

# GESTURE RECOGNITION

# REAL WORLD APPLICATIONS

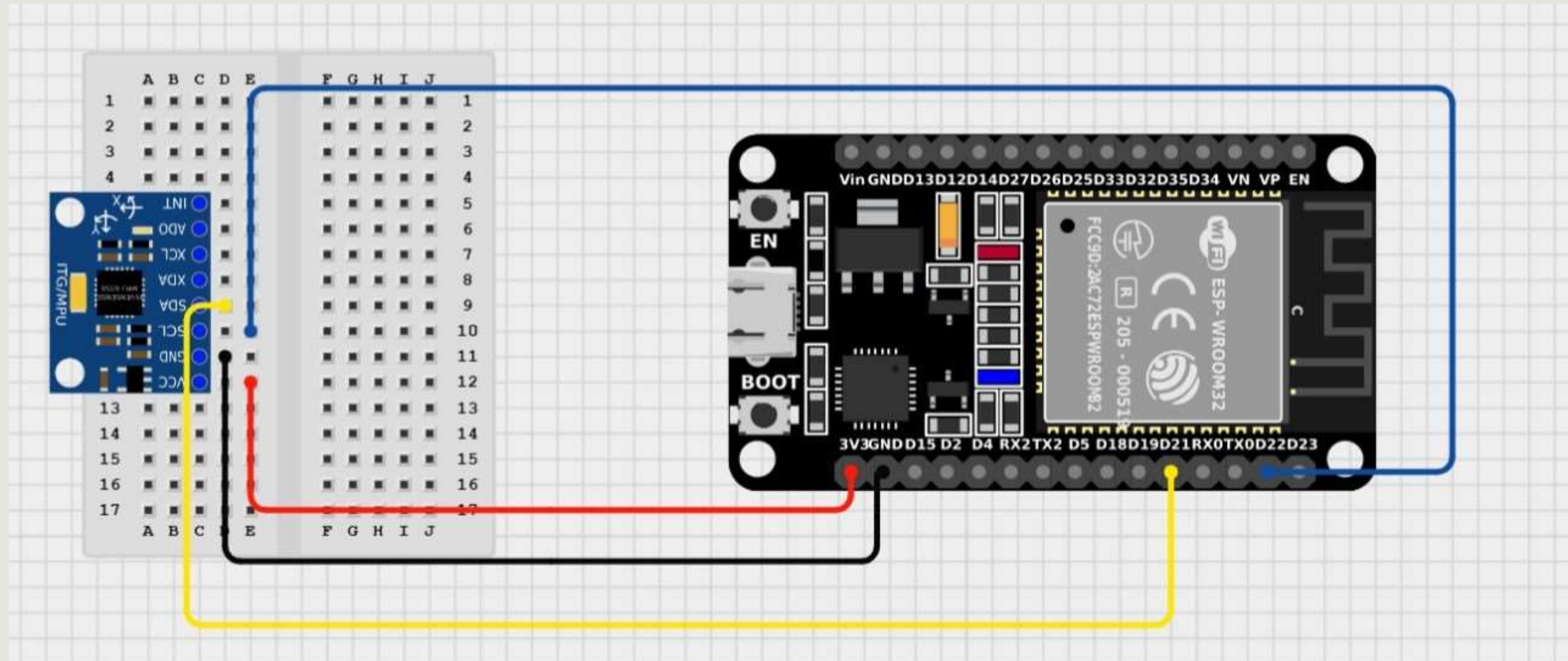
- **Human-Computer Interaction (HCI):** Control devices like smart TVs, gaming consoles, and VR systems using natural hand gestures without physical controllers.
- **Retail and Marketing:** Analyze customer behavior in stores by tracking gestures to optimize layout and product placement.
- **Virtual and Augmented Reality:** Enable immersive interactions within virtual spaces through hand and body motion tracking.
- **Robot Control:** Gesture commands help robots perform tasks and interact with humans in industries and assistive services.
- **Sign Language Recognition:** Translate sign languages in real-time, aiding communication for the deaf and mute.

# WORKING OF THE PROJECT

- Connect the MPU6050 sensor to the ESP32 and use Edge Impulse's data forwarder to send gesture data during motion.
- Label the collected data in Edge Impulse Studio by categorizing each sample according to its gesture.
- Design and train a model in Edge Impulse Studio using the labeled data with motion-optimized processing and classification blocks.
- Download the trained model and deploy it to Arduino IDE to enable real-time processing of live sensor data.
- The device now instantly detects and classifies hand gestures using new data from the MPU6050 sensor.



# PEHLE CONNECTION TOH DEKHLLO!!



# FIRMWARE CODE

THIS CODE IS USED A  
FIRMWARE TO YOUR  
ESP WHICH ENABLE  
IT TO SEND YOUR  
DATA THROUGH A  
BRIDGE CALLED  
EDGE IMPULSE  
CLI.THIS IS ALREADY  
UPLOADED IN YOUR  
ESP..

```
sketch_feb2a.ino
6
7 // 1. Include MPU6050 libraries
8 #include <Adafruit_MPU6050.h>
9 #include <Adafruit_Sensor.h>
10 #include <Wire.h>
11
12 // 2. Set the sampling frequency (Hz)
13 // Common frequencies for gesture recognition are 62.5Hz or 100Hz
14 #define FREQUENCY_HZ 100
15 #define INTERVAL_MS (1000 / FREQUENCY_HZ)
16
17 // Globals for sensor and timing
18 Adafruit_MPU6050 mpu;
19 unsigned long last_interval_ms = 0;
20
21 void setup() {
22   Serial.begin(115200);
23
24   // Initialize the MPU6050
25   if (!mpu.begin()) {
26     Serial.println("Failed to find MPU6050 chip");
27     while (1) {
28       delay(10);
29     }
30   }
31
32   // Set a reasonable accelerometer range
33   mpu.setAccelerometerRange(MPU6050_RANGE_8_G);
34
35   Serial.println("MPU6050 Found! Ready to collect data.");
36 }
37
38 void loop() {
```

```
sketch_feb2a.ino
23
24 // Initialize the MPU6050
25 if (!mpu.begin()) {
26   Serial.println("Failed to find MPU6050 chip");
27   while (1) {
28     delay(10);
29   }
30 }
31
32 // Set a reasonable accelerometer range
33 mpu.setAccelerometerRange(MPU6050_RANGE_8_G);
34
35 Serial.println("MPU6050 Found! Ready to collect data.");
36 }
37
38 void loop() {
39   // Ensure we are sampling at the correct frequency
40   if (millis() < last_interval_ms + INTERVAL_MS) {
41     return;
42   }
43   last_interval_ms = millis();
44
45   // Get a new sensor event
46   sensors_event_t a, g, temp;
47   mpu.getEvent(&a, &g, &temp);
48
49   // Print the accelerometer data in the required format (CSV)
50   Serial.print(a.acceleration.x);
51   Serial.print(',');
52   Serial.print(a.acceleration.y);
53   Serial.print(',');
54   Serial.println(a.acceleration.z);
55 }
```

# CLI INSTALLATION

CONNECT THE ESP SETUP TO YOUR COMPUTER AND  
OPEN THE COMMAND PROMPT. THEN ENTER THE  
COMMAND

**NPM INSTALL -G EDGE-IMPULSE-CLI --FORCE**

THIS WILL SUCCESSFULLY INSTALL THE EDGE IMPULSE  
CLI IN YOUR COMPUTER WHICH WILL HELP US TO  
FORWARD DATA TO EDGE IMPULSE



# ENABLING DATA FORWARDER

NOW IN YOUR EDGE IMPULSE CREATE A NEW PROJECT. THEN OPEN THE COMMAND PROMPT AND ENTER

## EDGE-IMPULSE-DATA-FORWARDER

```
C:\Windows\system32\cmd.exe - "node" "C:\Users\admin\AppData\Roaming\npm\node_modules\edge-impulse-cli\build\cli\data-forwarder.js"

Microsoft Windows [Version 10.0.19045.6216]
(c) Microsoft Corporation. All rights reserved.

C:\Users\admin>edge-impulse-data-forwarder
Edge Impulse data forwarder v1.34.0
? What is your user name or e-mail address (edgeimpulse.com)? _khushi2005
? What is your password? [hidden]
Endpoints:
  Websocket: wss://remote-mgmt.edgeimpulse.com
  API:       https://studio.edgeimpulse.com
  Ingestion: https://ingestion.edgeimpulse.com

? Which device do you want to connect to? ( type to search) COM9
[SER] Connecting to COM9
[SER] Serial is connected (00:01)
[WS ] Connecting to wss://remote-mgmt.edgeimpulse.com
[WS ] Connected to wss://remote-mgmt.edgeimpulse.com

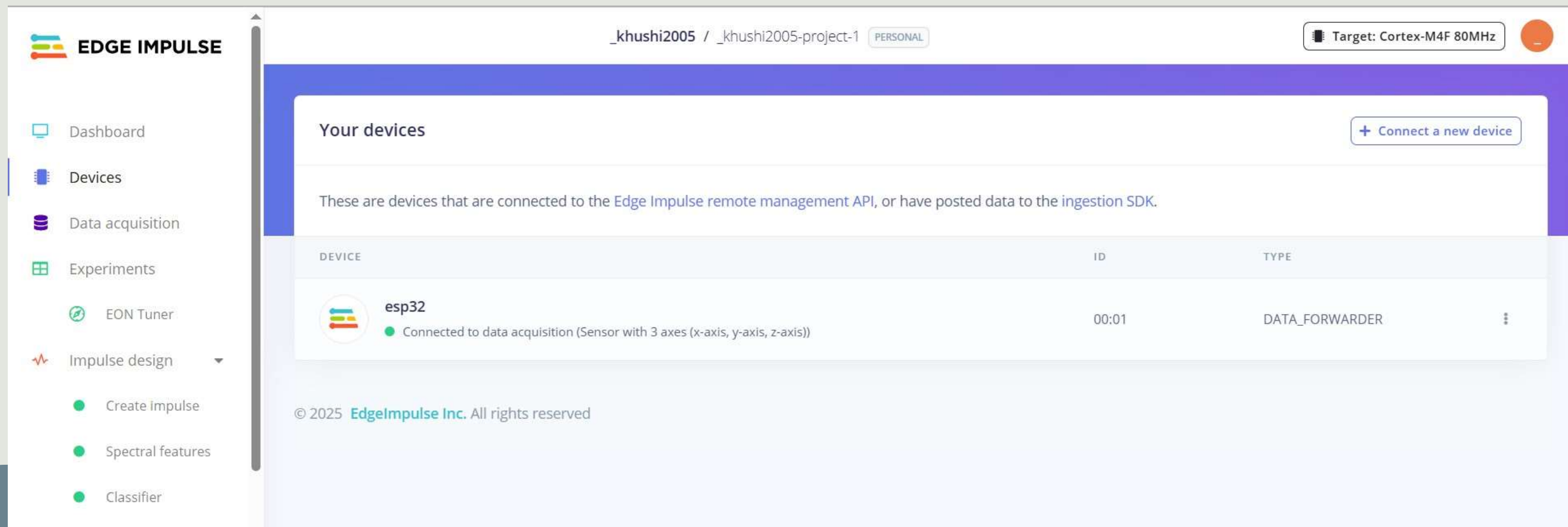
? To which project do you want to connect this device? ( type to search) 769868
[SER] Detecting data frequency...
[SER] Detected data frequency: 100Hz
? 3 sensor axes detected (example values: [0.08,-0.17,10.44]). What do you want to call them? Separate the names with ',': x-axis,y-axis,z-axis
[WS ] Device "esp32" is now connected to project "_khushi2005-project-1". To connect to another project, run `edge-impulse-data-forwarder --clean
[WS ] Go to https://studio.edgeimpulse.com/studio/769868/acquisition/training to build your machine learning model!
```

ENTER YOUR  
ID,PASSWORD,NEW  
PROJECT THAT YOU ARE  
WORKING ON AND IF  
ASKED THEN THE COM  
PORT IN WHICH ESP IS  
CONNECTED




# CONNECTED DEVICE IN EDGE IMPULSE

OPEN YOUR EDGE IMPULSE AND GO TO THE DEVICES OPTION. THERE YOU  
WILL SEE YOUR DEVICE NAME AND A GREEN DOT WITH IT



The screenshot displays the Edge Impulse web interface. On the left is a sidebar with navigation options: Dashboard, Devices, Data acquisition, Experiments, EON Tuner, and Impulse design. The 'Devices' option is selected. The main content area is titled 'Your devices' and includes a '+ Connect a new device' button. Below this, a table lists connected devices. One device, 'esp32', is shown with a green dot indicating it is connected. The table has columns for DEVICE, ID, and TYPE. At the bottom, there is a copyright notice: '© 2025 EdgeImpulse Inc. All rights reserved'.

DEVICE	ID	TYPE
 <b>esp32</b> Connected to data acquisition (Sensor with 3 axes (x-axis, y-axis, z-axis))	00:01	DATA_FORWARDER

# DATA ACQUISITION

GO TO THE DATA ACQUISITION OPTION. IN THE COLLECT DATA WINDOW, YOUR DEVICE NAME WILL BE SHOWN AUTOMATICALLY. SET THE SAMPLE LENGTH AS 10000 AND ENTER A LABEL FOR ANY GESTURE AND THEN PERFORM THAT GESTURE TO RECORD THE DATA.

The screenshot displays the Data Acquisition interface. On the left is a sidebar with navigation options: Dashboard, Devices, Data acquisition, Experiments, EON Tuner, and Impulse design. The main area has a top navigation bar with 'Dataset', 'Data explorer', 'Data sources', 'Synthetic data', 'AI labeling', and 'CSV Wizard'. Below this, there are two summary cards: 'DATA COLLECTED 12m 10s' and 'TRAIN / TEST SPLIT 75% / 25%'. The 'Dataset' section shows a table with columns: SAMPLE NAME, LABEL, ADDED, and LENGTH. The table lists several samples, all labeled 'rest', with a length of 10s. On the right, the 'Collect data' window is open. It contains fields for 'Device' (set to 'esp32'), 'Label' (set to 'rest'), 'Sample length (ms.)' (set to '10000'), and 'Sensor' (set to 'Sensor with 3 axes (x-axis, y-axis, z-axis)'). The 'Frequency' is set to '100Hz'. A blue circle highlights the 'Start sampling' button. At the bottom, there is a 'RAW DATA' section with the text 'Click on a sample to load...' and a 'Resume tutorial' button.

SAMPLE NAME	LABEL	ADDED	LENGTH
rest.63hp0deq	rest	Aug 27 2025, 1...	10s
rest.63hovtku	rest	Aug 27 2025, 1...	10s
rest.63hovc5l	rest	Aug 27 2025, 1...	10s
rest.63hoqcef	rest	Aug 27 2025, 1...	10s
rest.63hopt28	rest	Aug 27 2025, 1...	10s
rest.63hopcbj	rest	Aug 27 2025, 1...	10s
rest.63hoorre	rest	Aug 27 2025, 1...	10s

# DATA ACQUISITION

PROPERLY LABEL ALL THE DATA AND CREATE ATLEAST THREE CLASSES SUCH AS UP-DOWN, SIDE AND REST. ENSURE THAT MINIMUM 3 MINUTES OF DATA IS PRESENT IN EACH CLASS. ALSO ENTER SOME DATA IN THE TEST WINDOW. THE DATA SPLIT FOR EACH CLASS SHOULD BE MINIMUM 80/20 FOR TRAIN/TEST SPLIT

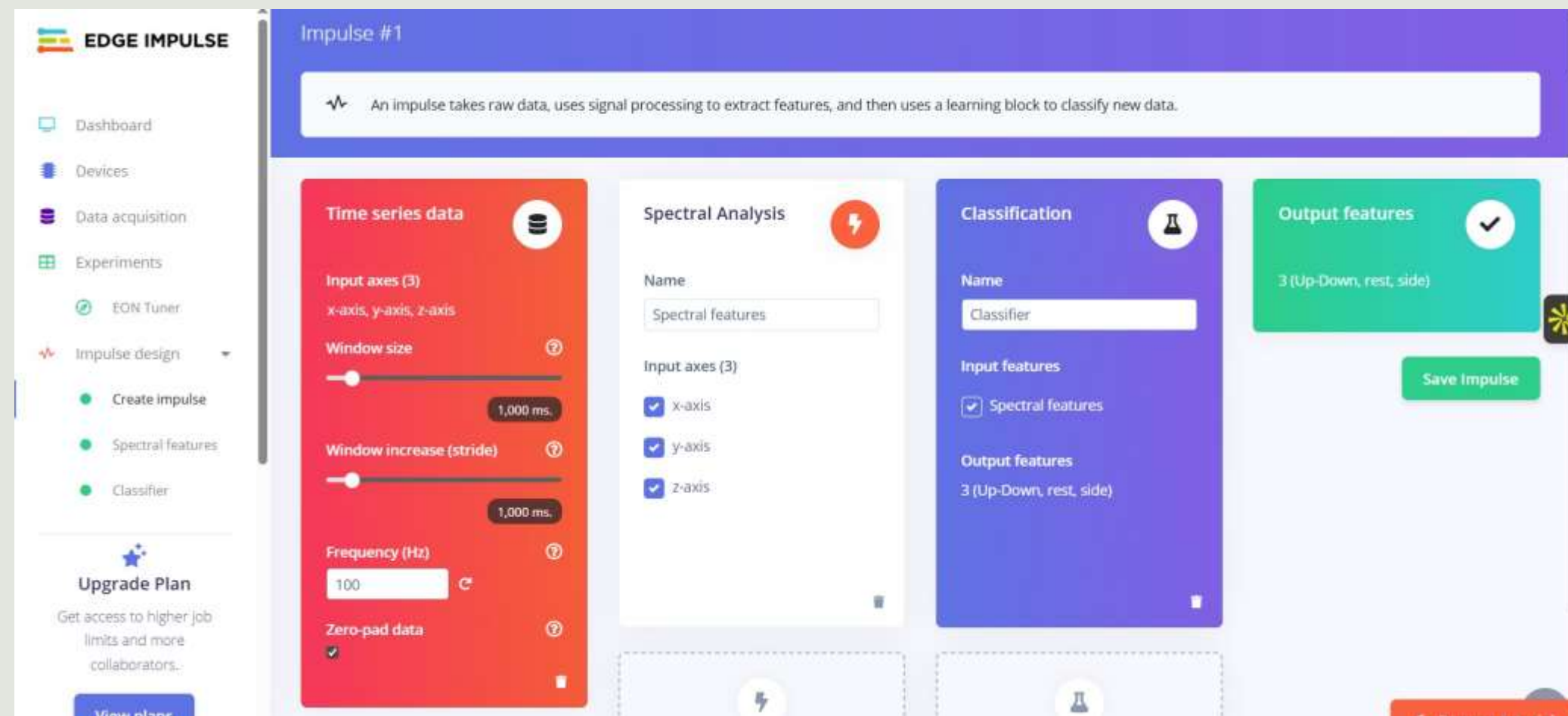
The screenshot displays the Edge Impulse Studio web interface. The top navigation bar includes links for Dataset, Data explorer, Data sources, Synthetic data, AI labeling, and CSV Wizard. The main dashboard shows 'DATA COLLECTED 12m 10s' and 'TRAIN / TEST SPLIT 75% / 25%'. The 'Dataset' section lists training samples (55) and test samples (118). A table of training samples is shown below.

SAMPLE NAME	LABEL	ADDED	LENGTH
side.63ho95eo	side	Aug 27 2025, 1...	10s
side.63ho8l7v	side	Aug 27 2025, 1...	10s
side.63ho83kt	side	Aug 27 2025, 1...	10s
side.63ho7er7	side	Aug 27 2025, 1...	10s
side.63ho6p8b	side	Aug 27 2025, 1...	10s
side.63ho6651	side	Aug 27 2025, 1...	10s
side.63ho5ku7	side	Aug 27 2025, 1...	10s
side.63ho53tr	side	Aug 27 2025, 1...	10s
side.63ho4ih9	side	Aug 27 2025, 1...	10s

The 'Collect data' panel on the right shows settings for 'Device' (esp32), 'Label' (rest), 'Sample length (ms.)' (10000), 'Sensor' (Sensor with 3 axes (x-axis, y-axis, z-axis)), and 'Frequency' (100Hz). A 'Start sampling' button is visible. Below this, a 'RAW DATA' section shows a waveform for 'side.63ho95eo' with a 'Resume tutorial' button.

# CREATING IMPULSE BLOCK

AFTER ALL THE DATA IS COLLECTED, CLICK ON THE CREATE NEW IMPULSE OPTION. CLICK ON THE ADD A PROCESSING BLOCK AND ADD 'SPECTRAL FEATURES' AS THE PROCESSING BLOCK. THEN CLICK ON ADD LEARNING BLOCK AND ADD 'CLASSIFIER' AS THE LEARNING BLOCK. THEN JUST SAVE THE IMPULSE





# SPECTRAL ANALYSIS AND FEATURE GENERATION

CLICK ON THE SPECTRAL FEATURES OPTION AND CLICK ON THE AUTOTUNE PARAMETERS. AFTER THE PARAMETERS ARE SET, JUST SAVE THE PARAMETERS.

The screenshot displays the Edge Impulse web interface for spectral analysis. The left sidebar contains navigation links: Dashboard, Devices, Data acquisition, Experiments, EON Tuner, Impulse design, Create impulse, Spectral features, and Classifier. The main panel is titled 'Parameters' and is divided into 'Filter' and 'Analysis' sections. The 'Filter' section includes fields for 'Scale axes' (0.042301185525321264), 'Input decimation ratio' (3), 'Type' (low), 'Cut-off frequency' (13.28125), and 'Order' (6). The 'Analysis' section includes 'Type' (FFT), 'FFT length' (32), 'Take log of spectrum?' (checked), 'Overlap FFT frames?' (unchecked), and 'Improve low frequency resolution?' (unchecked). A blue circle highlights the 'Autotune parameters' button at the top of the 'Parameters' section. Another blue circle highlights the 'Save parameters' button at the bottom of the 'Parameters' section. To the right of the parameter settings are three plots: a frequency response plot (dB vs Frequency (Hz)), an 'After filter' plot (Value vs Sample #), and a 'Spectral power (log)' plot (Energy vs Sample #). A 'Resume tutorial' button is located at the bottom right of the plots.

# SPECTRAL ANALYSIS AND FEATURE GENERATION

YOU WILL BE REDIRECTED TO THE FEATURE GENERATION WINDOW WHERE YOU WILL JUST VERIFY YOUR DATA PARAMETERS AND GENERATE THE FEATURES OF YOUR DATA.

The screenshot displays the Edge Impulse web interface. On the left is a sidebar with navigation links: Dashboard, Devices, Data acquisition, Experiments, EON Tuner, and Impulse design. Below these is an 'Upgrade Plan' section. The main area is titled 'Parameters' and 'Generate features'. It contains a 'Training set' section with the following details: Data in training set (9m 10s), Classes (3 (Up-Down, rest, side)), Training windows (550), Calculate feature importance (unchecked), and Normalize features (set to 'Don't normalize data'). A 'Generate features' button is present. To the right is a 'Feature explorer' section showing a scatter plot with three clusters: 'Up-Down' (blue), 'rest' (orange), and 'side' (green). At the bottom, there is a 'Feature generation output' section showing 0 results and an 'On-device performance' section with a 'Resume tutorial' button.

EDGE IMPULSE

\_khushi2005 / \_khushi2005-project-1 PERSONAL Target: Cortex-M4F 80MHz

Parameters Generate features

Training set

Data in training set 9m 10s

Classes 3 (Up-Down, rest, side)

Training windows 550

Calculate feature importance ☐

Normalize features ? Don't normalize data

Generate features

Feature explorer

Up-Down rest side

Feature generation output (0)

On-device performance ? Resume tutorial

# TRAINING OF THE MODEL

GO TO THE TRANSFER LEARNING OPTION AND SET THE NO. OF TRAINING CYCLES AS 30 AND THEN CLICK ON SAVE AND TRAIN. AFTER SOME TIME THE TRAINING STATISTICS WILL BE SHOWN WHICH CAN BE ANALYSED

The screenshot displays the Edge Impulse Studio web interface. On the left is a navigation sidebar with options: Dashboard, Devices, Data acquisition, Experiments, EON Tuner, Impulse design (selected), Create impulse, Spectral features, and Classifier. The main area is divided into two panels. The left panel contains training configuration settings: Learning rate (0.0005), Training processor (CPU), and a Neural network architecture diagram. The architecture consists of an Input layer (54 features), two Dense layers (20 and 10 neurons), an 'Add an extra layer' button, and an Output layer (3 classes). A 'Save & train' button is at the bottom. The right panel shows the results of a completed training job. It includes a 'Model' section with 'Model version: Quantized (int8)', a 'Last training performance' section showing 98.2% Accuracy and 0.07 Loss, and a 'Confusion matrix' table. Below these are 'Metrics' for the validation set, including an Area under ROC Curve of 1.00.

**EDGE IMPULSE**

Learning rate 0.0005

Training processor CPU

Advanced training settings

Neural network architecture

Input layer (54 features)

Dense layer (20 neurons)

Dense layer (10 neurons)

Add an extra layer

Output layer (3 classes)

Save & train

Model training complete

Model training complete

Job completed (success)

Model Model version: Quantized (int8)

Last training performance (validation set)

ACCURACY 98.2%

LOSS 0.07

Confusion matrix (validation set)

	UP-DOWN	REST	SIDE
UP-DOWN	100%	0%	0%
REST	0%	100%	0%
SIDE	4.9%	0%	95.2%
F1 SCORE	0.96	1.00	0.98

Metrics (validation set)

METRIC	VALUE
Area under ROC Curve	1.00
Weighted average Precision	0.98

# TESTING OF THE MODEL

YOU CAN CLICK ON THE MODEL TESTING OPTION TO TEST THE MODEL PREDICTIONS AND ITS ACCURACY

The screenshot displays the EdgeImpulse Studio web interface at the URL `studio.edgeimpulse.com/studio/769868/impulse/1/validation`. The interface is divided into several sections:

- Left Sidebar:** Contains navigation options: Data acquisition, Experiments, EON Tuner, Impulse design (with a dropdown), Retrain model, Live classification, Model testing (highlighted with a red checkmark), Deployment, and Versioning. An "Upgrade Plan" section is at the bottom.
- Test data:** A section with a "Classify all" button and instructions: "Set the 'expected outcome' for each sample to the desired outcome to automatically score the impulse." Below this is a table with columns: SAMPLE NA..., EXPECTED OUT..., LENG..., ACCURACY, and RESULT.
- Model testing output:** A log area showing the progress of the model testing, including messages like "Classifying data for Classifier...", "Job scheduled at 29 Aug 2025 17:44:52", and "Job completed (success)".
- Results:** A section showing the model version as "Unoptimized (float32)" and an overall accuracy of 90.00%.
- Metrics for Classifier:** A table listing various performance metrics and their values.

SAMPLE NA...	EXPECTED OUT...	LENG...	ACCURACY	RESULT
testing.63...	testing	10s		10 Up-Down
testing.63...	testing	10s		10 Up-Down
testing.63...	testing	10s		10 Up-Down
testing.63...	testing	10s		10 Up-Down
Up-Down....	Up-Down	10s	100%	10 Up-Down
Up-Down....	Up-Down	10s	100%	10 Up-Down
rest.63hpo...	rest	10s	0%	10 Up-Down

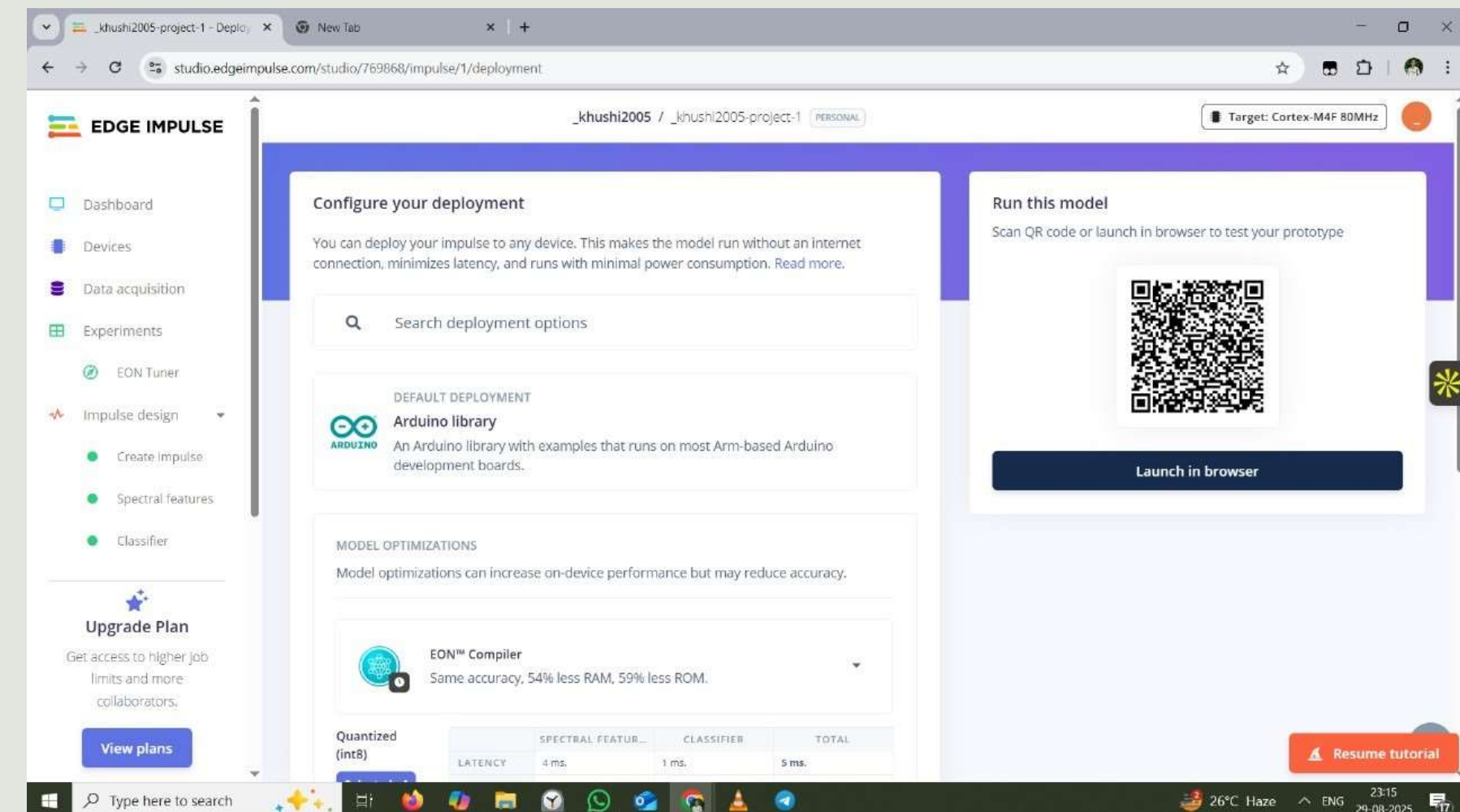
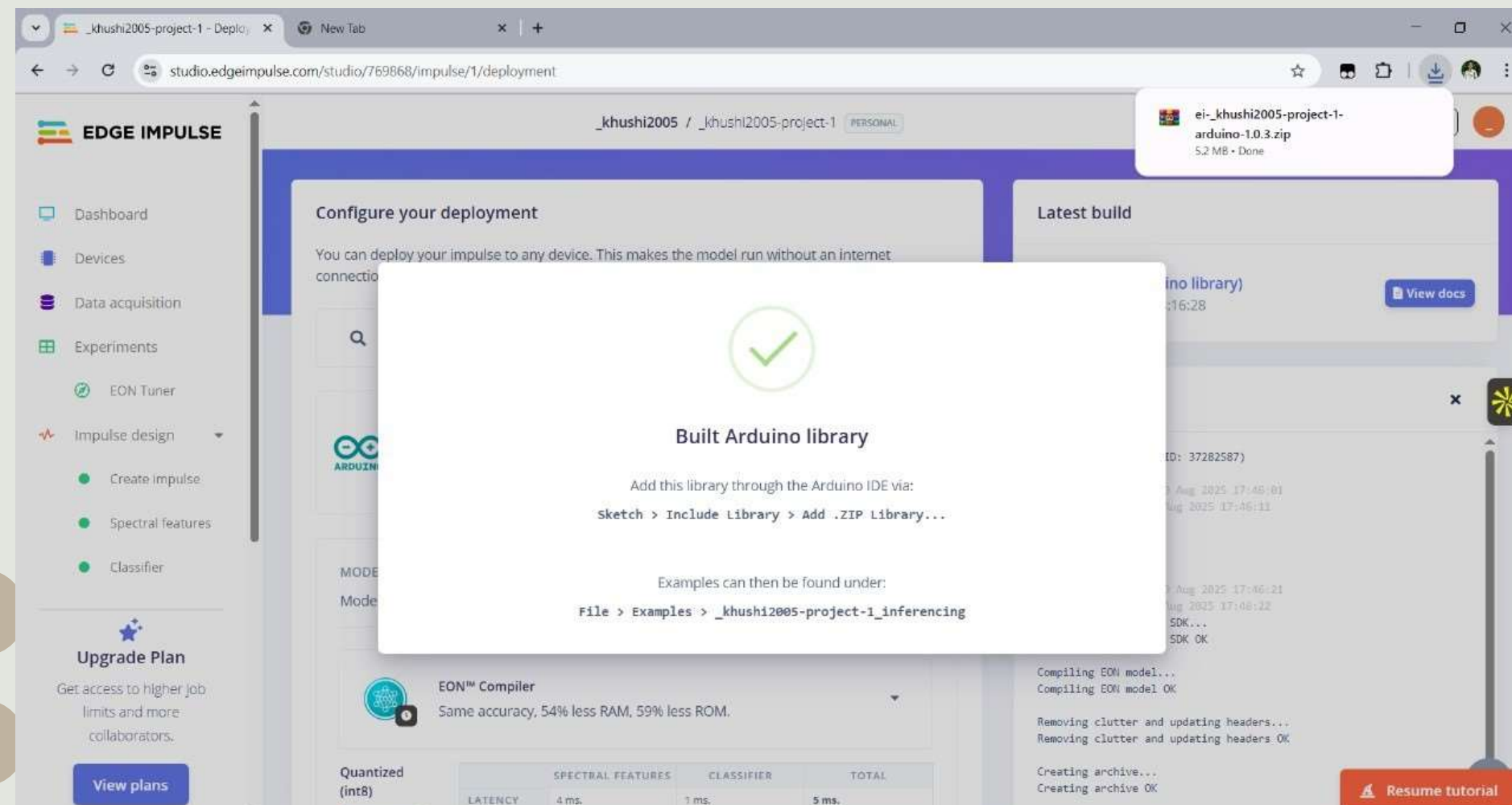
METRIC	VALUE
Area under ROC Curve ?	1.00
Weighted average Precision ?	0.94
Weighted average Recall ?	0.91
Weighted average F1 score ?	0.91

At the bottom right, there is a "Resume tutorial" button.



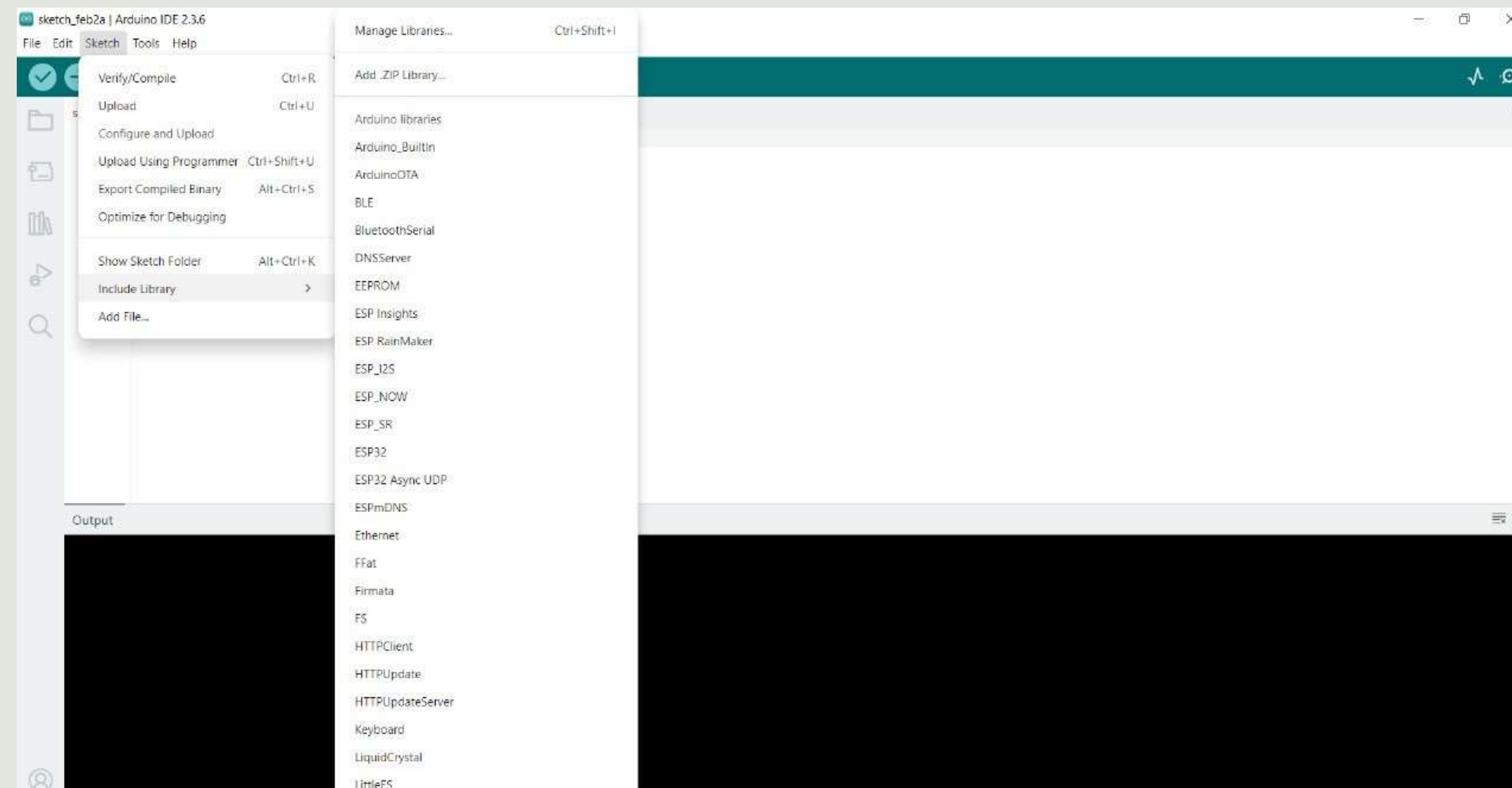
# DEPLOYMENT OF MODEL

DEPLOY THE MODEL USING THE MODEL DEPLOYMENT TAB AND SELECT THE DEFAULT DEPLOYMENT AS ARDUINO LIBRARY, WHICH WILL DOWNLOAD A LIBRARY FILE INTO YOUR LAPTOP



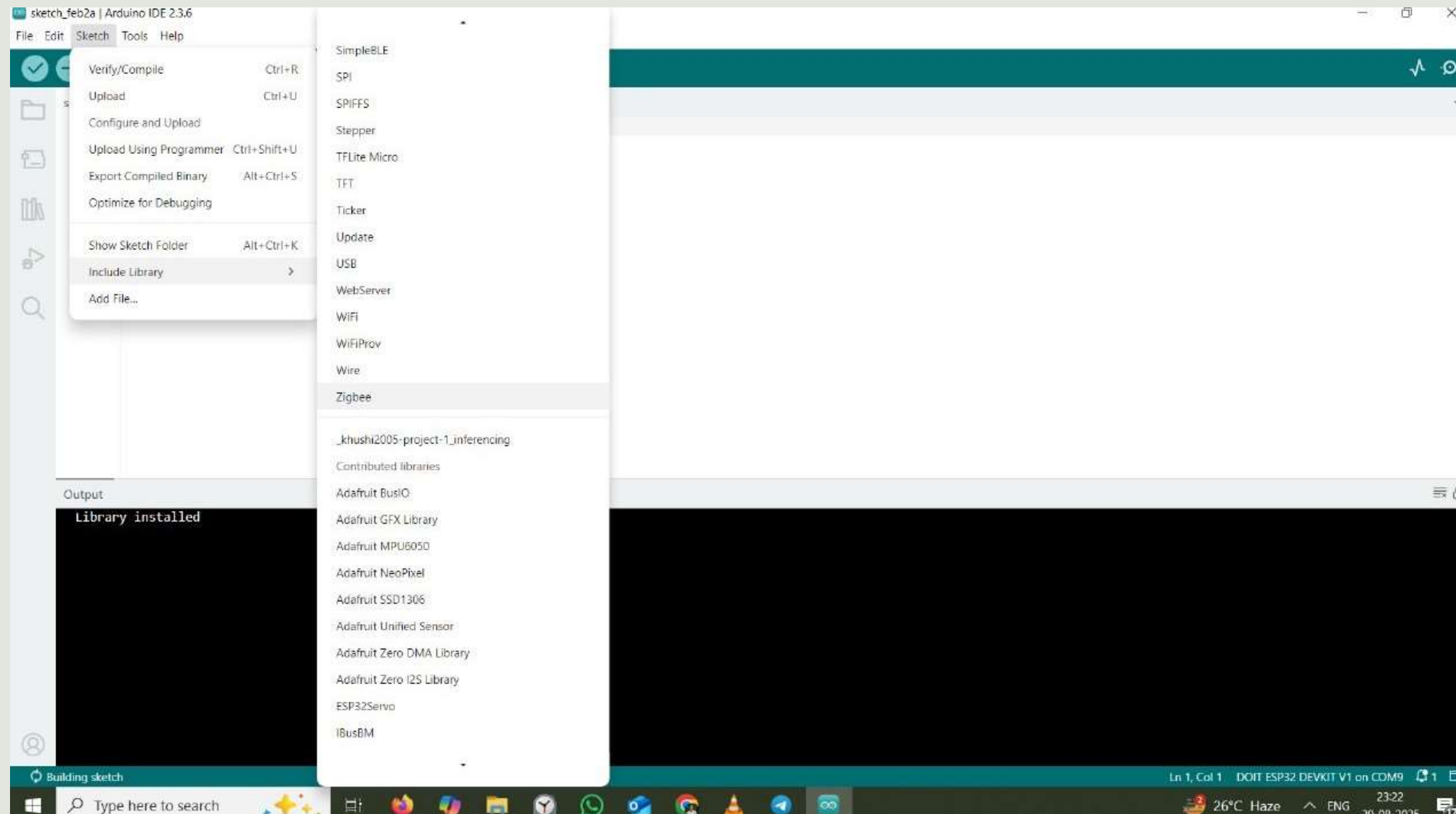
# LIBRARY INSTALLATION IN ARDUINO IDE

IN ARDUINO IDE, CLICK ON SKETCH, THEN INCLUDE LIBRARIES AND ON THE TOP, CLICK ON ADD .ZIP LIBRARY AND SELECT YOUR DOWNLOADED ZIP FILE. THIS WILL INSTALL THE MODEL LIBRARY ON YOUR ARDUINO IDE.



# USING THE MODEL

AFTER YOUR LIBRARY IS INSTALLED, ONCE AGAIN GO TO SKETCH AND INCLUDE LIBRARY. THEN CLICK ON YOUR MODEL LIBRARY.



# FURTHER CODE AND USE

WRITE THIS CODE WITH  
YOUR LIBRARY. THIS  
CODE WILL EXTRACT  
DATA FROM YOUR  
MPU6050 AND THEN  
CLASSIFY IT AND  
PREDICT THE GESTURE  
ON THE SERIAL  
MONITOR

```
Sketch_002a | Arduino IDE 2.3.6
File Edit Sketch Tools Help
DOIT ESP32 DEVKIT V1
Sketch_002a.ino
1 #include <Arduino.h>
2 #include <Wire.h>
3 #include <Adafruit_MPU6050.h>
4 #include <Adafruit_Sensor.h>
5
6 Adafruit_MPU6050 mpu;
7
8 // Buffer to hold accelerometer features
9 float features[ET_CLASSIFIER_DSP_INPUT_FRAME_SIZE];
10
11 // Setup edge impulse signal callback
12 int get_data(xlar_t offset, size_t length, float *out_ptr) {
13   memcpy(out_ptr, features + offset, length * sizeof(float));
14   return 0;
15 }
16
17 void setup() {
18   Serial.begin(115200);
19   pinMode(LED_PIN, OUTPUT);
20   digitalWrite(LED_PIN, LOW);
21
22   // Initialize MPU6050
23   if (!mpu.begin()) {
24     Serial.println("Failed to find MPU6050 chip");
25     while (1) {
26       delay(10);
27     }
28   }
29
30   mpu.setAccelerometerRange(MPU6050_RANGE_2_G);
31   mpu.setFilterBandwidth(MPU6050_BAND_5_HZ);
32   Serial.println("MPU6050 initialized");
33 }
```

Output

Library installed

```
Sketch_002a | Arduino IDE 2.3.6
File Edit Sketch Tools Help
DOIT ESP32 DEVKIT V1
Sketch_002a.ino
47 delay(40); // ~250Hz sampling rate (match training config)
48
49 // Prepare signal for classifier
50 signal_t signal;
51 signal.total_length = ET_CLASSIFIER_DSP_INPUT_FRAME_SIZE;
52 signal.get_data = &get_data;
53
54 ei_impulse_result_t result = { 0 };
55
56 // Run the classifier
57 EI_IMPULSE_ERROR res = run_classifier(&signal, &result, false);
58 if (res != EI_IMPULSE_OK) {
59   Serial.println("ERR: run_classifier failed");
60   Serial.println(res);
61   Serial.println("");
62   return;
63 }
64
65 // Print classification results
66 for (size_t ix = 0; ix < ET_CLASSIFIER_LABEL_COUNT; ix++) {
67   Serial.print(result.classification[ix].label);
68   Serial.print(" ");
69   Serial.println(result.classification[ix].value, 3);
70 }
71
72 delay(500); // Delay before next classification
73 }
```

Output


Library installed

```
Sketch_002a | Arduino IDE 2.3.6
File Edit Sketch Tools Help
DOIT ESP32 DEVKIT V1
Sketch_002a.ino
33 Serial.println("MPU6050 initialized");
34 delay(1000);
35
36
37 void loop() {
38   // Collect accelerometer data (X, Y, Z) into the features buffer
39   for (int i = 0; i < ET_CLASSIFIER_DSP_INPUT_FRAME_SIZE / 3; i++) {
40     sensors_event_t a;
41     mpu.getEvent(&a, &g, &temp);
42
43     features[i * 3 + 0] = a.acceleration.x;
44     features[i * 3 + 1] = a.acceleration.y;
45     features[i * 3 + 2] = a.acceleration.z;
46
47     delay(40); // ~250Hz sampling rate (match training config)
48   }
49
50   // Prepare signal for classifier
51   signal_t signal;
52   signal.total_length = ET_CLASSIFIER_DSP_INPUT_FRAME_SIZE;
53   signal.get_data = &get_data;
54
55   ei_impulse_result_t result = { 0 };
56
57   // Run the classifier
58   EI_IMPULSE_ERROR res = run_classifier(&signal, &result, false);
59   if (res != EI_IMPULSE_OK) {
60     Serial.println("ERR: run_classifier failed");
61     Serial.println(res);
62     Serial.println("");
63     return;
64   }
65 }
```

Output

Library installed





**THANK YOU!!!**

