

SIGNALZ V6

GETTING STARTED WITH
SIGNALZ.

NOVEMBER 2019.





TABLE OF CONTENTS

Introduction.

Getting started with SignalZ in 3 steps:

1. Connect your sensors to the SignalZ platform.
2. Analyze raw sensor data using the SignalZ platform.
3. Create algorithms, reports and dashboards.



INTRODUCTION

What is SignalZ?

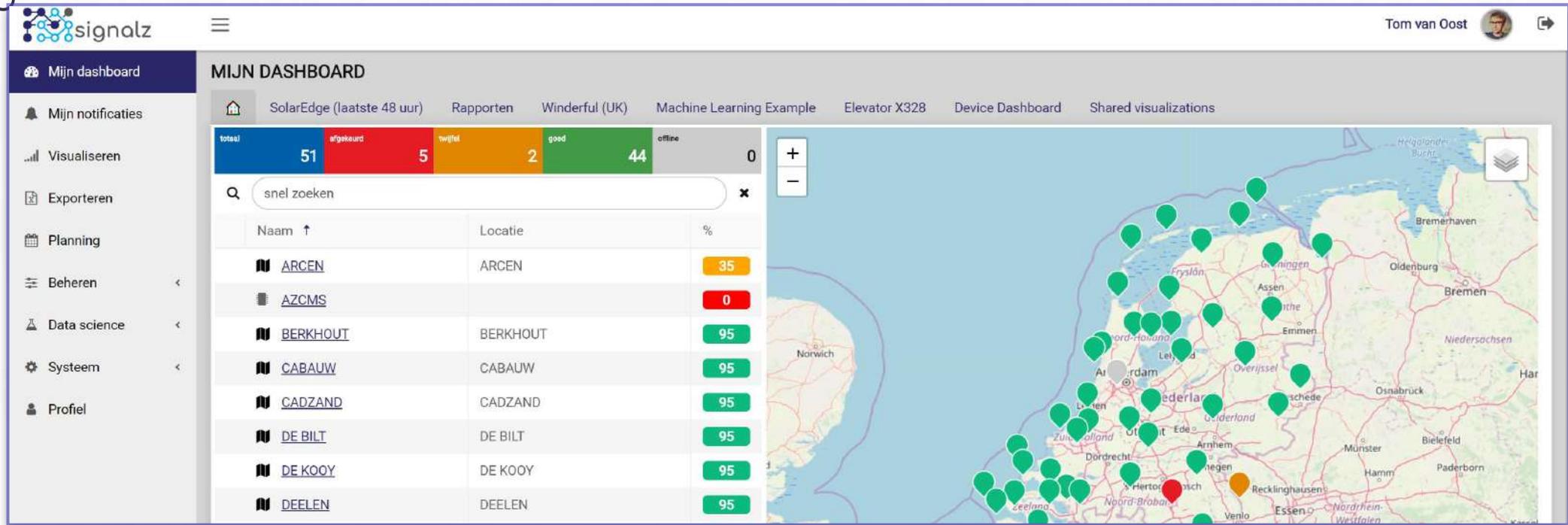
SignalZ is a scalable Internet-of-Things platform with an open architecture*.

It runs on-premise and in the cloud. (Azure)

What can I do with the SignalZ IoT platform?

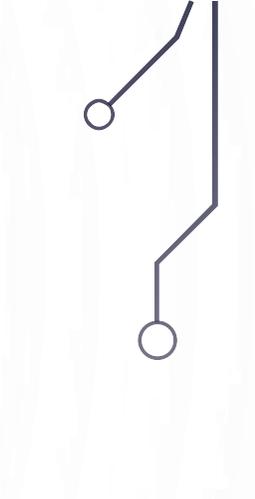
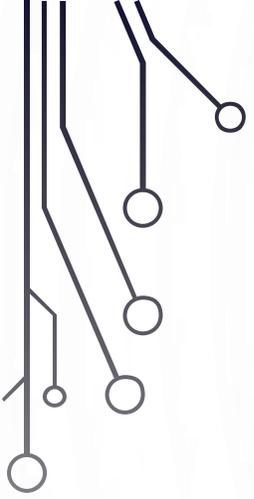
- Connect sensors and devices to SignalZ and start collecting data.
- Create algorithms using R and/or Python to analyze incoming data. (AI, machine learning)
- Build custom dashboards and reports to visualize your sensor/machine data.

*OData, RESTful API, Power BI integration.



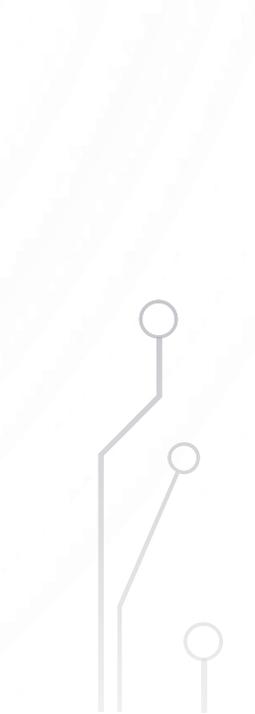
STEP 1: CONNECTING YOUR SENSORS





STEP 1: CONNECTING YOUR SENSORS

2 options:

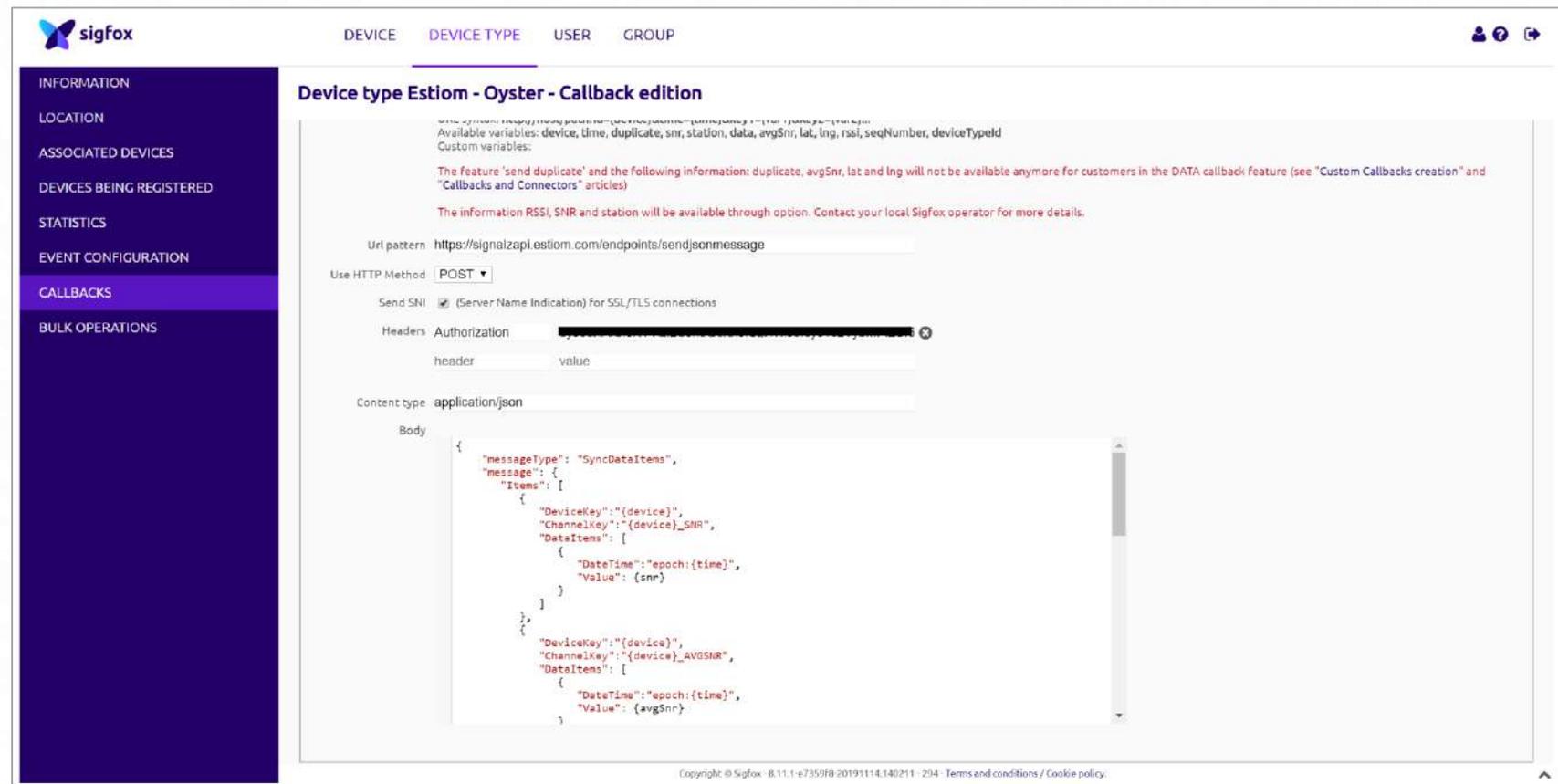
- **Push** sensordata directly to our secure RESTful API endpoint:
 - **Pull** sensordata from your devices/sensors using standard SignalZ IoT connectors or custom Python jobs.
- 
- 

STEP 1: CONNECTING YOUR SENSORS

Push example using Sigfox:

Configure the SignalZ API endpoint as a callback in Sigfox.

The connection is secured by SSL and you need a valid API token.



The screenshot displays the Sigfox web interface for configuring a device type callback. The left sidebar contains navigation options: INFORMATION, LOCATION, ASSOCIATED DEVICES, DEVICES BEING REGISTERED, STATISTICS, EVENT CONFIGURATION, CALLBACKS (highlighted), and BULK OPERATIONS. The main content area is titled "Device type Estiom - Oyster - Callback edition" and includes the following details:

- Available variables:** device, time, duplicate, snr, station, data, avgSnr, lat, lng, rssi, seqNumber, deviceTypeId
- Custom variables:** The feature 'send duplicate' and the following information: duplicate, avgSnr, lat and lng will not be available anymore for customers in the DATA callback feature (see "Custom Callbacks creation" and "Callbacks and Connectors" articles)
- Information:** The information RSSI, SNR and station will be available through option. Contact your local Sigfox operator for more details.
- Url pattern:** `https://signalzapi.estiom.com/endpoints/sendjsonmessage`
- Use HTTP Method:** POST
- Send SNI:** (Server Name Indication) For SSL/TLS connections
- Headers:** Authorization header with a redacted value.
- Content type:** application/json
- Body:** A JSON payload with the following structure:

```
{
  "messageType": "SyncDataItems",
  "message": {
    "Items": [
      {
        "DeviceKey": "{device}",
        "ChannelKey": "{device}_SNR",
        "DataItems": [
          {
            "DateTime": "epoch:{time}",
            "Value": {snr}
          }
        ]
      },
      {
        "DeviceKey": "{device}",
        "ChannelKey": "{device}_AVGSNR",
        "DataItems": [
          {
            "DateTime": "epoch:{time}",
            "Value": {avgSnr}
          }
        ]
      }
    ]
  }
}
```

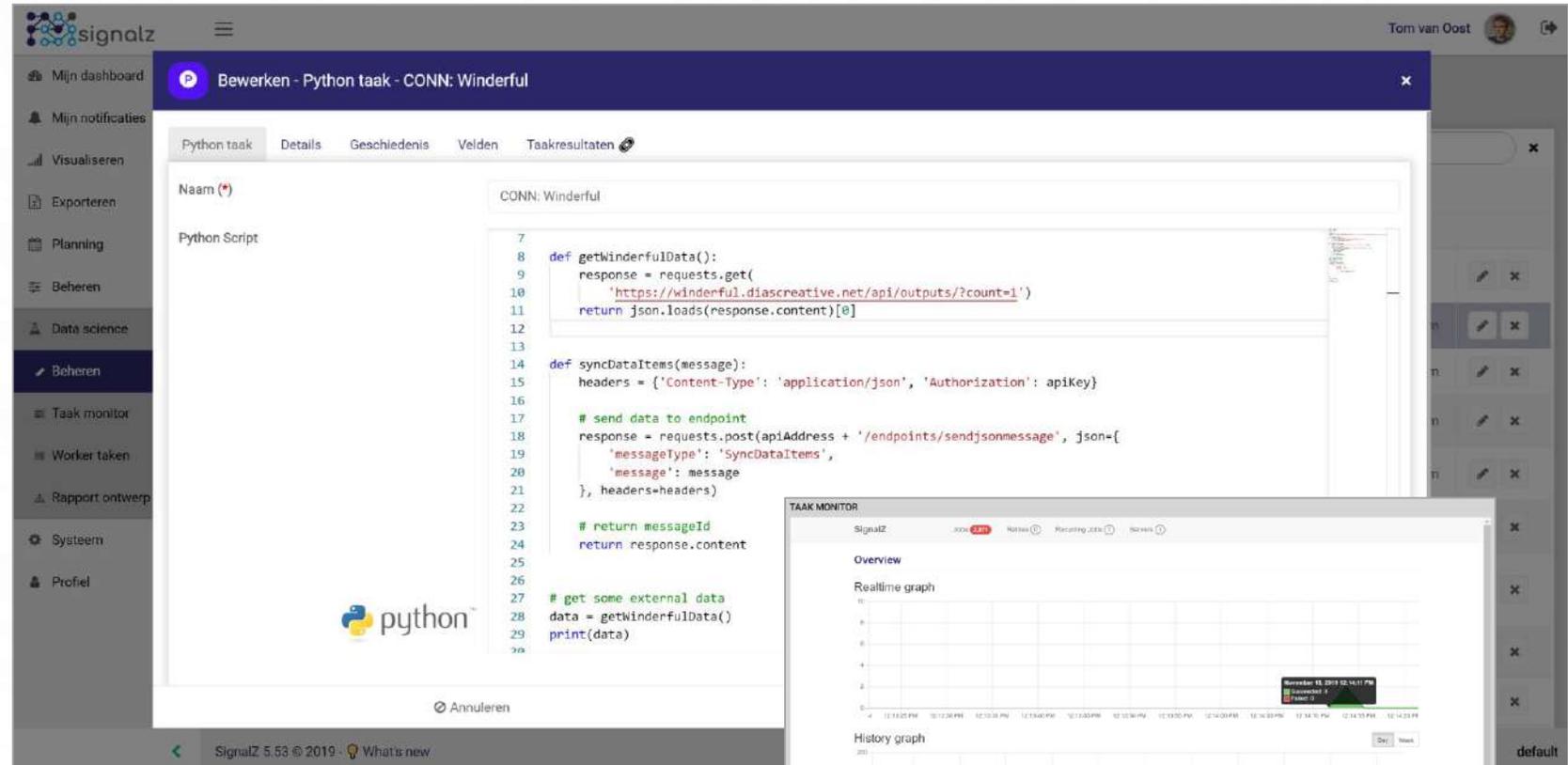
Copyright © Sigfox - 8.11.1-e7359f8 20191114.140211 - 294 - Terms and conditions / Cookie policy.

STEP 1: CONNECTING YOUR SENSORS

Pull example using a custom Python job:

Write your own Python job to pull data from your sensors, directly in SignalZ.

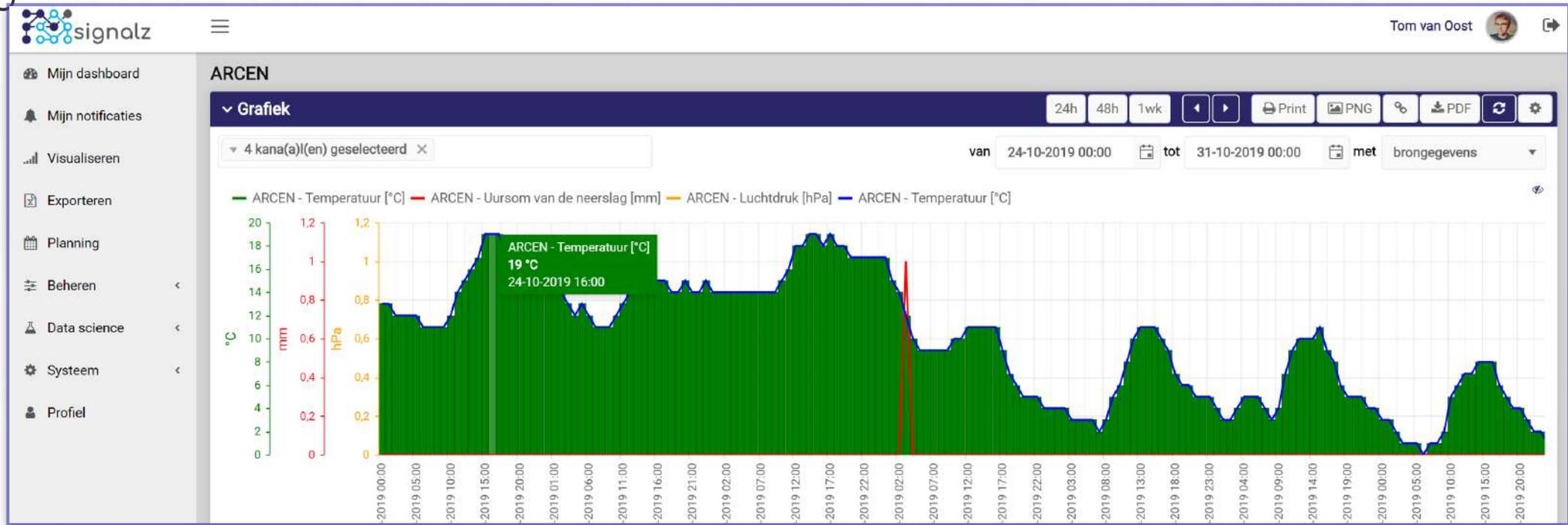
Schedule your job to run on the SignalZ job server.



The screenshot displays the SignalZ web interface. The main window is titled "Bewerken - Python taak - CONN: Winderful". The left sidebar contains navigation options: "Mijn dashboard", "Mijn notificaties", "Visualiseren", "Exporteren", "Planning", "Beheren", "Data science", "Beheren", "Taak monitor", "Worker taken", "Rapport ontwerp", "Systeem", and "Profiel". The main content area shows the configuration for a Python task. The "Naam (*)" field is set to "CONN: Winderful". The "Python Script" field contains the following code:

```
7
8 def getWinderfulData():
9     response = requests.get(
10         'https://winderful.diascreative.net/api/outputs/?count=1')
11     return json.loads(response.content)[0]
12
13
14 def syncDataItems(message):
15     headers = {'Content-Type': 'application/json', 'Authorization': apiKey}
16
17     # send data to endpoint
18     response = requests.post(apiAddress + '/endpoints/sendjsonmessage', json={
19         'messageType': 'SyncDataItems',
20         'message': message
21     }, headers=headers)
22
23     # return messageId
24     return response.content
25
26
27 # get some external data
28 data = getWinderfulData()
29 print(data)
30
```

Below the code editor is a "python" logo and an "Annuleren" button. A "TAASKONTROL" window is open in the bottom right corner, showing a "Realltime graph" and a "History graph". The "Realltime graph" is currently empty. The "History graph" shows a green area representing data over time, with a timestamp of "vrijdag, 13 oktober 2017 12:14:17 PM". The bottom status bar indicates "SignalZ 5.53 © 2019 - What's new".



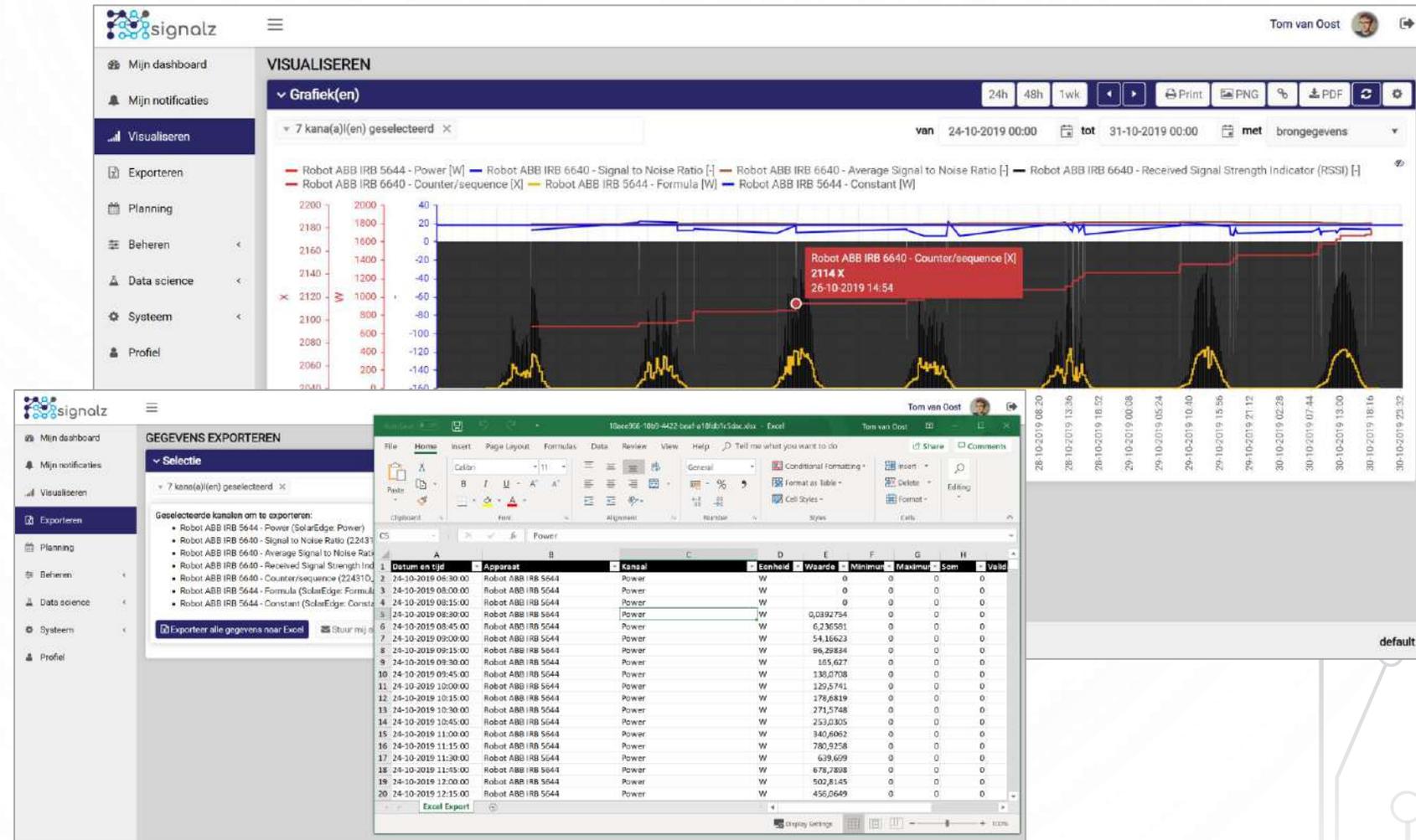
STEP 2: ANALYZE RAW SENSOR DATA USING THE SIGNALZ PLATFORM



STEP 2: ANALYZE RAW SENSOR DATA USING THE SIGNALZ PLATFORM

Your sensordata is now available in SignalZ!

Analyse the raw data using the SignalZ webinterface or export data to Microsoft Excel or Power BI.



STEP 2: ANALYZE RAW SENSOR DATA USING THE SIGNALZ PLATFORM.

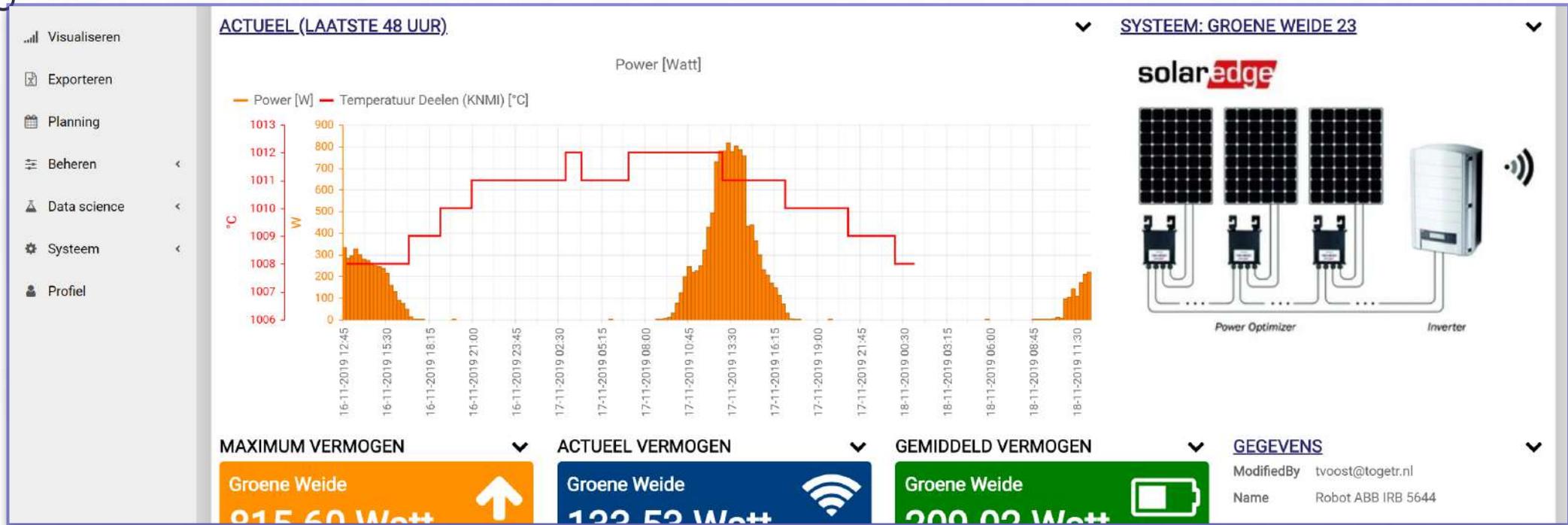
For more information watch the SignalZ BASICS demo video:

The image displays the SignalZ platform interface, which is used for monitoring and analyzing industrial equipment. The main dashboard is titled "ROBOT ABB IRB 6640" and features a sidebar with navigation options such as "Mijn dashboard", "Mijn notificaties", "Visualiseren", "Exporteren", "Planning", "Beheren", "Data science", "Systeem", and "Profiel". The main content area is divided into sections for "Grafiek", "Apparaatdetails", "Logboek", and "Model".

The "Logboek" section shows a table of events with columns for "Start datum/tijd", "Status", and "Bestanden". The table contains several entries, including one with the status "Open" and another with "Geen actie".

A modal window titled "Bewerken - Formule notificatie - SolarEdge Power" is open, allowing users to configure notification formulas. The "Formule" field contains the following code: `dataItems.GetLastValue("SE_290638", "SolarEdge: Power") > 0`. Other fields include "Naam" (SolarEdge Power), "Indien de waarde ... aantal keer gemeten wordt" (1), "Verstuur den elk(e) ... uur een notificatie" (24), and "Soort notificatie" (Email).

The "Model" section displays a 3D CAD model of the ABB IRB 6640 robot arm, showing its various components in different colors (blue, orange, grey). The interface also includes a footer with the text "SignalZ 5.53 © 2019 - Whats new" and "© Anvulieren" and "© Opslaan".



STEP 3: CREATE ALGORITHMS, REPORTS AND DASHBOARDS.



STEP 3: CREATE ALGORITHMS, REPORTS AND DASHBOARDS.

You can create custom charts, KPI's, reports and many other visualizations using the build-in dashboard editor.

DEVICE STATUS	SUMMARY
Error 2	Total trip counter 1.354.552
Warning 2	Average landing call waiting time 2.1 Min.
OK 44	Average Load 131.2 kg

Bewerken - Visualisatie - Actueel (laatste 48 uur)

ACTUEEL (LAATSTE 48 UUR)

Power [Watt]

Power [W] — Temperatuur Deelen (KNMI) [°C]

Bewerken - Visualisatie - Maximum vermogen

KPI visualisatie Details Geschiedenis Velden Preview

Naam (*) Maximum vermogen

Databron SolarEdge Power (laatste 48 uur)

Visualisatie grootte

KPI tekst Groene Weide

Pictogram ↑ fa-arrow-up

Kleur

Veld (waarde) Value (OData)

KPI aggregatie formule MAX()

KPI eenheid Watt

KPI aantal decimalen 2

Annuleren Opslaan



STEP 3: CREATE ALGORITHMS, REPORTS AND DASHBOARDS.

Create custom Python or R algorithms to apply Machine Learning and AI.

The image displays a dashboard and two task editors. The dashboard, titled "MIJN DASHBOARD", shows a "Machine Learning Example" tab with a table of training data and a list of Python versions. The training data table is as follows:

	Rain	Temp	Jacket
0	0.0	-10	a Winter Jacket
1	0.0	-5	a Winter Jacket
2	0.0	0	a Winter Jacket
3	0.0	5	a Spring Jacket
4	0.0	10	a Spring Jacket
5	0.0	15	a Summer Jacket
6	0.0	20	a Summer Jacket
7	0.0	25	no jacket
8	0.0	30	no jacket
9	0.5	10	a Raincoat
10	1.0	25	a Raincoat
11	0.3	-10	a Raincoat
12	0.1	5	a Raincoat

The Python versions list includes Python 3.5.4, scipy 1.2.1, numpy 1.16.3, and matplotlib 3.0.3. Below the dashboard, two task editors are shown. The first, "Aanmaken - R Script taak", displays an R script with the following code:

```
1 print(sample(1:3))
2 print(sample(1:3, size=3, replace=FALSE)) # same as previo
3 print(sample(c(2,5,3), size=4, replace=TRUE))
4 print(sample(1:2, size=10, prob=c(1,3), replace=TRUE))
5
6 barplot(table(sample(1:3, size=1000, replace=TRUE, prob=c(.
7
8
```

The second task editor, "Bewerken - Python taak - PYTHON: Machine Learning (Weather)", displays a Python script with the following code:

```
56
57 # Train ML algorithm using fixed dataset
58 array = dataset.values
59 trainInput = array[:,0:2]
60 trainOutput = array[:,2]
61
62 ml = LinearDiscriminantAnalysis()
63 ml.fit(trainInput, trainOutput)
64
65 # Predict based on live data
66 outcome = ''
67
68 hours = 168 * 2 # weeks
69 tempData = getWeatherData('275', 'KNMI: k1', hours)
70 rainData = getWeatherData('275', 'KNMI: k2', hours)
71 i = 0
72
73 for tempItem in tempData:
74     rainItem = rainData[i]
75     temp = tempItem['Value']
76     rain = rainItem['Value']
77     dateTime = (dateutil.parser.parse(tempItem['DateTime'])).strftime('%d-%m-%Y')
78     jacket = ml.predict([[rain, temp]])[0]
```

STEP 3: CREATE ALGORITHMS, REPORTS AND DASHBOARDS.

For more information watch the SignalZ DATA SCIENCE demo video:

The image displays the SignalZ software interface for a ROBOT ABB IRB 6640. The interface is divided into several sections:

- Dashboard:** Shows a sidebar with navigation options like 'Mijn dashboard', 'Mijn notificaties', 'Visualiseren', 'Exporteren', 'Planning', 'Beheren', 'Data science', 'Systeem', and 'Profiel'. The main content area includes a 'Grafiek' (Graph) section, 'Apparaatdetails' (Device details), and a 'Logboek' (Logbook).
- Logboek Table:** A table listing events with columns for 'Start datum/tijd', 'Status', and 'Bestanden'.

Start datum/tijd	Status	Bestanden
29-10-2019 10:07	Geen actie	landinq-door.jpg (56,97 KB), start.jpg (48,41 KB)
24-8-2019 08:38	Open	ABB-Robotic-product-range-brochure-4pages-2019-9AKK107.pdf (1,1 MB), Motion_Controller_Axis_Config4.jpg (83,26 KB), #b120 cod m
6-8-2019 10:37	Geen actie	ABB IRB 6640 in factory.jpg (523,32 KB), ABB IRB 6640.jpg (8...
- Bewerken - Formule notificatie - SolarEdge Power:** A configuration window for a formula notification. It includes fields for 'Naam (*)', 'Indien de waarde ... aantal keer gemeten wordt', 'Verstuur den elk(e) ... uur een notificatie', 'Soort notificatie', and 'Formule'. The formula field contains: `dataItems.GetLastValue("SE_2906198", "SolarEdge: Power") > 0`.
- 3D Model:** A 3D CAD model of the ABB IRB 6640 robot arm, shown in a semi-transparent blue view, with a control panel at the bottom.

SIGNALZ V6

WWW.SIGNALZ.EU

