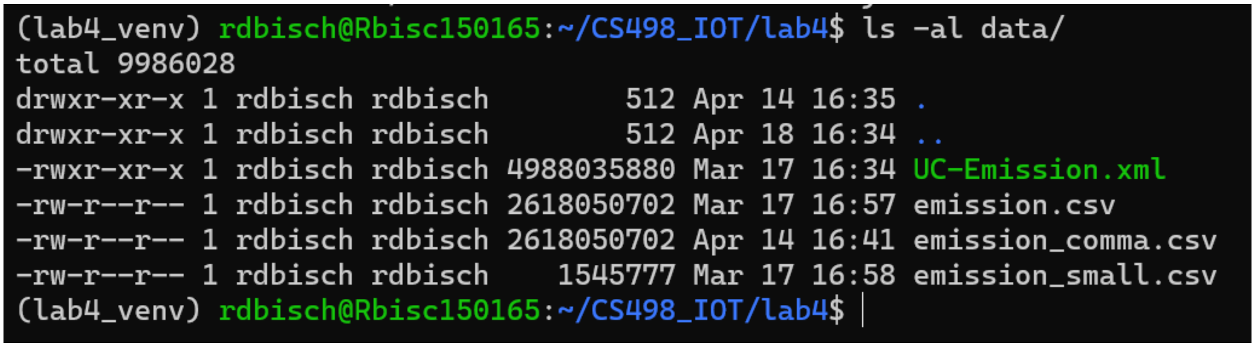
CS498 IoT Lab 4

Rick Bischoff (rdb4@illinois.edu)

https://gitlab.engr.illinois.edu/rdb4/sp21-cs498it-lab4
https://github.com/rdbisch/cs498_iot_lab4

Part 1 - Machine Learning

Step 1 - Acquiring Dataset



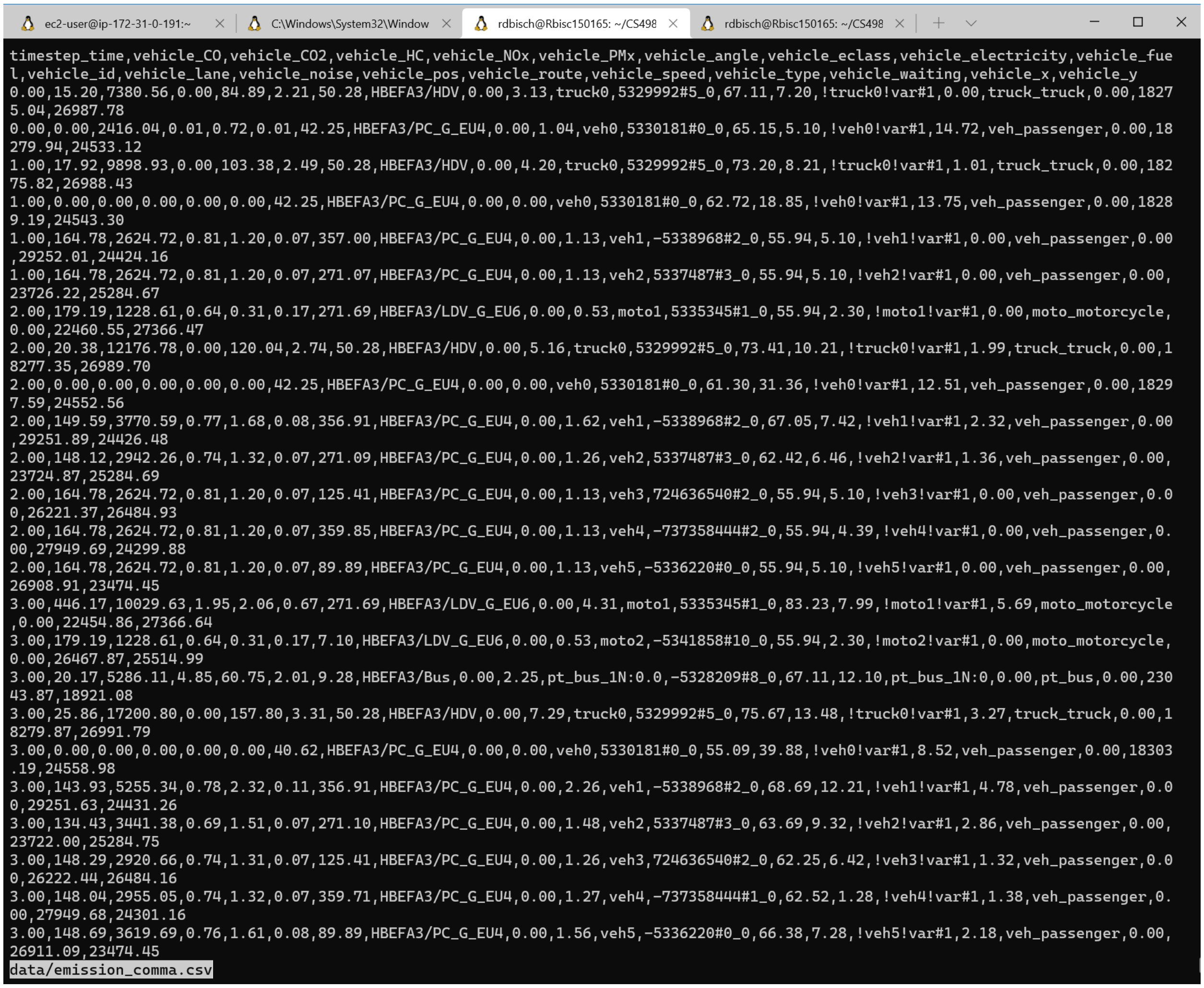
```
(lab4_venv) rdbisch@Rbisc150165:~/CS498_IOT/lab4$ ls -al data/
total 9986028
drwxr-xr-x 1 rdbisch rdbisch        512 Apr 14 16:35 .
drwxr-xr-x 1 rdbisch rdbisch        512 Apr 18 16:34 ..
-rwxr-xr-x 1 rdbisch rdbisch 4988035880 Mar 17 16:34 UC-Emission.xml
-rw-r--r-- 1 rdbisch rdbisch 2618050702 Mar 17 16:57 emission.csv
-rw-r--r-- 1 rdbisch rdbisch 2618050702 Apr 14 16:41 emission_comma.csv
-rw-r--r-- 1 rdbisch rdbisch    1545777 Mar 17 16:58 emission_small.csv
(lab4_venv) rdbisch@Rbisc150165:~/CS498_IOT/lab4$
```

Step 2 - Jupyter Notebook

See “Notebook_Practice.ipynb” in git repo.

Step 3 - Data Ingestion

See also Step1.



ec2-user@ip-172-31-0-191:~
C:\Windows\System32\Window
rdbisch@Rbisc150165: ~/CS498
rdbisch@Rbisc150165: ~/CS498

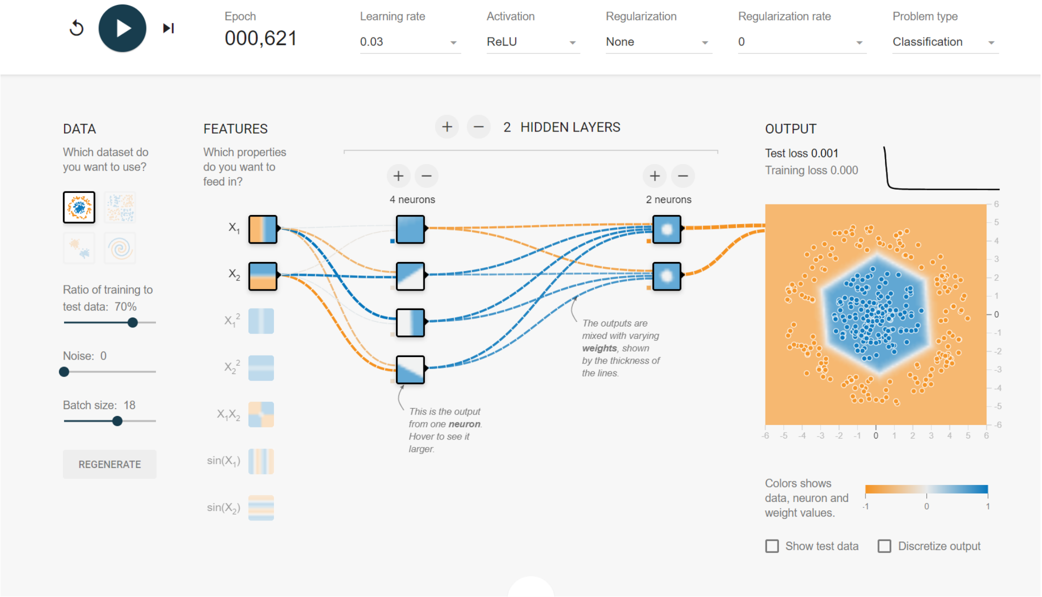
```
timestep_time,vehicle_CO,vehicle_CO2,vehicle_HC,vehicle_NOx,vehicle_PMx,vehicle_angle,vehicle_eclass,vehicle_electricity,vehicle_fue
l,vehicle_id,vehicle_lane,vehicle_noise,vehicle_pos,vehicle_route,vehicle_speed,vehicle_type,vehicle_waiting,vehicle_x,vehicle_y
0.00,15.20,7380.56,0.00,84.89,2.21,50.28,HBEFA3/HDV,0.00,3.13,truck0,5329992#5_0,67.11,7.20,!truck0!var#1,0.00,truck_truck,0.00,1827
5.04,26987.78
0.00,0.00,2416.04,0.01,0.72,0.01,42.25,HBEFA3/PC_G_EU4,0.00,1.04,veh0,5330181#0_0,65.15,5.10,!veh0!var#1,14.72,veh_passenger,0.00,18
279.94,24533.12
1.00,17.92,9898.93,0.00,103.38,2.49,50.28,HBEFA3/HDV,0.00,4.20,truck0,5329992#5_0,73.20,8.21,!truck0!var#1,1.01,truck_truck,0.00,182
75.82,26988.43
1.00,0.00,0.00,0.00,0.00,0.00,42.25,HBEFA3/PC_G_EU4,0.00,0.00,veh0,5330181#0_0,62.72,18.85,!veh0!var#1,13.75,veh_passenger,0.00,1828
9.19,24543.30
1.00,164.78,2624.72,0.81,1.20,0.07,357.00,HBEFA3/PC_G_EU4,0.00,1.13,veh1,-5338968#2_0,55.94,5.10,!veh1!var#1,0.00,veh_passenger,0.00
,29252.01,24424.16
1.00,164.78,2624.72,0.81,1.20,0.07,271.07,HBEFA3/PC_G_EU4,0.00,1.13,veh2,5337487#3_0,55.94,5.10,!veh2!var#1,0.00,veh_passenger,0.00,
23726.22,25284.67
2.00,179.19,1228.61,0.64,0.31,0.17,271.69,HBEFA3/LDV_G_EU6,0.00,0.53,moto1,5335345#1_0,55.94,2.30,!moto1!var#1,0.00,moto_motorcycle,
0.00,22460.55,27366.47
2.00,20.38,12176.78,0.00,120.04,2.74,50.28,HBEFA3/HDV,0.00,5.16,truck0,5329992#5_0,73.41,10.21,!truck0!var#1,1.99,truck_truck,0.00,1
8277.35,26989.70
2.00,0.00,0.00,0.00,0.00,0.00,42.25,HBEFA3/PC_G_EU4,0.00,0.00,veh0,5330181#0_0,61.30,31.36,!veh0!var#1,12.51,veh_passenger,0.00,1829
7.59,24552.56
2.00,149.59,3770.59,0.77,1.68,0.08,356.91,HBEFA3/PC_G_EU4,0.00,1.62,veh1,-5338968#2_0,67.05,7.42,!veh1!var#1,2.32,veh_passenger,0.00
,29251.89,24426.48
2.00,148.12,2942.26,0.74,1.32,0.07,271.09,HBEFA3/PC_G_EU4,0.00,1.26,veh2,5337487#3_0,62.42,6.46,!veh2!var#1,1.36,veh_passenger,0.00,
23724.87,25284.69
2.00,164.78,2624.72,0.81,1.20,0.07,125.41,HBEFA3/PC_G_EU4,0.00,1.13,veh3,724636540#2_0,55.94,5.10,!veh3!var#1,0.00,veh_passenger,0.0
0,26221.37,26484.93
2.00,164.78,2624.72,0.81,1.20,0.07,359.85,HBEFA3/PC_G_EU4,0.00,1.13,veh4,-737358444#2_0,55.94,4.39,!veh4!var#1,0.00,veh_passenger,0.
00,27949.69,24299.88
2.00,164.78,2624.72,0.81,1.20,0.07,89.89,HBEFA3/PC_G_EU4,0.00,1.13,veh5,-5336220#0_0,55.94,5.10,!veh5!var#1,0.00,veh_passenger,0.00,
26908.91,23474.45
3.00,446.17,10029.63,1.95,2.06,0.67,271.69,HBEFA3/LDV_G_EU6,0.00,4.31,moto1,5335345#1_0,83.23,7.99,!moto1!var#1,5.69,moto_motorcycle
,0.00,22454.86,27366.64
3.00,179.19,1228.61,0.64,0.31,0.17,7.10,HBEFA3/LDV_G_EU6,0.00,0.53,moto2,-5341858#10_0,55.94,2.30,!moto2!var#1,0.00,moto_motorcycle,
0.00,26467.87,25514.99
3.00,20.17,5286.11,4.85,60.75,2.01,9.28,HBEFA3/Bus,0.00,2.25,pt_bus_1N:0.0,-5328209#8_0,67.11,12.10,pt_bus_1N:0,0.00,pt_bus,0.00,230
43.87,18921.08
3.00,25.86,17200.80,0.00,157.80,3.31,50.28,HBEFA3/HDV,0.00,7.29,truck0,5329992#5_0,75.67,13.48,!truck0!var#1,3.27,truck_truck,0.00,1
8279.87,26991.79
3.00,0.00,0.00,0.00,0.00,0.00,40.62,HBEFA3/PC_G_EU4,0.00,0.00,veh0,5330181#0_0,55.09,39.88,!veh0!var#1,8.52,veh_passenger,0.00,18303
.19,24558.98
3.00,143.93,5255.34,0.78,2.32,0.11,356.91,HBEFA3/PC_G_EU4,0.00,2.26,veh1,-5338968#2_0,68.69,12.21,!veh1!var#1,4.78,veh_passenger,0.0
0,29251.63,24431.26
3.00,134.43,3441.38,0.69,1.51,0.07,271.10,HBEFA3/PC_G_EU4,0.00,1.48,veh2,5337487#3_0,63.69,9.32,!veh2!var#1,2.86,veh_passenger,0.00,
23722.00,25284.75
3.00,148.29,2920.66,0.74,1.31,0.07,125.41,HBEFA3/PC_G_EU4,0.00,1.26,veh3,724636540#2_0,62.25,6.42,!veh3!var#1,1.32,veh_passenger,0.0
0,26222.44,26484.16
3.00,148.04,2955.05,0.74,1.32,0.07,359.71,HBEFA3/PC_G_EU4,0.00,1.27,veh4,-737358444#1_0,62.52,1.28,!veh4!var#1,1.38,veh_passenger,0.
00,27949.68,24301.16
3.00,148.69,3619.69,0.76,1.61,0.08,89.89,HBEFA3/PC_G_EU4,0.00,1.56,veh5,-5336220#0_0,66.38,7.28,!veh5!var#1,2.18,veh_passenger,0.00,
26911.09,23474.45
data/emission_comma.csv
```

Step 4:

1. Try Relu, Sigmoid, Linear, and Tanh. Run for `600` Epochs. Note the output graphs and losses. **Q1: What do you observe? i.e. Do all activation functions give you a nice training result?**

   No. The TanH gave the best classification after 600 epochs, though Sigmoid was catching up fast if I had let it go longer. As it stands @ 600 epochs, ReLU has the 2nd best loss, but is seemingly only capturing a hexagon shape, and linear doesn't seem complicated enough to work.

i. ReLU:



Epoch 000,621

Learning rate 0.03

Activation ReLU

Regularization None

Regularization rate 0

Problem type Classification

DATA

Which dataset do you want to use?

Ratio of training to test data: 70%

Noise: 0

Batch size: 18

FEATURES

Which properties do you want to feed in?

2 HIDDEN LAYERS

4 neurons

2 neurons

*The outputs are mixed with varying **weights**, shown by the thickness of the lines.*

*This is the output from one **neuron**. Hover to see it larger.*

OUTPUT

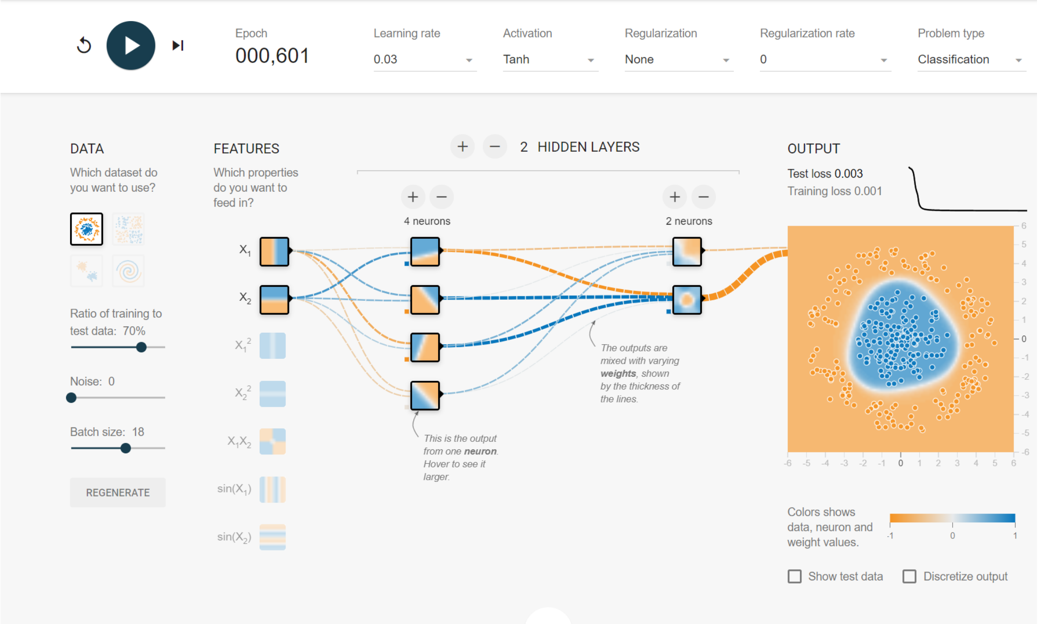
Test loss 0.001

Training loss 0.000

Colors shows data, neuron and weight values.

□ Show test data □ Discretize output

ii. Tanh:



Epoch 000,601

Learning rate 0.03

Activation Tanh

Regularization None

Regularization rate 0

Problem type Classification

DATA

Which dataset do you want to use?

Ratio of training to test data: 70%

Noise: 0

Batch size: 18

FEATURES

Which properties do you want to feed in?

2 HIDDEN LAYERS

4 neurons

2 neurons

*The outputs are mixed with varying **weights**, shown by the thickness of the lines.*

*This is the output from one **neuron**. Hover to see it larger.*

OUTPUT

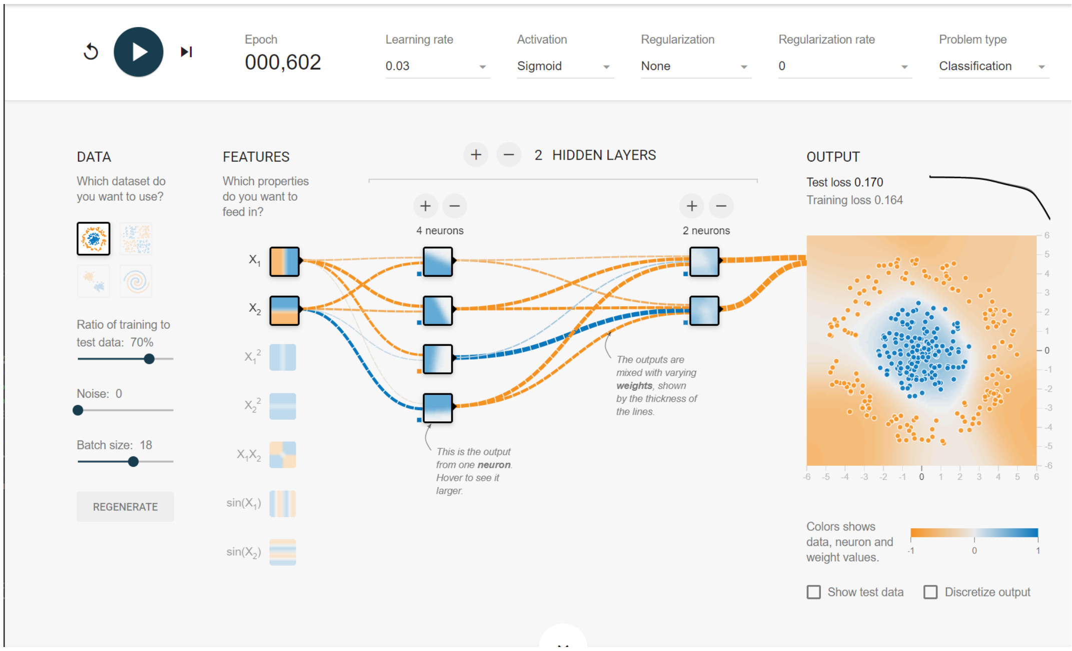
Test loss 0.003

Training loss 0.001

Colors shows data, neuron and weight values.

□ Show test data □ Discretize output

iii. Sigmoid



Epoch 000,602

Learning rate 0.03

Activation Sigmoid

Regularization None

Regularization rate 0

Problem type Classification

DATA

Which dataset do you want to use?

Ratio of training to test data: 70%

Noise: 0

Batch size: 18

REGENERATE

FEATURES

Which properties do you want to feed in?

2 HIDDEN LAYERS

4 neurons

2 neurons

*This is the output from one **neuron**. Hover to see it larger.*

*The outputs are mixed with varying **weights**, shown by the thickness of the lines.*

OUTPUT

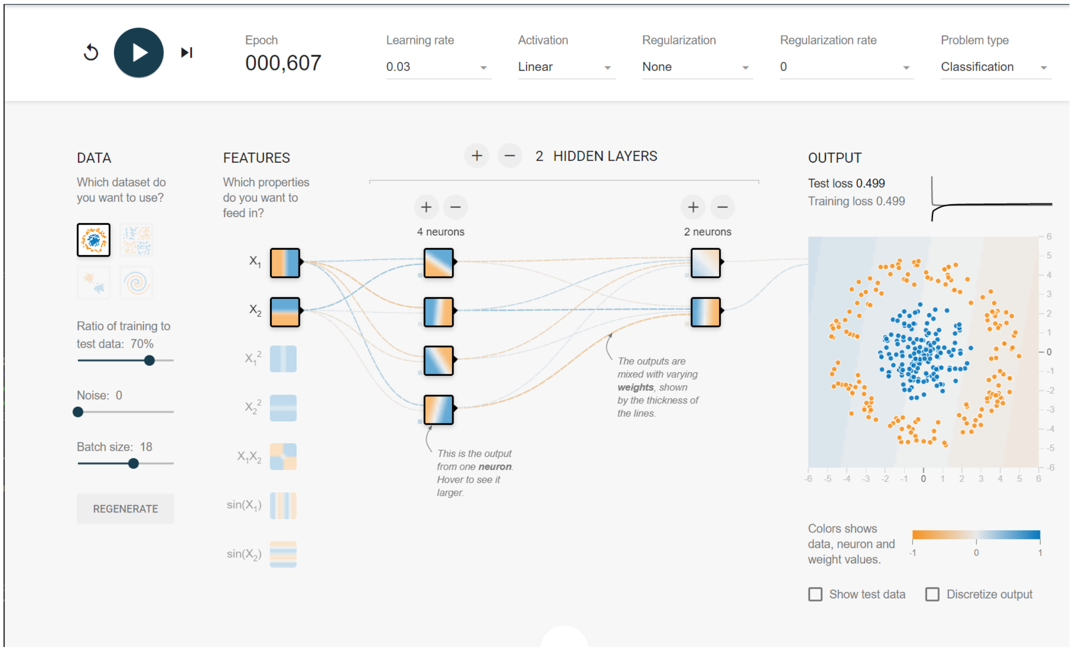
Test loss 0.170

Training loss 0.164

Colors shows data, neuron and weight values.

□ Show test data □ Discretize output

iv. Linear



Epoch 000,607

Learning rate 0.03

Activation Linear

Regularization None

Regularization rate 0

Problem type Classification

DATA

Which dataset do you want to use?

Ratio of training to test data: 70%

Noise: 0

Batch size: 18

REGENERATE

FEATURES

Which properties do you want to feed in?

2 HIDDEN LAYERS

4 neurons

2 neurons

*This is the output from one **neuron**. Hover to see it larger.*

*The outputs are mixed with varying **weights**, shown by the thickness of the lines.*

OUTPUT

Test loss 0.499

Training loss 0.499

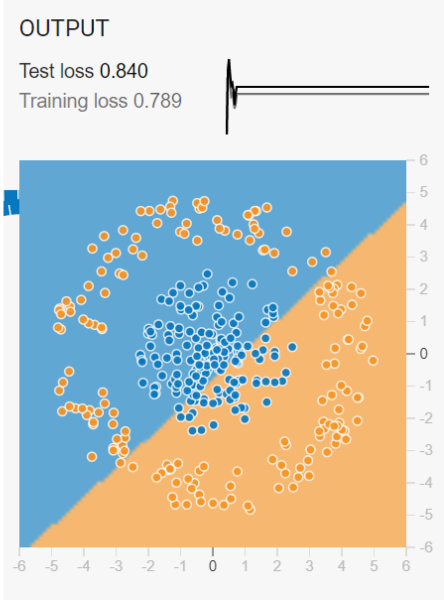
Colors shows data, neuron and weight values.

□ Show test data □ Discretize output

2. Change the learning rate. Try a large learning rate like `10` and a small learning rate like `0.00001`. **Q2: Look at the loss, epochs, etc. What do you observe?**

______

Large learning rates force an answer much too soon and the model never recovers from it. Our previous best, tanH, converges to something like



OUTPUT

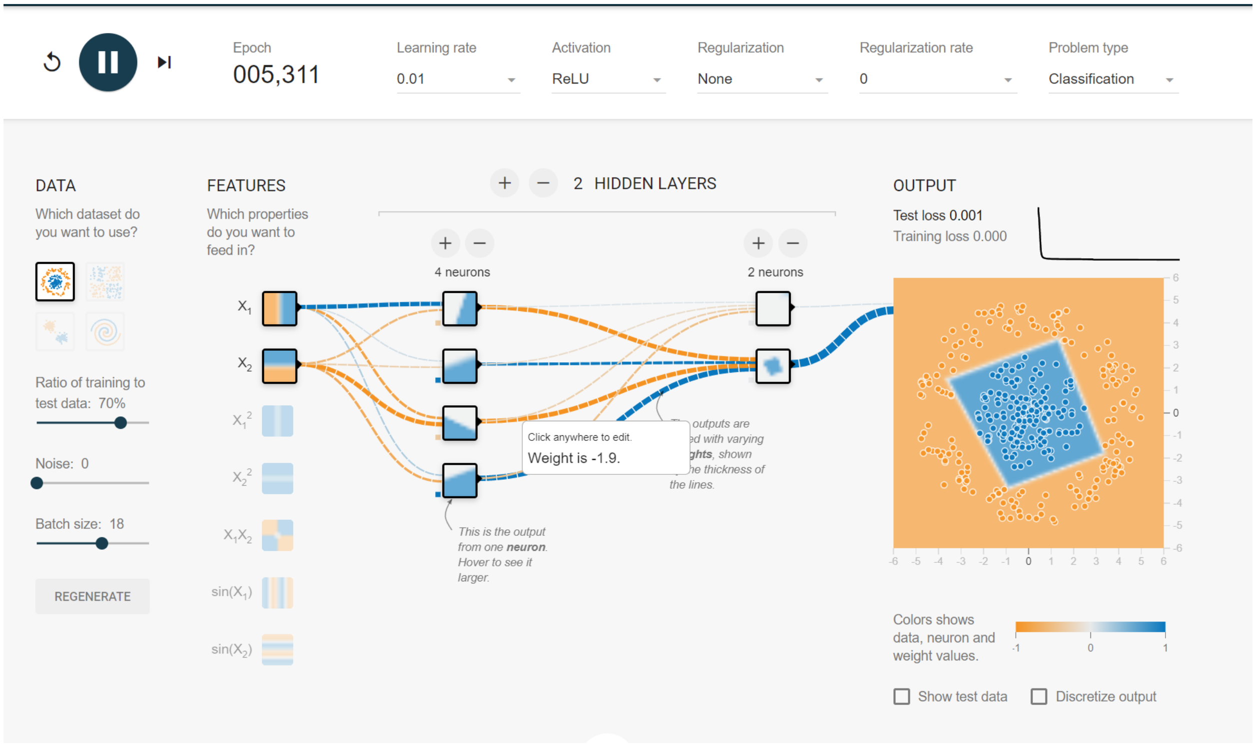
Test loss 0.840

Training loss 0.789

, whereas the smallest learning rates show signs of improvement but very very slow. This is really inefficient.

3. **Q3: What do you observe? i.e. Do they change over time? Can the weights be negative?** Notice that the thicker the dashed line (-----) is, the heavier the weight is.

Yes, the weights change over time. This is how the network adjusts its guesses by changing the weights. They can be negative just as well as a positive. Here is a screenshot



Epoch 005,311

Learning rate 0.01

Activation ReLU

Regularization None

Regularization rate 0

Problem type Classification

Ratio of training to test data: 70%

Noise: 0

Batch size: 18

2 HIDDEN LAYERS

Test loss 0.001

Training loss 0.000

Click anywhere to edit.

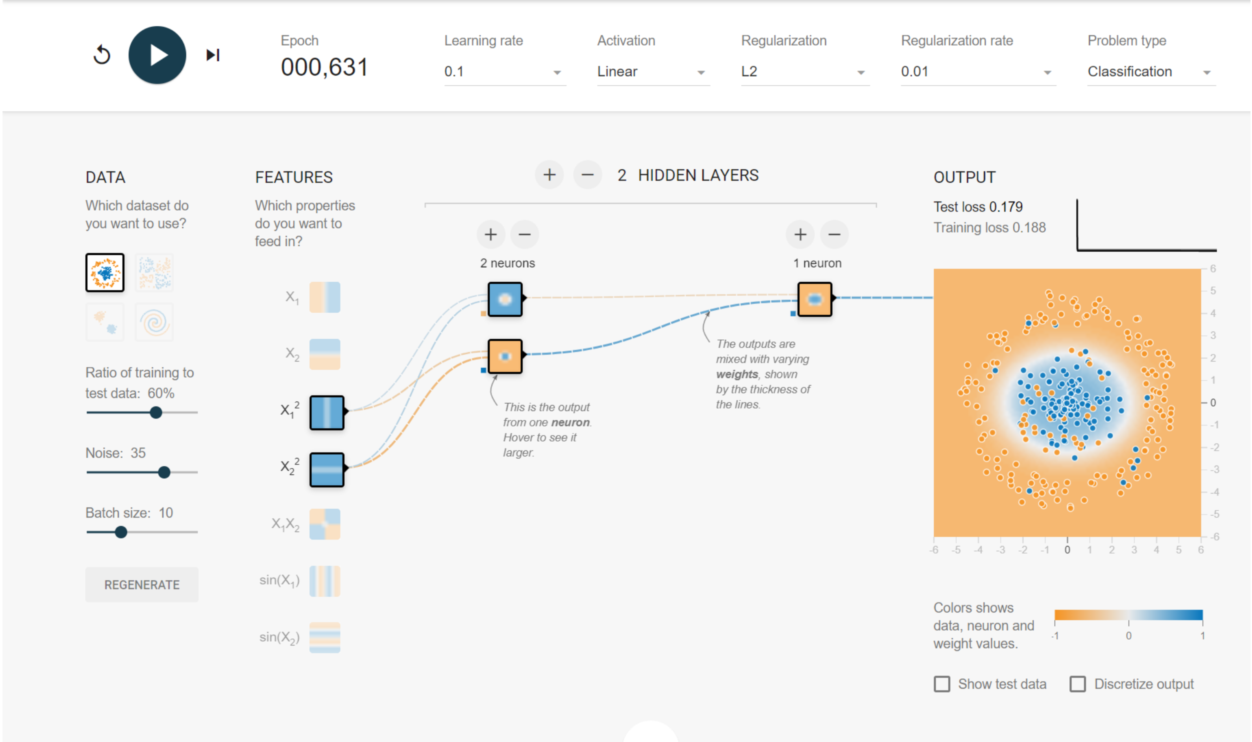
Weight is -1.9.

DATA
Which dataset do you want to use?
REGENERATE
FEATURES
Which properties do you want to feed in?
X1
X2
X1²
X2²
X1X2
sin(X1)
sin(X2)
+ −
4 neurons
2 neurons
outputs are ...ed with varying ...ghts, shown ...he thickness of the lines.
This is the output from one neuron. Hover to see it larger.
OUTPUT
6 5 4 3 2 1 0 -1 -2 -3 -4 -5 -6
-6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6
Colors shows data, neuron and weight values.
-1 0 1
□ Show test data
□ Discretize output

4. **Q4: Look at the test and training loss. What do you observe?**

Compared to the earlier ReLU fit without regularization the losses are much higher, but we can see that this one is fitting the data better visually. In this particular case, the test loss is higher than the training loss.

7. Lastly, adjust anything and get the best result you can for this dataset. Take a screenshot.



Epoch 000,631

Learning rate 0.1

Activation Linear

Regularization L2

Regularization rate 0.01

Problem type Classification

DATA

Which dataset do you want to use?

Ratio of training to test data: 60%

Noise: 35

Batch size: 10

FEATURES

Which properties do you want to feed in?

2 HIDDEN LAYERS

2 neurons

1 neuron

This is the output from one ***neuron***. Hover to see it larger.

The outputs are mixed with varying ***weights***, shown by the thickness of the lines.

OUTPUT

Test loss 0.179

Training loss 0.188

Colors shows data, neuron and weight values.

□ Show test data □ Discretize output

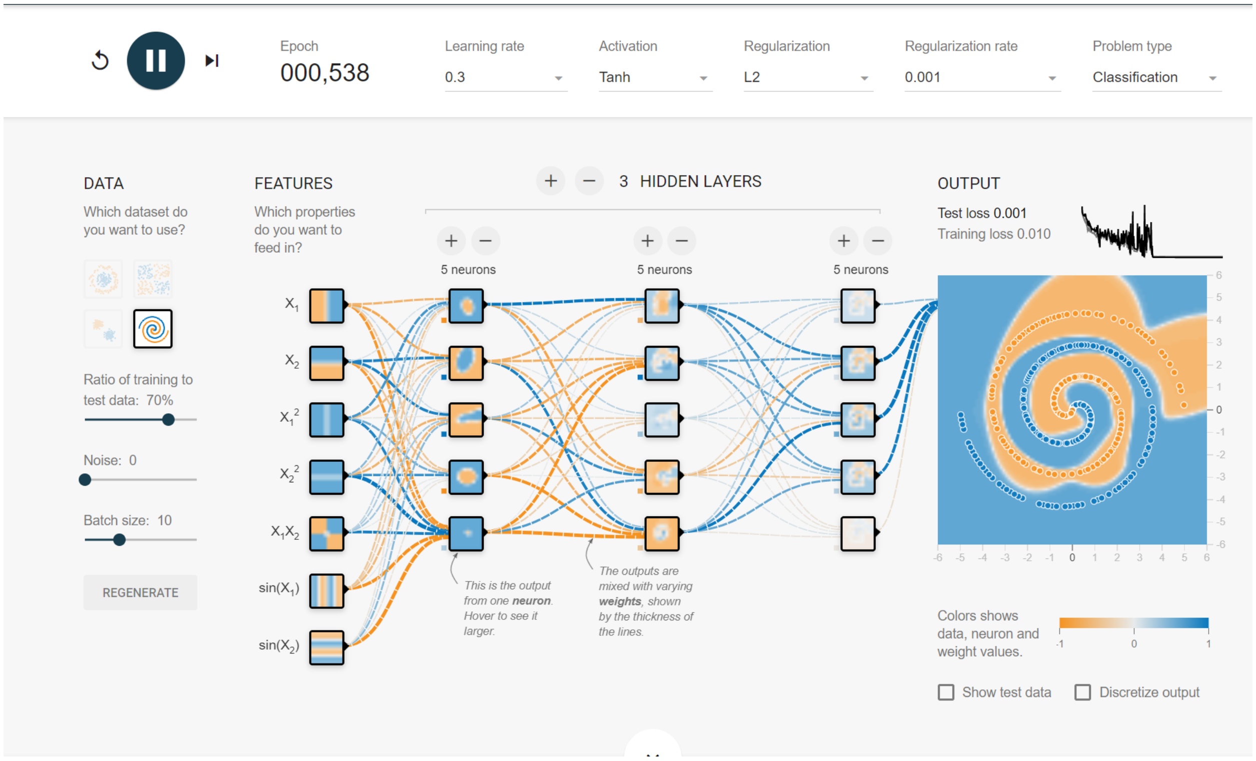
5. **Q5: Why are other features needed to get a better training result in this case?**

Other features are needed because the spiral cannot be defined as a function in just X1 and X2 mathematically (vertical line test!) regardless of how well the neural network works. So it needs extra features to be able to capture the complexity.

6. **Q6: What do you observe (with regards to training speed, loss, etc.)?**

Training speed slows down because there are more parameters (weights) to adjust.   Adding more layers or more nodes always makes the training loss lower because of a tendency to overfit, but the test error doesn't necessarily show the same behavior.

Lastly, adjust anything and get the best result you can for this dataset. Take a screenshot.


Epoch 000,538

Learning rate 0.3

Activation Tanh

Regularization L2

Regularization rate 0.001

Problem type Classification

Ratio of training to test data: 70%

Noise: 0

Batch size: 10

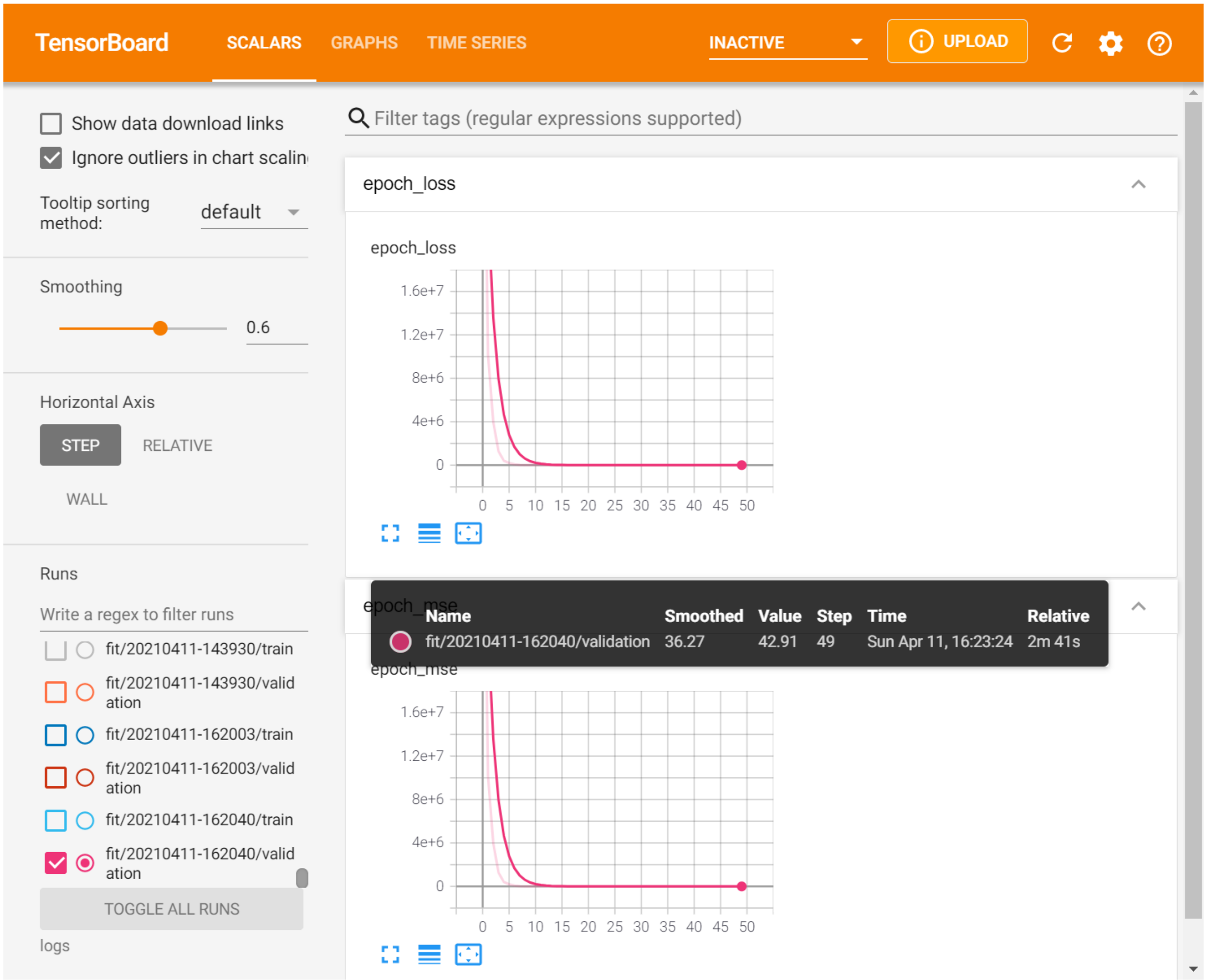
Test loss 0.001

Training loss 0.010

DATA
Which dataset do you want to use?
REGENERATE
FEATURES
Which properties do you want to feed in?
X_1
X_2
X_1^2
X_2^2
X_1X_2
$\sin(X_1)$
$\sin(X_2)$
3 HIDDEN LAYERS
5 neurons
5 neurons
5 neurons
This is the output from one neuron. Hover to see it larger.
The outputs are mixed with varying weights, shown by the thickness of the lines.
OUTPUT
Colors shows data, neuron and weight values.
-1
0
1
□ Show test data
□ Discretize output
-6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6

Step 5 - Machine Learning on Traffic Emission Data

Screen Shot of Small Dataset



□ Show data download links

☑ Ignore outliers in chart scalin

Tooltip sorting method: default

Smoothing 0.6

Horizontal Axis
[✓ STEP] RELATIVE WALL

Runs

Write a regex to filter runs

- □ ○ fit/20210411-143930/train
- □ ○ fit/20210411-143930/validation
- □ ○ fit/20210411-162003/train
- □ ○ fit/20210411-162003/validation
- □ ○ fit/20210411-162040/train
- ☑ ● fit/20210411-162040/validation

	Name	Smoothed	Value	Step	Time	Relative
●	fit/20210411-162040/validation	36.27	42.91	49	Sun Apr 11, 16:23:24	2m 41s

In [235]:

```
# Hyperparameters
learning_rate = 0.001 ### FILL IN A NUMBER
epochs = 50 ### FILL IN A NUMBER
batch_size = 125 ### FILL IN A NUMBER

# Label
label_name = "vehicle_CO2"
shuffle = True

#---Create a sequential model---#
model = tf.keras.models.Sequential([
    # Add the feature layer
    feature_layer,

    # First hidden layer with 20 nodes
    tf.keras.layers.Dense(units=20,
                          activation='relu',
                          kernel_regularizer=tf.keras.regularizers.l1(l=0.1),
                          name='Hidden1'),

    # First hidden layer with 20 nodes
    tf.keras.layers.Dense(units=20,
                          activation='relu',
                          kernel_regularizer=tf.keras.regularizers.l1(l=0.1),
                          name='Hidden3'),

    # Output layer
    tf.keras.layers.Dense(units=1,
                          activation='linear',
                          name='Output')

])


model.compile(optimizer=tf.keras.optimizers.Adam(lr=learning_rate),
              loss=tf.keras.losses.MeanSquaredError(),
              metrics=['mse'])


#---Train the Model---#
# Keras TensorBoard callback.
logdir = "logs/fit/" + datetime.now().strftime("%Y%m%d-%H%M%S")
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=logdir)

train_lbl = np.array(train_df_norm["vehicle_CO2"])
#train_df = train_df.drop(columns=["vehicle_CO2"])
# Split the datasets into features and label.
train_ft = {name:np.array(value) for name, value in train_df_norm.items()}
# train_lbl = np.array(train_ft.pop(label_name))

val_lbl =  np.array(valid_df_norm["vehicle_CO2"])
#val_df = val_df.drop(columns=["vehicle_CO2"])
val_ft = {name:np.array(value) for name, value in valid_df_norm.items()}


# Keras TensorBoard callback.
```

<u>Screen Shot of Big Dataset</u>

See Notebook_Learning2.ipynb for results for the entire data set.

Part 2

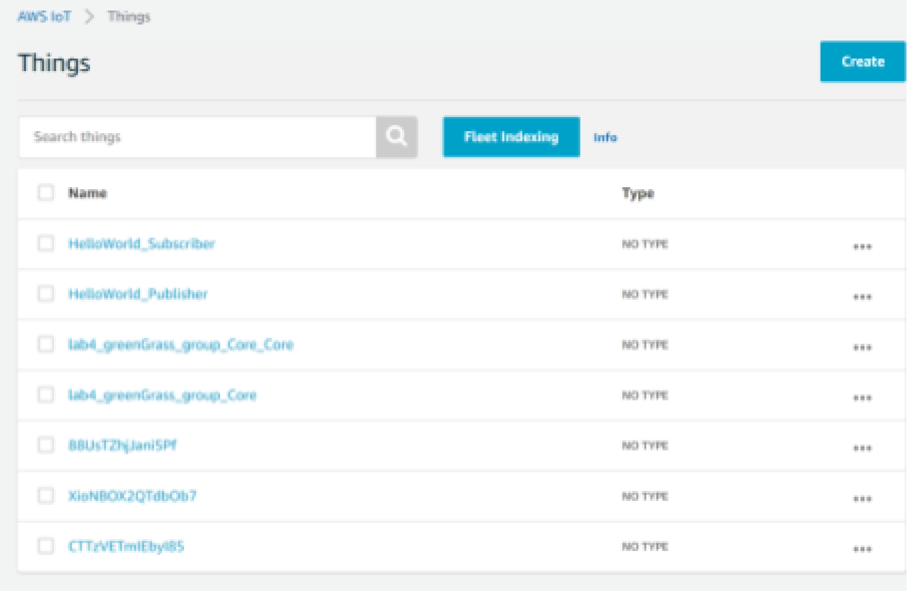
1. Compared with TCP-based client-server communication, what are the pros and cons for MQTT-based publish/subscribe.

MQTT offers a pretty handy abstraction for network communications. By abstracting communications away into “Topics”, and then further abstracting the communication to “Publish” and “Subscribe” patterns.

There are downsides though. Because it is so lightweight, it is not a very good solution for heavy payloads, like photos or videos. It also requires developers to expend more effort than they would otherwise under a fully managed protocol. Further, while the use of “topics” simplifies many things, the complexity of the system will dramatically increase over time as more and more topics are created. To the devices, this will be transparent, but to any central systems or human beings working on it, the complexity will become a bottleneck.

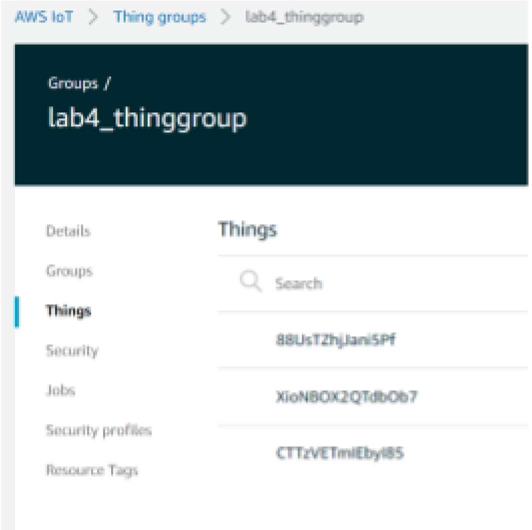
Section 1.3-1.4

IoT Core Setup, Rules, Policies, etc.



□ Name	Type	
□ HelloWorld_Subscriber	NO TYPE	•••
□ HelloWorld_Publisher	NO TYPE	•••
□ lab4_greenGrass_group_Core_Core	NO TYPE	•••
□ lab4_greenGrass_group_Core	NO TYPE	•••
□ BBUsTZhjJaniSPf	NO TYPE	•••
□ XioNBOX2QTdbOb7	NO TYPE	•••
□ CTTzVETmlEbyl85	NO TYPE	•••

Screenshot of available Things in AWS IoT



lab4_thinggroup

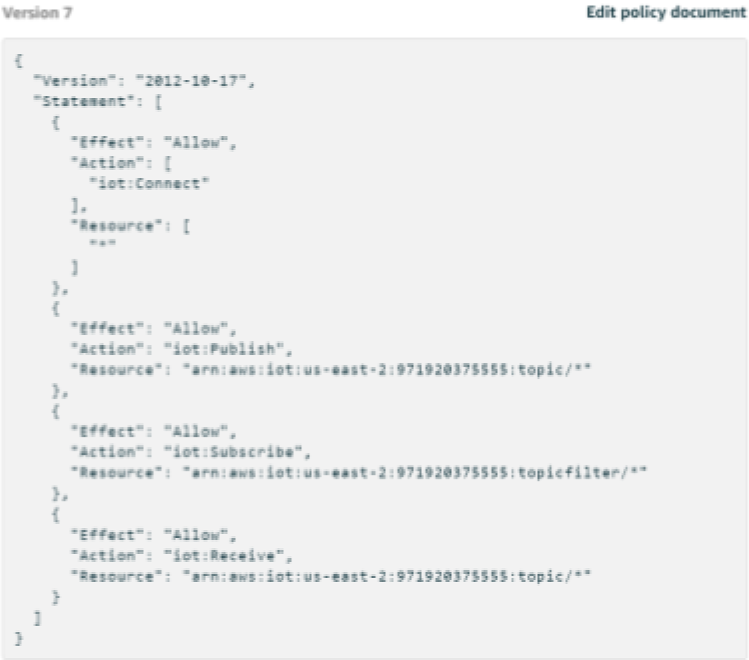
Things

88UsTZhjJani5Pf

XioNBOX2QTdbOb7

CTTzVETmlEbyl85

Screenshot of IoT Thing Group I created to manage policies easier

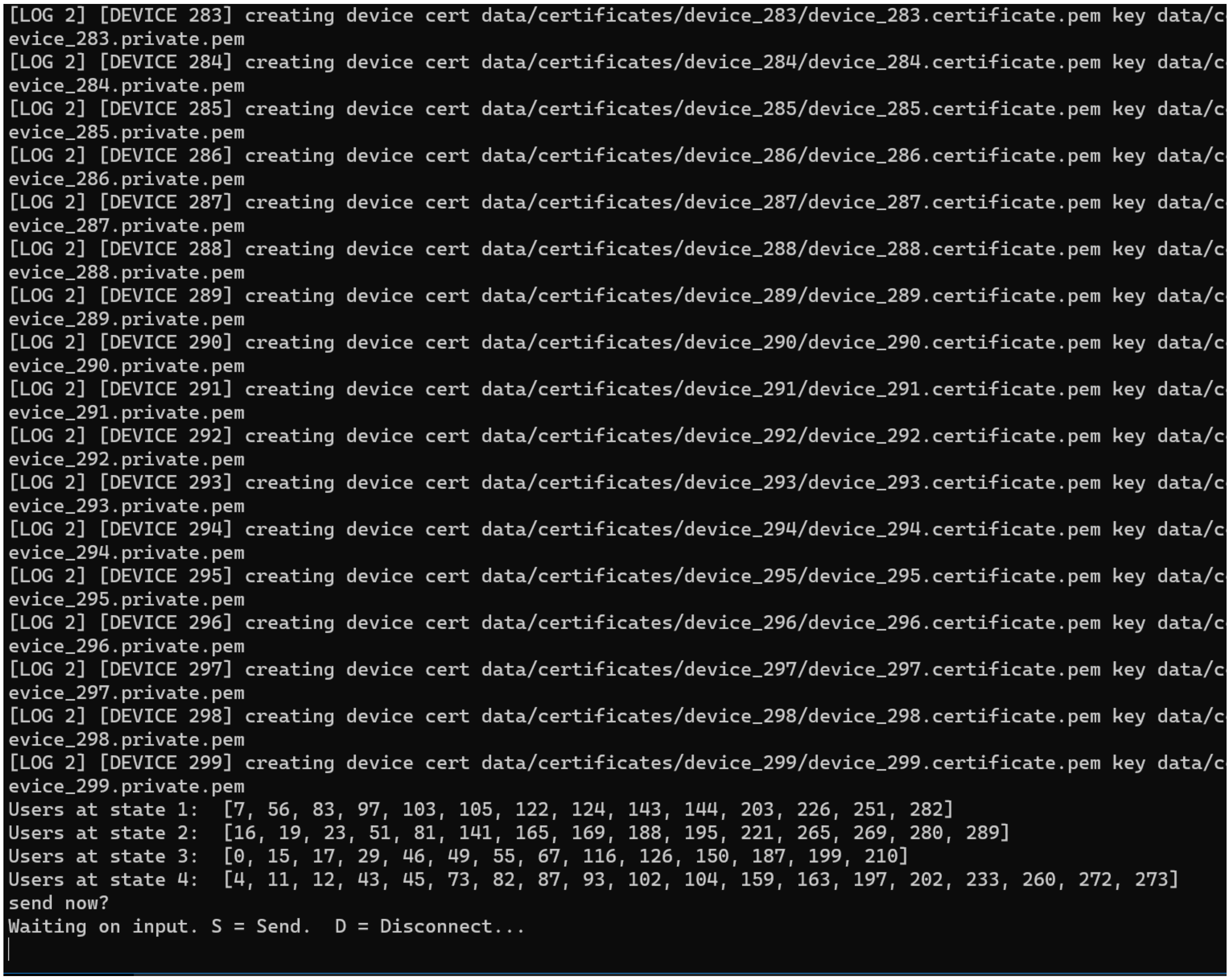


```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iot:Connect"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": "iot:Publish",
      "Resource": "arn:aws:iot:us-east-2:971920375555:topic/*"
    },
    {
      "Effect": "Allow",
      "Action": "iot:Subscribe",
      "Resource": "arn:aws:iot:us-east-2:971920375555:topicfilter/*"
    },
    {
      "Effect": "Allow",
      "Action": "iot:Receive",
      "Resource": "arn:aws:iot:us-east-2:971920375555:topic/*"
    }
  ]
}
```

Example policy created to allow IoT things to use MQTT

Section 1.5 Create Many Devices

I modified “createThing-Cert.py” in obvious ways to create “Things” en-masse, and make the certificates easier to manage. I also added code to automatically add the “Things” created here to the “Thing Group” mentioned above, to make management of policies easier.

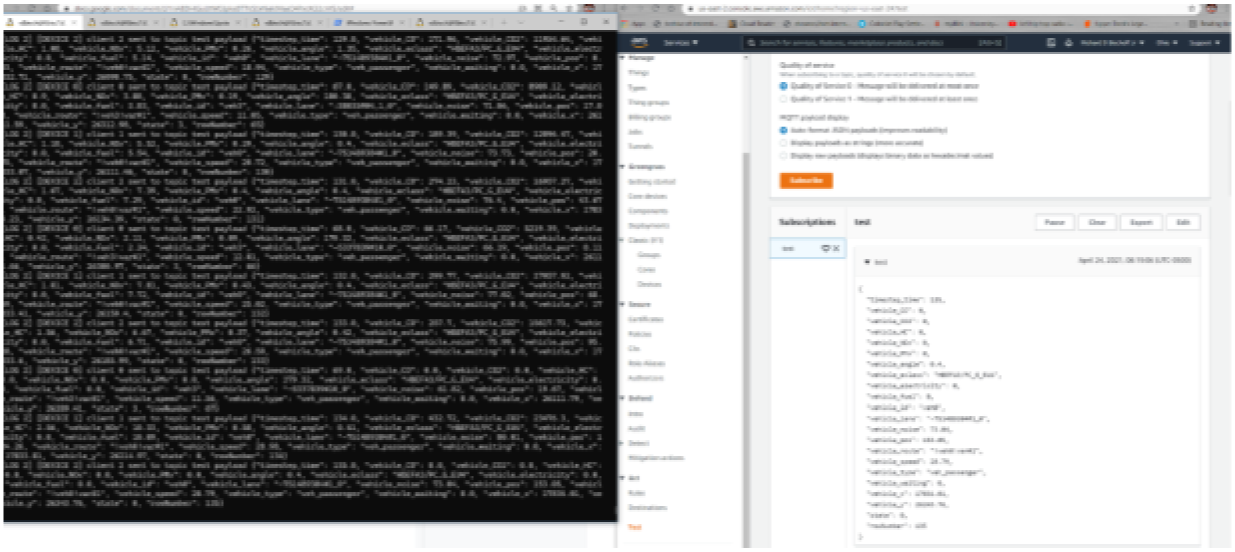


```
[LOG 2] [DEVICE 283] creating device cert data/certificates/device_283/device_283.certificate.pem key data/c
evice_283.private.pem
[LOG 2] [DEVICE 284] creating device cert data/certificates/device_284/device_284.certificate.pem key data/c
evice_284.private.pem
[LOG 2] [DEVICE 285] creating device cert data/certificates/device_285/device_285.certificate.pem key data/c
evice_285.private.pem
[LOG 2] [DEVICE 286] creating device cert data/certificates/device_286/device_286.certificate.pem key data/c
evice_286.private.pem
[LOG 2] [DEVICE 287] creating device cert data/certificates/device_287/device_287.certificate.pem key data/c
evice_287.private.pem
[LOG 2] [DEVICE 288] creating device cert data/certificates/device_288/device_288.certificate.pem key data/c
evice_288.private.pem
[LOG 2] [DEVICE 289] creating device cert data/certificates/device_289/device_289.certificate.pem key data/c
evice_289.private.pem
[LOG 2] [DEVICE 290] creating device cert data/certificates/device_290/device_290.certificate.pem key data/c
evice_290.private.pem
[LOG 2] [DEVICE 291] creating device cert data/certificates/device_291/device_291.certificate.pem key data/c
evice_291.private.pem
[LOG 2] [DEVICE 292] creating device cert data/certificates/device_292/device_292.certificate.pem key data/c
evice_292.private.pem
[LOG 2] [DEVICE 293] creating device cert data/certificates/device_293/device_293.certificate.pem key data/c
evice_293.private.pem
[LOG 2] [DEVICE 294] creating device cert data/certificates/device_294/device_294.certificate.pem key data/c
evice_294.private.pem
[LOG 2] [DEVICE 295] creating device cert data/certificates/device_295/device_295.certificate.pem key data/c
evice_295.private.pem
[LOG 2] [DEVICE 296] creating device cert data/certificates/device_296/device_296.certificate.pem key data/c
evice_296.private.pem
[LOG 2] [DEVICE 297] creating device cert data/certificates/device_297/device_297.certificate.pem key data/c
evice_297.private.pem
[LOG 2] [DEVICE 298] creating device cert data/certificates/device_298/device_298.certificate.pem key data/c
evice_298.private.pem
[LOG 2] [DEVICE 299] creating device cert data/certificates/device_299/device_299.certificate.pem key data/c
evice_299.private.pem
Users at state 1:  [7, 56, 83, 97, 103, 105, 122, 124, 143, 144, 203, 226, 251, 282]
Users at state 2:  [16, 19, 23, 51, 81, 141, 165, 169, 188, 195, 221, 265, 269, 280, 289]
Users at state 3:  [0, 15, 17, 29, 46, 49, 55, 67, 116, 126, 150, 187, 199, 210]
Users at state 4:  [4, 11, 12, 43, 45, 73, 82, 87, 93, 102, 104, 159, 163, 197, 202, 233, 260, 272, 273]
send now?
Waiting on input. S = Send.  D = Disconnect...
```

Section 1.6 Use the Device Simulator

I modified “emulator_client.py” to reference the certificates created above. I added logging to make it easier to track topics and devices on the screen. I also added code so that the “Things” would emit a single row from the vehicle datasets per event. Then I modified the loop to run until there is no more rows to read, and then disconnect it from AWS..

This is the same data I will be analyzing in later parts of the lab.



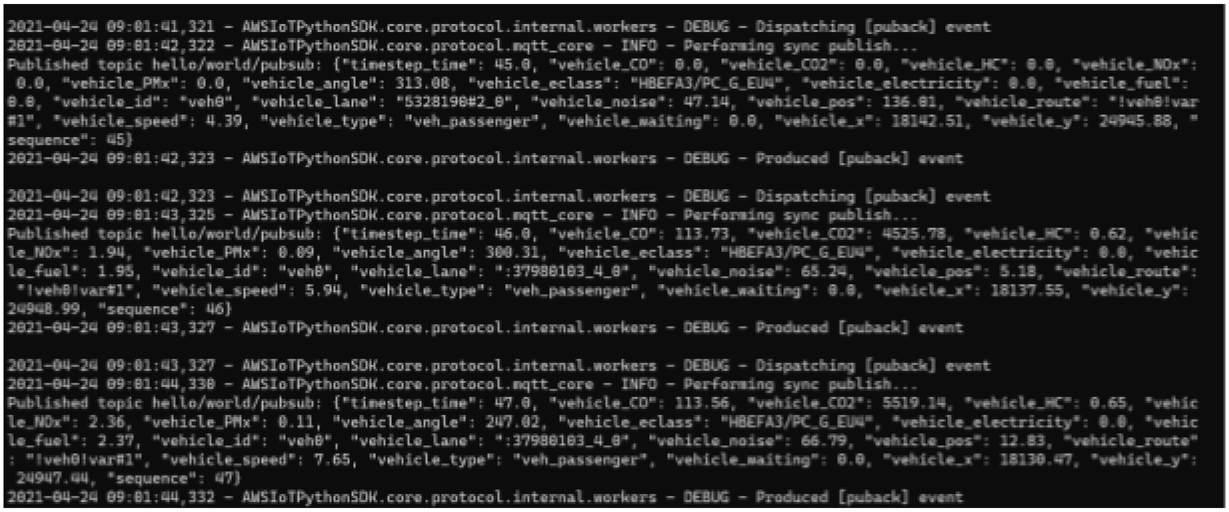
Snapshot of the device simulator publishing data

Section 1.6 Greengrass

Note: In the repository, this code lives in the greengrass_module4/ sub-directory.

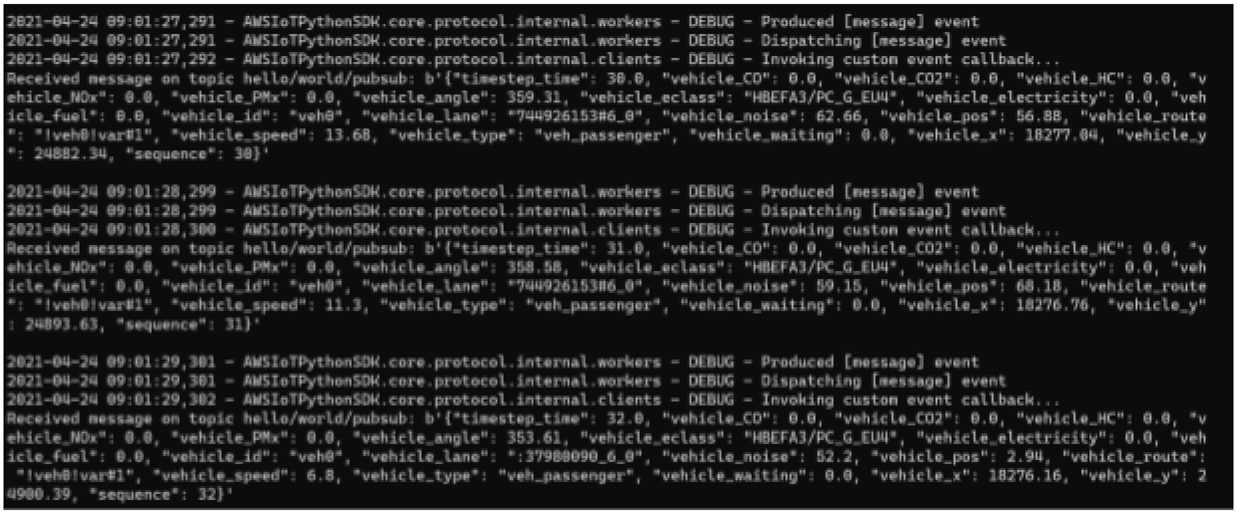
I am using my Raspberry Pi for my final project, so rather than risk that project, I opted to use an EC2 instance instead.   The configuration of this was relatively straightforward, and had the GreenGrass publisher/subscriber tutorial running OK.

I then modified “basicDiscovery.py”. I removed the payload option from the command line and instead changed it to a data argument. Then, I would send over the data row by row, similar to how emulator_client works.



```
2021-04-24 09:01:41,321 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Dispatching [puback] event
2021-04-24 09:01:42,322 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Performing sync publish...
Published topic hello/world/pubsub: {"timestep_time": 45.0, "vehicle_CO": 0.0, "vehicle_CO2": 0.0, "vehicle_HC": 0.0, "vehicle_NOx":
 0.0, "vehicle_PMx": 0.0, "vehicle_angle": 313.08, "vehicle_eclass": "HBEFA3/PC_G_EU4", "vehicle_electricity": 0.0, "vehicle_fuel":
0.0, "vehicle_id": "veh0", "vehicle_lane": "5328190#2_0", "vehicle_noise": 47.14, "vehicle_pos": 136.01, "vehicle_route": "!veh0!var
#1", "vehicle_speed": 4.39, "vehicle_type": "veh_passenger", "vehicle_waiting": 0.0, "vehicle_x": 18142.51, "vehicle_y": 24945.88, "
sequence": 45}
2021-04-24 09:01:42,323 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Produced [puback] event

2021-04-24 09:01:42,323 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Dispatching [puback] event
2021-04-24 09:01:43,325 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Performing sync publish...
Published topic hello/world/pubsub: {"timestep_time": 46.0, "vehicle_CO": 113.73, "vehicle_CO2": 4525.78, "vehicle_HC": 0.62, "vehic
le_NOx": 1.94, "vehicle_PMx": 0.09, "vehicle_angle": 300.31, "vehicle_eclass": "HBEFA3/PC_G_EU4", "vehicle_electricity": 0.0, "vehic
le_fuel": 1.95, "vehicle_id": "veh0", "vehicle_lane": ":37980103_4_0", "vehicle_noise": 65.24, "vehicle_pos": 5.18, "vehicle_route":
 "!veh0!var#1", "vehicle_speed": 5.94, "vehicle_type": "veh_passenger", "vehicle_waiting": 0.0, "vehicle_x": 18137.55, "vehicle_y":
24948.99, "sequence": 46}
2021-04-24 09:01:43,327 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Produced [puback] event

2021-04-24 09:01:43,327 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Dispatching [puback] event
2021-04-24 09:01:44,330 - AWSIoTPythonSDK.core.protocol.mqtt_core - INFO - Performing sync publish...
Published topic hello/world/pubsub: {"timestep_time": 47.0, "vehicle_CO": 113.56, "vehicle_CO2": 5519.14, "vehicle_HC": 0.65, "vehic
le_NOx": 2.36, "vehicle_PMx": 0.11, "vehicle_angle": 247.02, "vehicle_eclass": "HBEFA3/PC_G_EU4", "vehicle_electricity": 0.0, "vehic
le_fuel": 2.37, "vehicle_id": "veh0", "vehicle_lane": ":37980103_4_0", "vehicle_noise": 66.79, "vehicle_pos": 12.83, "vehicle_route"
: "!veh0!var#1", "vehicle_speed": 7.65, "vehicle_type": "veh_passenger", "vehicle_waiting": 0.0, "vehicle_x": 18130.47, "vehicle_y":
 24947.44, "sequence": 47}
2021-04-24 09:01:44,332 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Produced [puback] event
```

Greengrass, publish.sh



```
2021-04-24 09:01:27,291 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Produced [message] event
2021-04-24 09:01:27,291 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Dispatching [message] event
2021-04-24 09:01:27,292 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Invoking custom event callback...
Received message on topic hello/world/pubsub: b'{"timestep_time": 30.0, "vehicle_CO": 0.0, "vehicle_CO2": 0.0, "vehicle_HC": 0.0, "v
ehicle_NOx": 0.0, "vehicle_PMx": 0.0, "vehicle_angle": 359.31, "vehicle_eclass": "HBEFA3/PC_G_EU4", "vehicle_electricity": 0.0, "veh
icle_fuel": 0.0, "vehicle_id": "veh0", "vehicle_lane": "744926153#6_0", "vehicle_noise": 62.66, "vehicle_pos": 56.88, "vehicle_route
": "!veh0!var#1", "vehicle_speed": 13.68, "vehicle_type": "veh_passenger", "vehicle_waiting": 0.0, "vehicle_x": 18277.04, "vehicle_y
": 24882.34, "sequence": 30}'

2021-04-24 09:01:28,299 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Produced [message] event
2021-04-24 09:01:28,299 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Dispatching [message] event
2021-04-24 09:01:28,300 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Invoking custom event callback...
Received message on topic hello/world/pubsub: b'{"timestep_time": 31.0, "vehicle_CO": 0.0, "vehicle_CO2": 0.0, "vehicle_HC": 0.0, "v
ehicle_NOx": 0.0, "vehicle_PMx": 0.0, "vehicle_angle": 358.58, "vehicle_eclass": "HBEFA3/PC_G_EU4", "vehicle_electricity": 0.0, "veh
icle_fuel": 0.0, "vehicle_id": "veh0", "vehicle_lane": "744926153#6_0", "vehicle_noise": 59.15, "vehicle_pos": 68.18, "vehicle_route
": "!veh0!var#1", "vehicle_speed": 11.3, "vehicle_type": "veh_passenger", "vehicle_waiting": 0.0, "vehicle_x": 18276.76, "vehicle_y"
: 24893.63, "sequence": 31}'

2021-04-24 09:01:29,301 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Produced [message] event
2021-04-24 09:01:29,301 - AWSIoTPythonSDK.core.protocol.internal.workers - DEBUG - Dispatching [message] event
2021-04-24 09:01:29,302 - AWSIoTPythonSDK.core.protocol.internal.clients - DEBUG - Invoking custom event callback...
Received message on topic hello/world/pubsub: b'{"timestep_time": 32.0, "vehicle_CO": 0.0, "vehicle_CO2": 0.0, "vehicle_HC": 0.0, "v
ehicle_NOx": 0.0, "vehicle_PMx": 0.0, "vehicle_angle": 353.61, "vehicle_eclass": "HBEFA3/PC_G_EU4", "vehicle_electricity": 0.0, "veh
icle_fuel": 0.0, "vehicle_id": "veh0", "vehicle_lane": ":37980090_6_0", "vehicle_noise": 52.2, "vehicle_pos": 2.94, "vehicle_route":
 "!veh0!var#1", "vehicle_speed": 6.8, "vehicle_type": "veh_passenger", "vehicle_waiting": 0.0, "vehicle_x": 18276.16, "vehicle_y": 2
4900.39, "sequence": 32}'
```

Greengrass, sub.sh

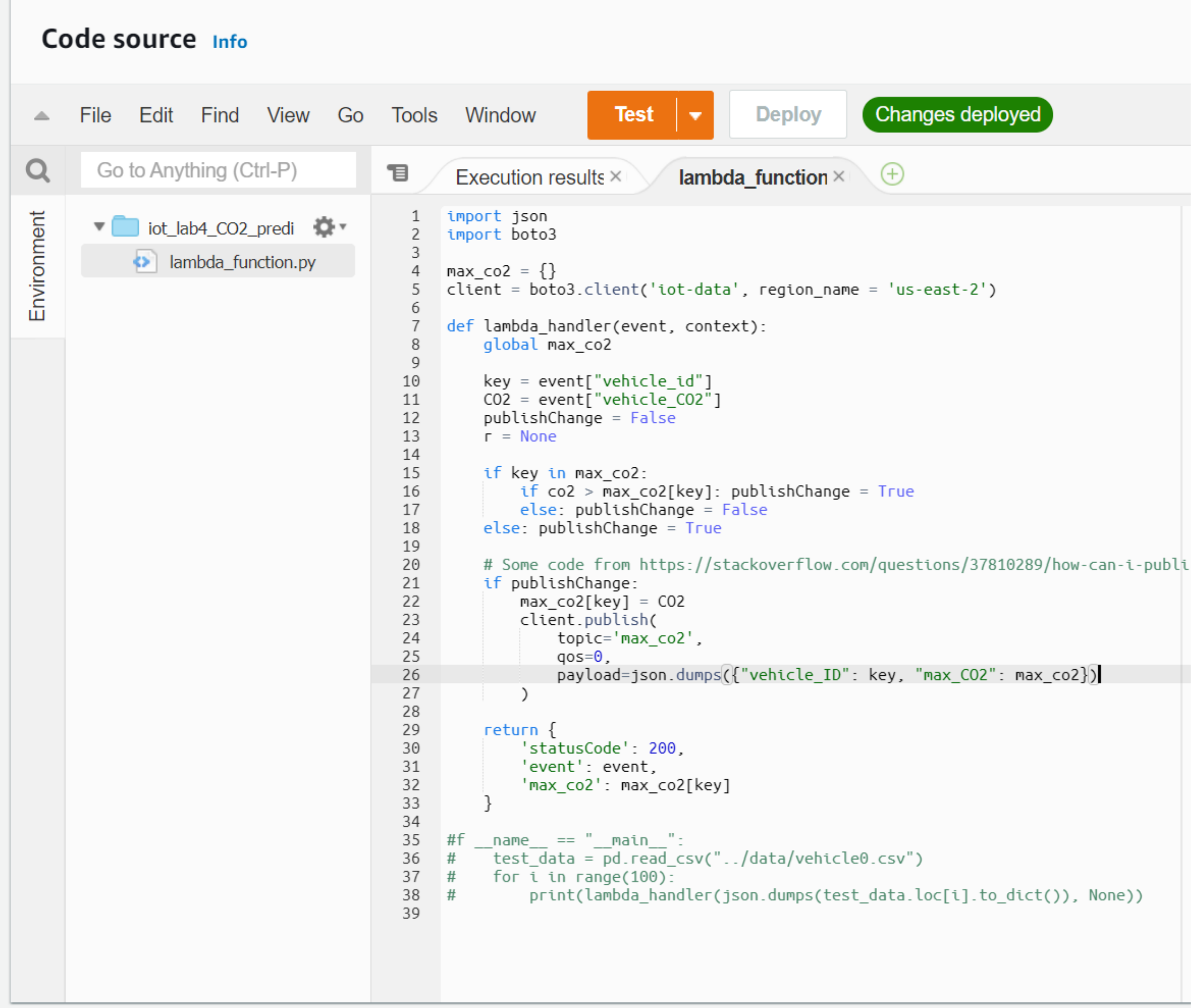
Section 2 Data Inference

Section 2.1 Lambda

Due to problems with python2 compatibility. I threw away the provided deploy_package lambda and started my own from scratch.

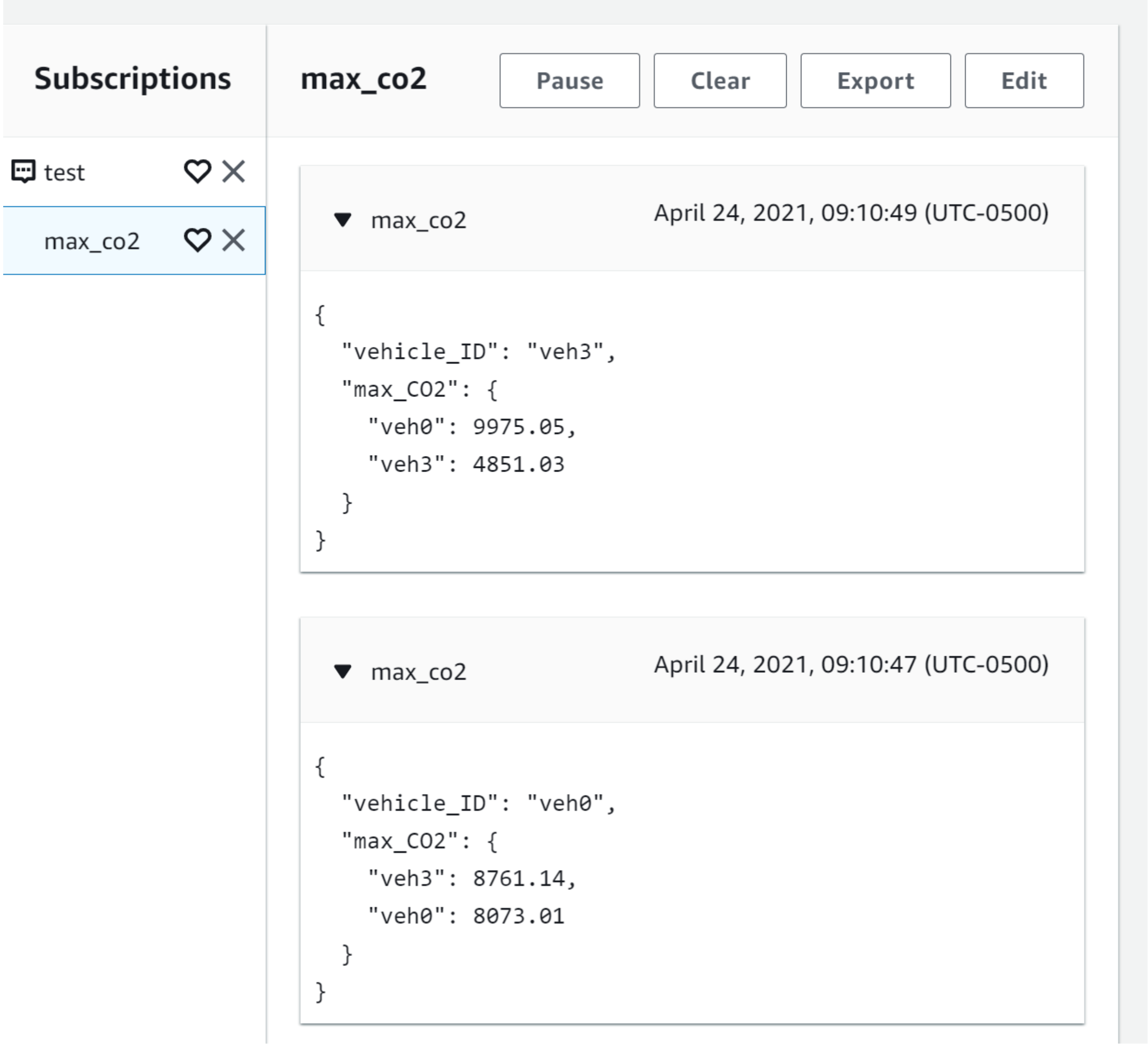
I first misunderstood the lab and actually created a python script that took the model from part 1 and scored it. I was, however, unfortunately unable to get that to work in AWS Lambdas because of the environment restrictions.

So instead I simply implemented a max-CO2 filter as instructed.



```python
import json
import boto3

max_co2 = {}
client = boto3.client('iot-data', region_name = 'us-east-2')

def lambda_handler(event, context):
    global max_co2

    key = event["vehicle_id"]
    CO2 = event["vehicle_CO2"]
    publishChange = False
    r = None

    if key in max_co2:
        if co2 > max_co2[key]: publishChange = True
        else: publishChange = False
    else: publishChange = True

    # Some code from https://stackoverflow.com/questions/37810289/how-can-i-publi
    if publishChange:
        max_co2[key] = CO2
        client.publish(
            topic='max_co2',
            qos=0,
            payload=json.dumps({"vehicle_ID": key, "max_CO2": max_co2})
        )

    return {
        'statusCode': 200,
        'event': event,
        'max_co2': max_co2[key]
    }

#f __name__ == "__main__":
#    test_data = pd.read_csv("../data/vehicle0.csv")
#    for i in range(100):
#        print(lambda_handler(json.dumps(test_data.loc[i].to_dict()), None))
```

Here is a screenshot of it working using the messages published from emulator_client.py.



max_co2 April 24, 2021, 09:10:49 (UTC-0500)

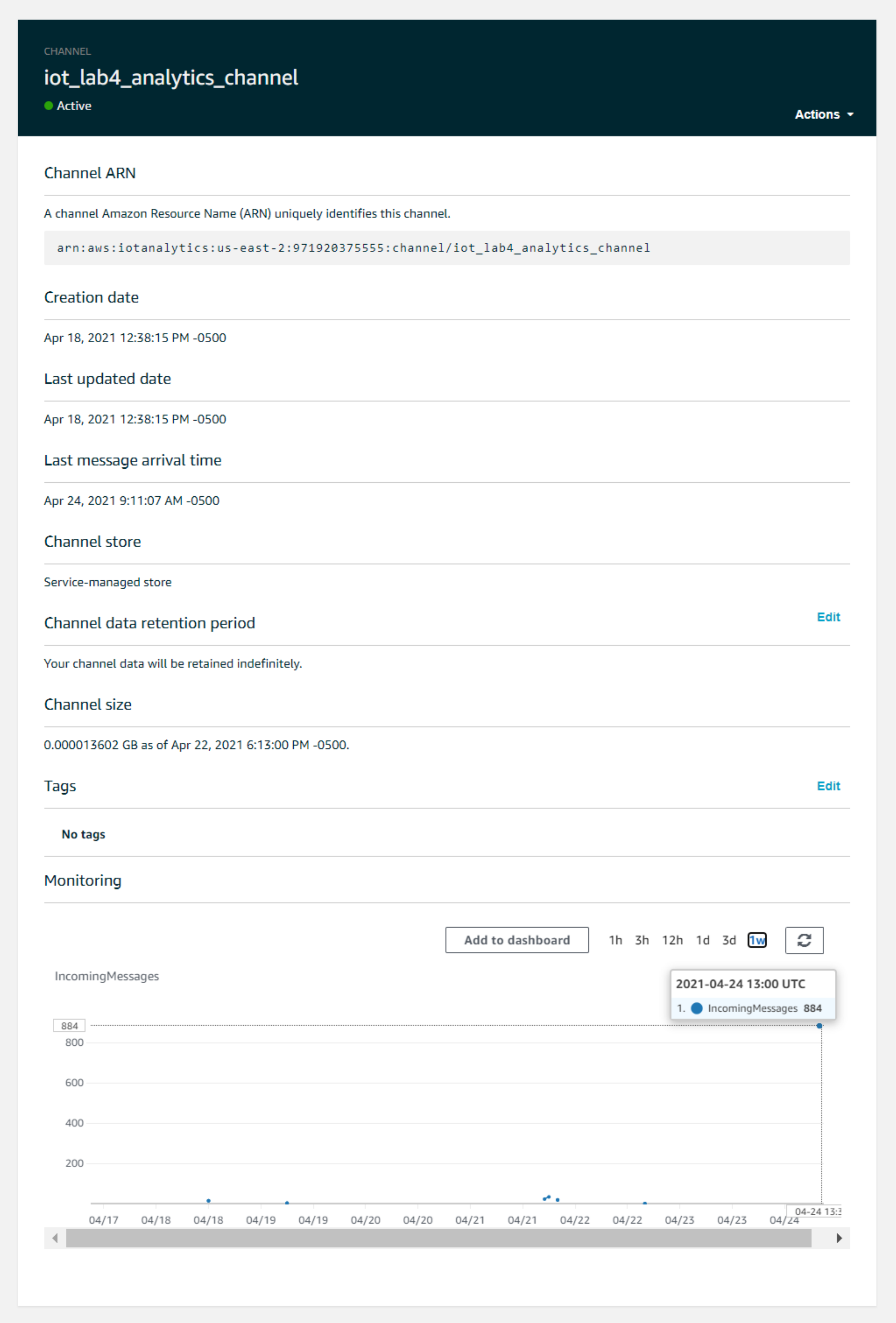
```
{
  "vehicle_ID": "veh3",
  "max_CO2": {
    "veh0": 9975.05,
    "veh3": 4851.03
  }
}
```

max_co2 April 24, 2021, 09:10:47 (UTC-0500)

```
{
  "vehicle_ID": "veh0",
  "max_CO2": {
    "veh3": 8761.14,
    "veh0": 8073.01
  }
}
```

Section 3 - IoT Analytics

I setup the channels, pipelines, data stores, and datasets as instructed.



CHANNEL

iot_lab4_analytics_channel

● Active

Channel ARN

A channel Amazon Resource Name (ARN) uniquely identifies this channel.

```
arn:aws:iotanalytics:us-east-2:971920375555:channel/iot_lab4_analytics_channel
```

Creation date

Apr 18, 2021 12:38:15 PM -0500

Last updated date

Apr 18, 2021 12:38:15 PM -0500

Last message arrival time

Apr 24, 2021 9:11:07 AM -0500

Channel store

Service-managed store

Channel data retention period

Your channel data will be retained indefinitely.

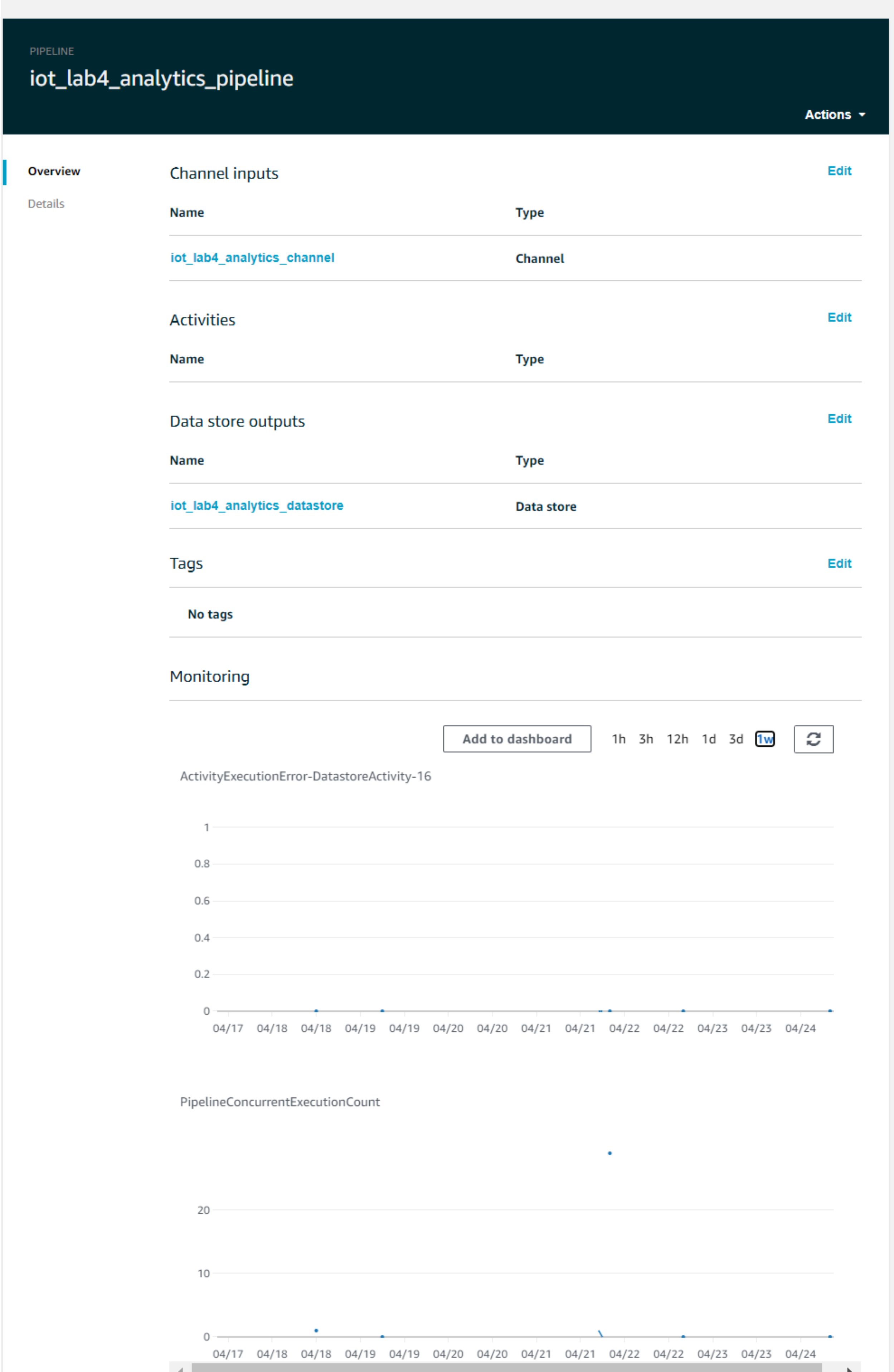
Channel size

0.000013602 GB as of Apr 22, 2021 6:13:00 PM -0500.

Tags

No tags

Monitoring



iot_lab4_analytics_pipeline

Channel inputs

Name	Type
iot_lab4_analytics_channel	Channel

Activities

Name	Type

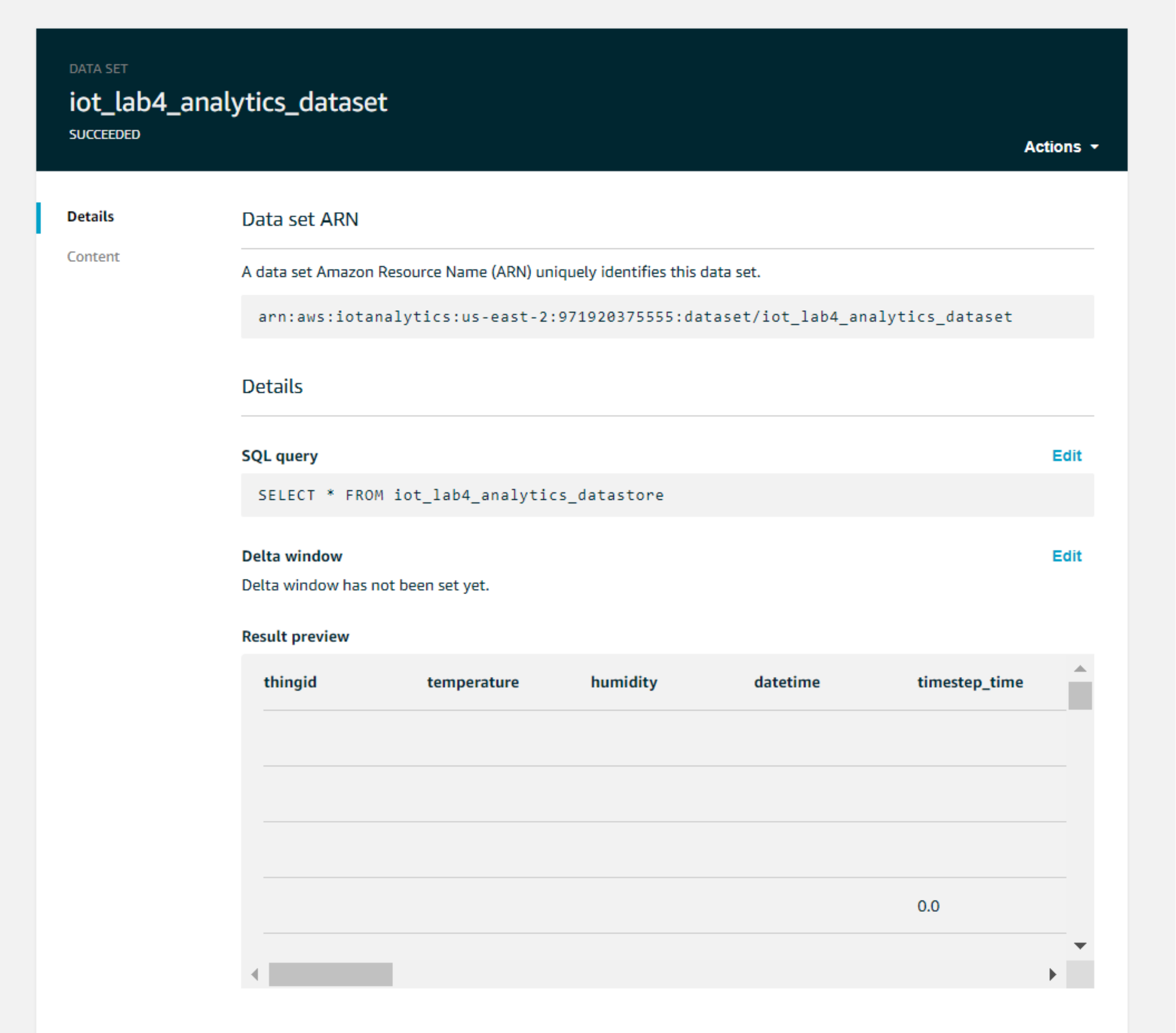
Data store outputs

Name	Type
iot_lab4_analytics_datastore	Data store

Tags

No tags

Monitoring



iot_lab4_analytics_dataset

Data set ARN

A data set Amazon Resource Name (ARN) uniquely identifies this data set.

```
arn:aws:iotanalytics:us-east-2:971920375555:dataset/iot_lab4_analytics_dataset
```

Details

SQL query

```
SELECT * FROM iot_lab4_analytics_datastore
```

Delta window

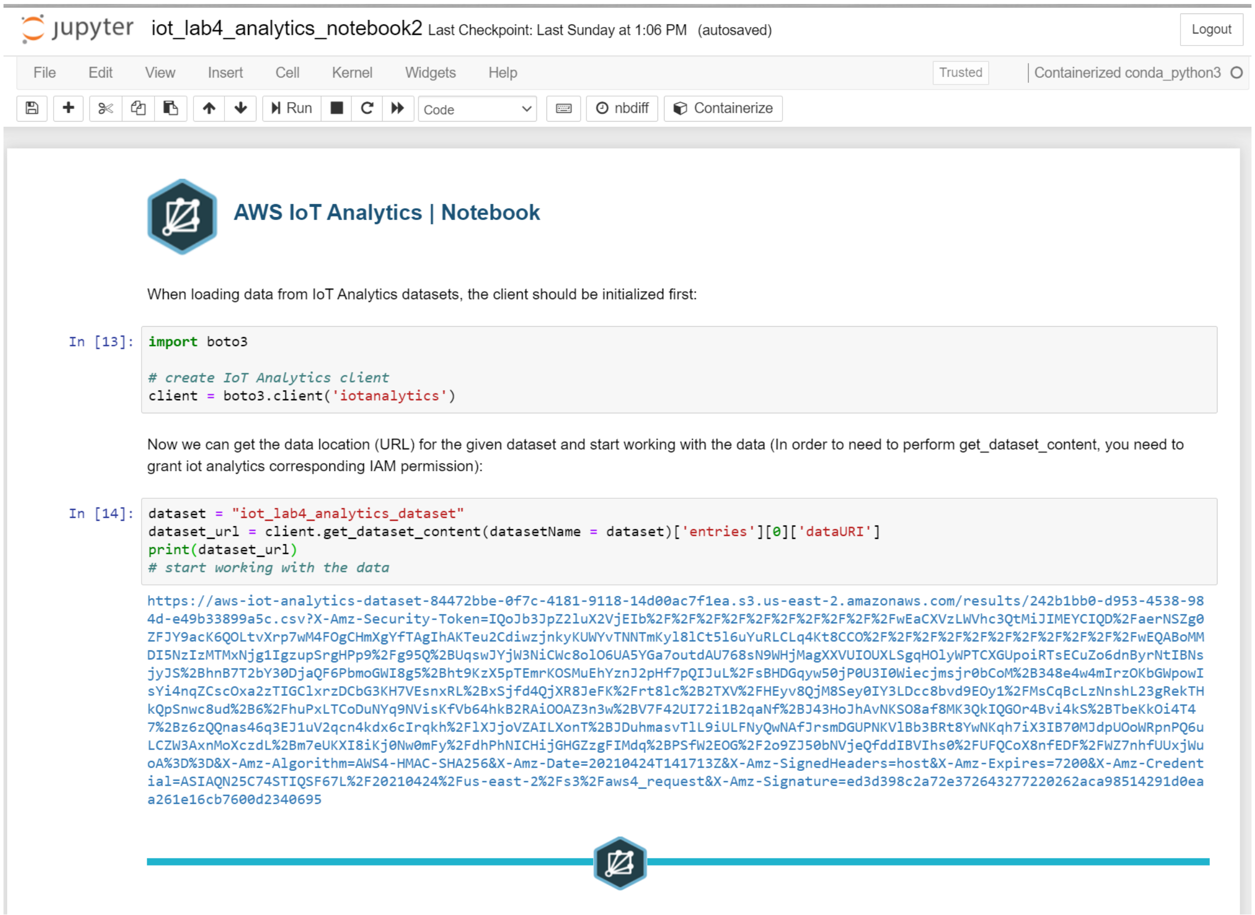
Delta window has not been set yet.

Result preview

thingid	temperature	humidity	datetime	timestep_time
				0.0

The default schema is pretty terrible because it is using all of the test data I sent through during development to create it.

I did not spend a ton of time creating nifty visualizations in Jupyter, but I did play around with the dataset and used some visualizations to help debug data flows. The print out is available in the zip-archive and github as “iot_lab4_analytics_notebook2.pdf”.



jupyter
iot_lab4_analytics_notebook2 Last Checkpoint: Last Sunday at 1:06 PM (autosaved)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Containerized conda_python3
Run
Code
nbdiff
Containerize

AWS IoT Analytics | Notebook

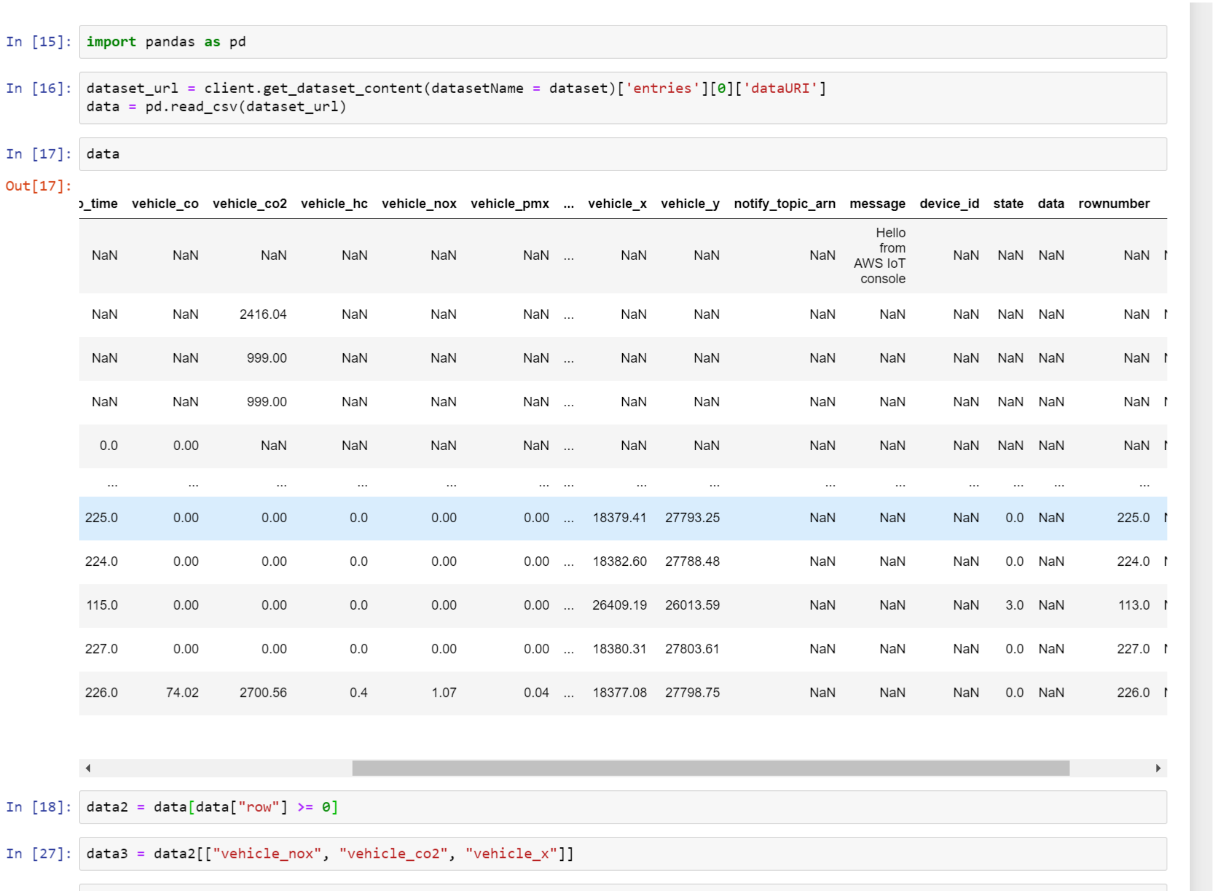
When loading data from IoT Analytics datasets, the client should be initialized first:

```
In [13]: import boto3

         # create IoT Analytics client
         client = boto3.client('iotanalytics')
```

Now we can get the data location (URL) for the given dataset and start working with the data (In order to need to perform get_dataset_content, you need to grant iot analytics corresponding IAM permission):

```
In [14]: dataset = "iot_lab4_analytics_dataset"
         dataset_url = client.get_dataset_content(datasetName = dataset)['entries'][0]['dataURI']
         print(dataset_url)
         # start working with the data
```

```
https://aws-iot-analytics-dataset-84472bbe-0f7c-4181-9118-14d00ac7f1ea.s3.us-east-2.amazonaws.com/results/242b1bb0-d953-4538-98
4d-e49b33899a5c.csv?X-Amz-Security-Token=IQoJb3JpZ2luX2VjEIb%2F%2F%2F%2F%2F%2F%2F%2F%2F%2FwEaCXVzLWVhc3QtMiJIMEYCIQD%2FaerNSZg0
ZFJY9acK6QOLtvXrp7wM4FOgCHmXgYfTAgIhAKTeu2CdiwzjnkyKUWYvTNNTmKyl8lCt5l6uYuRLCLq4Kt8CCO%2F%2F%2F%2F%2F%2F%2F%2F%2F%2F%2FwEQABoMM
DI5NzIzMTMxNjg1IgzupSrgHPp9%2Fg95Q%2BUqswJYjW3NiCWc8olO6UA5YGa7outdAU768sN9WHjMagXXVUIOUXLSgqHOlyWPTCXGUpoiRTsECuZo6dnByrNtIBNs
jyJS%2BhnB7T2bY30DjaQF6PbmoGWI8g5%2Bht9KzX5pTEmrKOSMuEhYznJ2pHf7pQIJuL%2FsBHDGqyw50jP0U3I0Wiecjmsjr0bCoM%2B348e4w4mIrzOKbGWpowI
sYi4nqZCscOxa2zTIGClxrzDCbG3KH7VEsnxRL%2BxSjfd4QjXR8JeFK%2Frt8lc%2B2TXV%2FHEyv8QjM8Sey0IY3LDcc8bvd9EOy1%2FMsCqBcLzNnshL23gRekTH
kQpSnwc8ud%2B6%2FhuPxLTCoDuNYq9NVisKfVb64hkB2RAiOOAZ3n3w%2BV7F42UI72i1B2qaNf%2BJ43HoJhAvNKSO8af8MK3QkIQGOr4Bvi4kS%2BTbeKkOi4T4
7%2Bz6zQQnas46q3EJ1uV2qcn4kdx6cIrqkh%2FlXJjoVZAILXonT%2BJDuhmasvTlL9iULFNyQwNAfJrsmDGUPNKVlBb3BRt8YwNKqh7iX3IB70MJdpUOoWRpnPQ6u
LCZW3AxnMoXczdL%2Bm7eUKXI8iKj0Nw0mFy%2FdhPhNICHijGHGZzgFIMdq%2BPSfW2EOG%2F2o9ZJ50bNVjeQfddIBVIhs0%2FUFQCoX8nfEDF%2FWZ7nhfUUxjWu
oA%3D%3D&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Date=20210424T141713Z&X-Amz-SignedHeaders=host&X-Amz-Expires=7200&X-Amz-Credent
ial=ASIAQN25C74STIQSF67L%2F20210424%2Fus-east-2%2Fs3%2Faws4_request&X-Amz-Signature=ed3d398c2a72e372643277220262aca98514291d0ea
a261e16cb7600d2340695
```



```
In [15]: import pandas as pd
```

```
In [16]: dataset_url = client.get_dataset_content(datasetName = dataset)['entries'][0]['dataURI']
         data = pd.read_csv(dataset_url)
```

```
In [17]: data
```

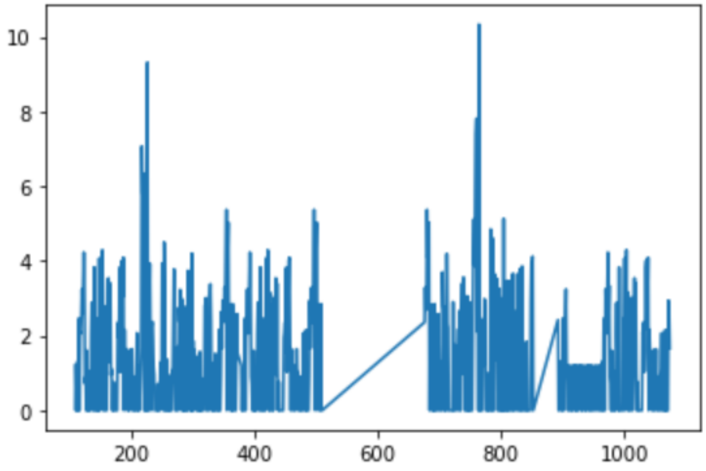
Out[17]:

_time	vehicle_co	vehicle_co2	vehicle_hc	vehicle_nox	vehicle_pmx	...	vehicle_x	vehicle_y	notify_topic_arn	message	device_id	state	data	rownumber
NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	Hello from AWS IoT console	NaN	NaN	NaN	NaN
NaN	NaN	2416.04	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
NaN	NaN	999.00	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
NaN	NaN	999.00	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
0.0	0.00	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
225.0	0.00	0.00	0.0	0.00	0.00	...	18379.41	27793.25	NaN	NaN	NaN	0.0	NaN	225.0
224.0	0.00	0.00	0.0	0.00	0.00	...	18382.60	27788.48	NaN	NaN	NaN	0.0	NaN	224.0
115.0	0.00	0.00	0.0	0.00	0.00	...	26409.19	26013.59	NaN	NaN	NaN	3.0	NaN	113.0
227.0	0.00	0.00	0.0	0.00	0.00	...	18380.31	27803.61	NaN	NaN	NaN	0.0	NaN	227.0
226.0	74.02	2700.56	0.4	1.07	0.04	...	18377.08	27798.75	NaN	NaN	NaN	0.0	NaN	226.0

```
In [18]: data2 = data[data["row"] >= 0]
```

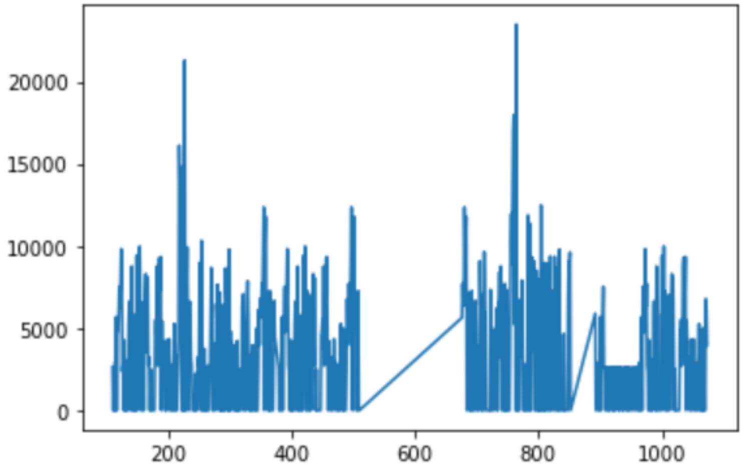
```
In [27]: data3 = data2[["vehicle_nox", "vehicle_co2", "vehicle_x"]]
In [28]: data3["vehicle_nox"].plot()
```

```
Out[28]: <AxesSubplot:>
```



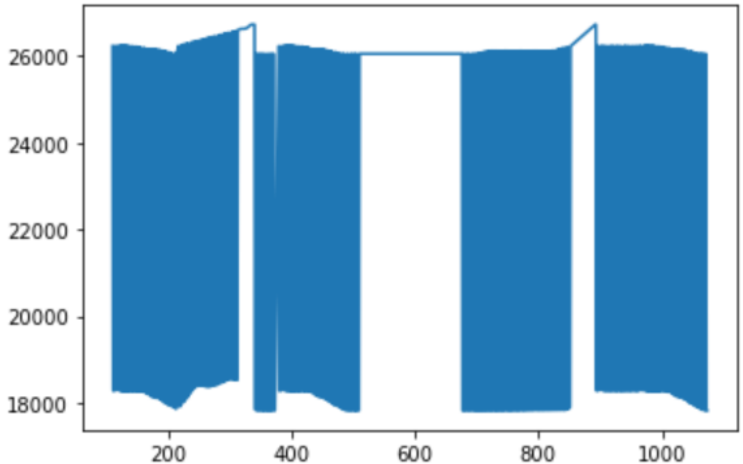
```
In [29]: data3["vehicle_co2"].plot()
```

```
Out[29]: <AxesSubplot:>
```

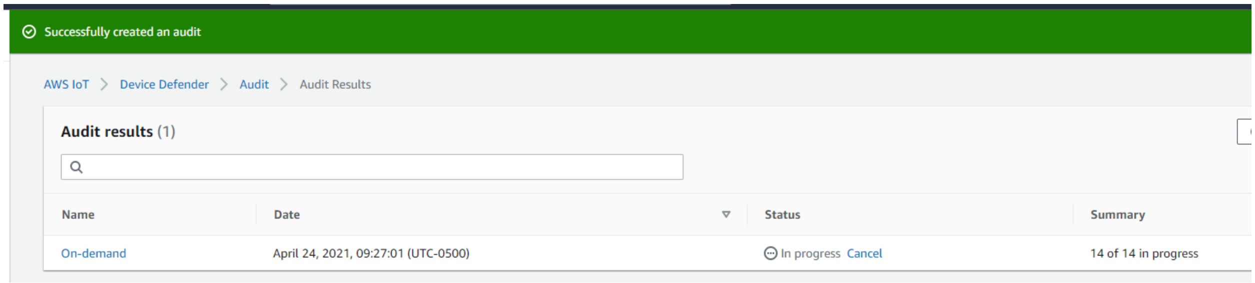


```
In [31]: data3["vehicle_x"].plot()
```

```
Out[31]: <AxesSubplot:>
```



Section 4 - AWS IoT Device Defender



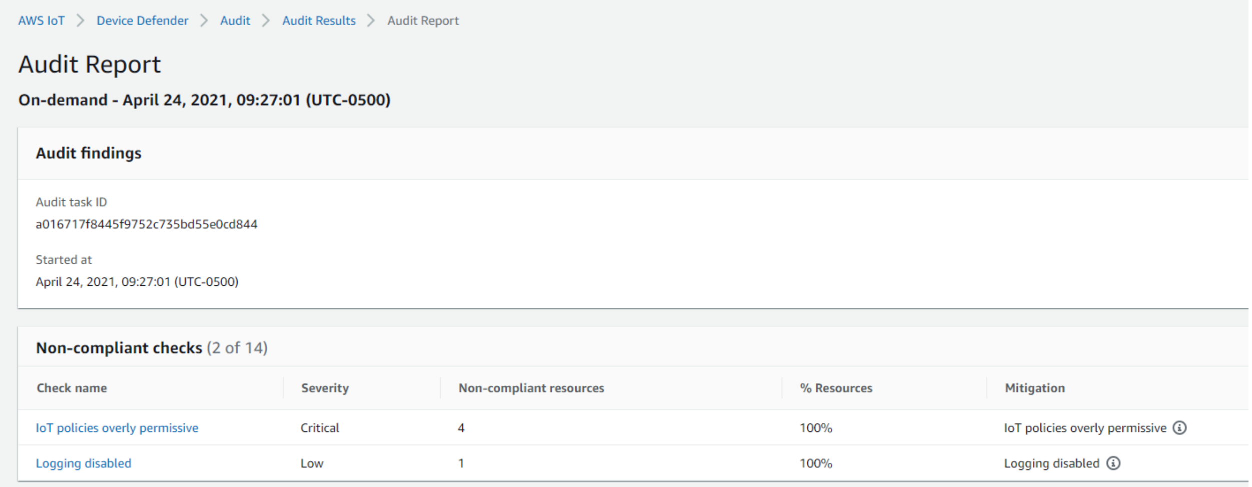
Successfully created an audit

AWS IoT > Device Defender > Audit > Audit Results

Audit results (1)

Name	Date	Status	Summary
On-demand	April 24, 2021, 09:27:01 (UTC-0500)	In progress Cancel	14 of 14 in progress

Uh-oh, I failed some audit items!



AWS IoT > Device Defender > Audit > Audit Results > Audit Report

Audit Report

On-demand - April 24, 2021, 09:27:01 (UTC-0500)

Audit findings

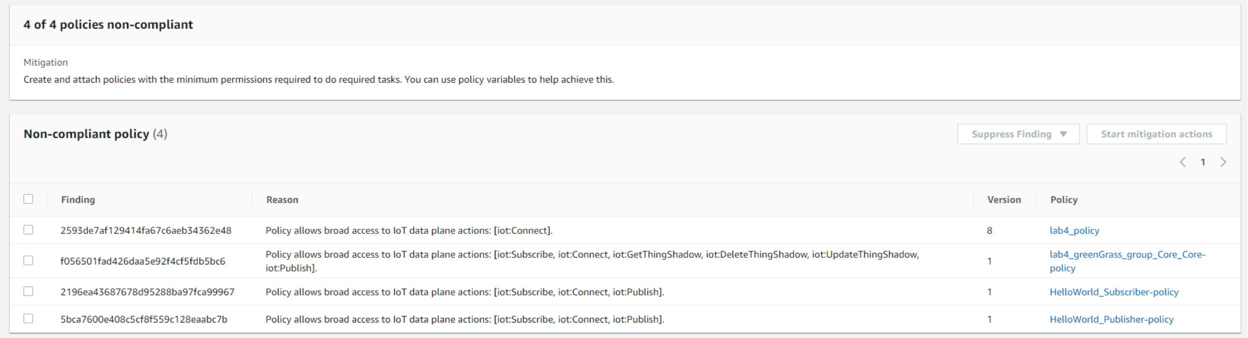
Audit task ID
a016717f8445f9752c735bd55e0cd844

Started at
April 24, 2021, 09:27:01 (UTC-0500)

Non-compliant checks (2 of 14)

Check name	Severity	Non-compliant resources	% Resources	Mitigation
IoT policies overly permissive	Critical	4	100%	IoT policies overly permissive ⓘ
Logging disabled	Low	1	100%	Logging disabled ⓘ

The failed audit on IoT policies looks interesting. Taking a deeper look:



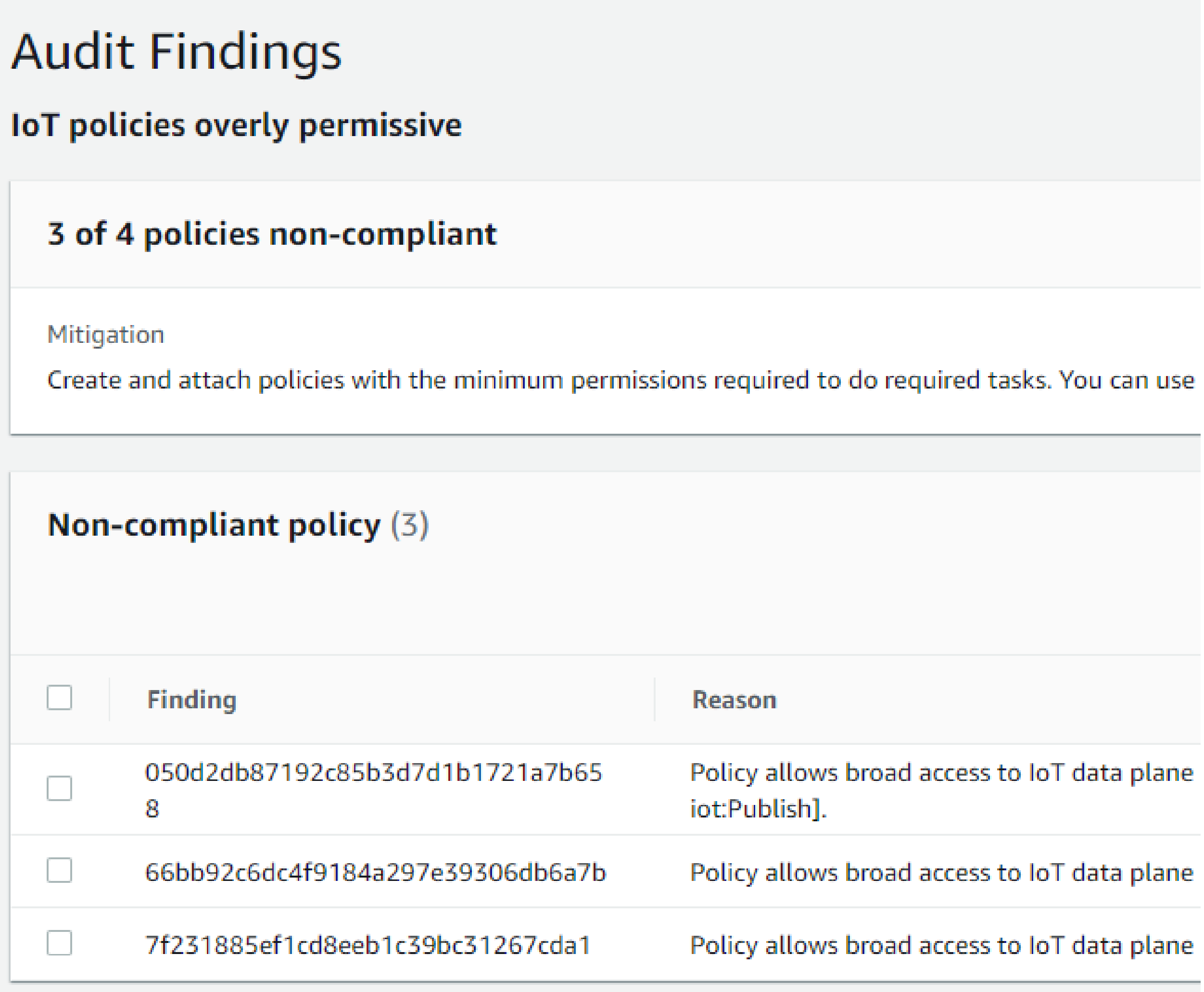
4 of 4 policies non-compliant

Mitigation
Create and attach policies with the minimum permissions required to do required tasks. You can use policy variables to help achieve this.

Non-compliant policy (4)

□	Finding	Reason	Version	Policy
□	2593de7af129414fa67c6aeb34362e48	Policy allows broad access to IoT data plane actions: [iot:Connect].	8	lab4_policy
□	f056501fad426daa5e92f4cf5fdb5bc6	Policy allows broad access to IoT data plane actions: [iot:Subscribe, iot:Connect, iot:GetThingShadow, iot:DeleteThingShadow, iot:UpdateThingShadow, iot:Publish].	1	lab4_greenGrass_group_Core_Core-policy
□	2196ea43687678d95288ba97fca99967	Policy allows broad access to IoT data plane actions: [iot:Subscribe, iot:Connect, iot:Publish].	1	HelloWorld_Subscriber-policy
□	5bca7600e408c5cf8f559c128eaabc7b	Policy allows broad access to IoT data plane actions: [iot:Subscribe, iot:Connect, iot:Publish].	1	HelloWorld_Publisher-policy

The last 3 were Amazon suggested policies, so… but the first line item is the one I made for the earlier part of the lab. Better fix that. I modified the policy to allow ONLY my Things, and then restricted it to the “test” topic I used throughout. Wa-la, that passes the audit!



Audit Findings

IoT policies overly permissive

3 of 4 policies non-compliant

Mitigation

Create and attach policies with the minimum permissions required to do required tasks. You can use

Non-compliant policy (3)

□	Finding	Reason
□	050d2db87192c85b3d7d1b1721a7b658	Policy allows broad access to IoT data plane iot:Publish].
□	66bb92c6dc4f9184a297e39306db6a7b	Policy allows broad access to IoT data plane
□	7f231885ef1cd8eeb1c39bc31267cda1	Policy allows broad access to IoT data plane

Appendix

This is a list of the things that went wrong configuring Part 2!

Part 2

Difficulties So far:

1. Github code was python 2 not python3
2. Configuration difficulties….what was my region!?
3. Can’t figure out how to run the emulator…. What is my endpoint
4. Ok I found the endpoint but it’s timing out what the heck
5. Turns out my policies are wrong….

6. OK Got policy fixed…. But I noticed there doesn't seem to be a policy attached to my things, which is weird… I added everything to a new thing group and attached a policy to that. Let's see if that worked?
7. Noooooooooope. My policy wasn't correct, still being blocked.. I know this because I found a "non-policy" policy that made it work! Ugh.
8. Now I've discovered that the emulator code is not python3 compat either..fixing….ack that wasn't it. I actually have to fill in the publish call with something useful.
9. Ok that's all set. Things are connecting and the code works, but not seeing any messages coming through on AWS console
10. Alright! Success. This worked by changing the policy to all topics, not just "test". But now I have no clarity why "test" didn't work…
11. Spent a few hours playing around with pub/sub
12. Creating an EC2 instance to test Green Grass…..how do I connect!? An hour later….
13. Greengrass installed and running on EC2!
14. OK many hours passed….Finally got greengrass to work I think. The daemon was publishing its local network address, as well as the amazon private IP address, and so on. I had to manually go change it in the IoT console to the public IP and it worked.
15. Trying to understand the "Deploy_service" lambda instructions. All we're provided with is a scikit learn model with no model specifications….
16. Skipping scikit learn lambda problem until TA gets back to me...so working on AWS IoT analytics instead. Had to go through instructions twice to get it to work. Now I am in a Sagemaker notebook with a giant presigned URL
17. OK Got sagemaker to work now back to data. I spent a few hours writing tensorflow scoring code to code my model from part #1...only to find out that you can't do that with amazon lambdas, but instead you have to use ECR or another service!
18. Someone on slack posted that we dont need to score a model at all (#15).... Ugh! Rewrote lambda just to aggregate max co2.
19. Have a new lambda written, but getting forbidden errors. Have to figure out how to allow lambda to publish to MQTT