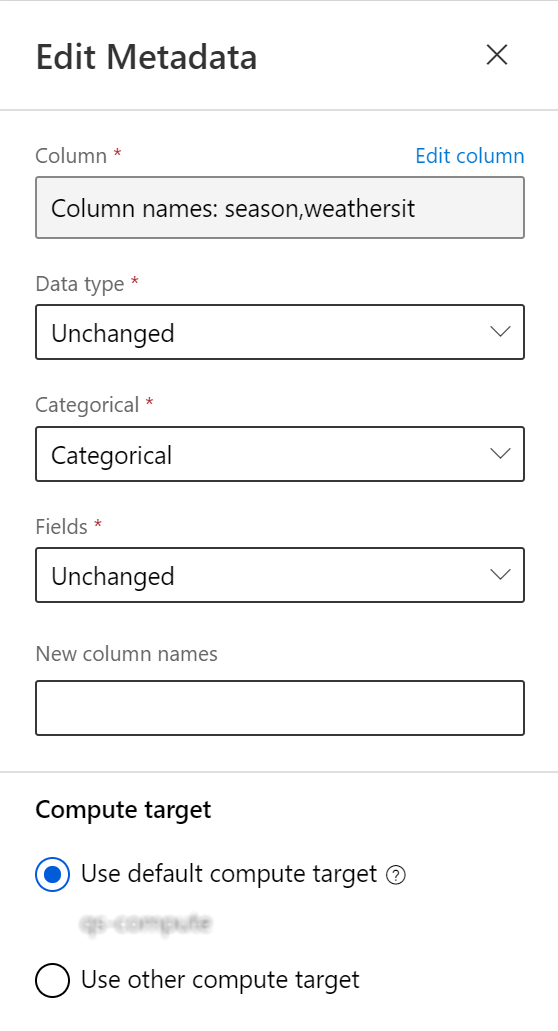
1. Under the **Data transformation** category drag and drop the Edit metadata module.



1. Edit the column list by selecting **Edit columns** on the right pane. Add the season and weathersit column and select **Save**.

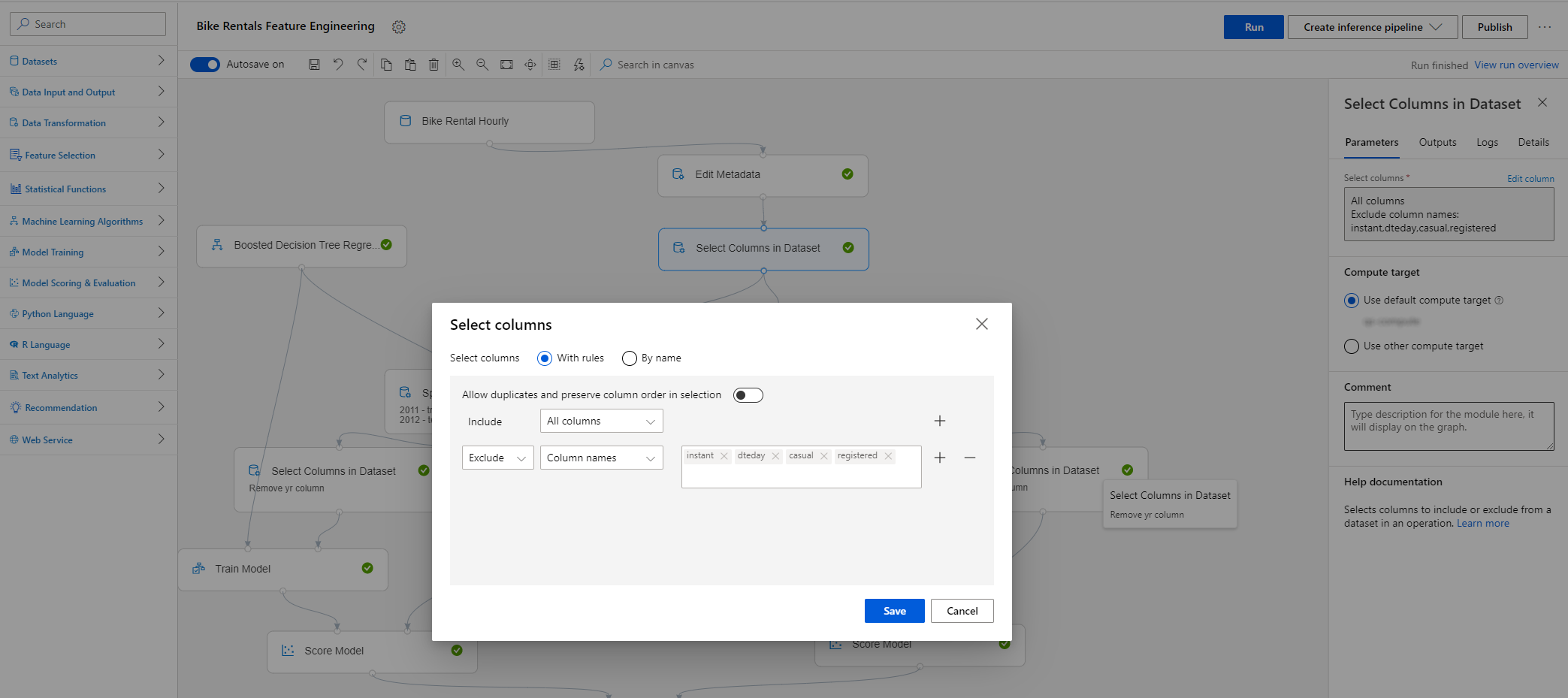


1. Configure the Edit metadata module by selecting the Categorical attribute for the two columns.

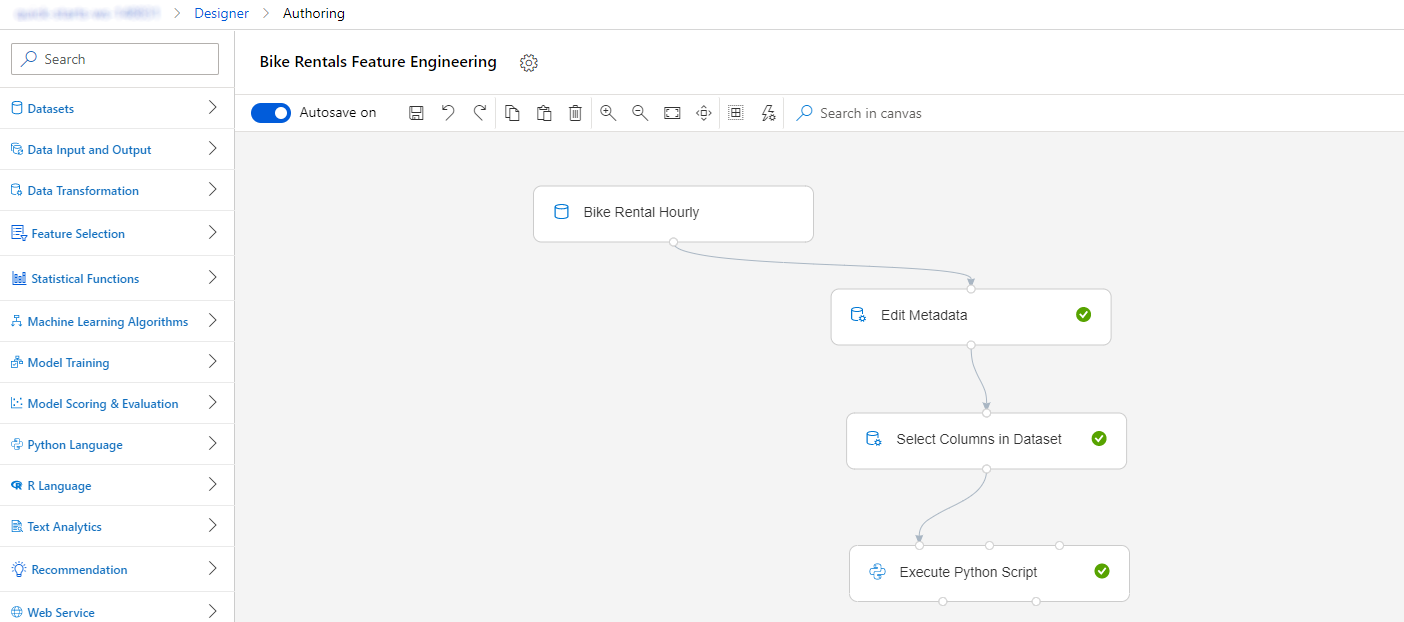


1. Next, use the **Select columns in Dataset** module under the **Data transformation** category and configure it as follows:

* Include all columns
* Exclude column names: instant, dteday, casual ,registered
* Use default compute target: qs-compute



1. Connect the output from the **Edit columns** module to the input of the **Select columns in Dataset** module.
2. Under the **Python Language** category on the left, select the **Execute Python Script** module and connect it with the **Select Columns in Dataset** module. Make sure the connector is connected to the very first input of the **Execute Python Script** module.



1. We are using the Python script to append a new set of features to the dataset: number of bikes that were rented in each of the previous 12 hours. Feature set B captures very recent demand for the bikes. This will be the B set in the described feature engineering approach.

Select **Edit code** and use the following lines of code:

# The script MUST contain a function named azureml\_main

# which is the entry point for this module.

# imports up here can be used to

import pandas as pd

import numpy as np

# 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(f'Input pandas.DataFrame #1: {dataframe1}')

# If a zip file is connected to the third input port,

# 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

for i in np.arange(1, 13):

prev\_col\_name = 'cnt' if i == 1 else 'Rentals in hour -{}'.format(i-1)

new\_col\_name = 'Rentals in hour -{}'.format(i)

dataframe1[new\_col\_name] = dataframe1[prev\_col\_name].shift(1).fillna(0)

# Return value must be of a sequence of pandas.DataFrame

# E.g.

# - Single return value: return dataframe1,

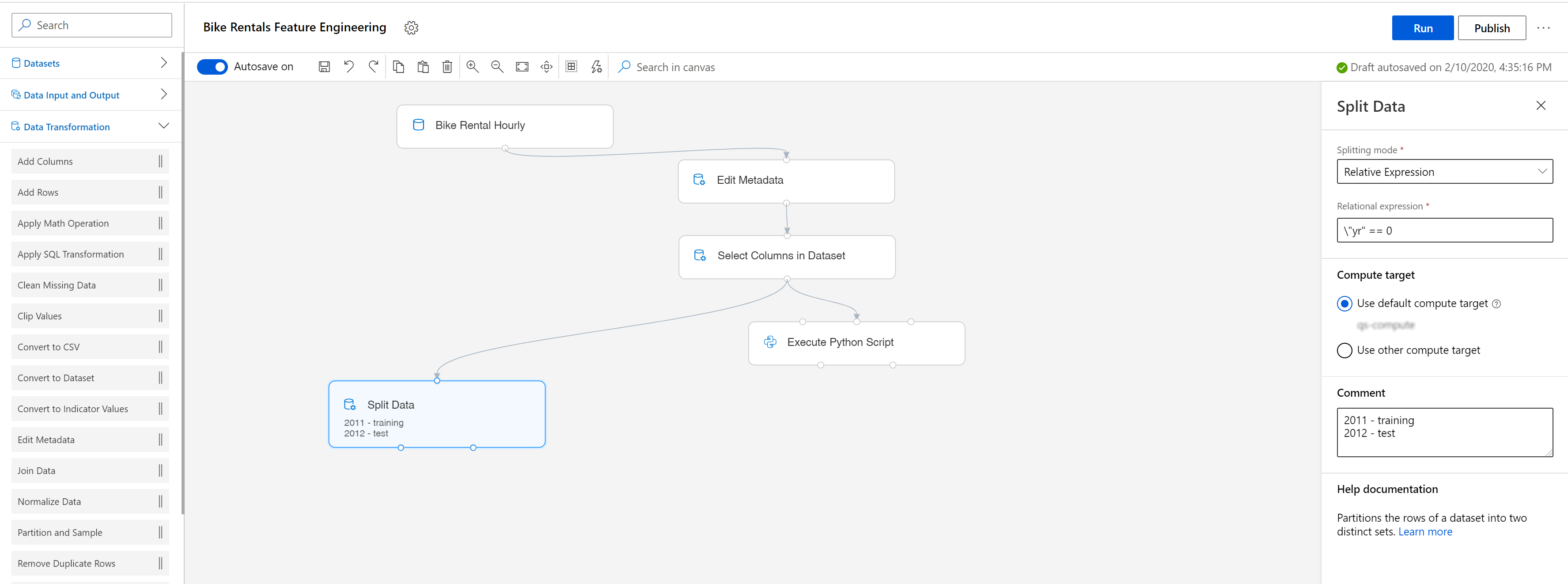
# - Two return values: return dataframe1, dataframe2

return dataframe1,

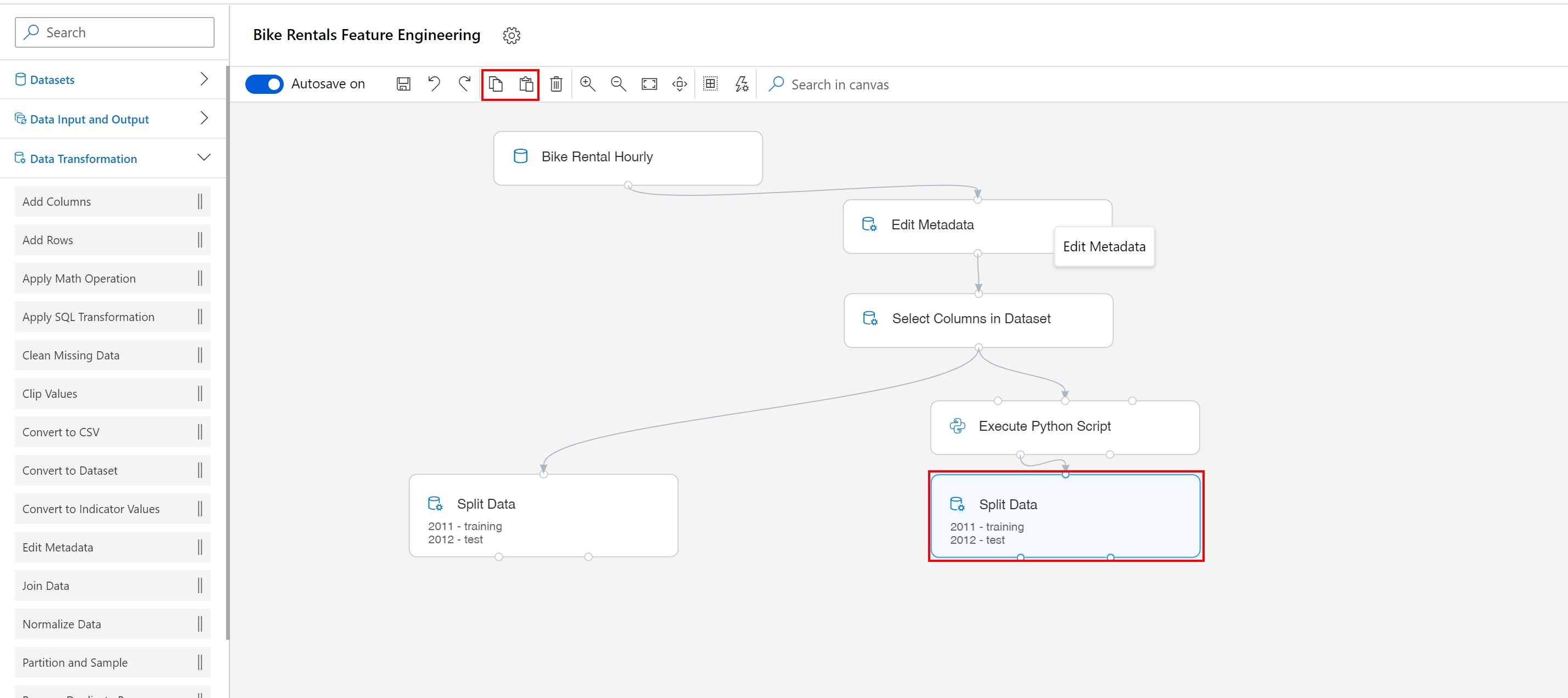
Don’t worry if you do not fully understand the details of the Python code above. For now, it’s enough to keep in mind that is adds 12 new columns to your dataset containing the number of bikes that were rented in each of the previous 12 hours.

Task 5: Split data into train and test datasets

1. Use the **Split Data** module under the **Data Transformation** module and connect its input with output from the **Select Columns in Dataset** module. Use the following configuration:
   * Splitting mode: Relative Expression
   * Relational expression: \"yr" == 0

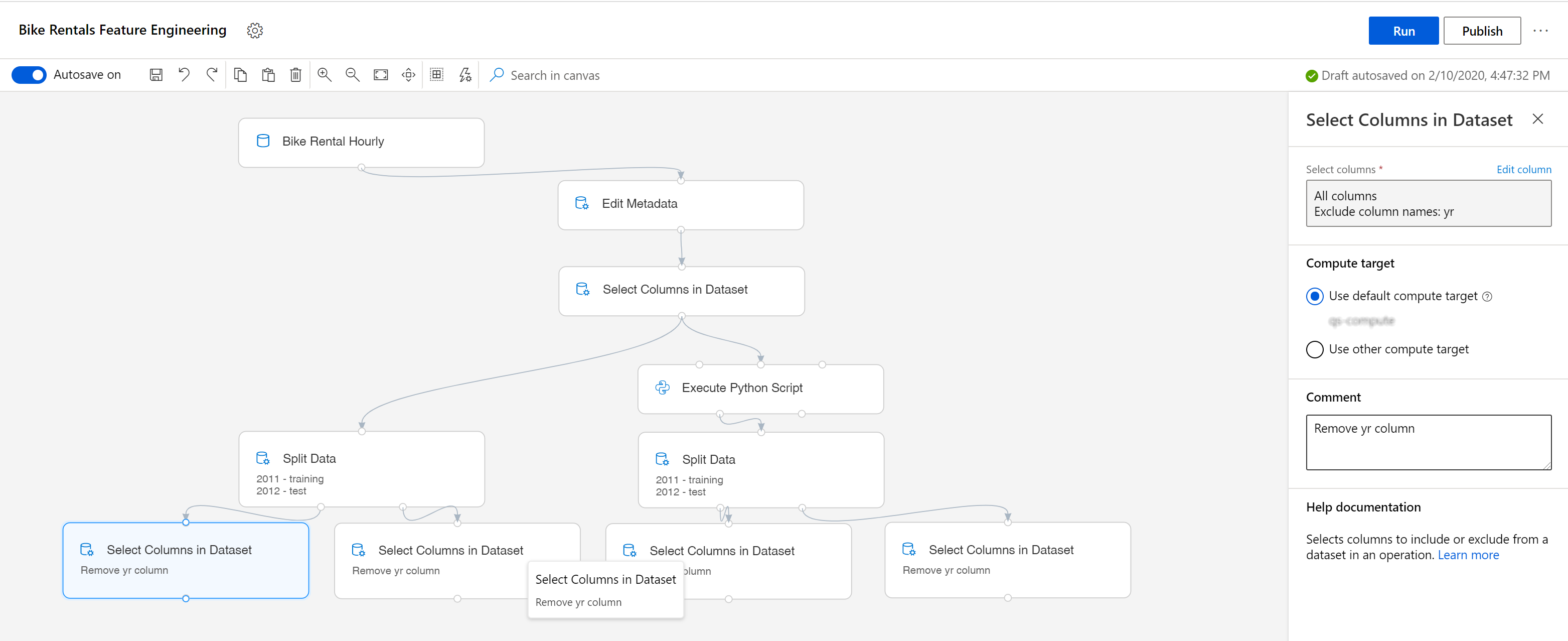


1. Select the **Split Data** module block and use the menu buttons to Copy and Paste it on the canvas. Connect the second one to the output of the Python Script execution step, which is the featured B set.

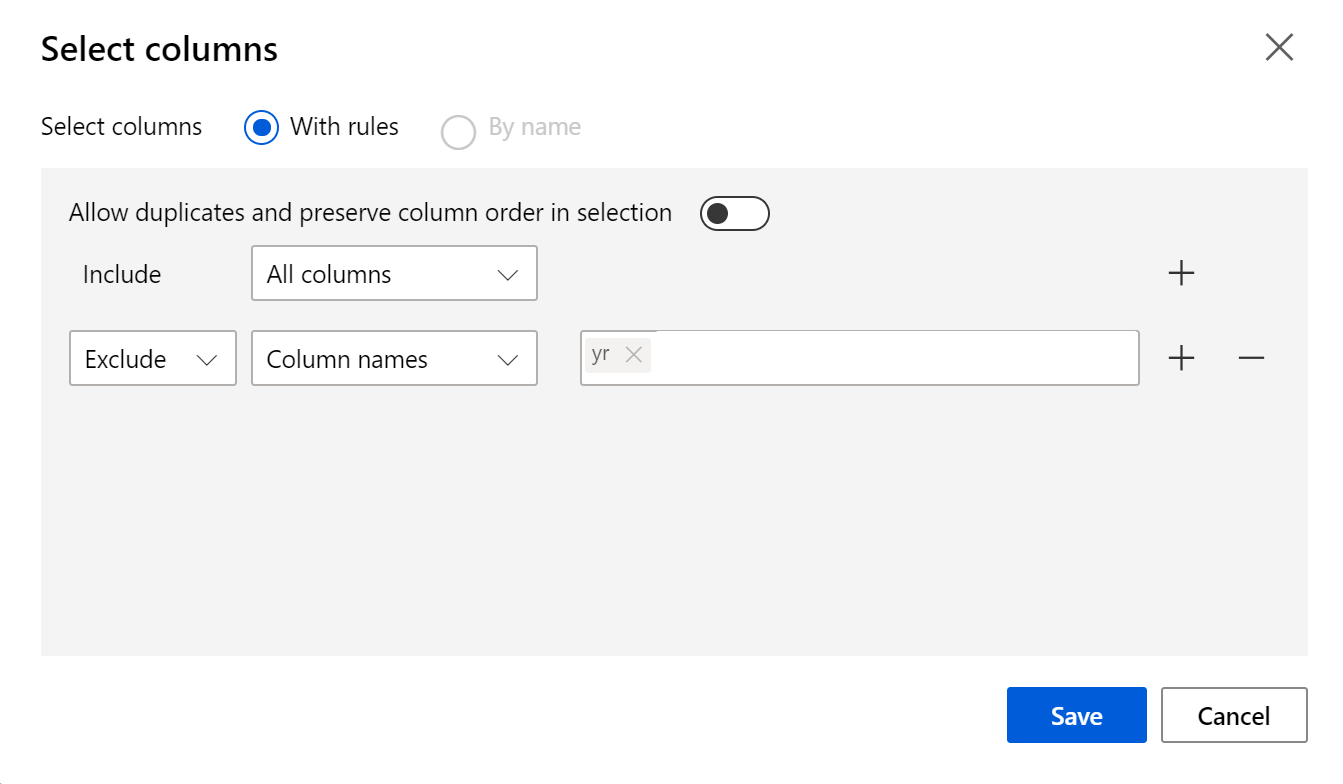


Task 6: Select columns from the test and training resulted sets

1. Next, using the **Select columns** module under the **Data transformation** category, create four identical modules to exclude the yr column from all the outputs: test and training sets in both branches: A and A+B.

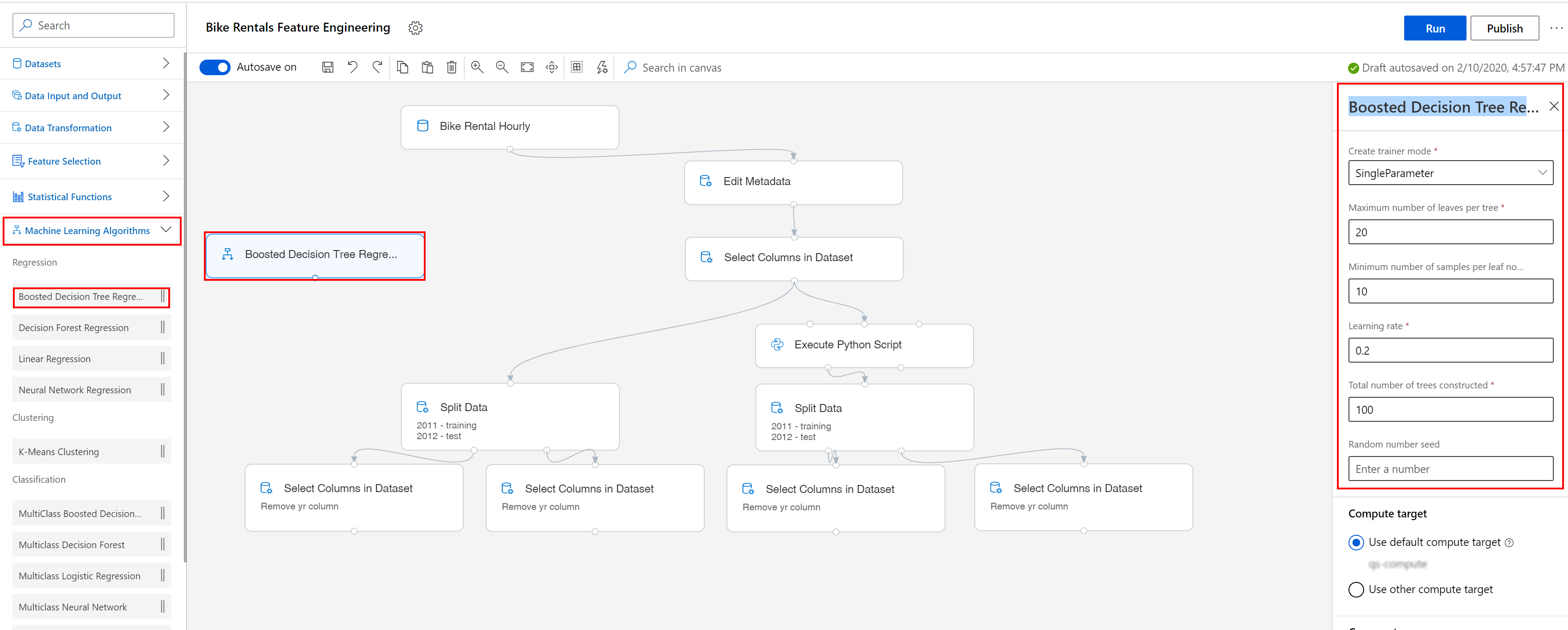


1. Use the following structure for the columns field in each module:

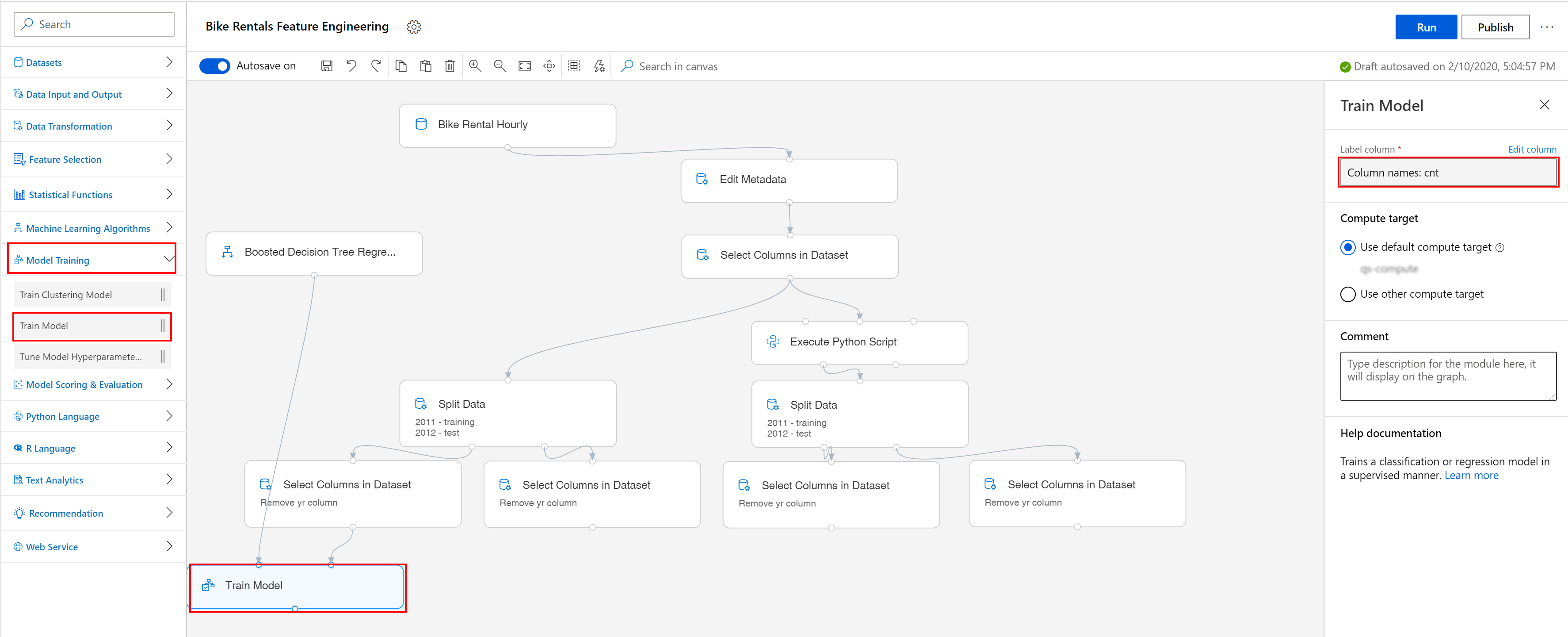


Task 7: Create the regression model

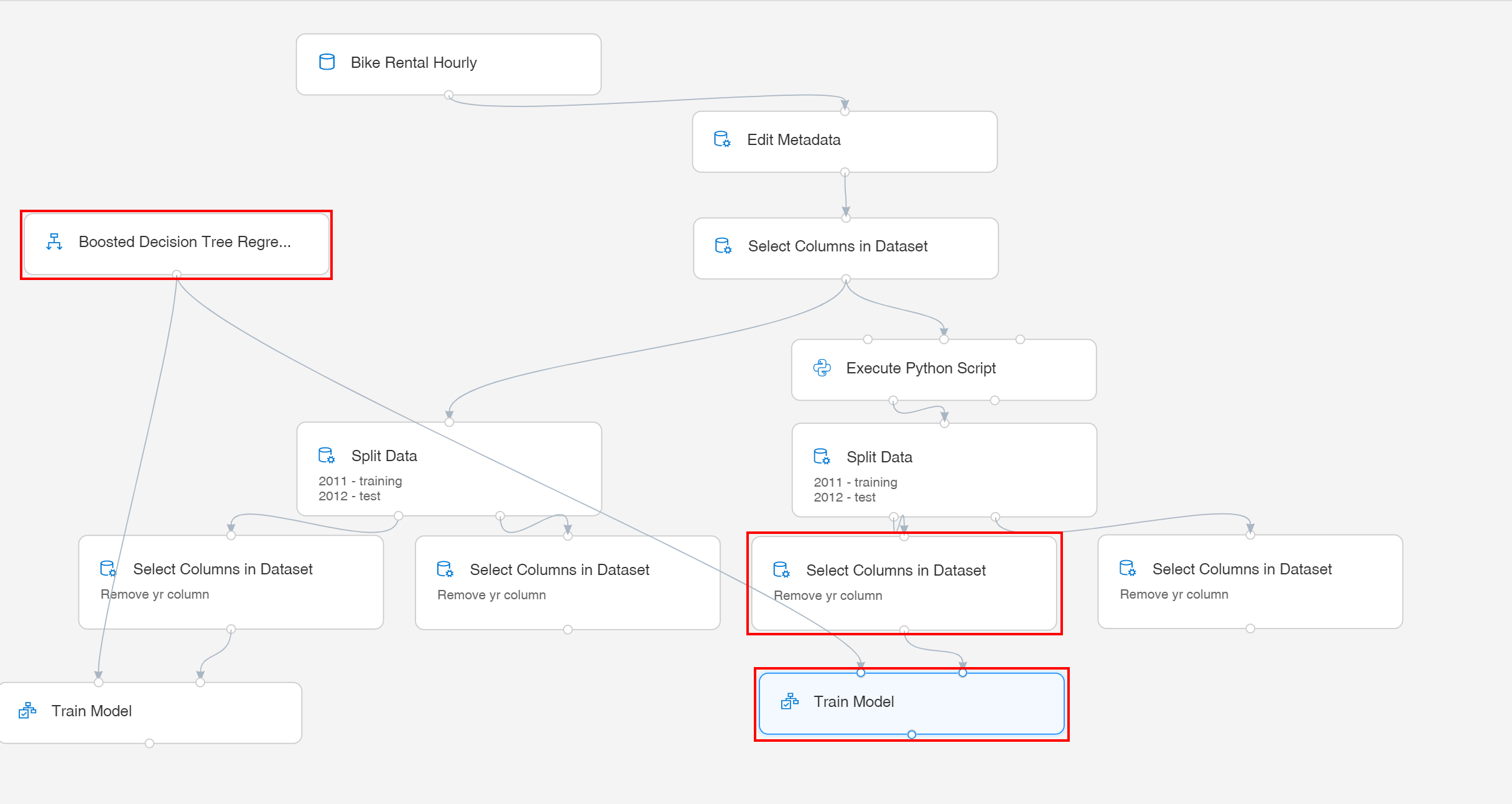
1. Under the **Machine Learning Algorithms, Regression** category, select the **Boosted Decision Tree Regression** module. Drag and drop it on the canvas and use the default settings provided.



1. Next, use the **Train model** module under the **Model training** category and enter the cnt column in the **Label column** field.
2. Link the **Boosted Decision Tree Regression** module as the first input and the training dataset as the second input like in the image below.

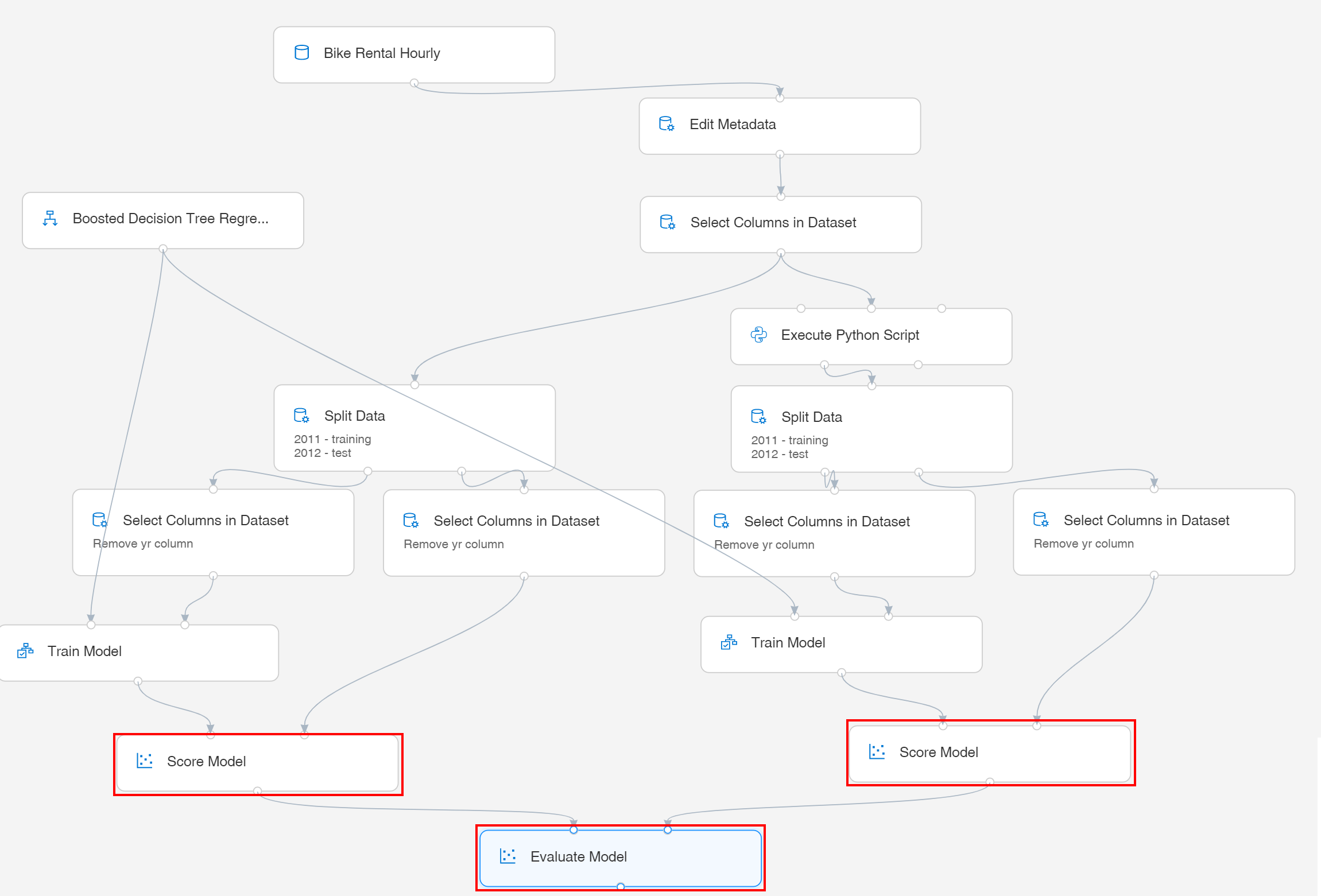


1. Use the exact same configuration on the right branch that uses the output from the Python Script.

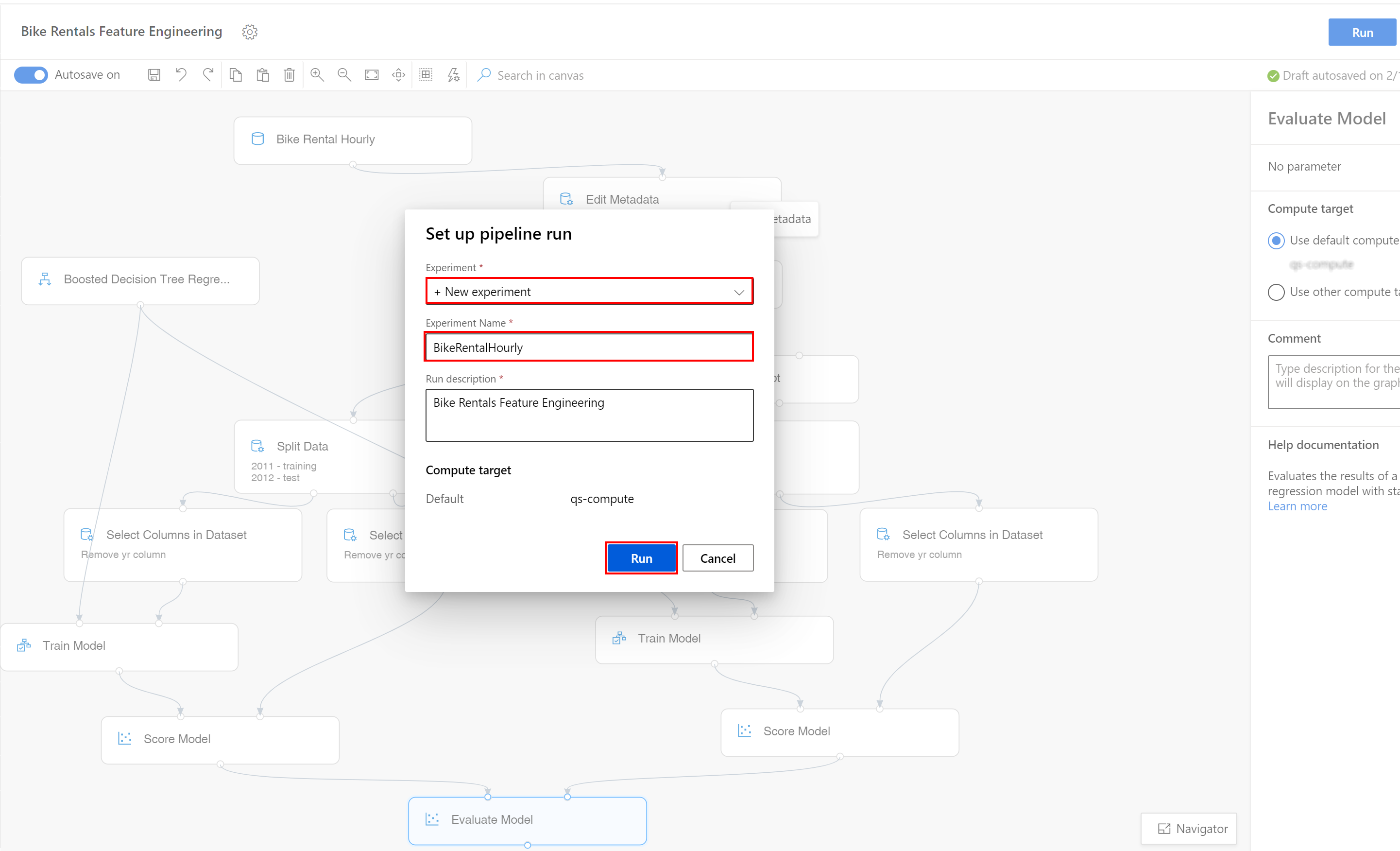


Task 8: Evaluate and score models

1. Use two **Score Model** modules (under the **Model Scoring and Evaluation** category) and link on the input the two trained models and the test datasets.
2. Drag the **Evaluate Model** module which stands in the same category, **Model Scoring and Evaluation** and link it to the two **Score Model** modules.

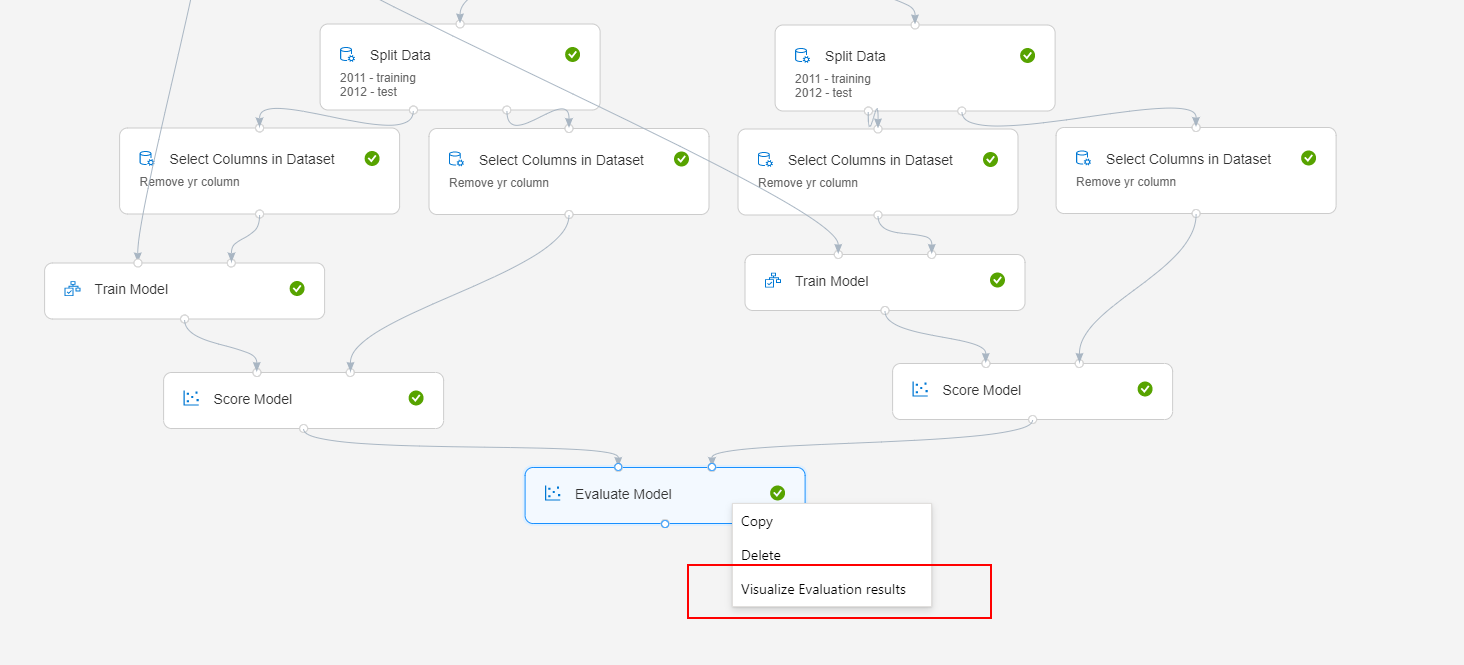


1. Select **Submit** to open the Setup pipeline run editor. In the Setup pipeline run editor, select **Experiment, Create new** and provide New experiment name: **BikeRentalHourly**.

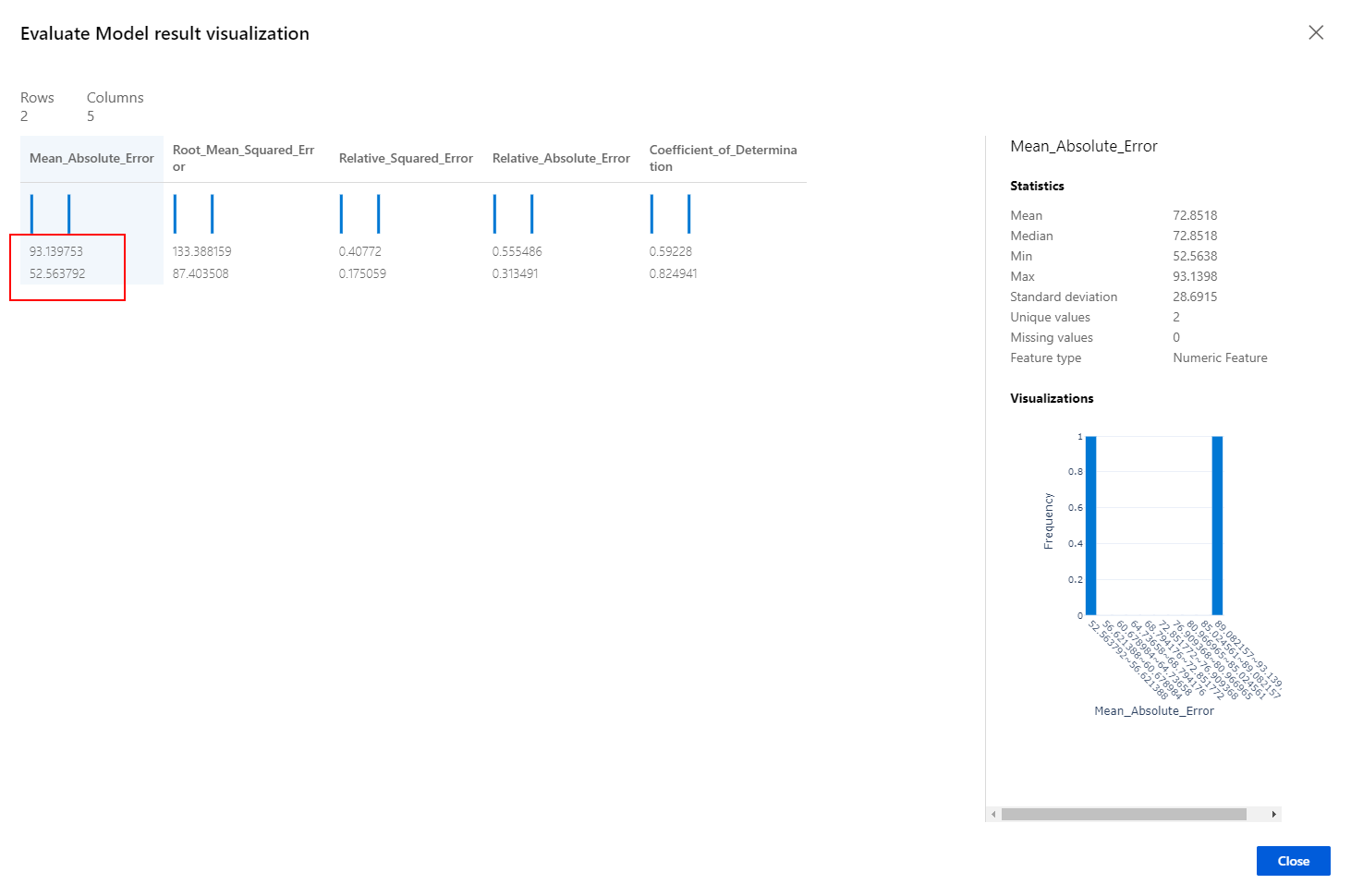


Please note that the button name in the UI is changed from **Run** to **Submit**.

1. Wait for pipeline run to complete. It will take around **10 minutes** to complete the run.
2. Once the pipeline execution completes, right click on the **Evaluate Model** module and select **Visualize Evaluation results**.



1. The **Evaluate Model result visualization** popup shows the results of the evaluation.



Notice the values for the \*\*Mean\_Absolute\_Error\*\* metric. The first value (the bigger one) corresponds to the model trained on feature set A. The second value (the smaller one) corresponds to the model trained on feature sets A + B.

It is remarkable how, using simple feature engineering to derive new features from the existing data set, a new context was created that allowed the model to better understand the dynamics of the data and hence, produce a better prediction.

**Next Steps**

Congratulations! You have trained and compared performance of two models using the same algorithm, but with two different datasets. You can continue to experiment in the environment but are free to close the lab environment tab and return to the Udacity portal to continue with the lesson.