Azure Red shirt dev tour

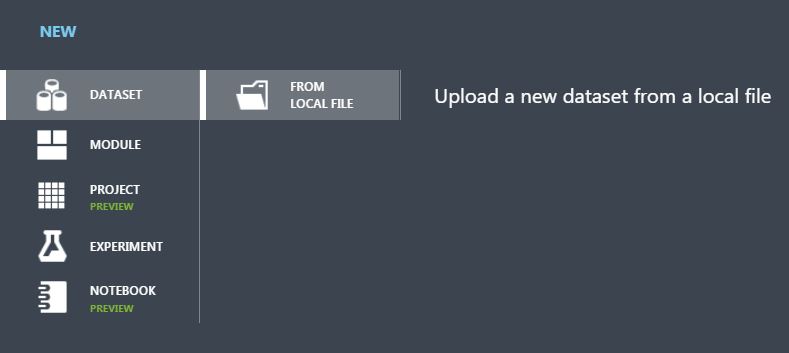
Hands On Lab Cognitive Services & AI

# IOT

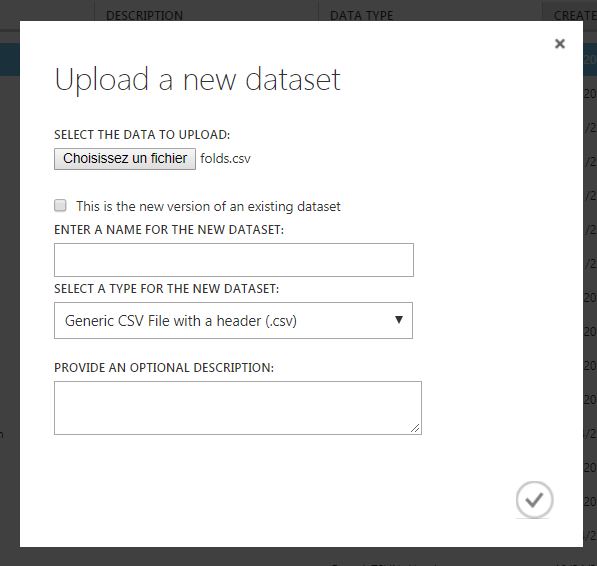
The dataset contains 9568 data points collected from a Combined Cycle Power Plant over 6 years (2006-2011), when the power plant was set to work with full load.

Features consist of hourly average ambient variables Temperature (T), Ambient Pressure (AP), Relative Humidity (RH), Exhaust Vacuum (V) and the net hourly electrical energy output (EP) of the plant.

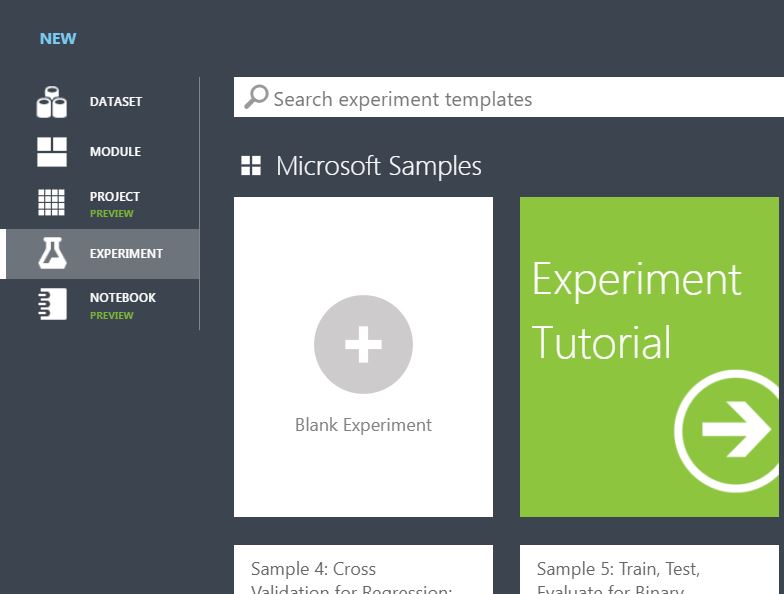
* Go to <https://studio.azureml.net/>
* Create a free account or sign in
* Click New, Dataset



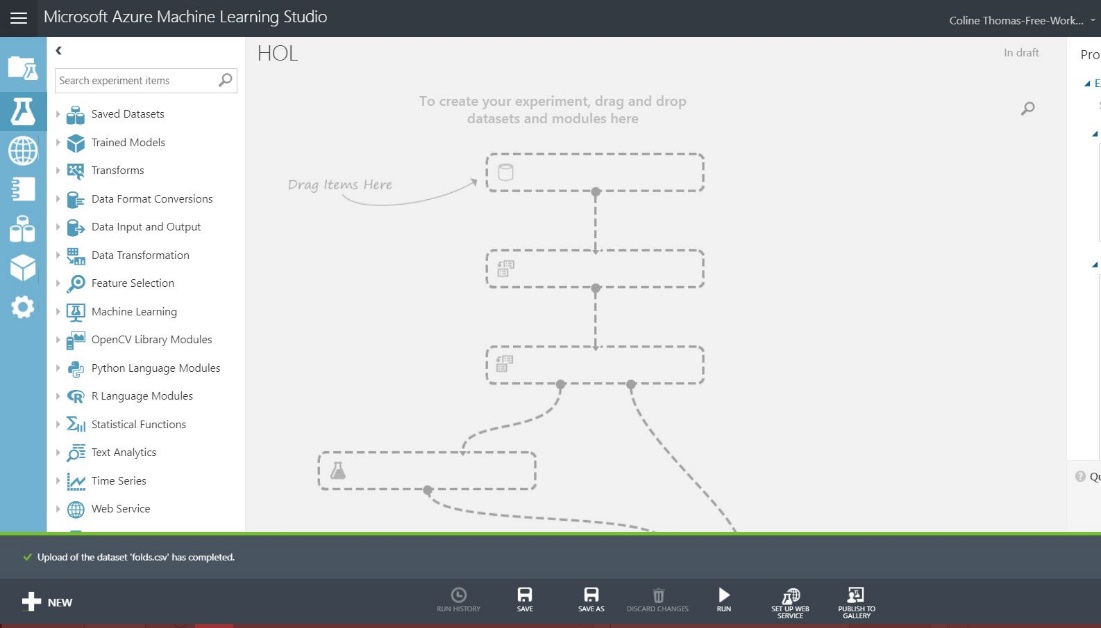
* Add the dataset “folds.csv”

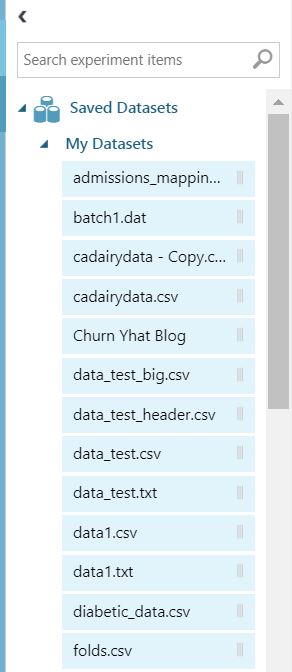


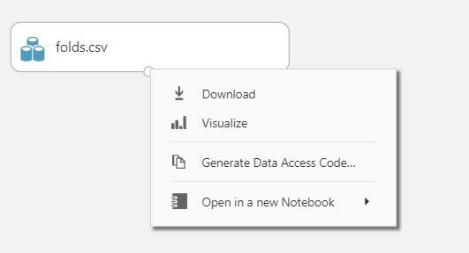
* Click New again
* Create a new Experiment



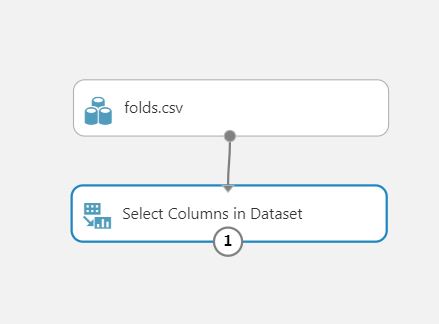
* Rename your new experiment



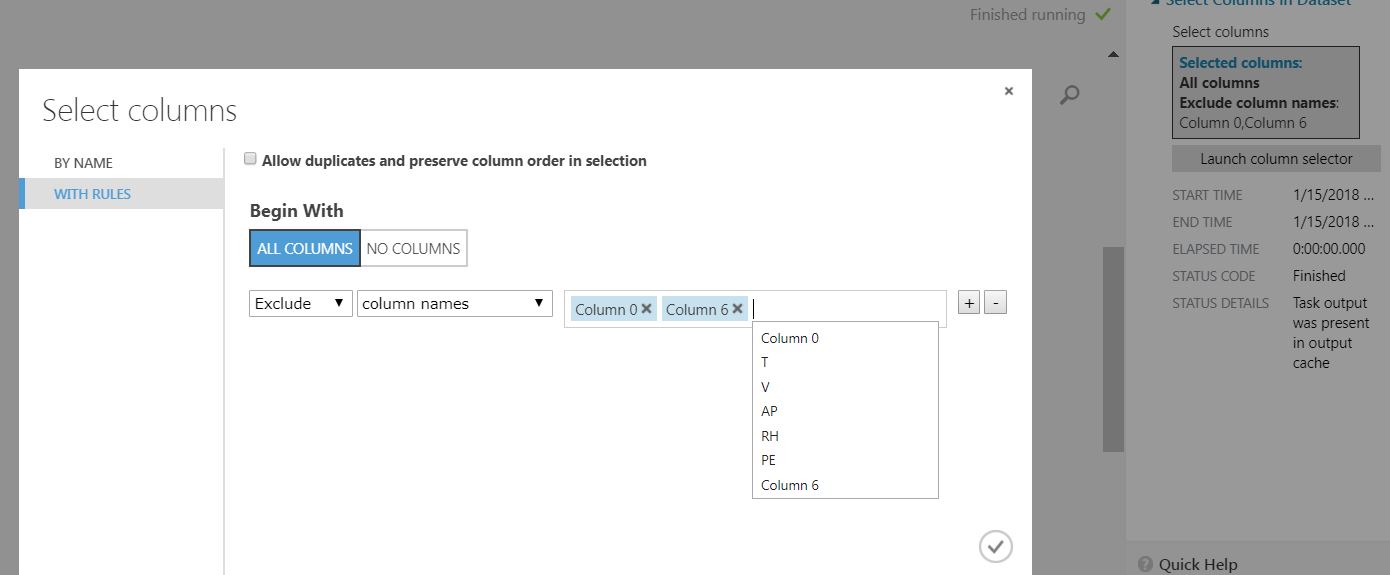
* Add to this experiment your newly added dataset
* Click on Visualize



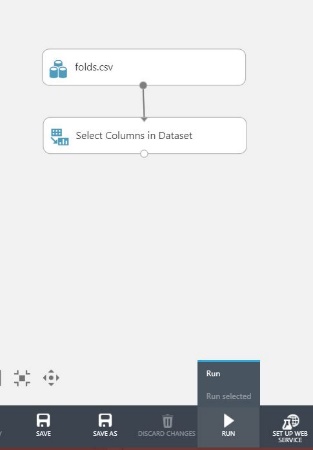
* We have 2 useless columns to remove
* Add a Select Column in Dataset



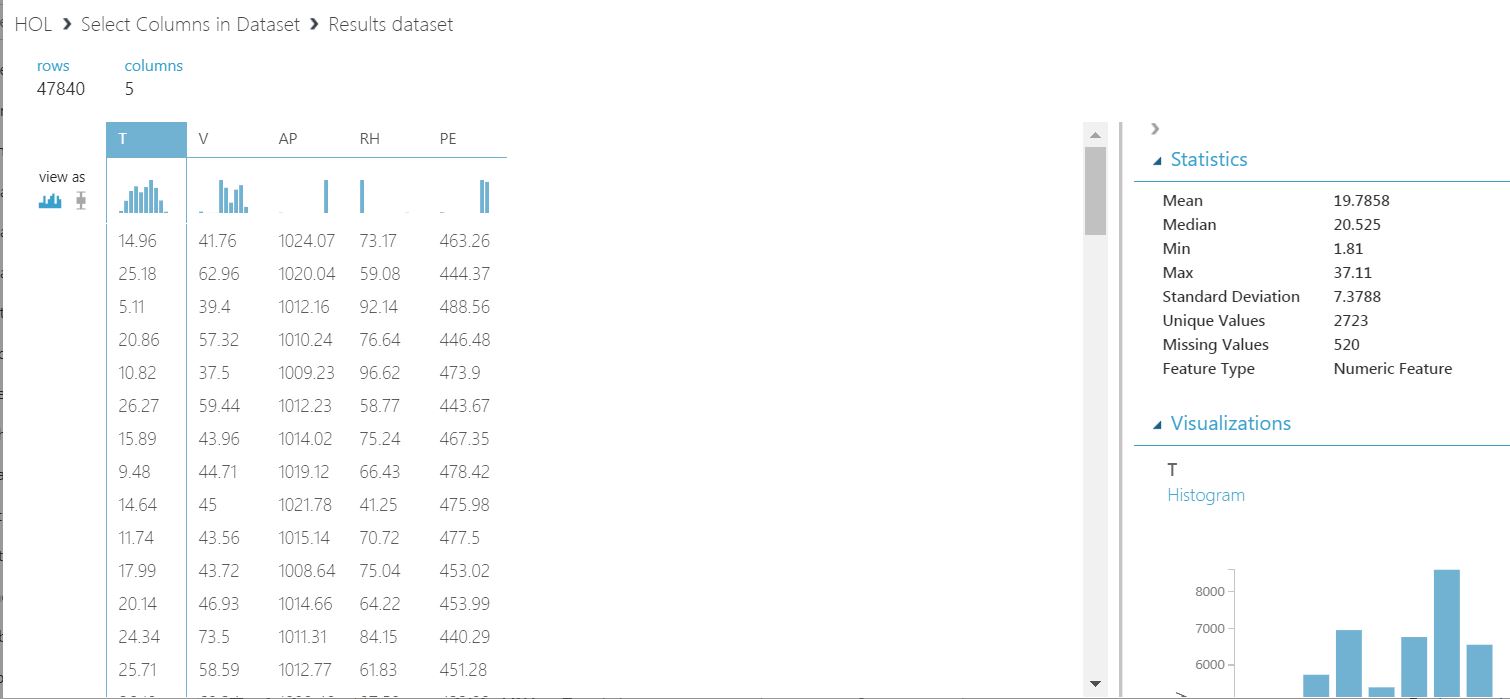
* Remove Column 0 and Column 6



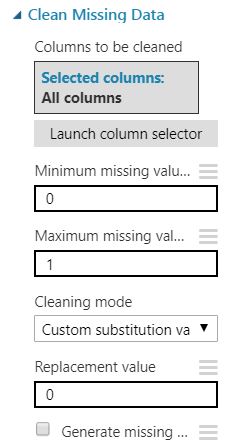
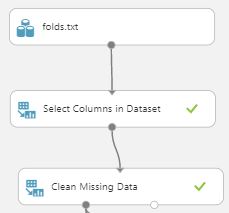
* Run



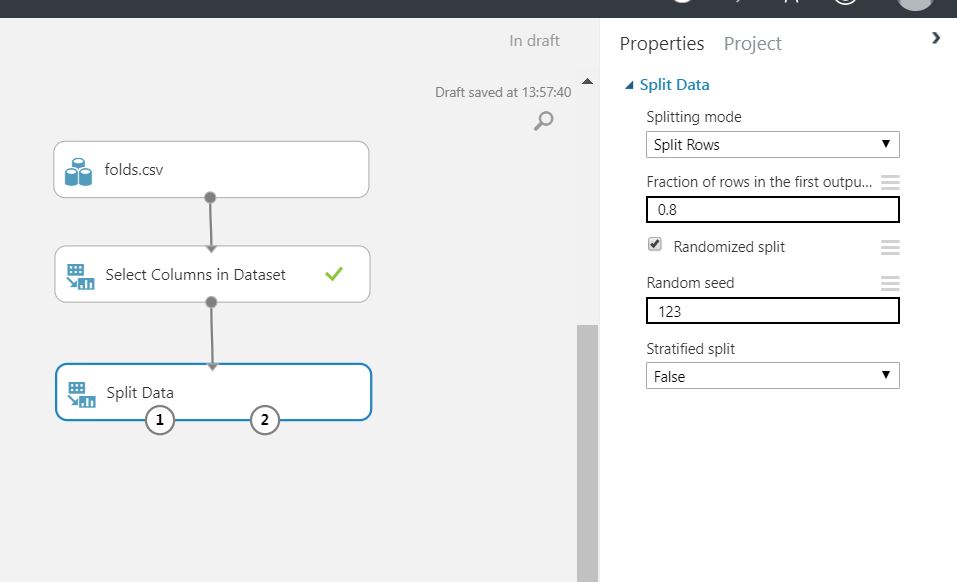
* Visualize. You have kind of a good dataset.

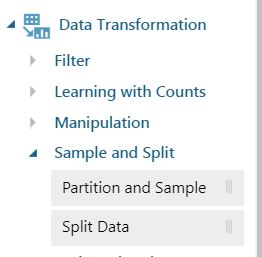


* We have some missing values. Our algorithm will ignore the entire line and predict nothing here…
* Add a Clean Missing Data with 0 for the replacement value

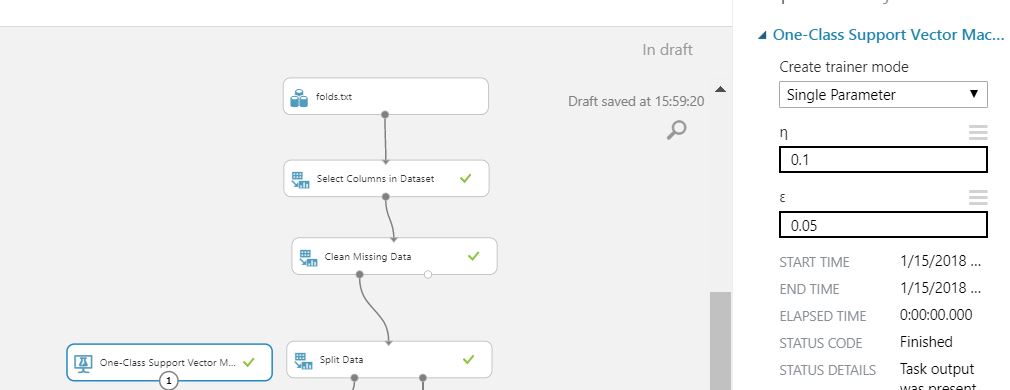


* To train then test our algorithm we need to split our data
* Add a Split Data

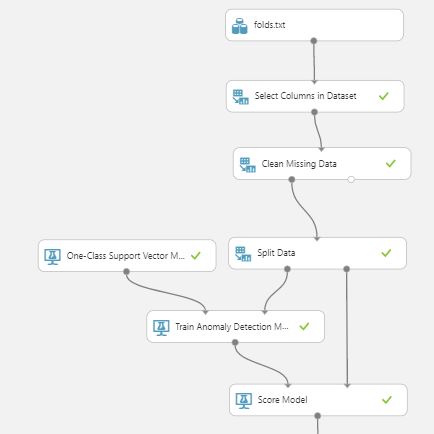




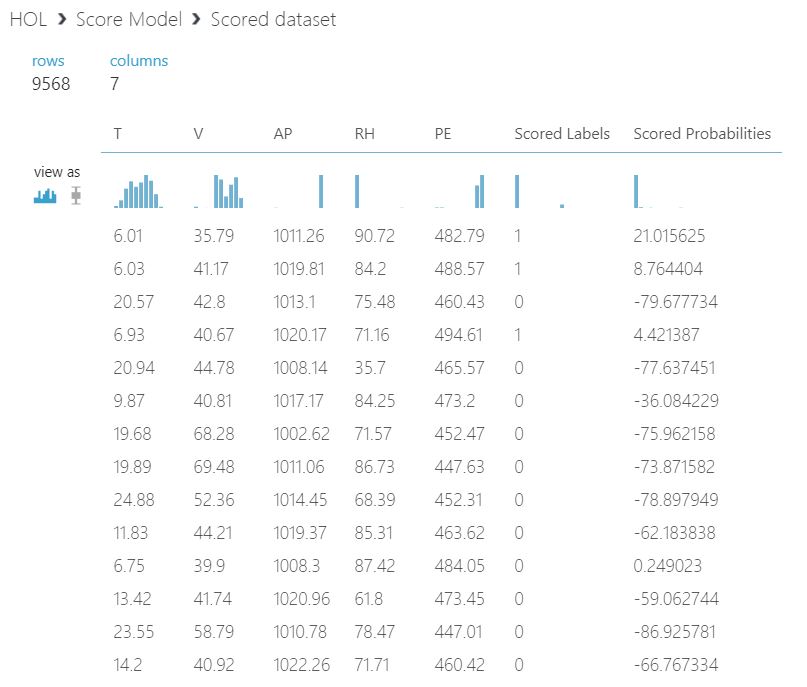
* We want to detect anomalies on future data using the dataset we have.
* We are going to use an algorithm called One-Class Support Vector Machine (<https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/one-class-support-vector-machine>) which enables unsupervised training on a dataset with few anomalies.
* Add it to the flow



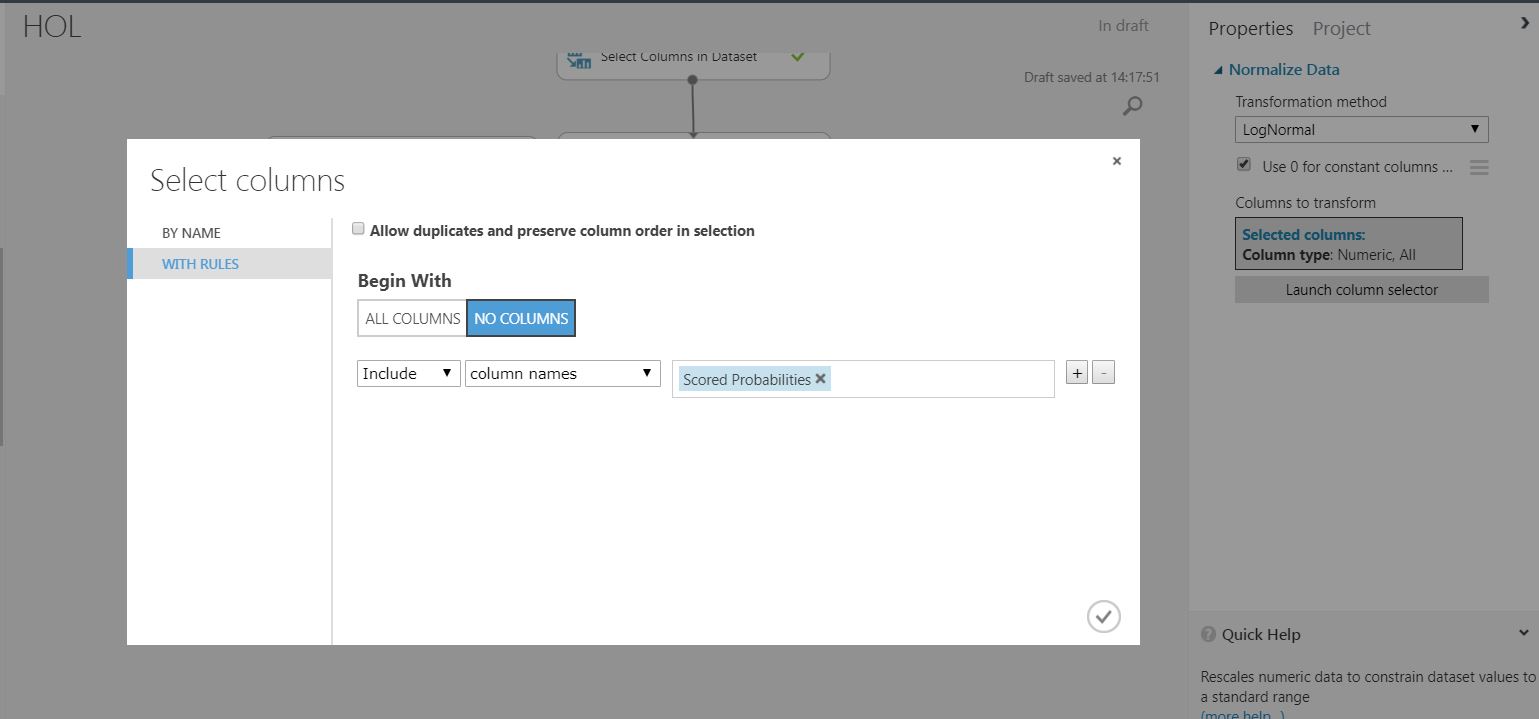
* Add a Train Anomaly Detection Model
* Add a Score Model



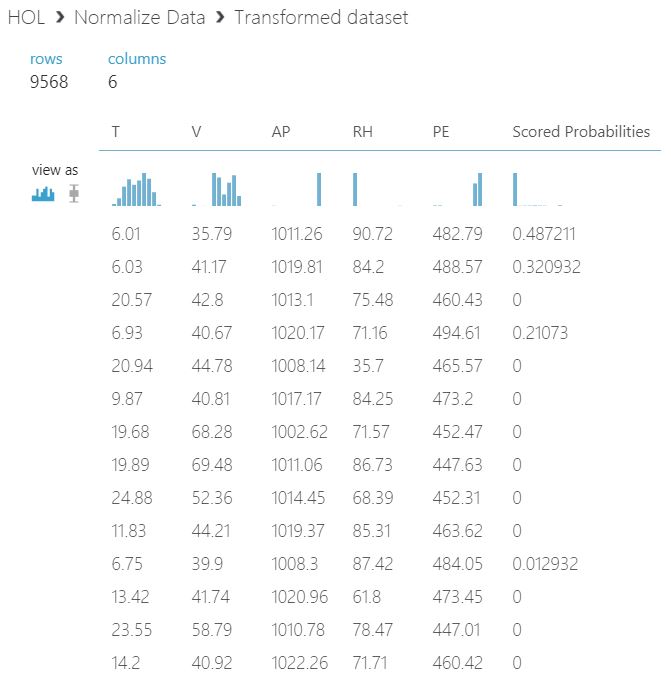
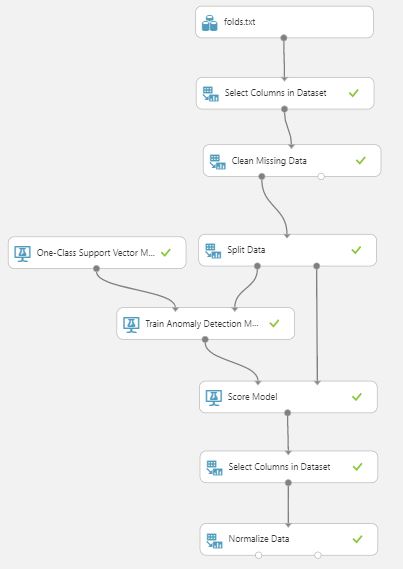
* Run it
* Check your output



* We have the Scored Labels Column which is useless
* Add a Select Column in Dataset and remove the Scored Labels column
* We would like the column Scored Probabilities to be more “expressive”
* Add a Normalize Data with a LogNormal transformation method on the Scored Probabilities column



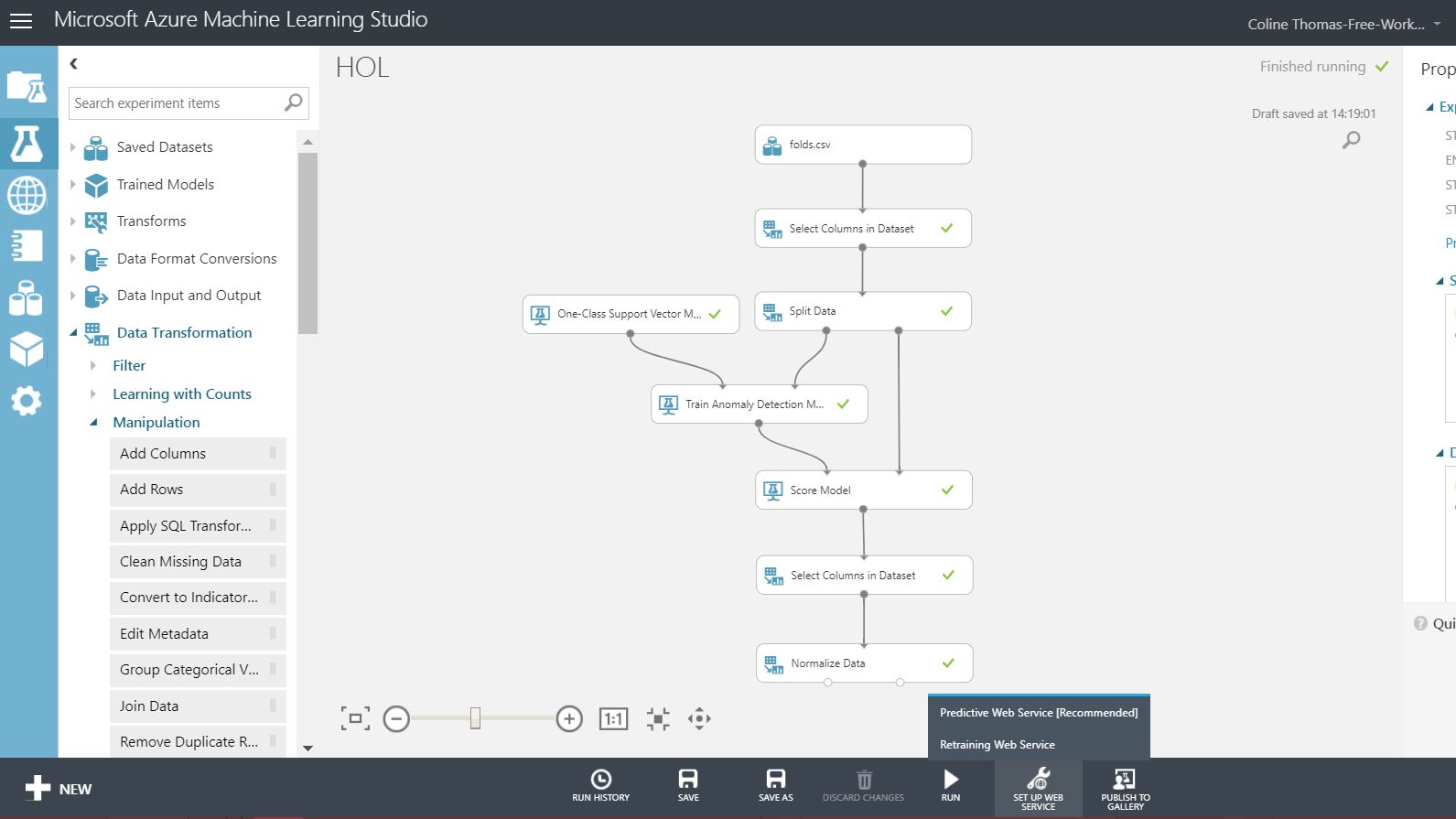
* Run it again and check your output



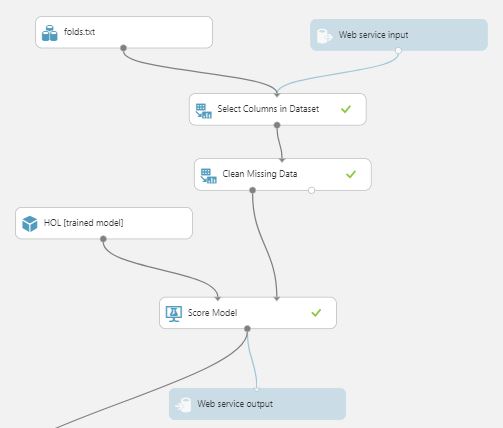
* We have a model which is able to detect anomaly risk in a dataset

We now need to use this model outside of Azure ML

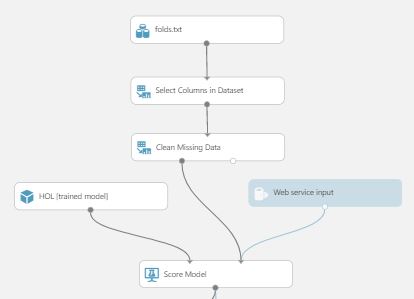
* Click on Set Up Web Service, Predictive Web Service



* Run the predictive experiment



* Check the input and the output. None of them is valid.
* Put the Web Service Input AFTER the transformation

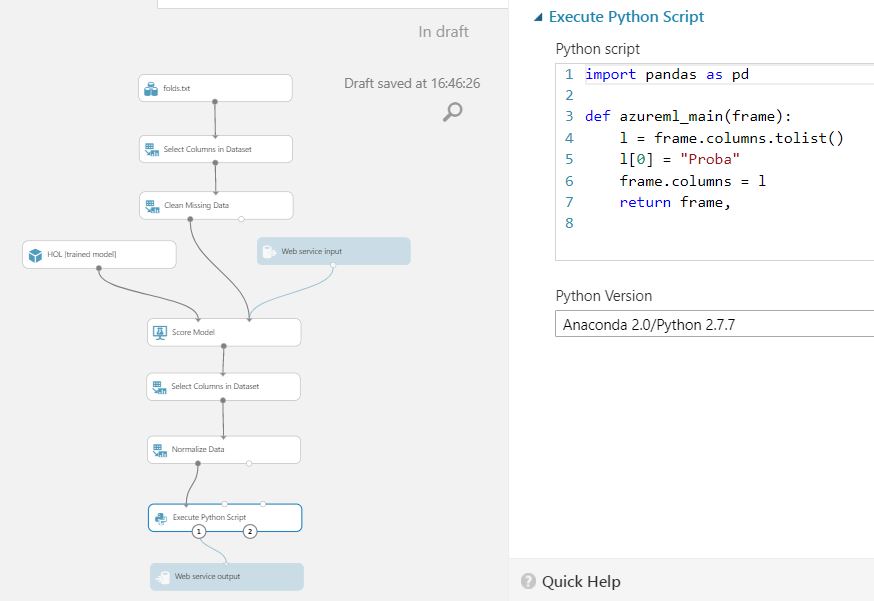


* Add a Select Column in Dataset and keep only the Scored Probabilities column at the end. This is the only thing we want.
* Add a Normalize Data just as before.
* We now should rename the output column to make it easier to use.
* Add an Execute Python Script and paste the following code

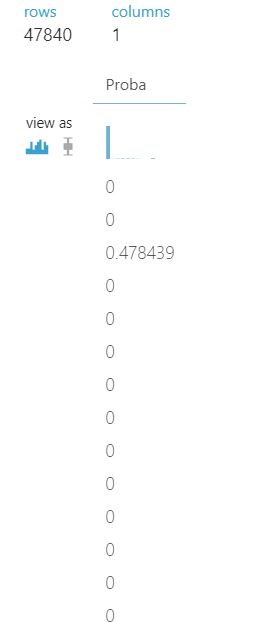
*import* pandas *as* pd  
  
*def* azureml\_main(*frame*):

l = *frame*.columns.tolist()  
  
 l[0] = "Proba"  
 frame.columns = l

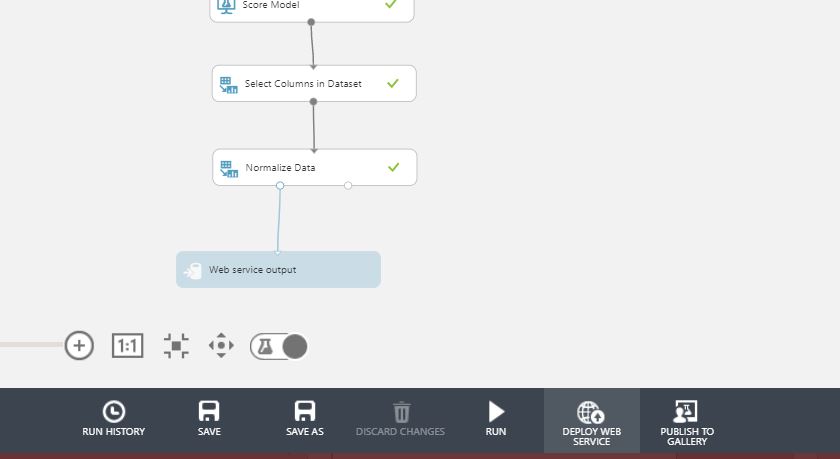
*return* frame,



* Run it. You should have a perfect output.

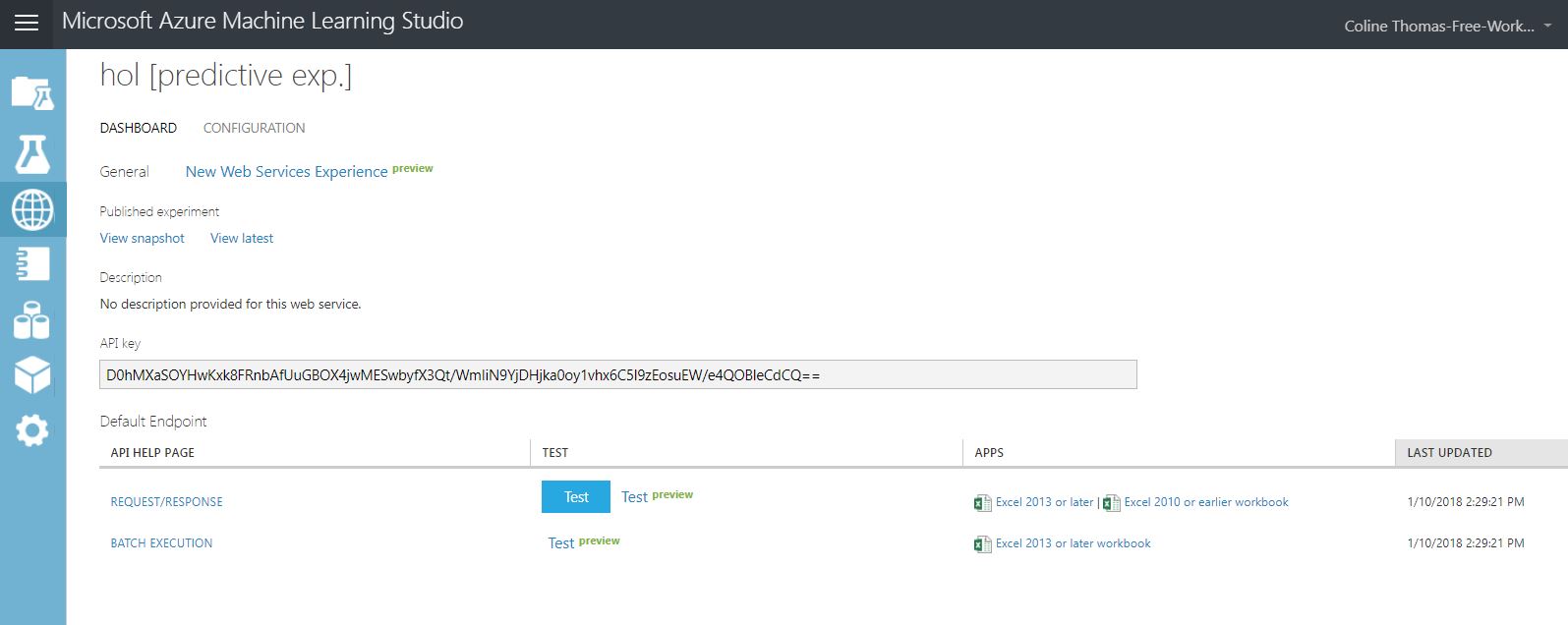


* Click on Deploy Web Service



You now have access to your model throw an API which takes 5 floats as input and output a 0-1 probability

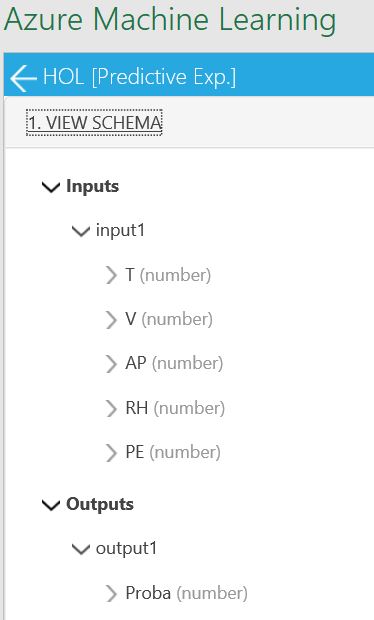
* Click on the link Excel 2013 or later



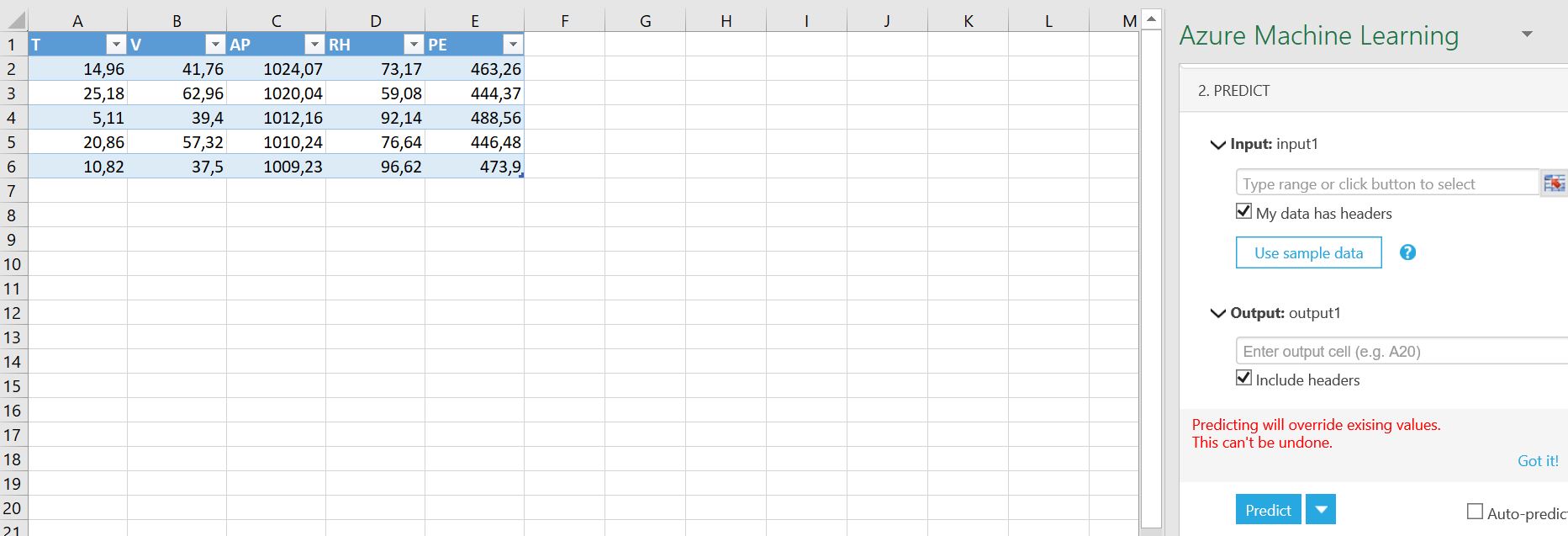
* Open the file
* Click on Activate Modifications



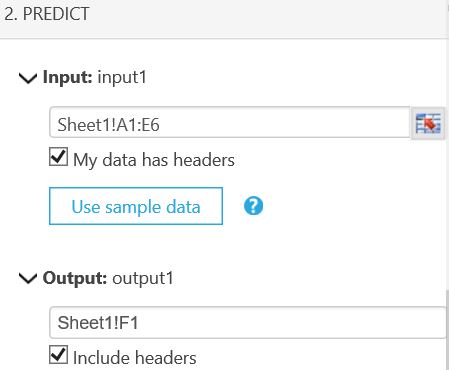
* You are now able to test your model throw Excel
* Click on HOL[predictive experiment]
* Check the Schema first



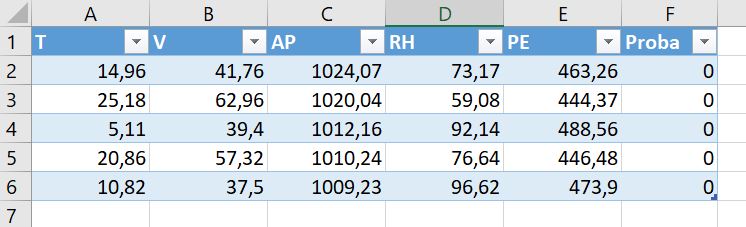
* Click on use sample data

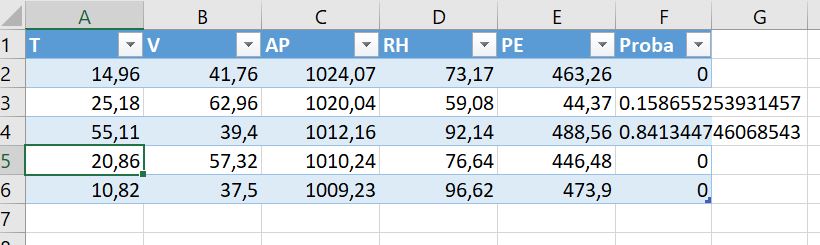


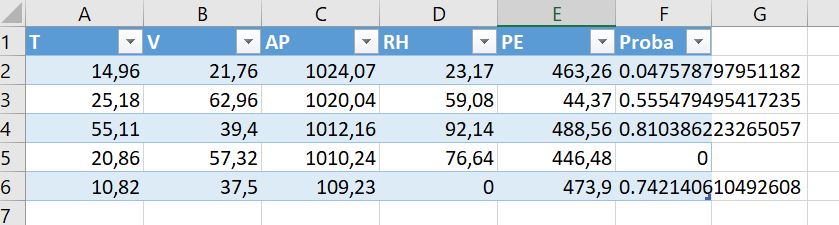
* Select the entire created table as input and F1 as output



* Click predict
* You can modify values and click predict again to test your model

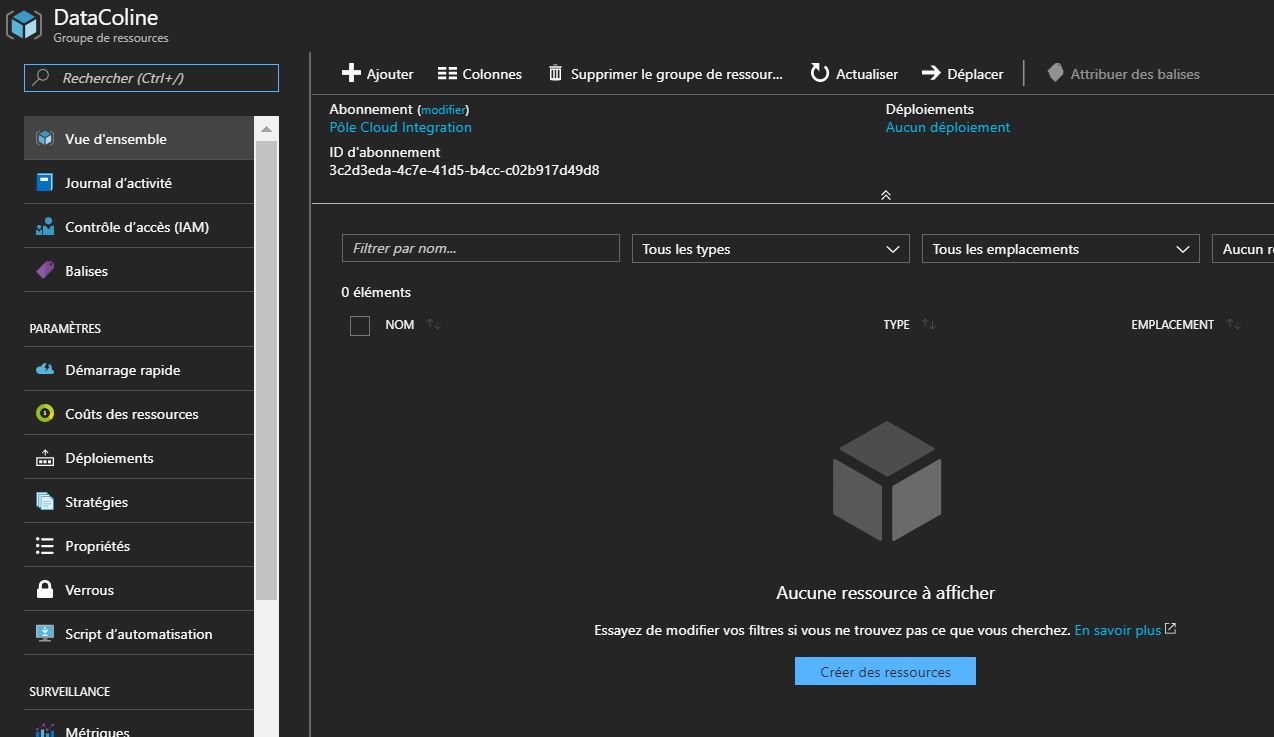




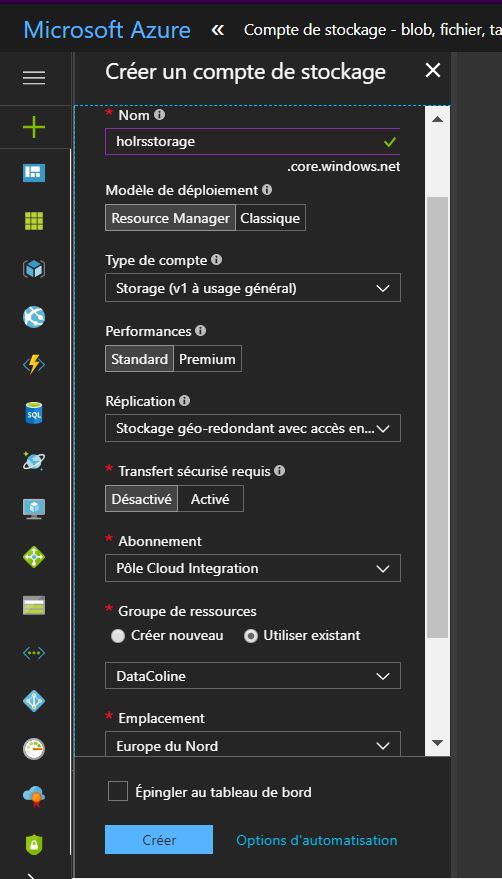
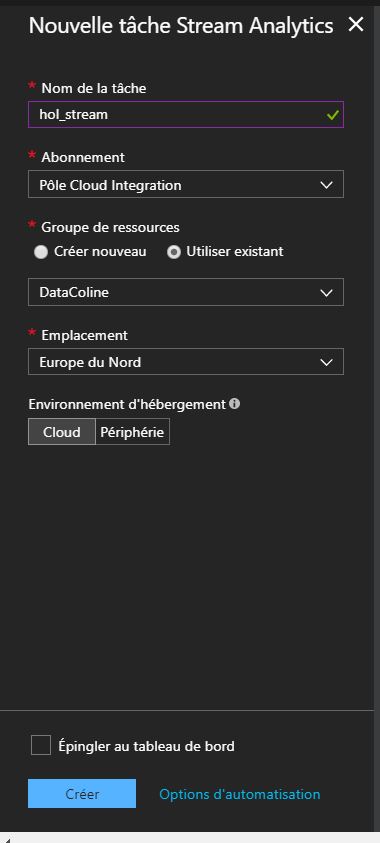


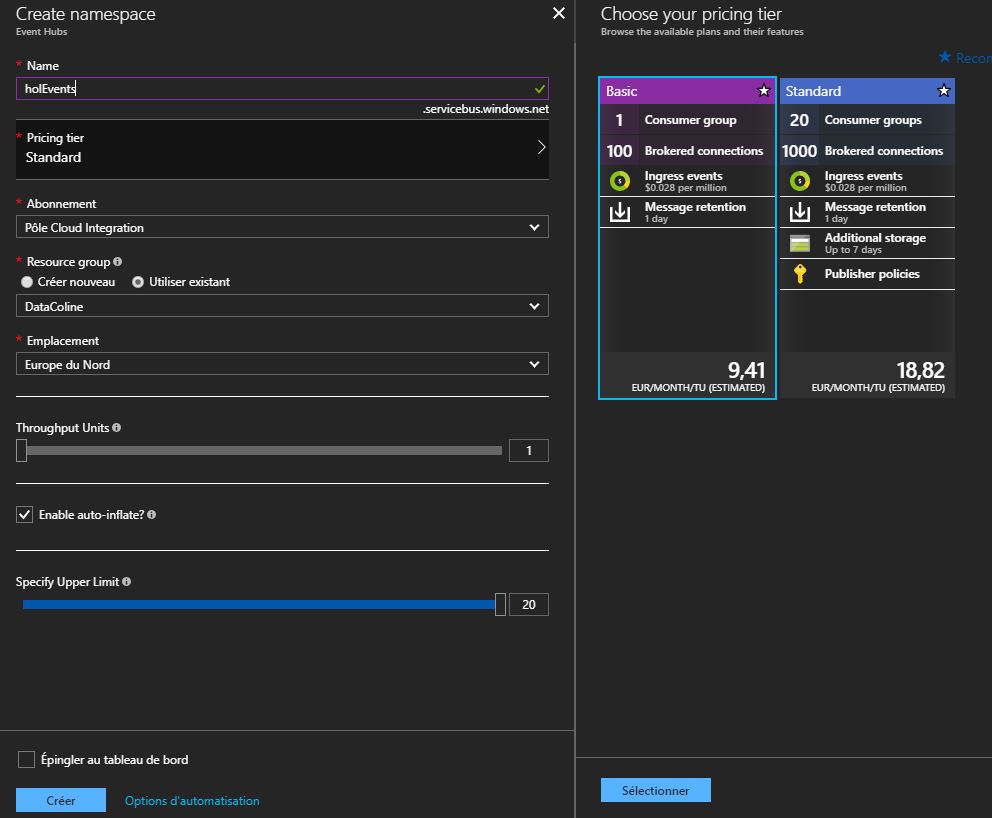
We want to use it on streaming data to determine in real time if there are anomalies in our plant.

* Go to <https://portal.azure.com> and sign in using your account or create a free one.
* Create a new Resource Group

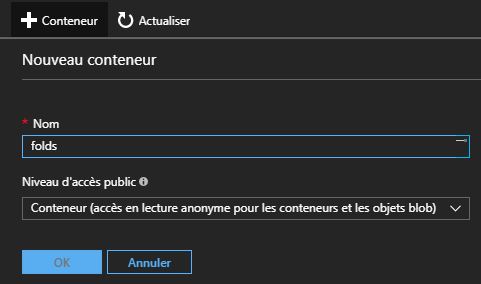


* Add a Storage Account, an Azure Stream Analytics and an Event Hubs

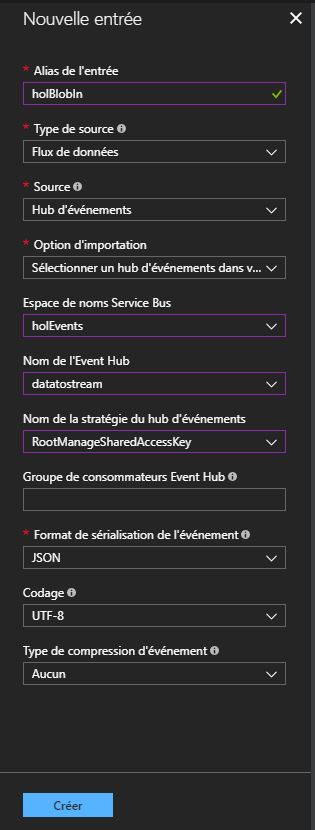
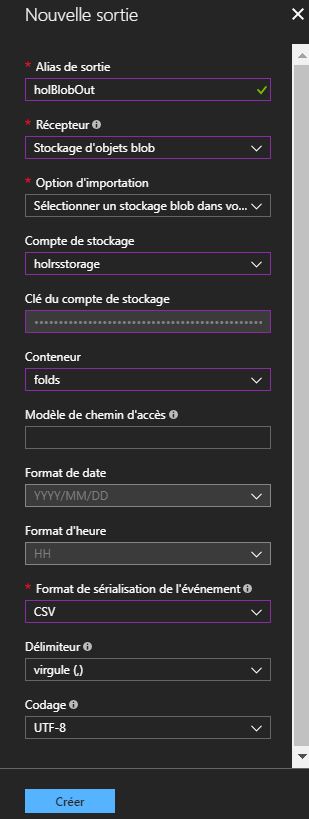
 



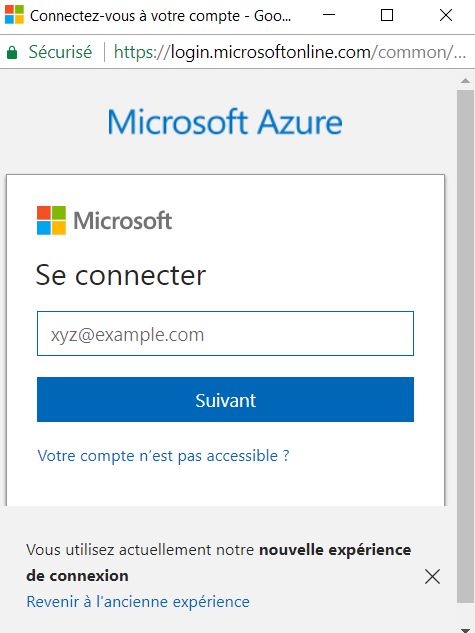
* Go back to your Resource Group
* Open the newly created Event Hubs and click on add an Event Hub, call it datatostream
* Go back to your Resource Group
* Open your newly created Storage Account and open Blob Objects
* Add a new Container



* Go back to your Resource Group
* Open your Stream Analytics
* Click on input and add an input from your Event Hub
* Go back to your Stream Analytics
* Click on output and add an output to your Blob Storage to archive outputs

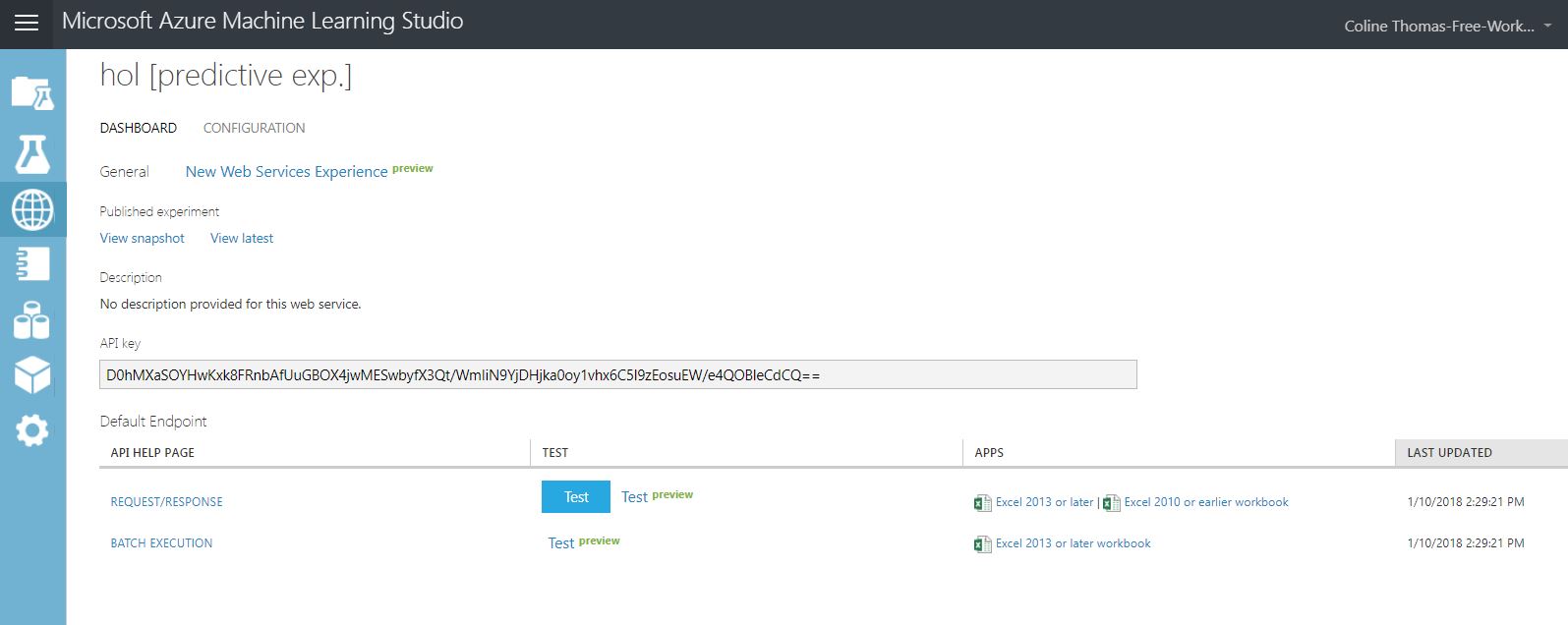
* Add an output to your PowerBi Account (you can create a free one) to visualize your data in real time



* Go back to your Stream Analytics
* Click on Functions
* Add a new function connected to your predictive API



* You can find your key there



* And the URL by clicking on REQUEST/RESPONSE and by copying the POST URL
* Go back to your Stream Analytics
* Click on Request
* Add the following request

WITH anomaly AS (

SELECT cast(T as float) as T,

cast(V as float) as V,

cast(AP as float) as AP,

cast(RH as float) as RH,

cast(PE as float) as PE,

cast(anomaly(T, V, AP, RH, PE) as float) as result

from holBlobIn

)

Select System.Timestamp as date, 'Paris' as location, T, V, AP, RH, PE, result \* 100 as result

Into holBlobOut

From anomaly

Select System.Timestamp as date,T, V, AP, RH, PE, result

Into holPbiOut

From anomaly

* Now save it
* And launch your streaming task
* Nothing is now happening because our Event Hub does not receive anything
* Open any Python IDE
* Copy the following code

*import* json  
*import* random  
*from* datetime *import* datetime  
*from* azure.servicebus *import* ServiceBusService  
  
*def* rd(*mu*, *sigma*):  
 *return* abs(round(random.normalvariate(*mu*, *sigma*), 2))  
  
  
*def* main():  
 sbs = ServiceBusService(service\_namespace='holEvents',  
 shared\_access\_key\_name='RootManageSharedAccessKey',  
 shared\_access\_key\_value='ugV/8wxg/Z0ZoTWBZWRUP5j2cgaEDiJC26ZLuoshotY=')  
 turn = 0  
 *while* turn >= 0:  
 t = rd(19.6, 67.6)  
 ap = rd(1002.6, 101.1)  
 rh = rd(54, 13.6)  
 v = rd(83.5, 99.1)  
 pe = rd(445.6, 61.1)  
  
 now = datetime.now().strftime("%M")  
  
 *if* turn == 0:  
 time = now  
 *else*:  
 *if* now != time:  
  
 data = {"T":str(t), "V":str(v), "AP":str(ap), "RH":str(rh), "PE":str(pe)}  
  
 body = str.encode(json.dumps(data))  
 *print*(body)  
 sbs.send\_event('datatostream', body)  
  
 time = now  
  
 turn += 1  
  
*if* \_\_name\_\_ == '\_\_main\_\_':  
 main()

* And launch it
* It will send continuously values similar to what the plant is supposed to send but with some errors because it’s not exactly the same.
* Let it turn and go to Power BI