Chapter 8:   Edge Impulse

After completing this chapter, you will understand how to use the Edge Impulse machine learning solution to develop and train ML models for use with Infineon MCUs. You will work with solutions using inputs from motion and audio sensors.

Table of contents

Document conventions

Convention	Usage	Example
Courier New	Displays code and text commands	`CY_ISR_PROTO(MyISR);` `make build`
Italics	Displays file names and paths	*sourcefile.hex*
[**bracketed, bold**]	Displays keyboard commands in procedures	[**Enter**] or [**Ctrl**] [**C**]
Menu > Selection	Represents menu paths	**File > New Project > Clone**
Bold	Displays GUI commands, menu paths and selections, and icon names in procedures	Click the **Debugger** icon, and then click **Next**.

8.1   Overview

In this chapter we will be using one of Infineon's Machine Learning partners - Edge Impulse - to help facilitate and automate the steps of capturing raw data from a sensor, labeling the data, generating and training a machine learning model, generating embedded code optimized for a PSoC™ 6 MCU and validating the model on the target hardware.
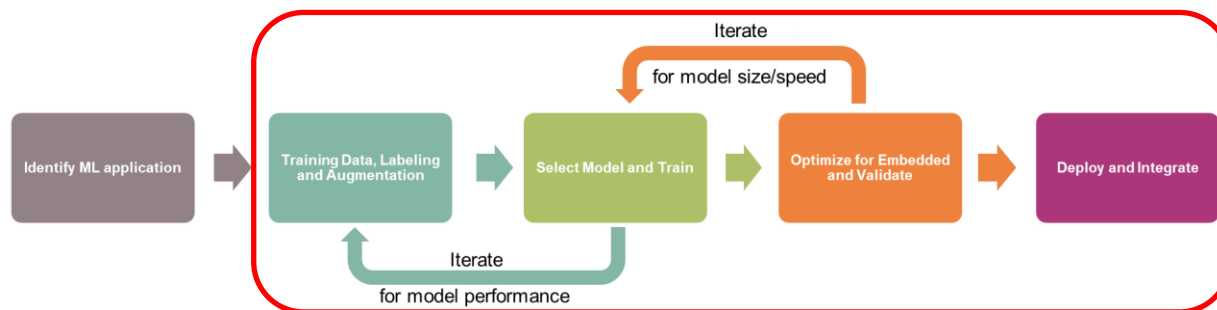
The main Edge Impulse tool is called Edge Impulse Studio. There is also a Linux SDK and a command line interface. We will focus on using Edge Impulse Studio solution in this training, but documentation is available for the other methods on the Edge Impulse website.
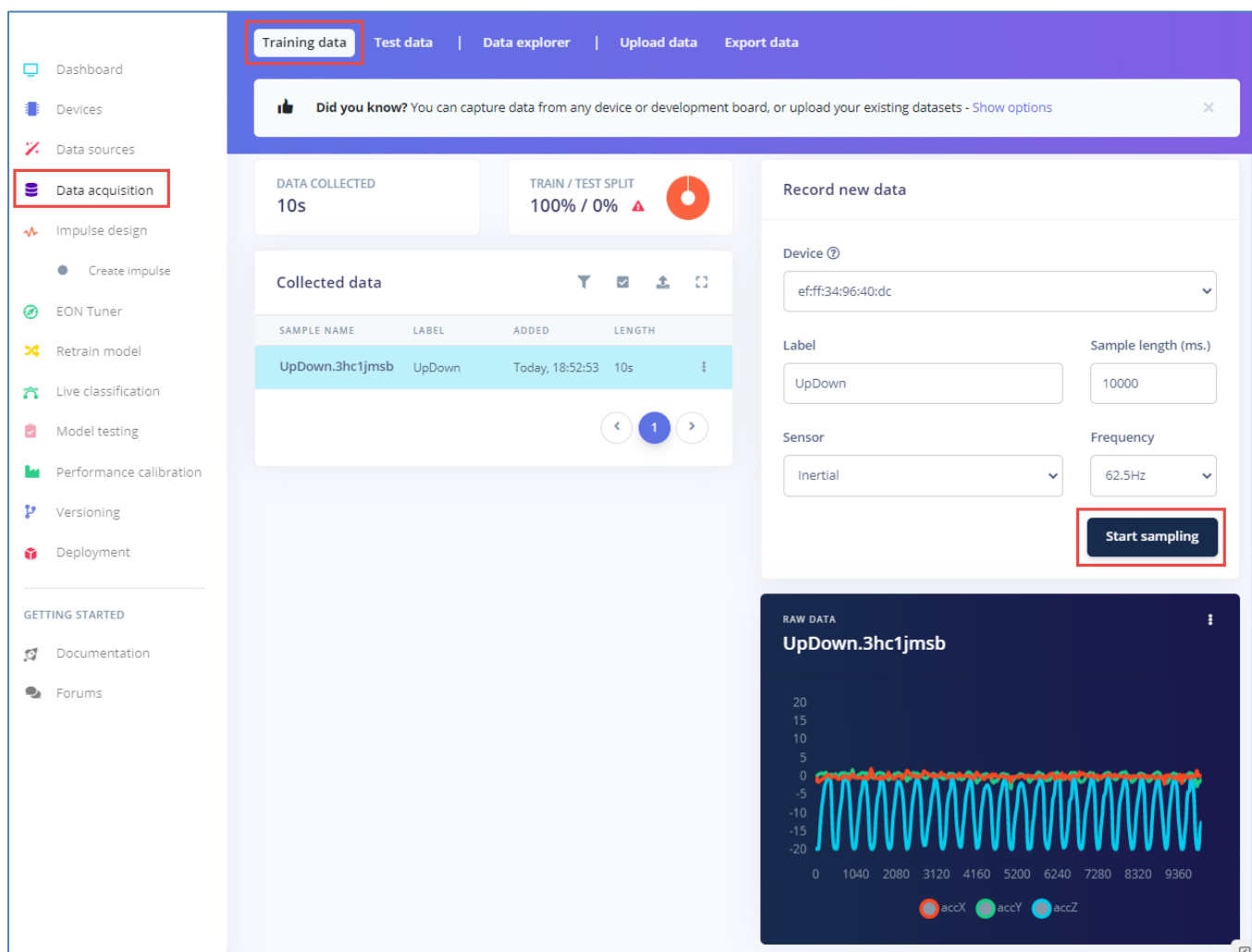
8.2   Workflow

As with all machine learning flows, the main steps are: (1) capture data; (2) build a model; (3) analyze the model; and (4) test the model on the target device. Each step is discussed briefly below.

8.2.1 Capture and label data

Data can be captured in a variety of ways. In some cases, you will find pre-defined datasets that can be used, or you may have existing data that you collected. In that scenario, you can import the data. If you need to capture new data, you can use resources on your computer (such as the microphone) or even on your cellphone as an input. However, the best way to collect new data is by using the same sensor that your final application will use. That's the method we will use by connecting the Infineon kit to Edge Impulse with firmware designed to stream the sensor data using a UART.

In Edge Impulse, each file will contain data for one classification. You also need to separate the training data and test data into different sets of files. Note that the training data also contains validation data, but that's done internally by the tool, meaning that you will set the percentage of the dataset used for validation in the settings for the training step.
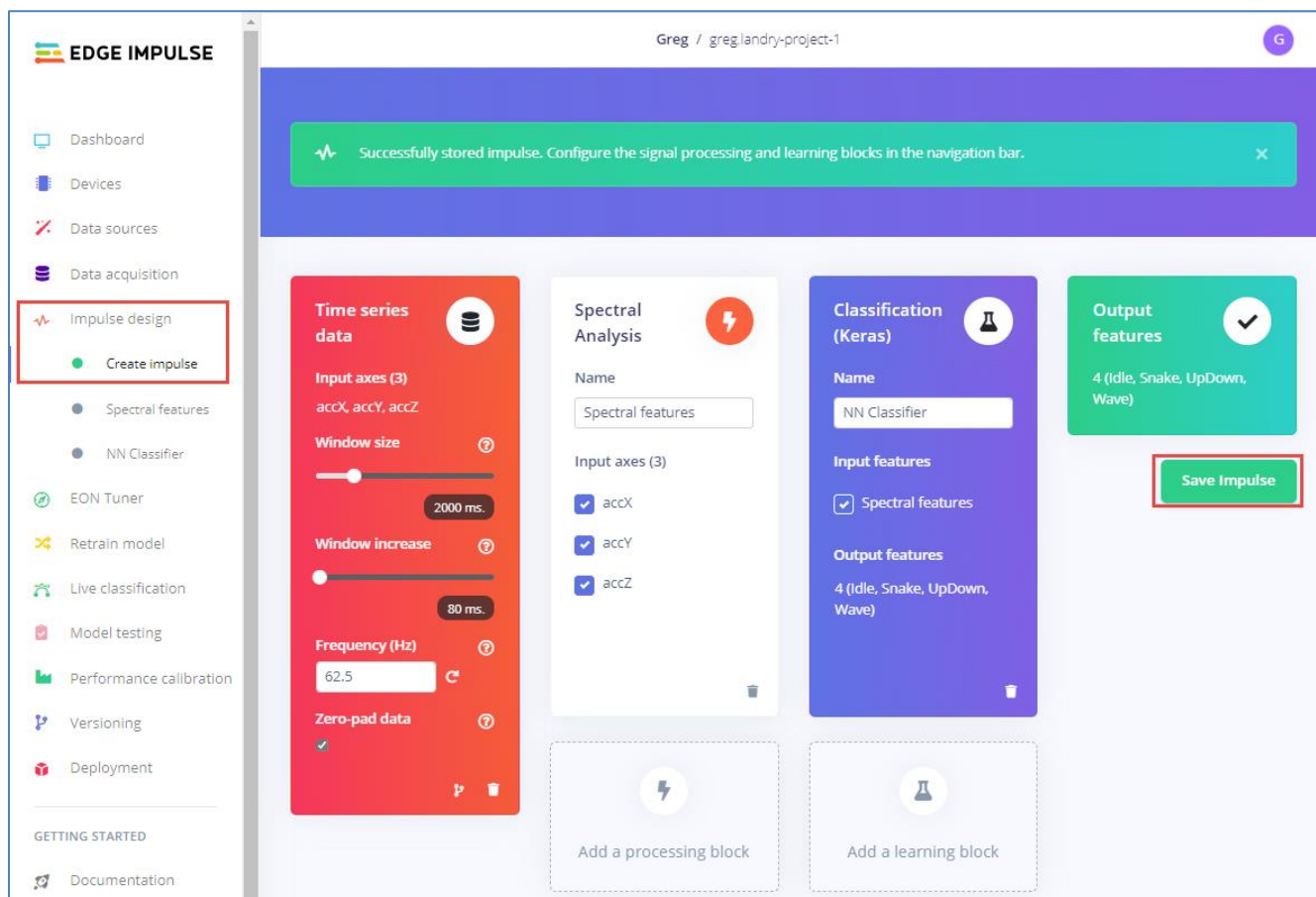
8.2.2 Build a model

Once you have data, you create an "impulse." This is the term that Edge Impulse uses for the instructions given to create and train the model. The impulse is used by selecting various processing blocks (such as filters) and learning blocks based on what the model will need to do. At this point, some knowledge of the dataset and machine learning algorithms is very useful in selecting the right options. It's a good idea to look at examples that solve a similar problem (e.g. continuous motion detection examples or audio recognition examples) to see the types of blocks that are likely to be relevant for your model.

In the example below, the impulse has an input block with two second overlapping windows that are each shifted 80ms from the previous window. For continuous data this impulse block will give you multiple samples for the model to work with (remember Augmentation from chapter 2?). After that there is a spectral analysis block that does low pass filtering to remove jitter in the motion sensor data. Next is the neural network – that's the actual machine learning piece. Finally, an output block is used to provide the final results. You will adjust parameters in each block and you can add additional processing blocks and learning blocks as needed.

Once you save the impulse, you will see selections to adjust the parameters for the Spectral Analysis and NN Classifier. The impulse is set up in stages. That is, first you set up the processing block(s) and adjust their parameters to get maximum separation in the data categories and then feed that into the learning block(s).

On the Spectral Features page, you should select which data file you want to view and then you can slide the window across the raw data to examine the results of different filter settings. It's useful to look at each type of label (each class) with different settings. The goal is to choose settings that maximize the differences between the classes. By adjusting the processing block parameters, you can minimize the amount of work that the learning block has to do. This will ultimately result in smaller, faster, and more accurate models.
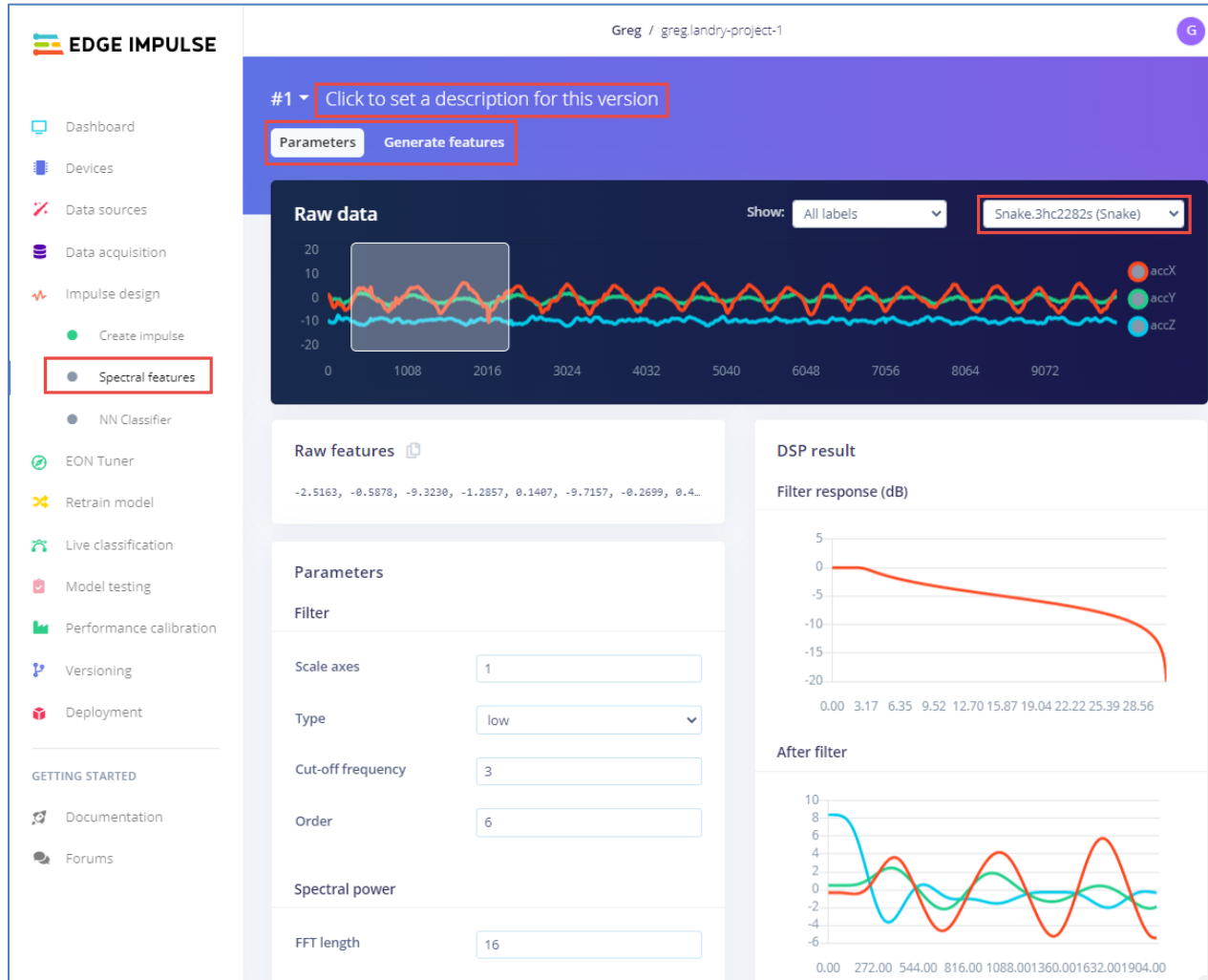
When you save the parameters, it will take you to the "Generate features" tab. Click the **Generate features** button to generate the features from your filter settings and then use the feature explorer to see how well the classes are separated. You can zoom in and out in the feature explorer (mouse wheel) and move the chart around (left-click and drag). If you click on a sample, it will show you the where it was previously located in the dataset.
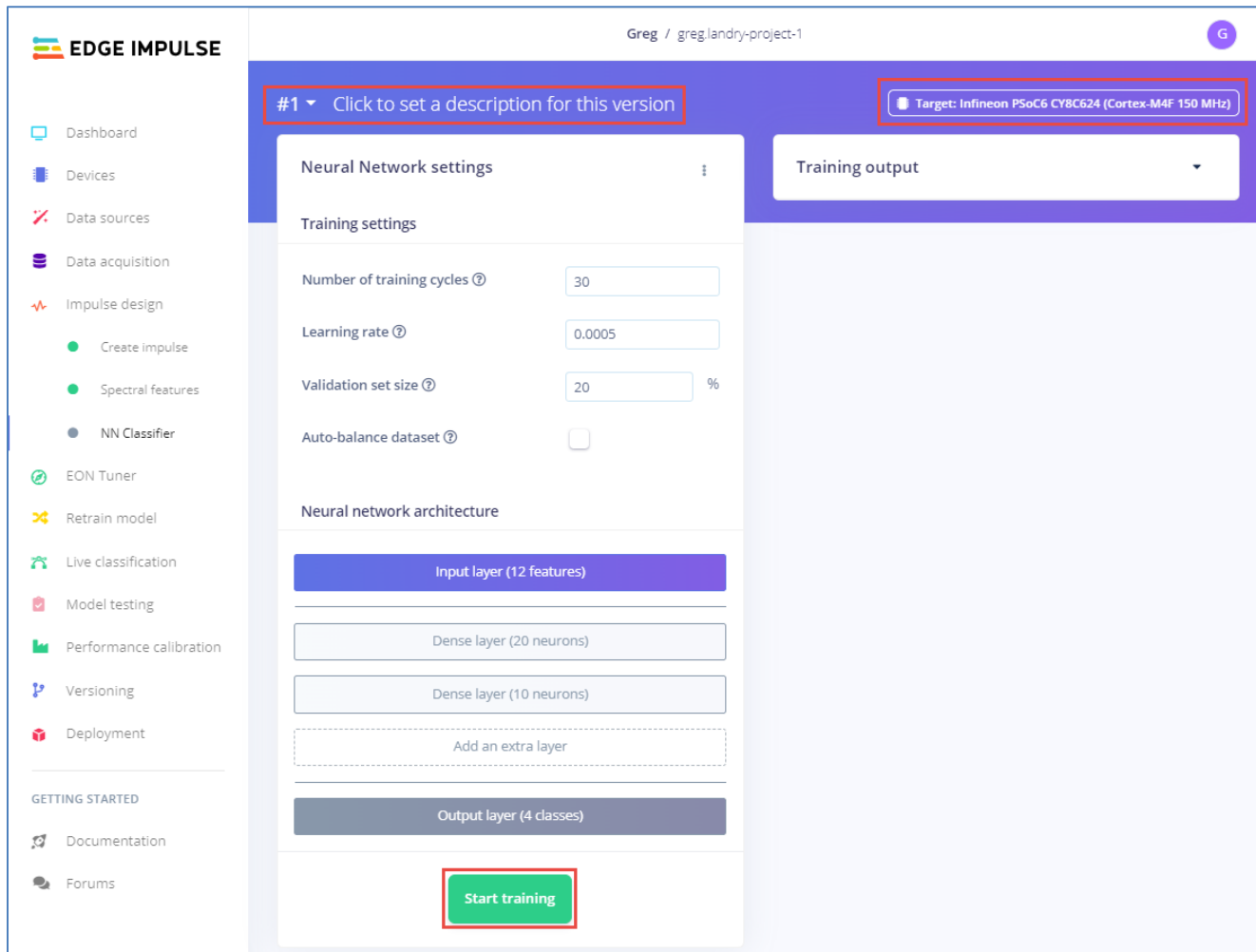
At the bottom of the window you can see how much time and memory the filtering will require on the device.
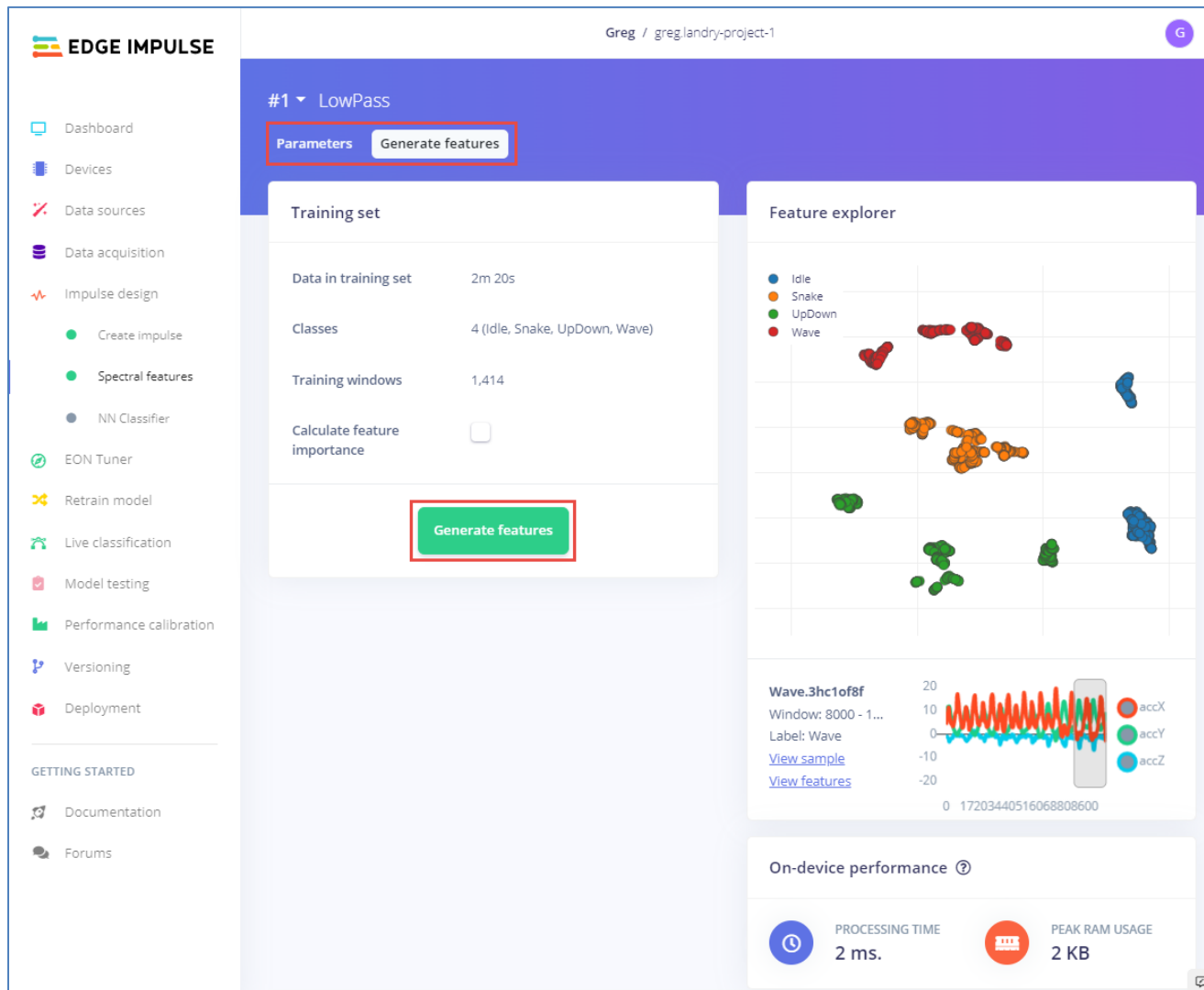
You can go back and forth between the Parameters and Generate features pages to iterate with different filter settings to get the best results. You clone different versions (click the down arrow next to #1 and select "Clone as new version") so you can compare different results.

Once you are satisfied with the processing blocks, you run the training step. You can decide what learning rate to use and how many iterations to run. Some experimentation here will be useful to optimize the results. As with the filter block, you can clone new versions and re-run with different settings if you don't want to over-write the previous training session's results.

> Note: Be sure to select the correct target device before starting training. This is necessary for the model to be optimized for the PSoC™ 6 device that we are using.

8.2.3 Analyze the model

After the training finishes, you will see the end results displayed.

In addition to accuracy, you can view memory use and expected time for this part of the model to run on the target device. You can see the results for the quantized model or for the unquantized (floating point) model. Incorrect results are shown in red on the chart. You can zoom in and out and move the chart around to see the results more clearly. Click on any sample to see where it came from in the dataset.
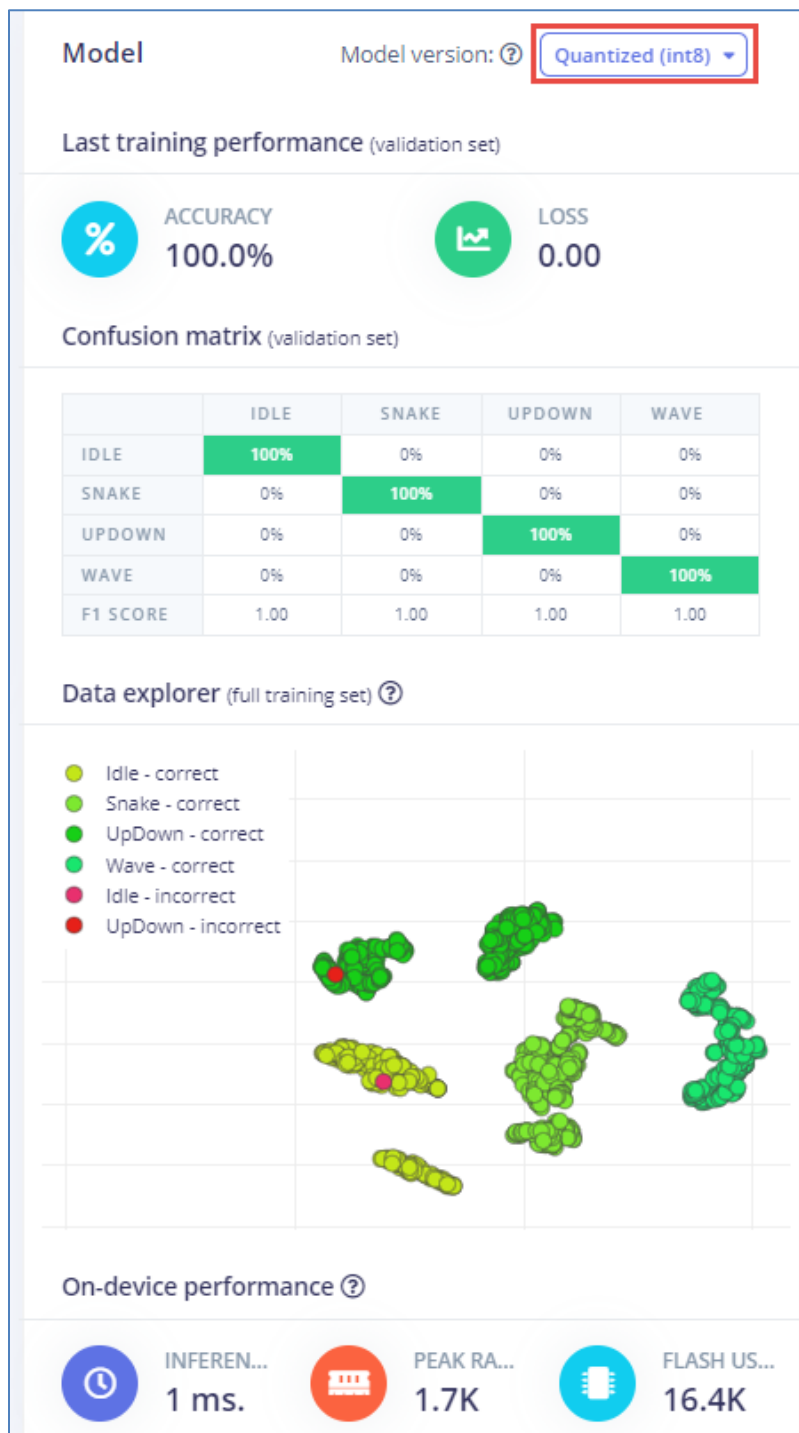
8.2.4   Anomaly detection

One shortcoming of a machine learning classification model is that it won't perform well on data that is outside the envelope of the samples it was trained on. When given an input, the model will always provide a classification even if it's uncertain about the correct classification. You can handle this issue in the firmware by rejecting results that are below a threshold confidence value. Another way to do this in Edge Impulse is through an anomaly detection learning block. This allows the model to automatically reject any uncertain data points.
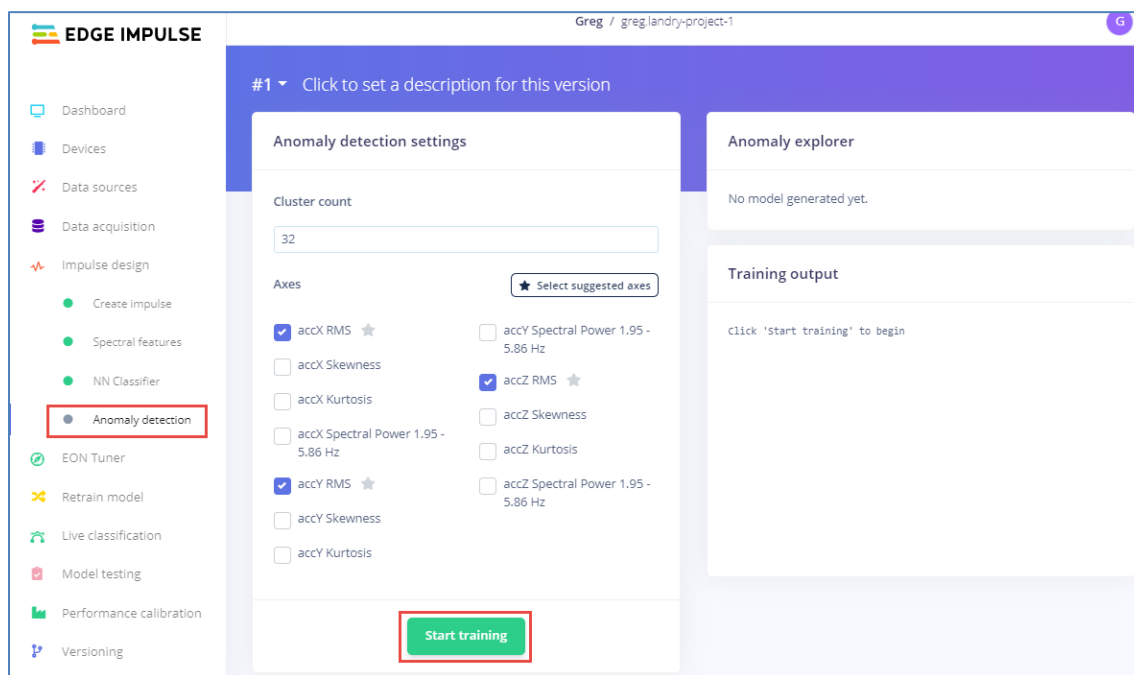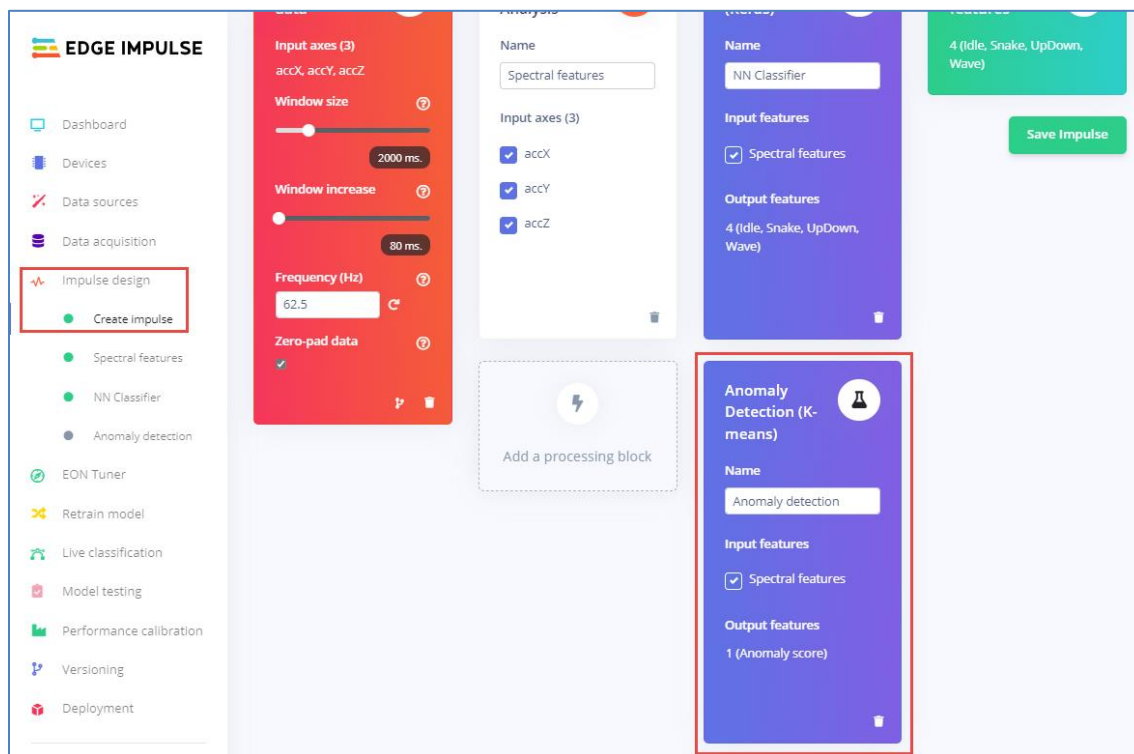
8.2.5 Test and retrain the model

On the "Model Testing" page you can easily test your model against the test datasets that you collected.

If there are misclassifications from some files that you want to fix, you can edit the expected outcome (if necessary) and move those files to the training data (use the three dots on the right side of the datasets) and retrain the model from the "Retrain model" page.

Note: This method adds an entirely new file of samples to the training dataset. If you want to add individual samples to the training data set, use the Live Classification page, which we will discuss next.

From the "Live classification" page, you can test the model against datasets that you have already collected or you can collect new samples from the kit, similar to when you initially captured data by using the "Live classification" page.
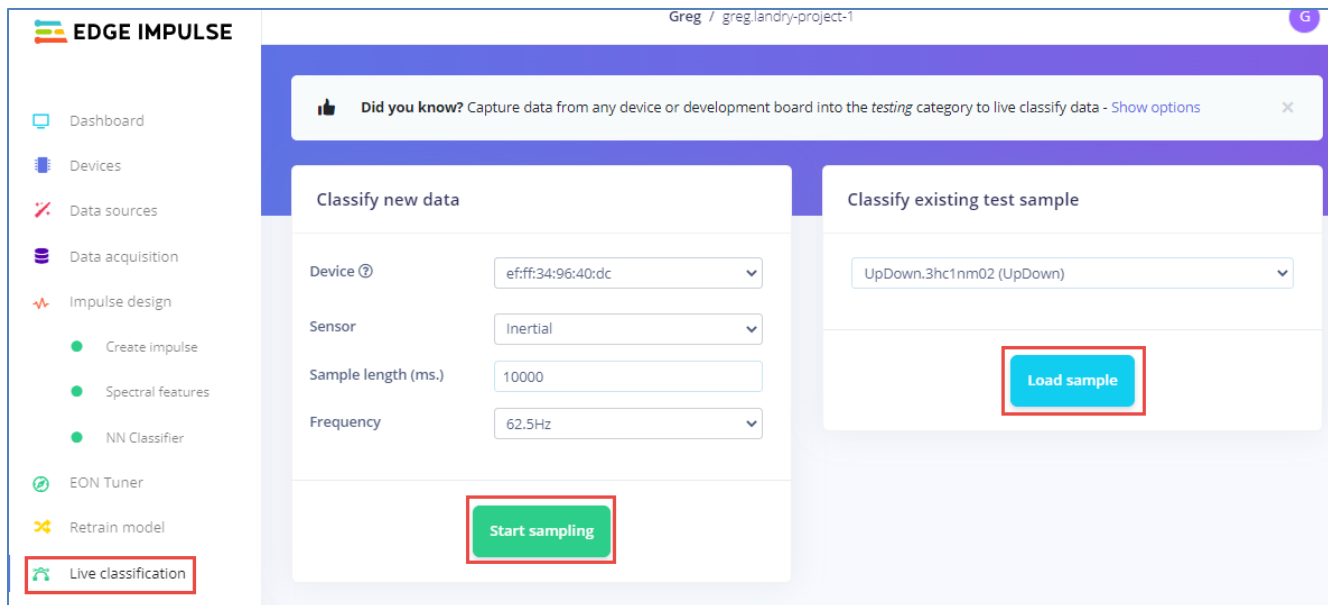
Regardless of the option you choose, the new samples are plotted on the chart with the rest of the samples from the training results. Click on any of the individual samples to see their information. In the summary section, you can provide the correct label for the expected outcome for these samples. Once you have provided the expected outcome you can include the samples in the training dataset (click on the three dots next to summary). You can then retrain the model from the "Retrain model" page.
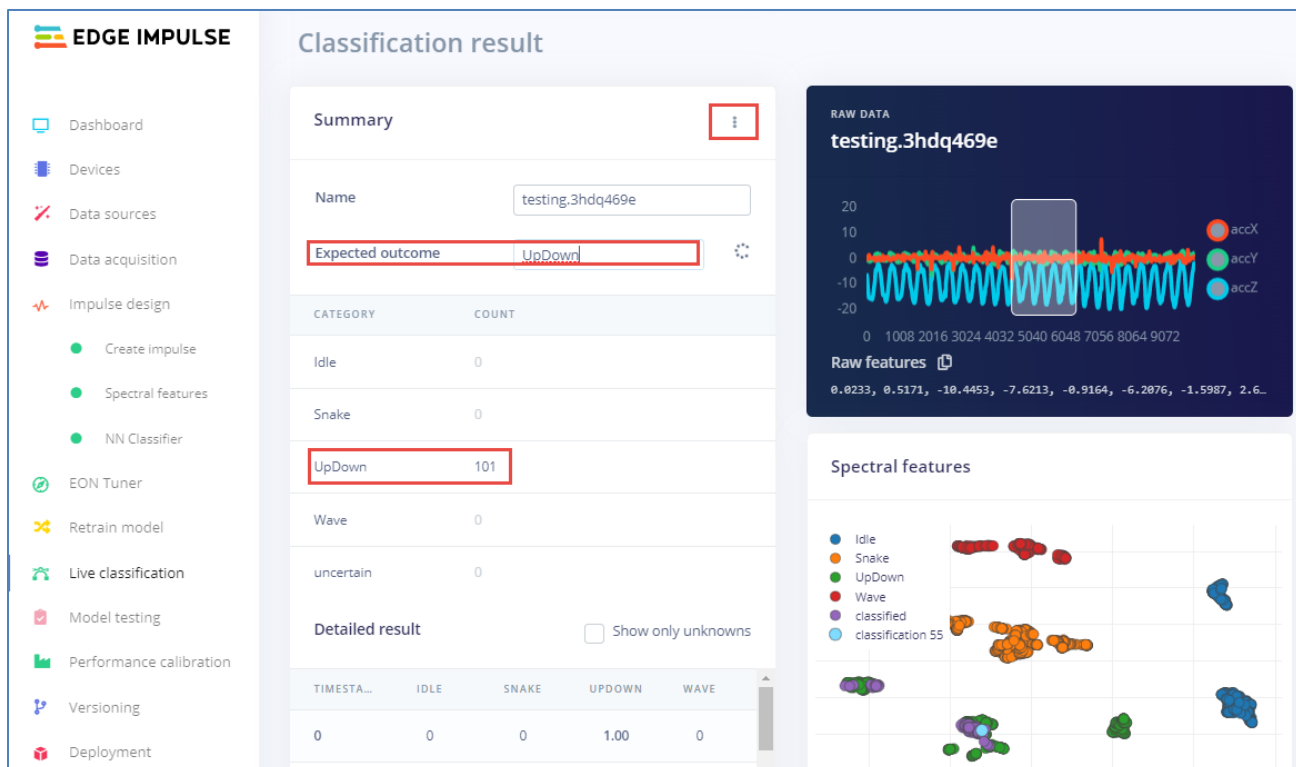
8.2.6   Test on the device

When you are satisfied with the model's performance using Edge Impulse Studio, you can make use of the "Deployment" page to generate the complete binary code (hex file) for a test application that can be programmed directly to the PSoC™ kit with no integration required. Under the **Build Firmware** section, you will select **Infineon PSoC 62S2 Wi-Fi BT Pioneer Kit**.

Once you click build, the test application including the model is downloaded in a zip file. You can unzip the file and then program the hex file to the kit using Cypress Programmer. After programming, open a UART terminal to interact with the kit. Enter AT+RUNIMPULSE to start collecting data.

By default, a new output will be calculated every two seconds based on which gesture you are performing with the kit. The blue LED will flash when the kit is sampling the sensor so you can identify when to perform a given gesture to test its recognition.

```
COM114 - PuTTY                                              —   □   ×
Type AT+HELP to see a list of commands.
 AT+RUNIMPULSE
Inferencing settings:
        Interval: 16.0000ms.
        Frame size: 375
        Sample length: 2000.00 ms.
        No. of classes: 4
Starting inferencing, press 'b' to break
Starting inferencing in 2 seconds...
Predictions (DSP: 25 ms., Classification: 1 ms., Anomaly: 0 ms.):
    Idle:         0.996094
    Snake:        0.000000
    UpDown:       0.000000
    Wave:         0.000000
    anomaly score: -0.184449
Starting inferencing in 2 seconds...
Predictions (DSP: 26 ms., Classification: 0 ms., Anomaly: 1 ms.):
    Idle:         0.000000
    Snake:        0.000000
    UpDown:       0.996094
    Wave:         0.000000
    anomaly score: 0.145422
Starting inferencing in 2 seconds...
Predictions (DSP: 25 ms., Classification: 1 ms., Anomaly: 0 ms.):
    Idle:         0.000000
    Snake:        0.250000
    UpDown:       0.000000
    Wave:         0.750000
    anomaly score: 2.114249
Starting inferencing in 2 seconds...
Predictions (DSP: 26 ms., Classification: 0 ms., Anomaly: 1 ms.):
    Idle:         0.000000
    Snake:        0.996094
    UpDown:       0.000000
    Wave:         0.003906
    anomaly score: 0.691605
Starting inferencing in 2 seconds...
Inferencing stopped by user
>
```
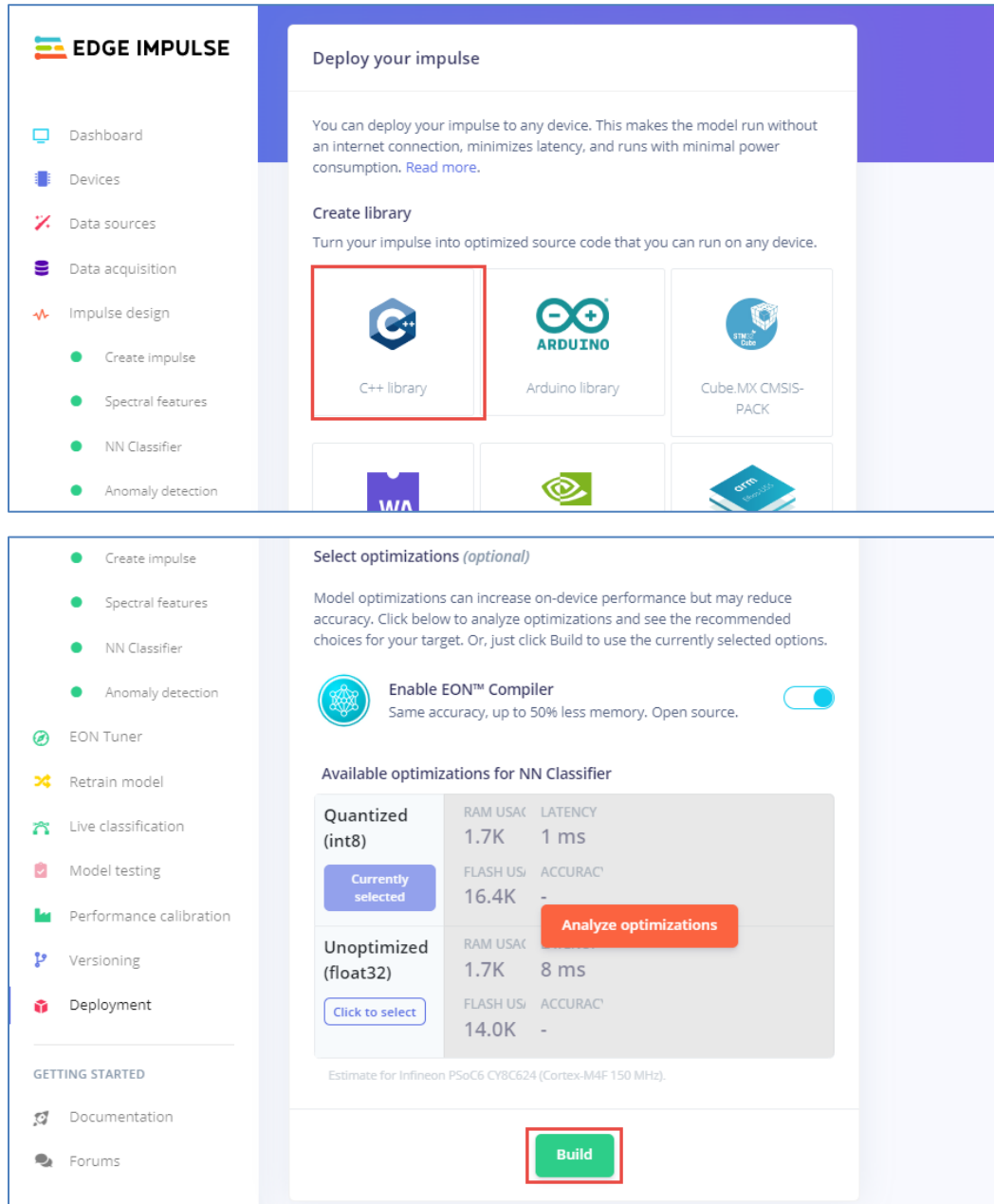
Press 'b' to stop the inferencing engine. You can also enter AT+HELP to list other commands.

> *Note: You will likely need to reset the kit after programming it or plugging it in to make sure the sensors get initialized properly. If you see error messages that say "ERR: no Accel data!", reset the kit and try again.*

8.2.7 Deployment

Once you have the final model, you will want to deploy it in your end application. You will use the same "Deployment" page as the previous step, but in this case, you will select **C++ Library** to generate a library that can be included into your own application.


It's useful to review the Edge Impulse code example to see how the model is used.

You can find documentation on the Edge Impulse API used to interface with the model from the website. On the left panel, click on **Documentation**. Scroll down to the **Deployment** section (not the one in Edge Impulse Studio section) and select **C++ Library > As a generic C++ library**.
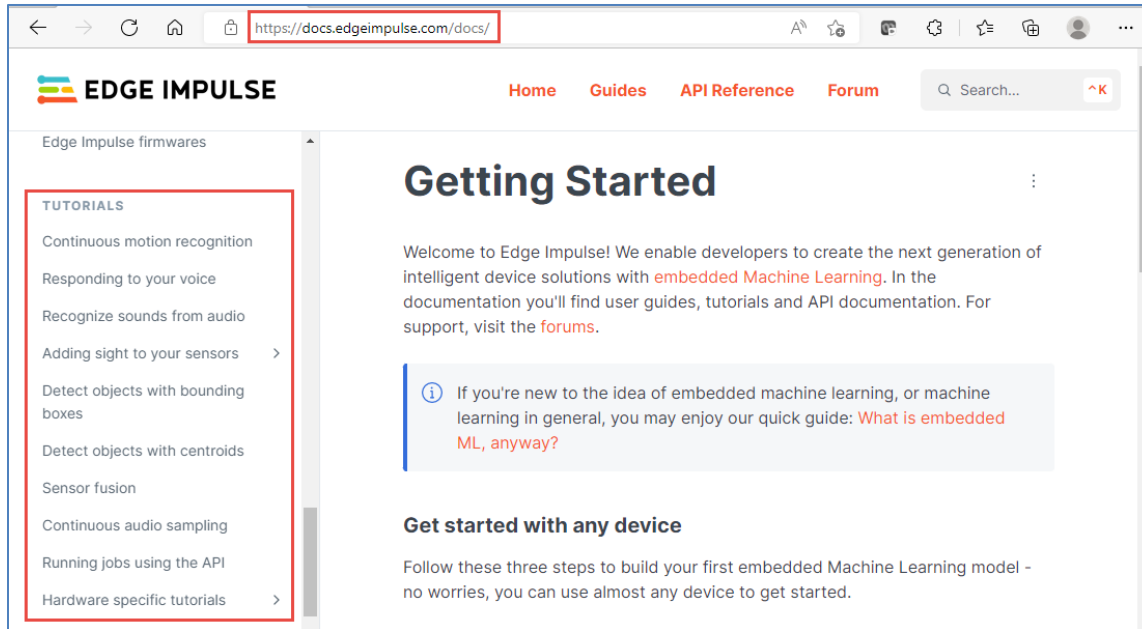
8.3   Getting started

For your first step, go to the Edge Impulse website, click Login, and create a new account. When you first create the account, it will take you through a tutorial in Edge Impulse Studio to quickly create a keyword detection model. It briefly goes through each of the steps. Though it is a high-level tutorial, it is worth trying it to see what the overall flow looks like.

Once you have finished the tutorial, you can click on the "Documentation" link from the left panel in Edge Impulse Studio.
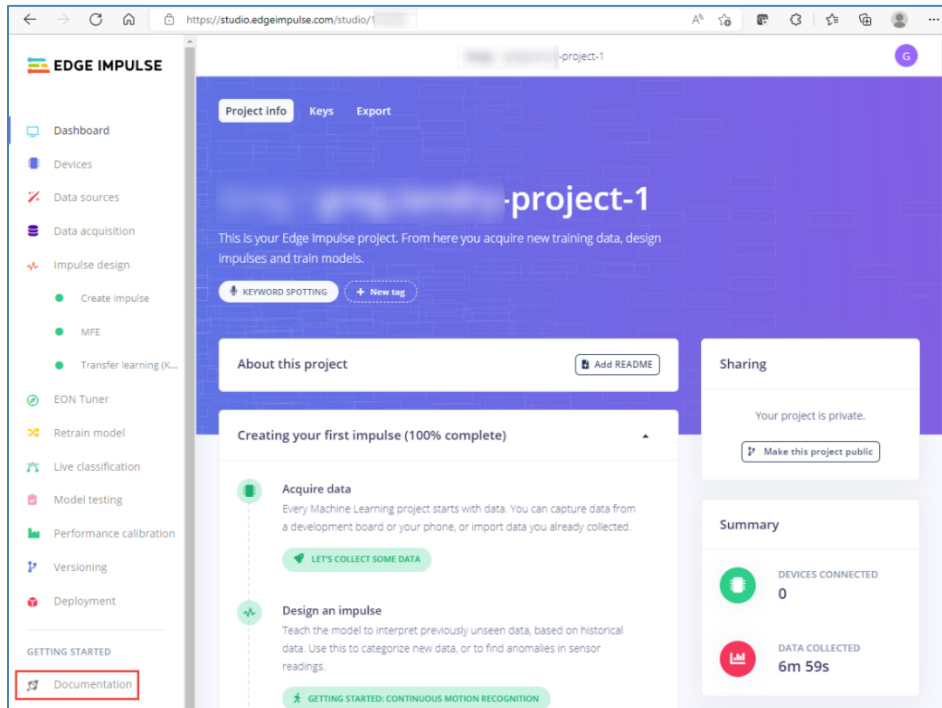
Note: You can also get to the documentation from https://edgeimpulse.com/ by going to Developers > Docs or you can go directly to https://docs.edgeimpulse.com/docs/.

The documentation opens in a new tab and starts out in the "Getting Started" page. There are many other links on the left panel including documentation for the tools, a set of tutorials with example applications, pre-built datasets, supported development platforms and more.

The tutorials are a useful way to learn about Edge Impulse Studio. Many of them even include YouTube video versions of the tutorial. You might find these easier to follow and more informative than the Getting Started section of the documentation. One such example is the: Continuous motion recognition tutorial.
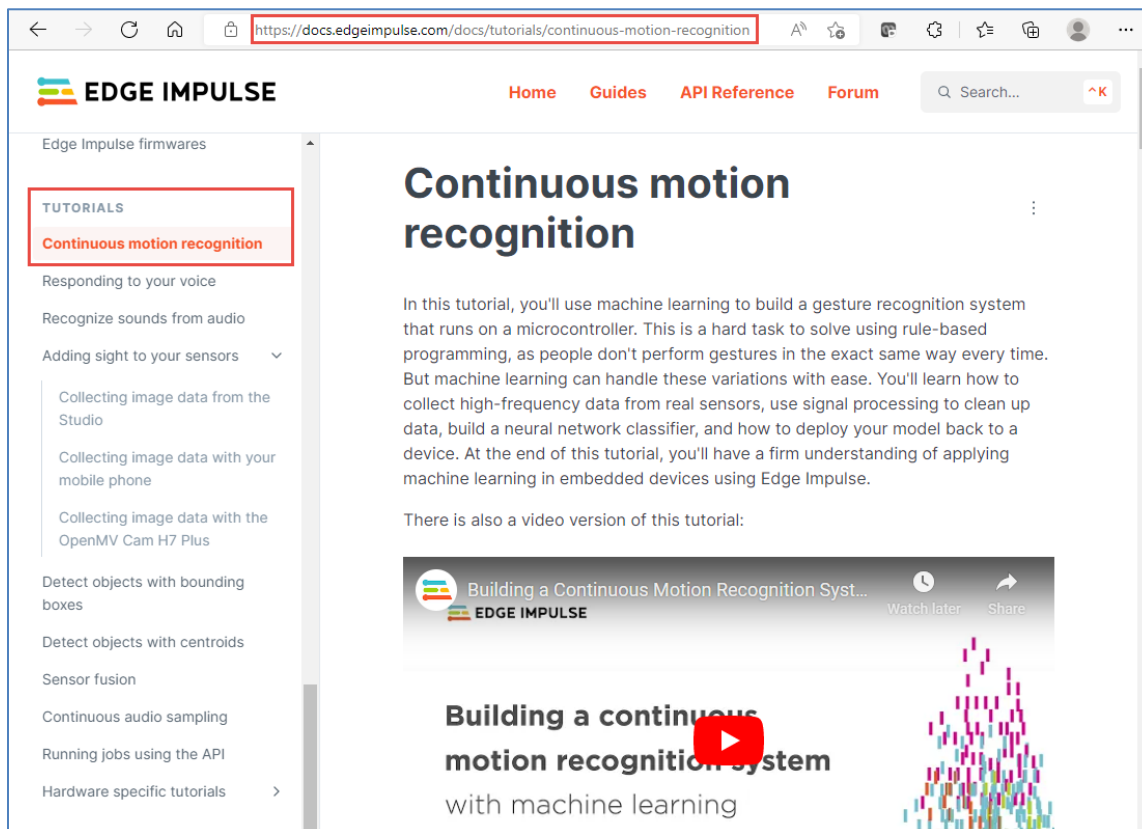
8.4   Using the Infineon kit with Edge Impulse

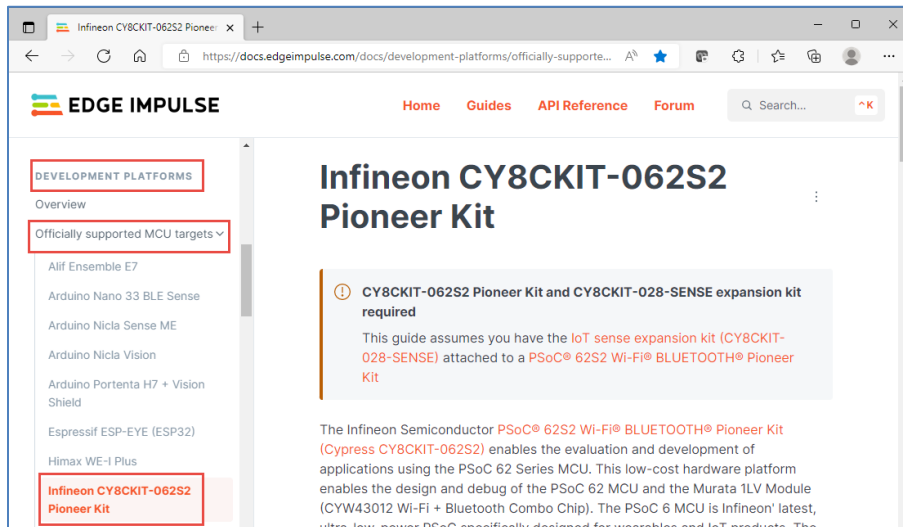While the list of supported devices for many of the tutorials does not include the Infineon PSoC™ 6 kit, it is in fact supported. You can find instructions on connecting the Infineon PSoC™ 6 kit from the documentation, using the link in the left panel for **Development Platforms > Official supported MCU targets > Infineon CY8CKIT-062S2 Pioneer Kit.**

If you follow the instructions, you will install Infineon's Cypress Programmer and then use it to program a hex file containing the Edge Impulse base firmware onto the kit. This allows the kit to connect directly to Edge Impulse Studio.

The Edge Impulse base firmware is also available directly in ModusToolbox™ as a code example. The name in Project Creator is "EdgeImpulse Continuous Motion recognition." If you create that application, you will be able to program it using ModusToolbox™ instead of Cypress Programmer. It is also a good reference on using an Edge Impulse model.

Though the code example uses the BMI160_driver library to interface with the motion sensor, the motion sensor on your shield may have a different chip ID than the default value in the library. Therefore, after creating the application, you may need to edit the *Makefile* to add the following lines after the existing `PREBUILD=` line:

```
# Copy the the BMI160 motion sensor driver files to the application and
# update the chip ID to match what is on the CY8CKIT-028-SENSE shield.
PREBUILD+=rm -rf bmi160;
PREBUILD+=mkdir bmi160/;
PREBUILD+=cp $(SEARCH_BMI160_driver)/bmi160* bmi160/;
PREBUILD+=cp $(SEARCH_BMI160_driver)/LICENSE bmi160/;
PREBUILD+=sed -i 's/UINT8_C(0xD1)/UINT8_C(0xD8)/' bmi160/bmi160_defs.h;
CY_IGNORE+=$(SEARCH_BMI160_driver)
```

> Note: If the silk screen on the back of the shield says "9-axis IMU" and "600-60608-01 REV03" it is v1. If the silk screen says "6-axis IMU" and "600-60608-01 REV04" it is v2. The code above is necessary only for v1 shields.

Note:If your machine uses csh instead of bash terminals (such as a Mac) change the second to last line above to the following:

```
PREBUILD+=sed -i  ''  's/UINT8_C(0xD1)/UINT8_C(0xD8)/' bmi160/bmi160_defs.h;
```

As described in the workflow section, once you have a completed model, you can generate two types of outputs from Edge Impulse. 1) A C library file containing the model, or 2) a hex file containing a complete test application for the target hardware with your model embedded in it.

8.5 Exercises

Exercise 1: Install Edge Impulse tools

Most of Edge Impulse's tools are web-based, but Infineon's Cypress Programmer is used for programming their test firmware onto the PSoC™. You can skip this step if you don't plan to evaluate Edge Impulse.

Note:If you already did the installation in Chapter 1 you can skip this exercise.

☐   1.   Go to the following page: https://softwaretools.infineon.com/tools/com.ifx.tb.tool.cypressprogrammer

☐   2.   Download and install Cypress Programmer.

Exercise 2: Setup account and follow the quick start tutorial

In this exercise, you will setup an Edge Impulse account and will complete the online Getting Started tutorial.

☐   1.   Go to https://edgeimpulse.com/

☐   2.   Click the **Login** button and follow the instructions to create a new account.

☐   3.   The tutorial should launch automatically once you log into your account.

Note: You will need to record audio from your computer's microphone for 38 seconds, so find a place where you can do that without too much background noise.

Exercise 3: Program the Edge Impulse firmware onto the kit

This section will prepare the kit for collecting data and to test out completed models on the target hardware.

☐   1.   Go to https://docs.edgeimpulse.com/docs/development-platforms/officially-supported-mcu-targets/infineon-cy8ckit-062s2 (You can find this in the Edge Impulse documentation along the left side under Development Platforms > Official supported MCU targets).

Note: Since there are some differences from the directions written on the page, it is suggested to follow the details below for each section.

Installing Dependencies

You can skip this section because you should have already installed Cypress Programmer and the Edge Impulse CLI is no longer required to connect the kit to Edge Impulse Studio.

Updating the Firmware

Rather than follow the instructions on the web page for updating the firmware, we'll use the ModusToolbox™ code example instead. If you prefer to use the hex file with Cypress Programmer, you can follow the web page instructions.

a. Run the ModusToolbox™ Project Creator and select the **CY8CKIT-062S2-43012** BSP. On the application page, select **EdgeImpulse Continuous Motion recognition** from the Machine Learning category. You can keep the default name or change it if you want.

b. Once the application has been created, open the application's *Makefile* to add the following lines after the existing PREBUILD= line:

```
# Copy the BMI160 motion sensor driver files to the application and
# update the chip ID to match what is on the CY8CKIT-028-SENSE shield.
PREBUILD+=rm -rf bmi160;
PREBUILD+=mkdir bmi160/;
PREBUILD+=cp $(SEARCH_BMI160_driver)/bmi160* bmi160/;
PREBUILD+=cp $(SEARCH_BMI160_driver)/LICENSE bmi160/;
PREBUILD+=sed -i 's/UINT8_C(0xD1)/UINT8_C(0xD8)/' bmi160/bmi160_defs.h;

CY_IGNORE+=$(SEARCH_BMI160_driver)
```

c. Build the application and program it to the device.

d. Open a UART terminal to the kit with a baud rate of 115200.

e. The UART will display `Type AT+HELP to see a list of commands`. You can do that if you want to see all of the available commands.

f. Enter the command AT+RUNIMPULSE to start running the default model included in the code example. A set of samples will be taken every 2 seconds. Press the **b** key to stop.
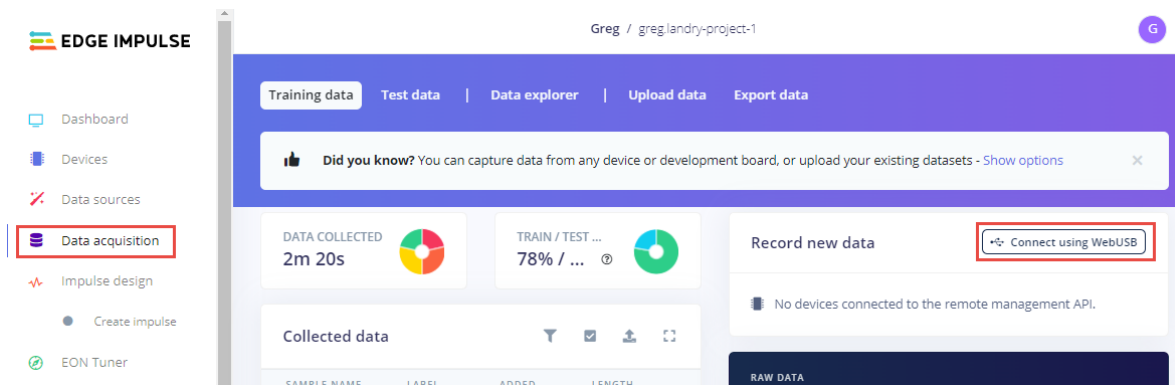
Note: If you see `ERR: no Accel data,` reset the kit and try again.

Connecting to Edge Impulse

☐   g.   Connect the kit to Edge Impulse Studio and verify the connection.

Note: In Edge Impulse Studio, go to "Data acquisition" and click on "Connect using WebUSB." Select the COM port for the KitProg3 and click "Connect."

Exercise 4: Continuous motion recognition (i.e. Gestures)

In this exercise, you will follow the "Continuous motion recognition" tutorial to understand how to build a machine learning model using Edge Impulse Studio.

1. Go to the Continuous motion detection tutorial at
    https://docs.edgeimpulse.com/docs/tutorials/continuous-motion-recognition. You can find this in the
    Edge Impulse documentation along the left side under Tutorials.

   Note: *We recommended watching the video first before trying this for yourself.*

Exercise 5: Try other tutorials

In this exercise, you will try out some additional tutorials.

1. Read through the descriptions for each of the tutorials to understand what each one does. Select one or
    two to follow based on your interest.

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.