CE 6302.001 – Microprocessor and Embedded Systems – F22

Dhavalashri Prasad Net-ID : dxp210085	Sanmati Marabad Net-ID : sxm210368

TinyML-Motion Recognition

The main aim of this project is to train TICC1352P Launchpad connected to BoostExcel-sensors board for different kinds of motion recognition.

Softwares Used

- Edge Impulse CLI
- NodeJS
- Texas Instruments Uniflash

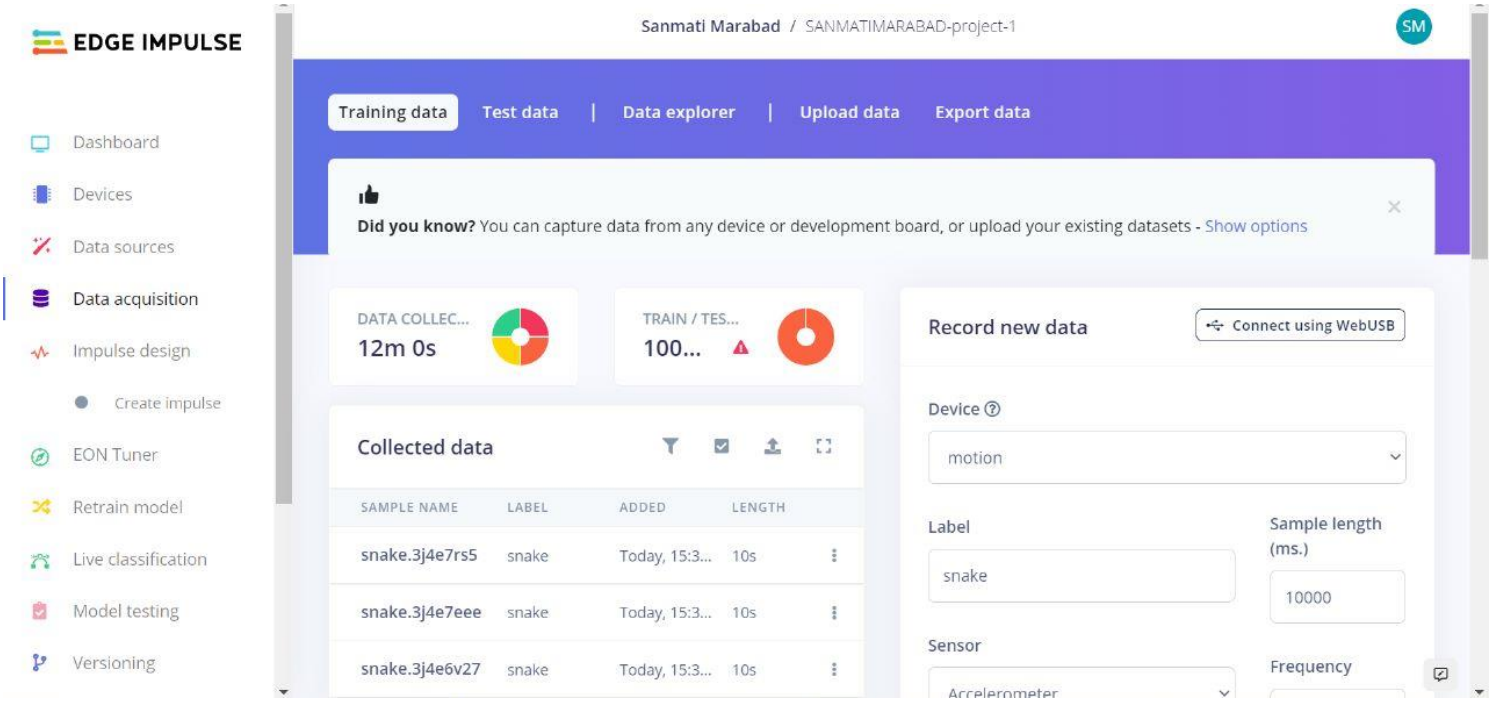
Steps to connect development board to Edge Impulse

- Flash the board using Texas Instruments Flash Batch 5.
- Create an account on Edge Impulse which is used to collect the data and train the system.
- Connect to the board using Edge Impulse daemon command.
- Now we can access the device on the Edge Impulse website
- After the connection is succeeded, we start training the system for different kind of movements.

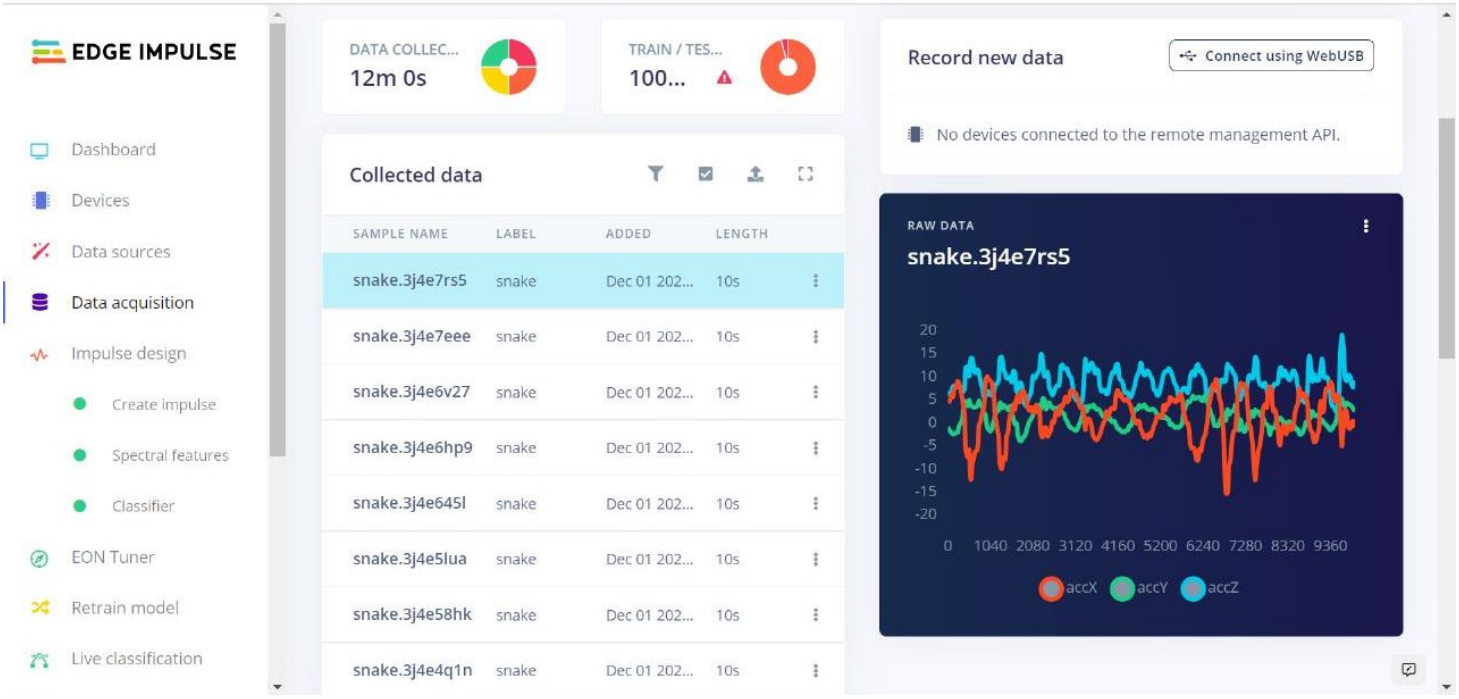
Data acquisition

This is the process of collecting different data sets in order to train the system.

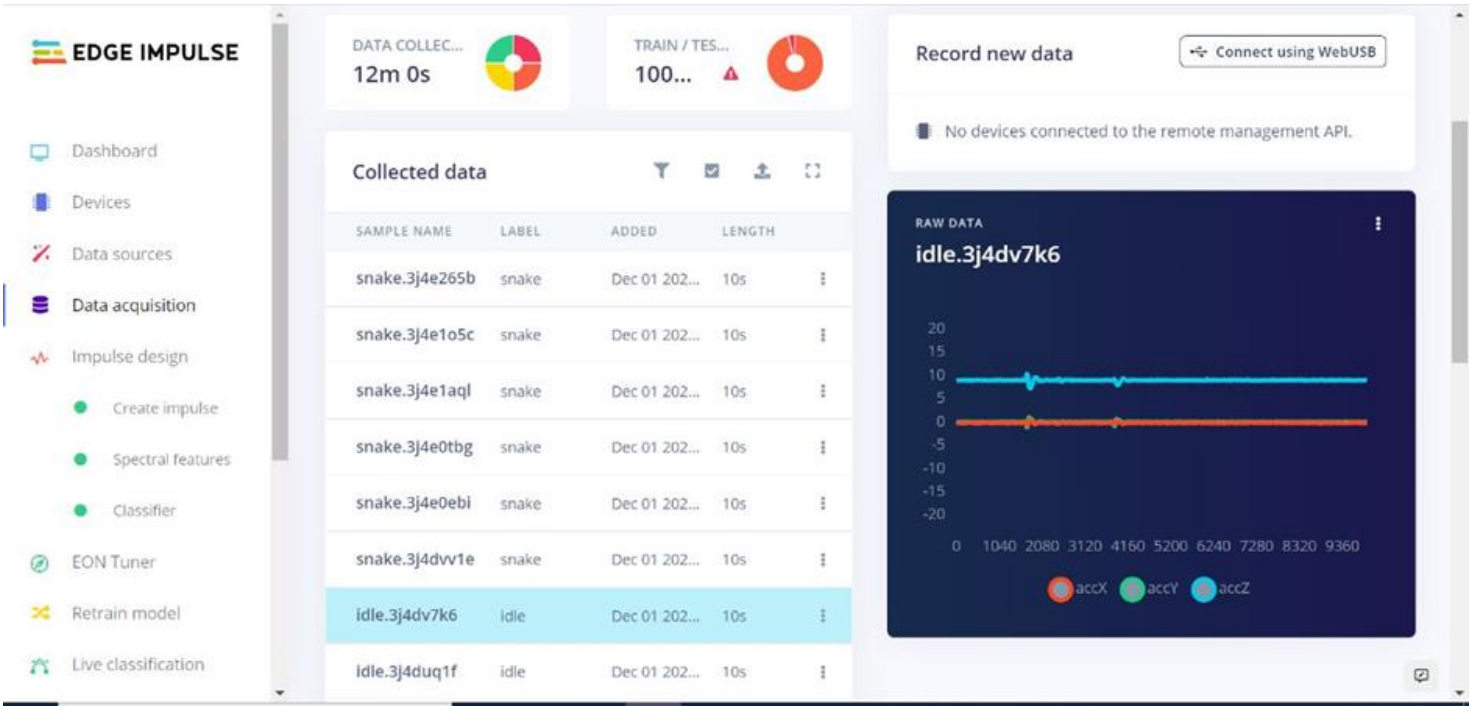
- For each kind of movement, we create a different label like up-down, wave, snake and idle.
- The sample length is set to 10,000 and frequency is set to 62.5Hz
- We use the built-in accelerometer in the booster sensor pack for motion recognition.
- Each sample data was 30 seconds long. Similarly, for each movement, data was collected 6 times (6 samples).



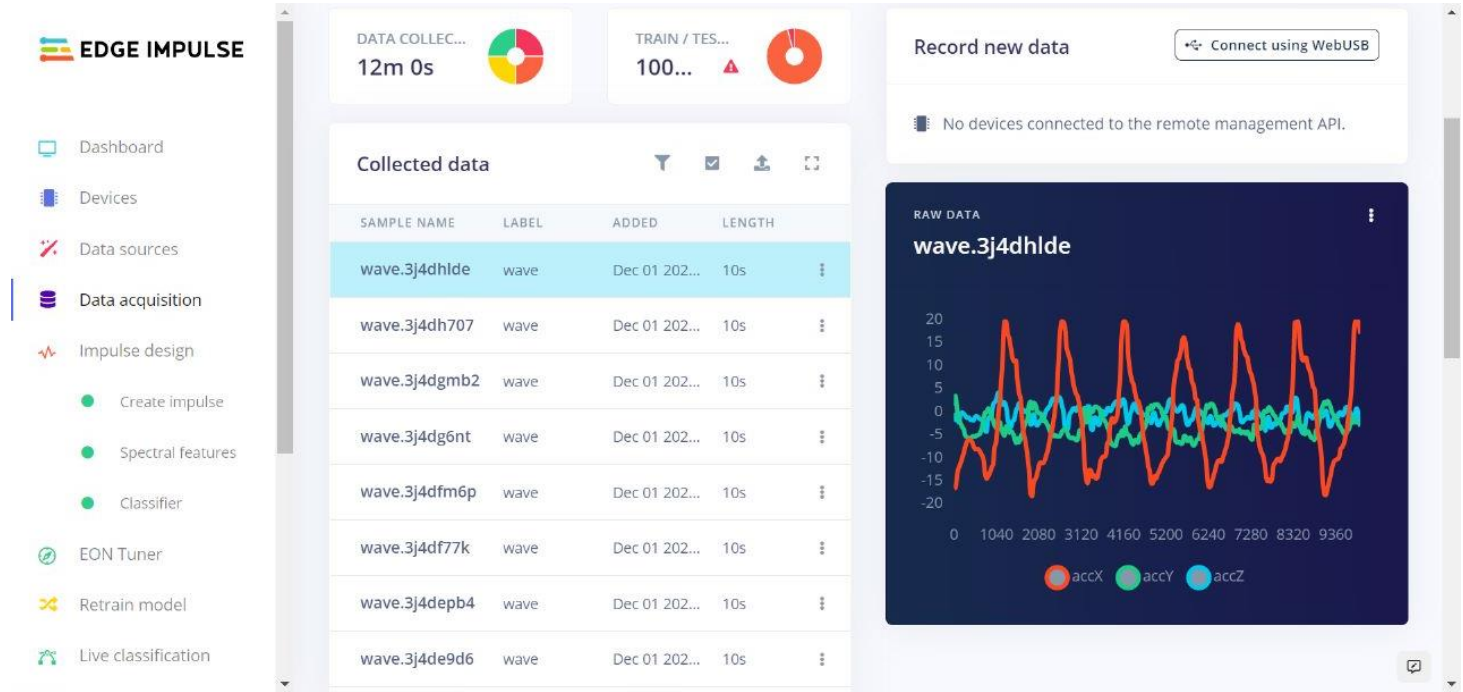
Data Acquisition



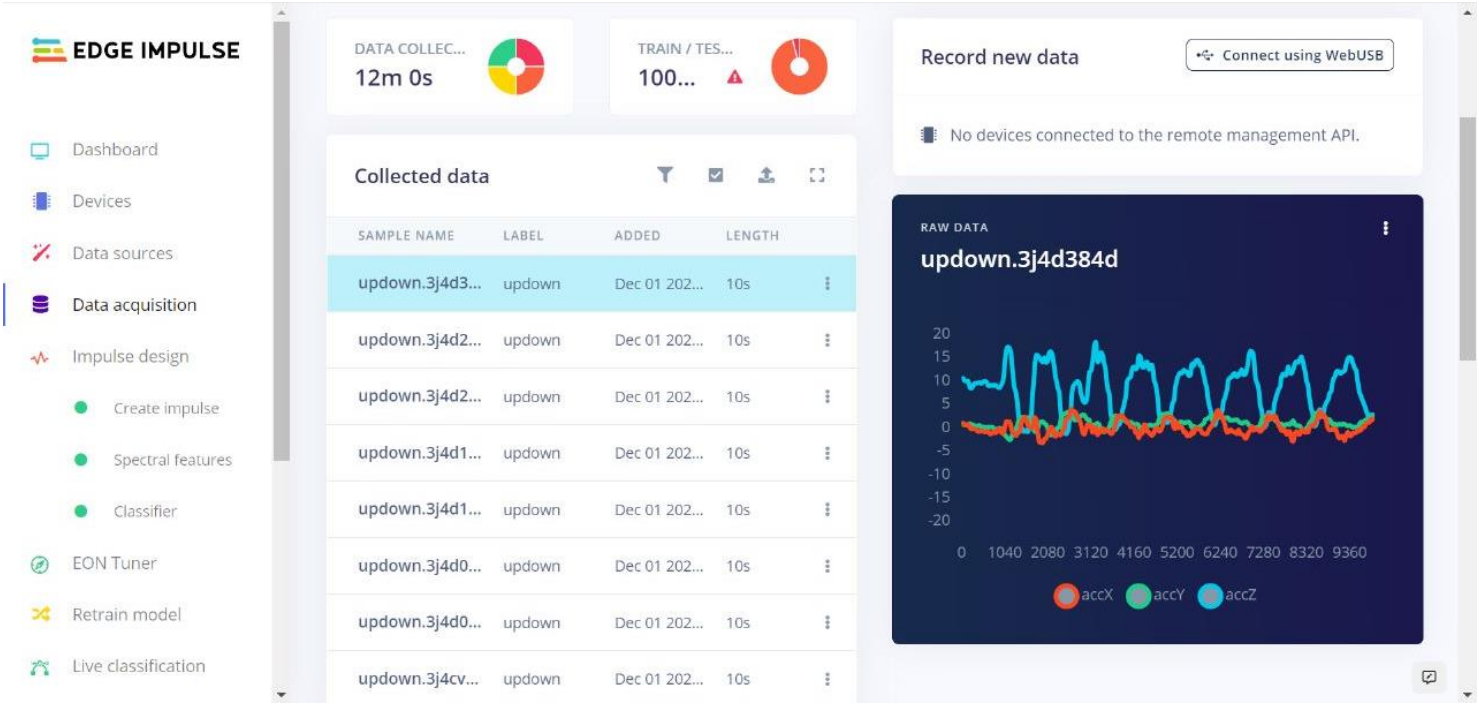
Raw data acquisition for Snake training sample



Raw data acquisition for idle training sample



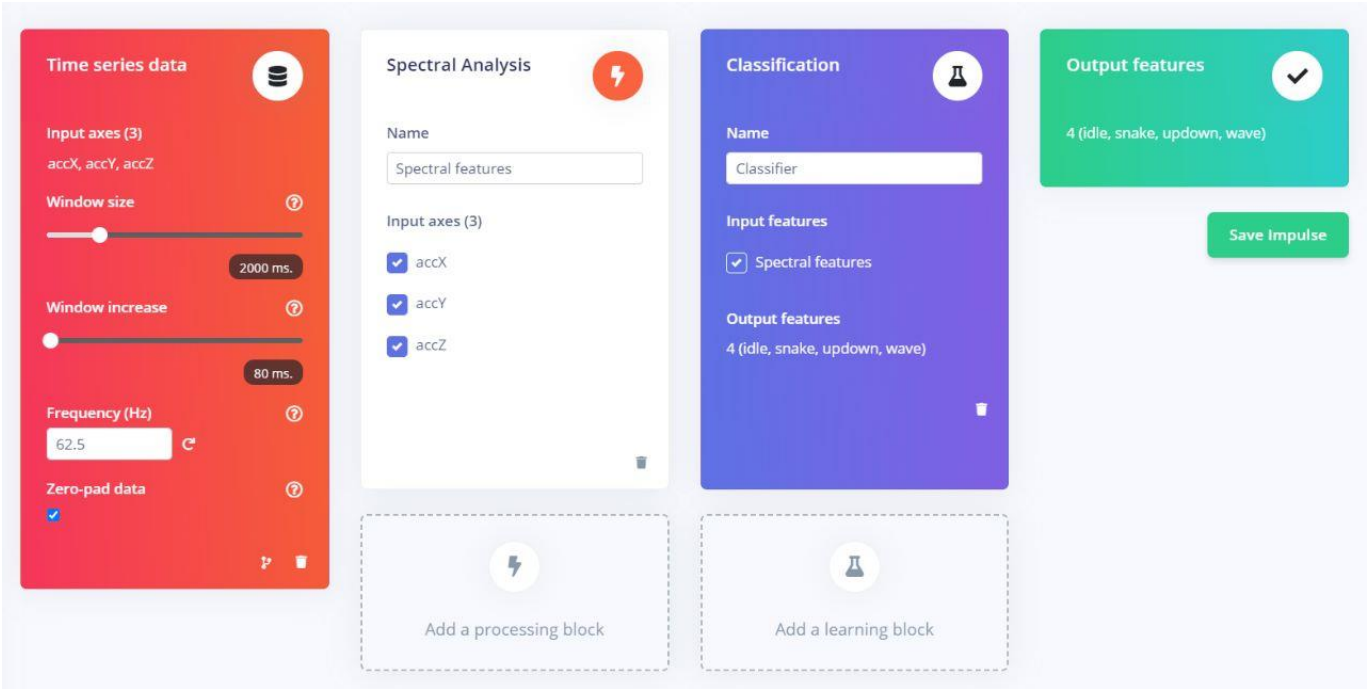
Raw data acquisition for Wave training sample



Raw data acquisition for Up-Down training sample

Impulse Design

Your Impulse may now be made when you have gathered the necessary facts for your project. Three primary building components will make up a full Impulse: an input block, a processing block, and a learning block.



❖ **Input Block**

- The input block indicates the type of input data that the system is getting trained. This can be audio, motion or images.
- The inputs to input block are 3 axes: Accx, Accy and Accz and a window size of 2000ms and window increase of 80ms.
- The frequency is set to 62.5Hz.

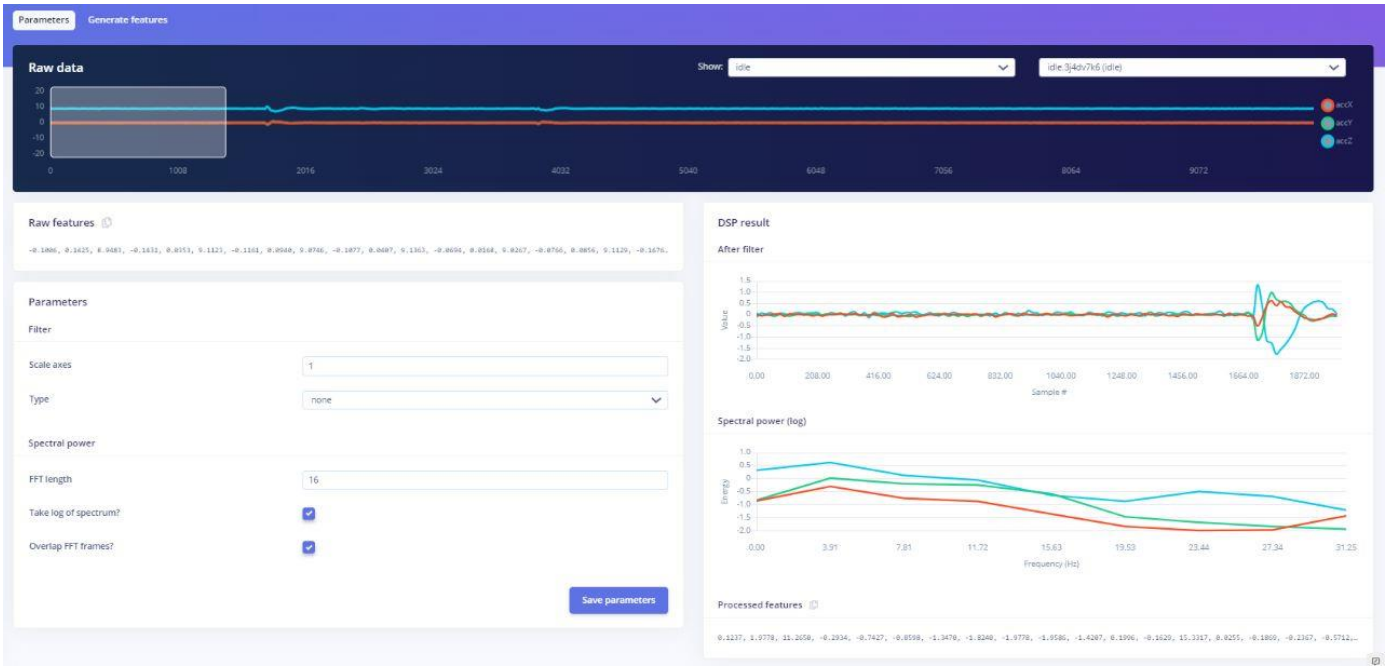
❖ **Processing Block**

- A processing block is a feature extractor which processes the input data and extracts the features from it so that the machine learning model can be trained using the extracted features.
- For motion recognition, we use the spectral analysis as a processing block.
- The inputs to spectral analysis are 3 axes: Accx, Accy and Accz

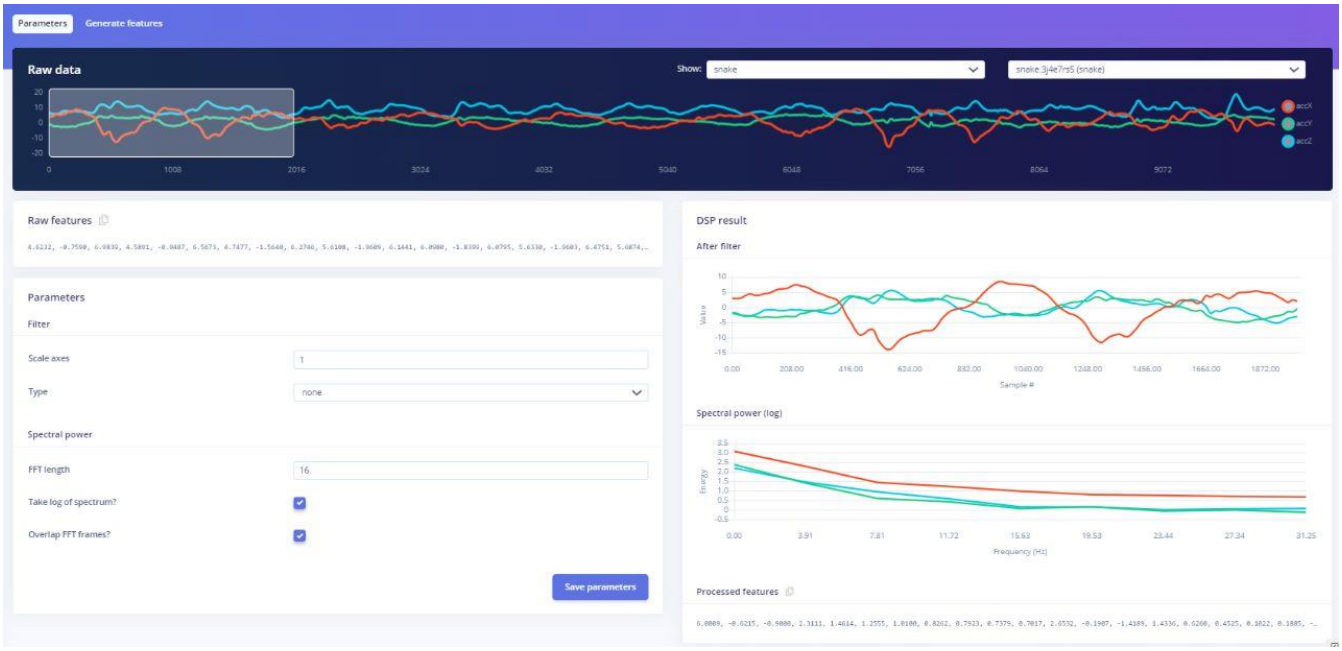
❖ **Learning Block**

- A learning block is a neural network that is trained to learn the input data.
- In the current project, we use classification as the learning block where it classifies and outputs four features: idle, snake, up-down and wave.

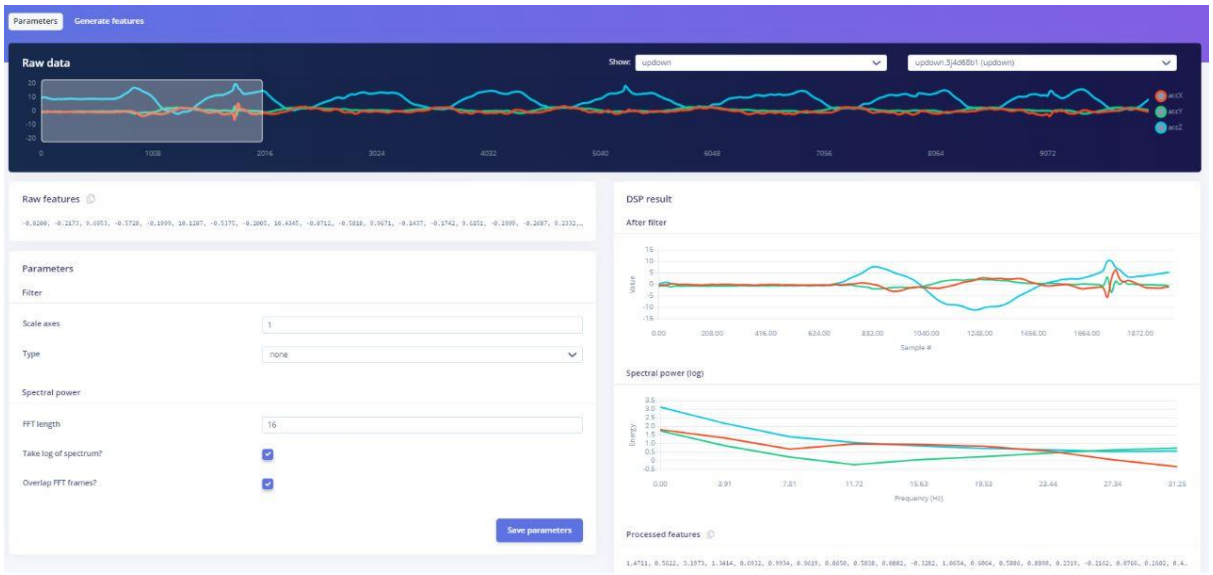
After selecting the impulse blocks, we need to save the impulse and next step is to generate the features using spectral features. This step extracts and generates features from the raw input data.



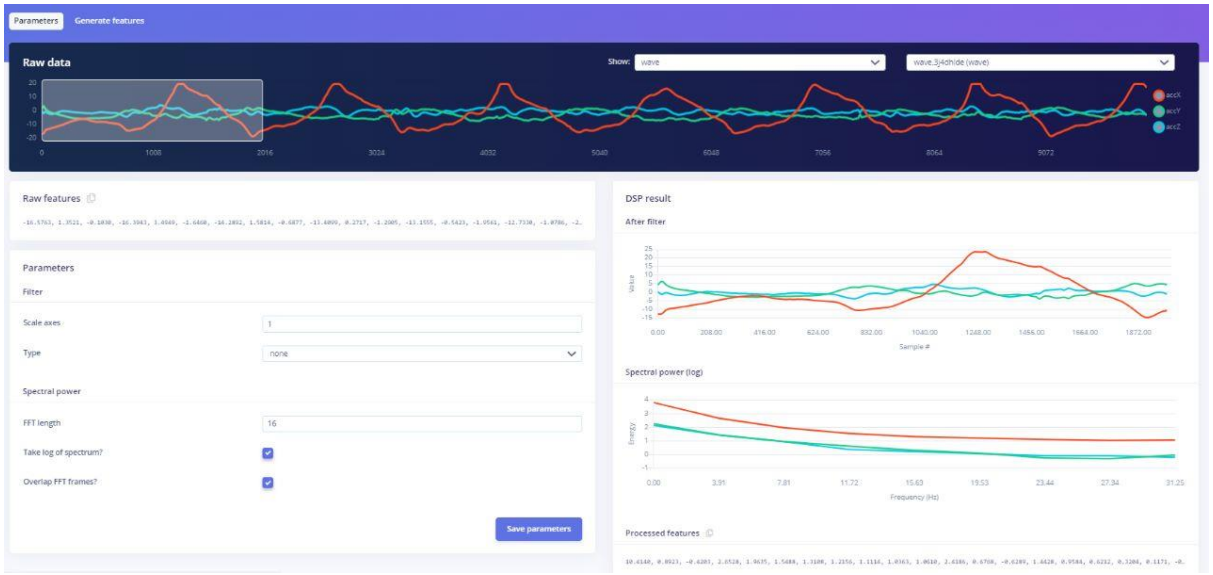
Parameters for idle class



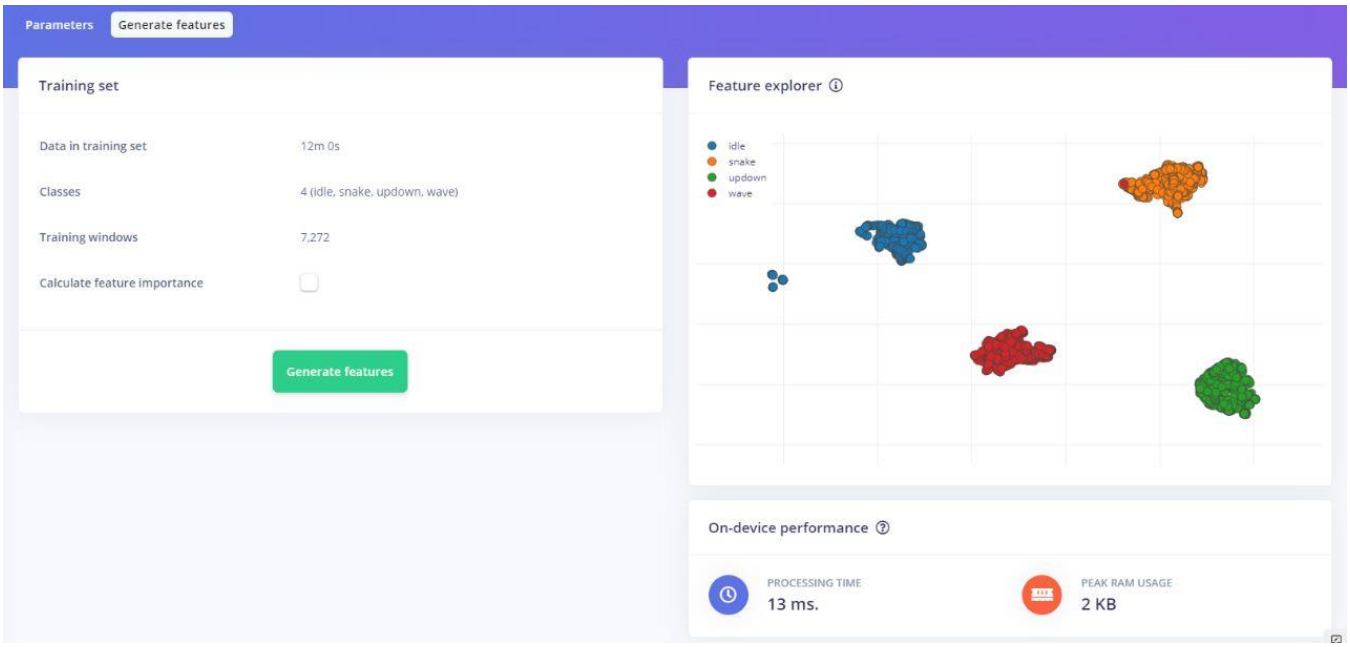
Parameters for Snake class



Parameters for Up-Down class



Parameters for Wave class



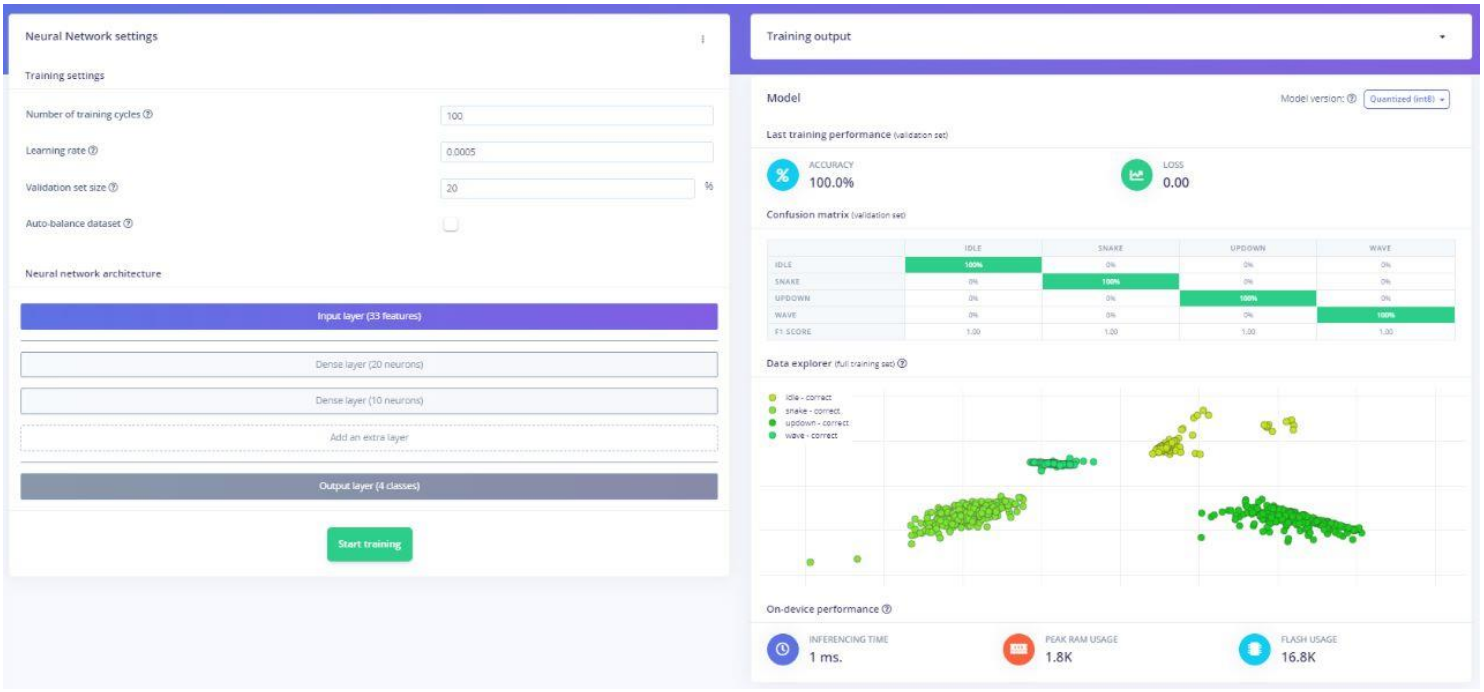
Features generated for different types inputs

Classifier

In order for the system to learn the features extracted from the input data, we need to train it using classifier.

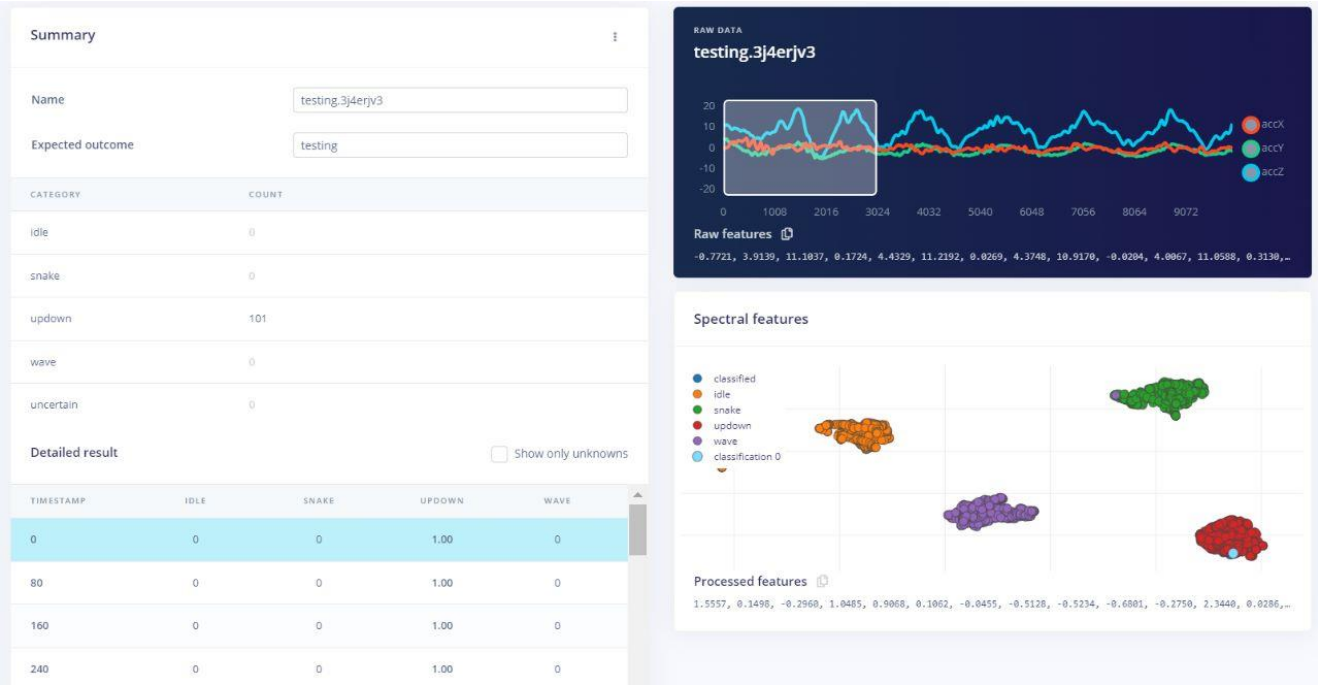
Results of training

The accuracy for each class is found to be 100% and there was no loss of data observed since the selected features were distinct from each other.

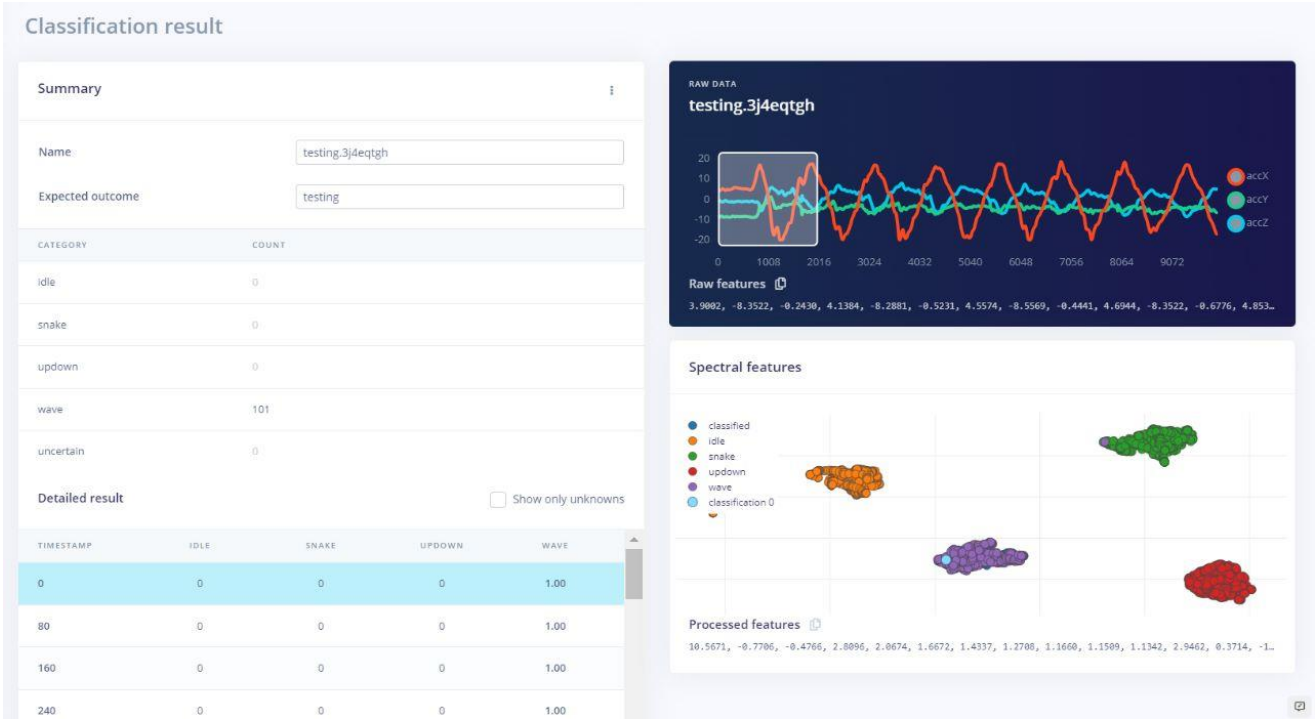


Testing using Live Classification

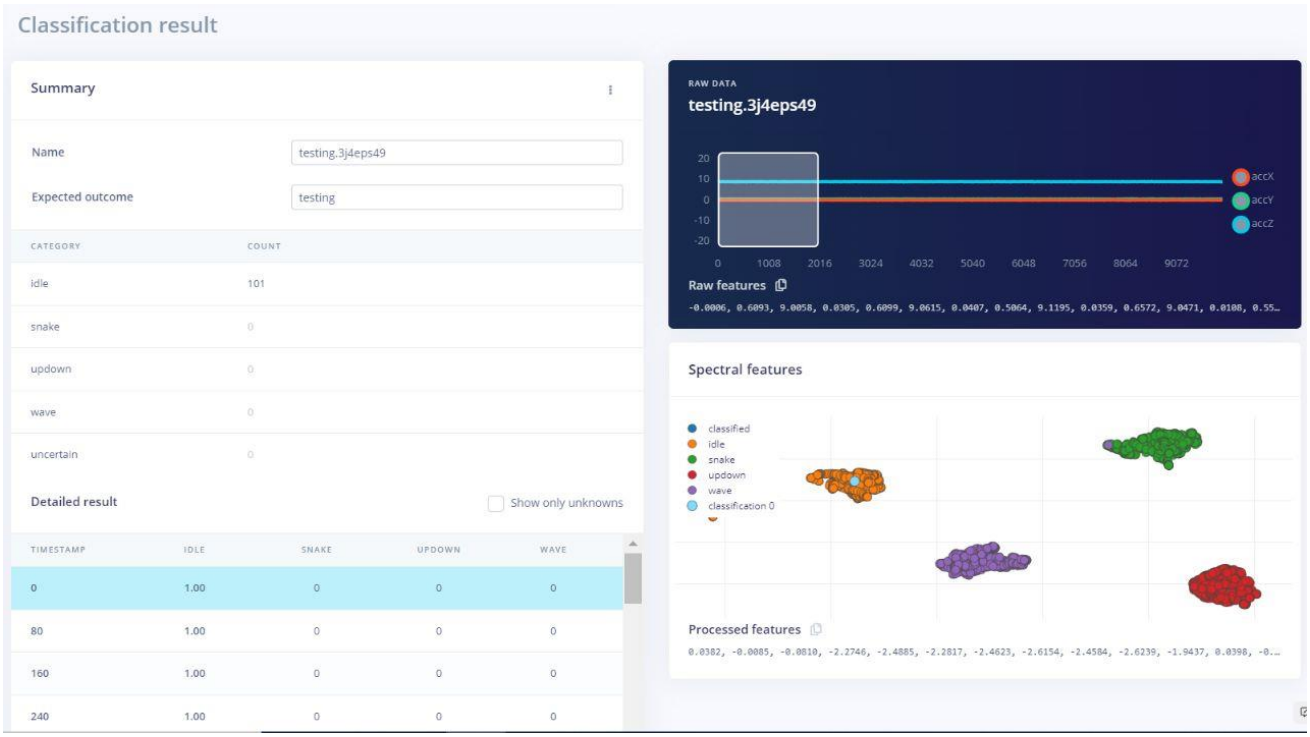
Before dumping the trained model into the board, testing it using Live Classification is necessary. The classification results were observed to be 100% accurate for all the different inputs that were given.



Testing of Up-Down feature

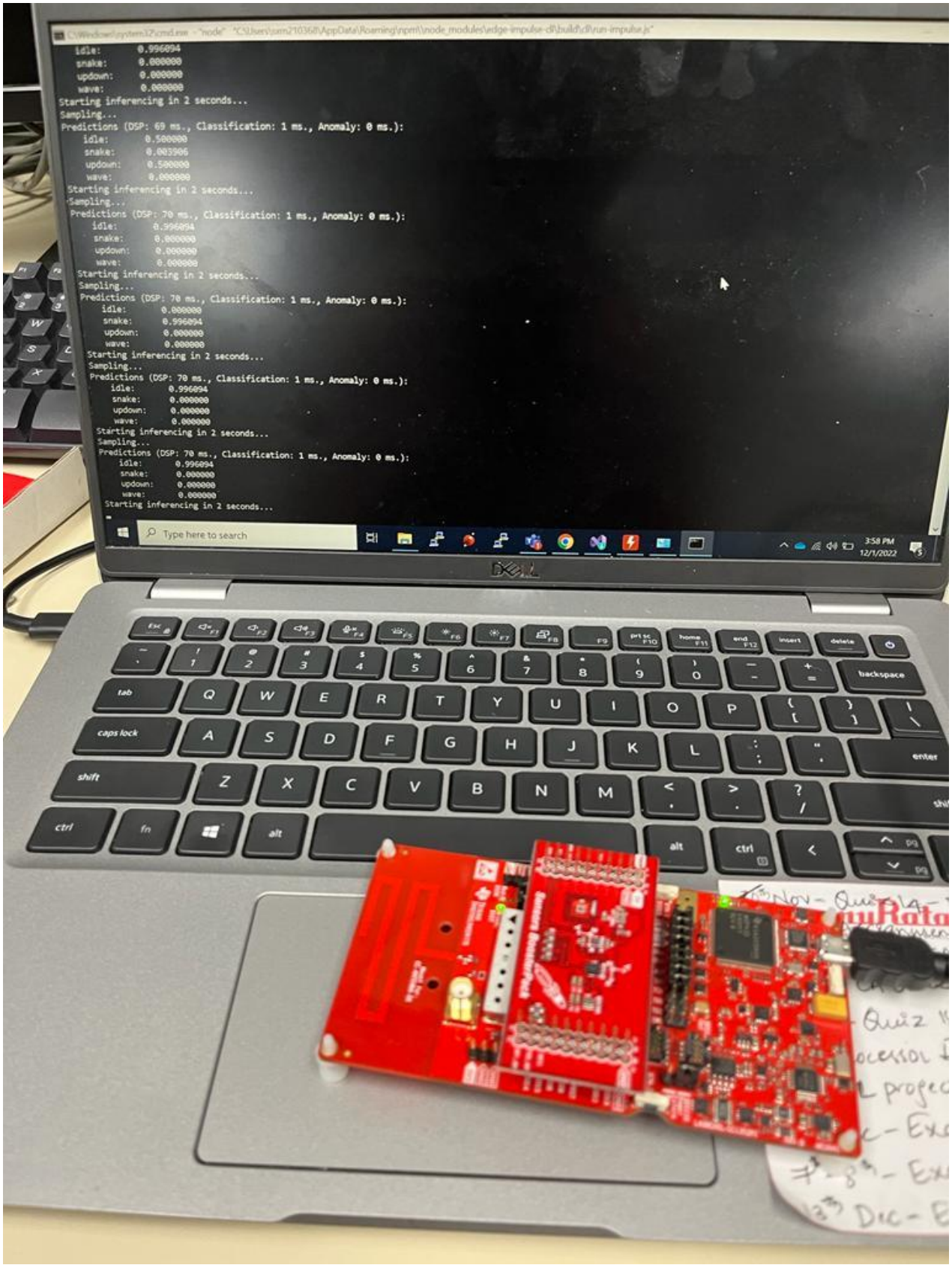


Testing of Wave feature



Testing of Idle feature

After live classification is tested, the code is dumped to the board and now we can use the board as a trained model to classify the input data.



Testing the trained-model the development board

From the output, we observed 100% accuracy in the results of the trained model.