

DsPIC33CK Predictive Maintenance in Motor Control Applications

Introduction:

In today's industrial landscape, the efficient operation of machinery is paramount. Unforeseen downtime can lead to significant losses in productivity and revenue. To mitigate such risks, predictive maintenance has emerged as a valuable solution, allowing for proactive identification of potential issues before they escalate into costly problems.

In this technical blog, we delve into the realm of predictive maintenance for industrial motors, leveraging the powerful capabilities of the Microchip MPLAB Machine Learning Development Suite. Specifically, we focus on utilizing the suite's features to build a classification model tailored for monitoring the operational state of motors in real-time.

Application User Guide:

The following sections will guide you through the process of developing a predictive maintenance application for industrial motors using the MPLAB Machine Learning Development Suite for dsPIC LVMC Motor control board. We'll start by exploring a compelling industrial use case example that underscores the significance of this technology in today's manufacturing environment. Then, we'll delve into the technical aspects, outlining the steps to build a robust data pipeline for feature extraction and train a classification model using the Machine learning development suite's AutoML feature. Following model training, we'll discuss methods for evaluating its performance within the MPLAB ML Dev Suite environment. Finally, we'll demonstrate how to convert the trained model into a Knowledge Pack that can be seamlessly deployed and tested on the dsPIC LVMC Motor control board, bringing predictive maintenance capabilities directly to the heart of industrial operations. Let's embark on this journey to harness the power of machine learning for enhanced machinery reliability and efficiency.

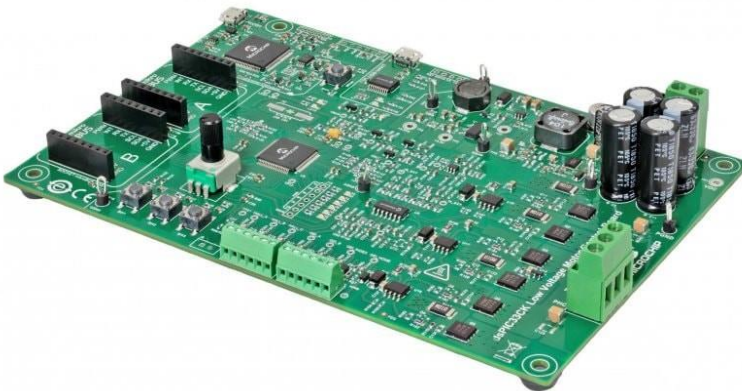
Hardware:

In our predictive maintenance setup, we utilize the following hardware components:

AC300020 - 24V 3-PHASE BRUSHLESS DC MOTOR: This motor serves as the primary equipment for our predictive maintenance application. It provides the mechanical motion that we aim to monitor and maintain.



DSPIC33CK LVMC DEVELOPMENT BOARD (Part Number: 2740-ac300020): The dsPIC33CK LVMC Development Board acts as the core controller for our system. It provides the computational power and interface necessary to interact with the motor and execute our predictive maintenance algorithms.

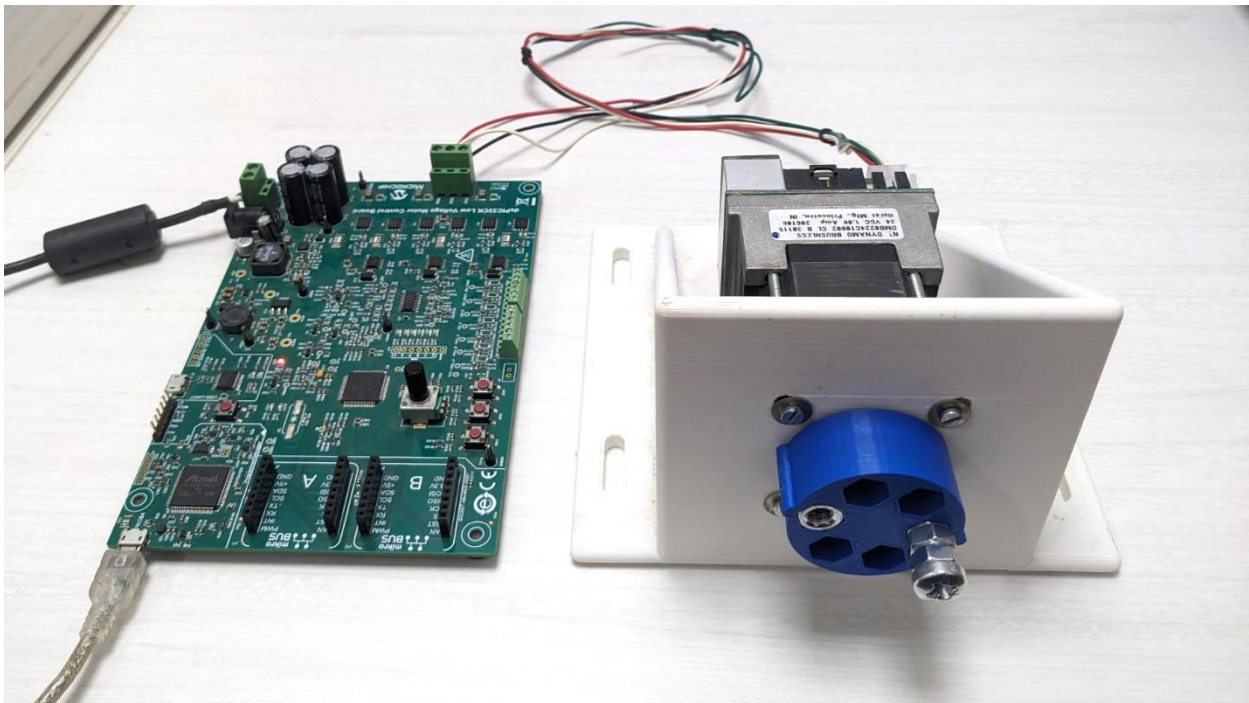


DM330031 component complements the development board and enhances its functionality for our specific application.

3D PRINTED MODULE: The 3D printed module plays a crucial role in our setup by simulating an unbalanced load scenario. It introduces controlled variations in the motor's operating conditions, allowing us to test the efficacy of our predictive maintenance algorithms under different circumstances.



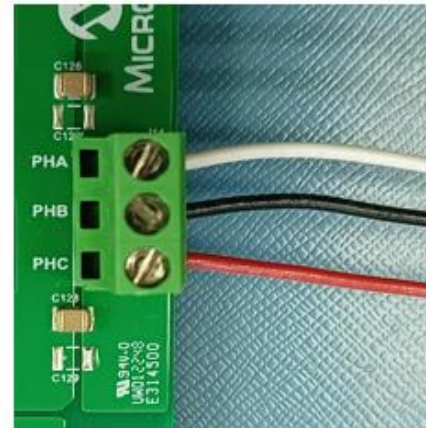
Hardware Set-Up:



To assemble the hardware components, follow these steps:

- Connect the 3-phase wires from the motor to the connector J14, provided on the dsPIC33CK LVMC Motor Control Development Board, in the specified order as shown below:

LVMC Board	Hurst075
J14 Connector	Phase Terminals
PHA	White
PHB	Black
PHC	Red



[Diagram depicting the specified order]

Ensure that the connections are secure and properly aligned to facilitate smooth communication between the motor and the development board. With the hardware set up in place, we are ready to proceed with configuring the software components and implementing our predictive maintenance solution.

Software Tools:

This project has been verified to work with the following software tool versions:

- [MPLAB X IDE 6.15](#) and above
- [MPLAB ML Development Suite](#)
 - [MPLAB® ML Data Collector 1.0](#)
 - [MPLAB® ML Model Builder 1.0](#)
- [XC16: 2.00](#)
- [MPLAB Code Configurator: 5.3.7](#)
- [MCC Core Version: 2.3.3](#)
- [motorBench® Development Suite: 2.45](#)
- DsPIC33CK_MP_DFP 1.10.341

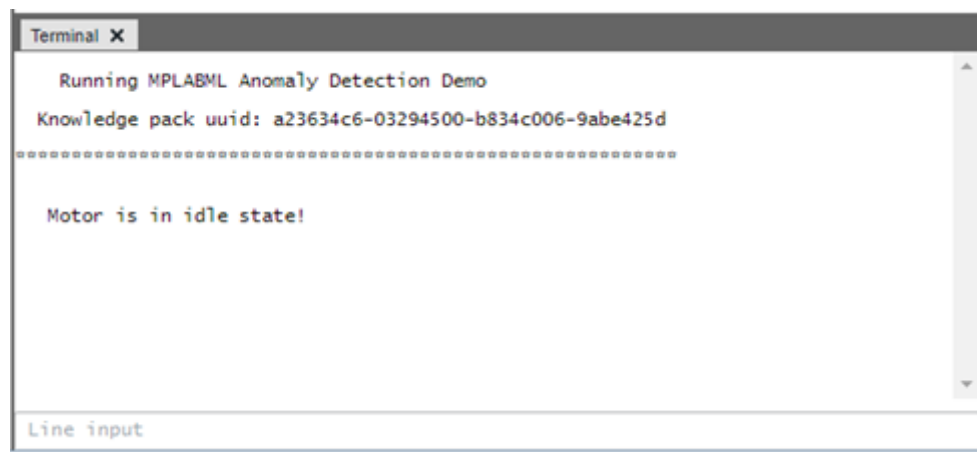
Running the Application Demo:

To run the application demo, follow these steps:

- Start MPLAB X IDE and open the project "anomaly-detection-ml-33ck256mp508-lvmc.X" with device selection dsPIC33CK256MP508.
- Set the project "anomaly-detection-ml-33ck256mp508-lvmc.X" as the main project.
- Open the "app_config.h" file located under Header Files.
- Ensure that the macro DATA_STREAMER_FORMAT is set as DATA_STREAMER_FORMAT_NONE.

```
61 | // Data streaming formatting selection
62 | #ifndef DATA_STREAMER_FORMAT
63 | #define DATA_STREAMER_FORMAT DATA_STREAMER_FORMAT_NONE
64 | #endif
```

- Open project properties and ensure that the selected MPLAB® XC16 Compiler and Device Pack support the device configured in the firmware.
- Build the project and program the device.
- Open the terminal window on Data Visualizer, select the COM port of the LVMC board, and set the baud rate to 115200.
- Below message will be printed right after device reset, indicating the Knowledge Pack number and the motor status.



- Press switch SW1 to start the motor. If the motor runs, the output below will be shown.

```
Terminal X
Running MPLABML Anomaly Detection Demo
Knowledge pack uuid: a23634c6-03294500-b834c006-9abe425d
*****

Normal Operation

Line input
```

- If an unbalanced load is detected, a corresponding message will be displayed.

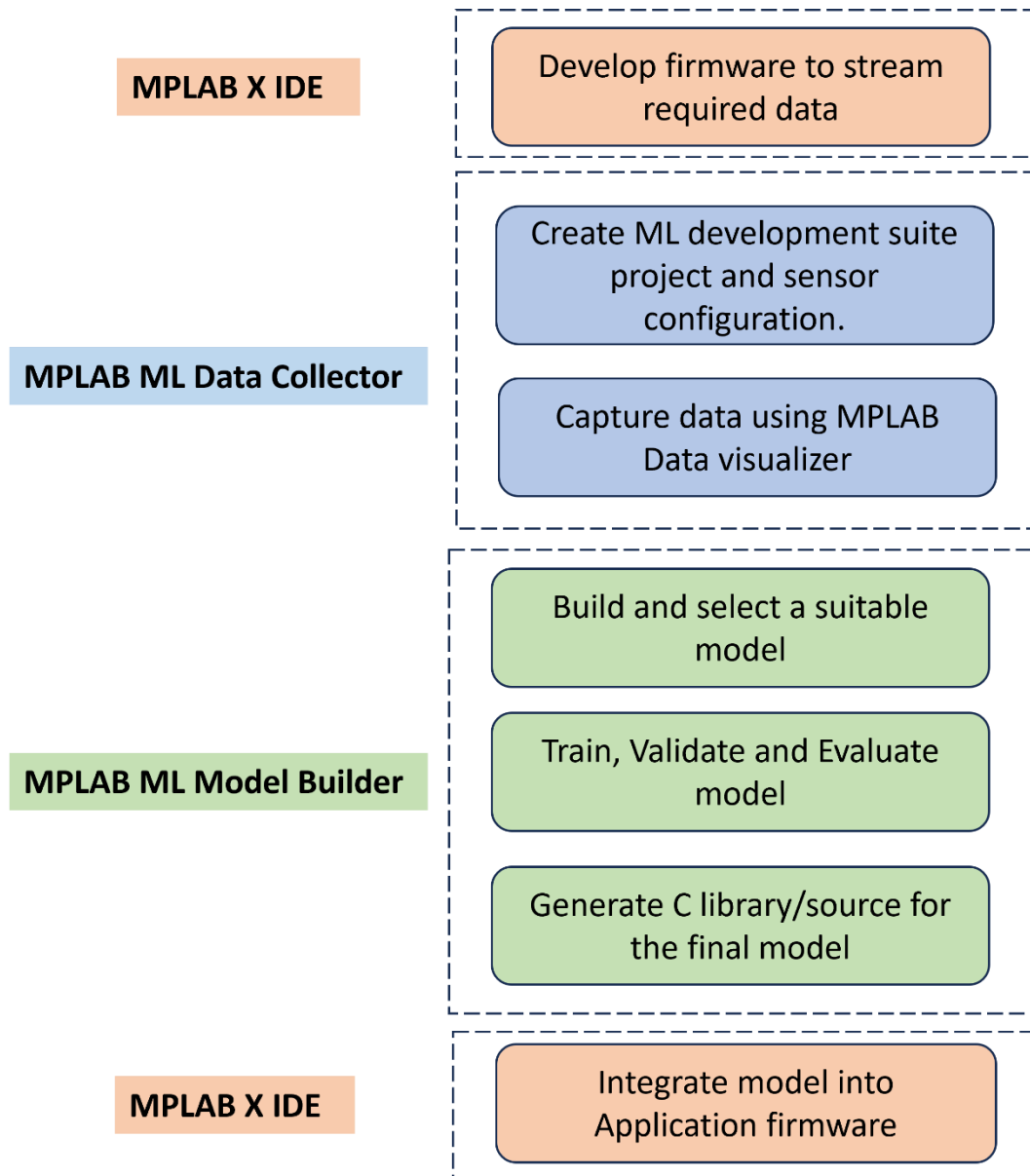
```
Terminal X
Running MPLABML Anomaly Detection Demo
Knowledge pack uuid: a23634c6-03294500-b834c006-9abe425d
*****

Unbalanced_Load

Line input
```

Note: This application project can detect anomalies up to 1200 RPM, and the motor speed has also been restricted accordingly. Users can revert to maximum speed if required. Please refer to the "mcaf_sample_application.c" file for speed modifications.

Procedure to Build ML Model from Scratch: MPLAB Machine Learning Development Suite



To develop this application, follow these steps:

- Start MPLAB X IDE and open the project "anomaly-detection-ml-33ck256mp508-lvmc.X" with device selection dsPIC33CK256MP508.
- Set the project "anomaly-detection-ml-33ck256mp508-lvmc.X" as the main project.
- Open the "app_config.h" file located under Header Files.

- Ensure that the macro `DATA_STREAMER_FORMAT` is set as `DATA_STREAMER_FORMAT_MDV`.

```
// Data streaming formatting selection
#ifdef DATA_STREAMER_FORMAT
#define DATA_STREAMER_FORMAT DATA_STREAMER_FORMAT_MDV
#endif
```

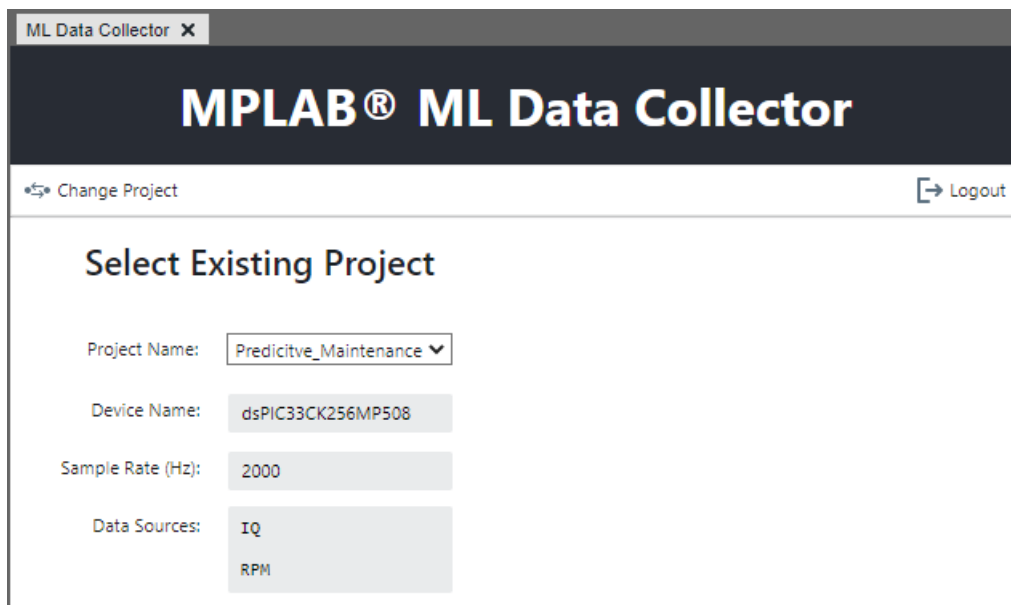
- Build and program the device. Now MPLAB Data Visualizer streaming firmware is running on the target device.

Capturing Data using MPLAB ML Data Collector:

[Getting Started](#) chapter of the MPLAB® Machine Learning Development Suite user's guide details about different configuration options. to get more details about each option.

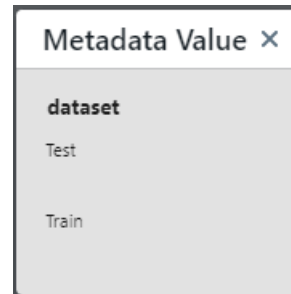
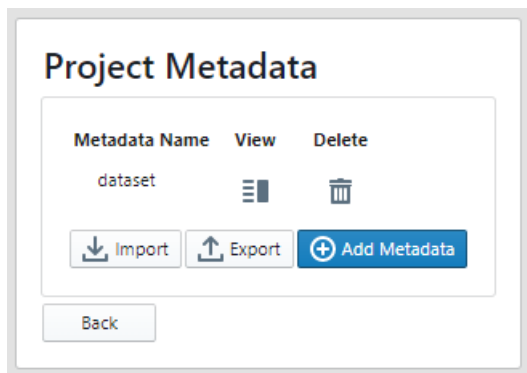
The following steps show the configuration for this specific predictive maintenance application.

- For creating new project in MPLAB ML Data Collector, please refer to the user's guide chapter [1.1.3 Creating a Project](#)



The screenshot shows the MPLAB ML Data Collector web interface. At the top, there is a dark header with the text "MPLAB® ML Data Collector". Below the header, there is a navigation bar with a "Change Project" link and a "Logout" button. The main content area is titled "Select Existing Project". It contains four configuration fields: "Project Name" with a dropdown menu showing "Predictive_Maintenance", "Device Name" with a text input showing "dsPIC33CK256MP508", "Sample Rate (Hz)" with a text input showing "2000", and "Data Sources" with a list box showing "IQ" and "RPM".

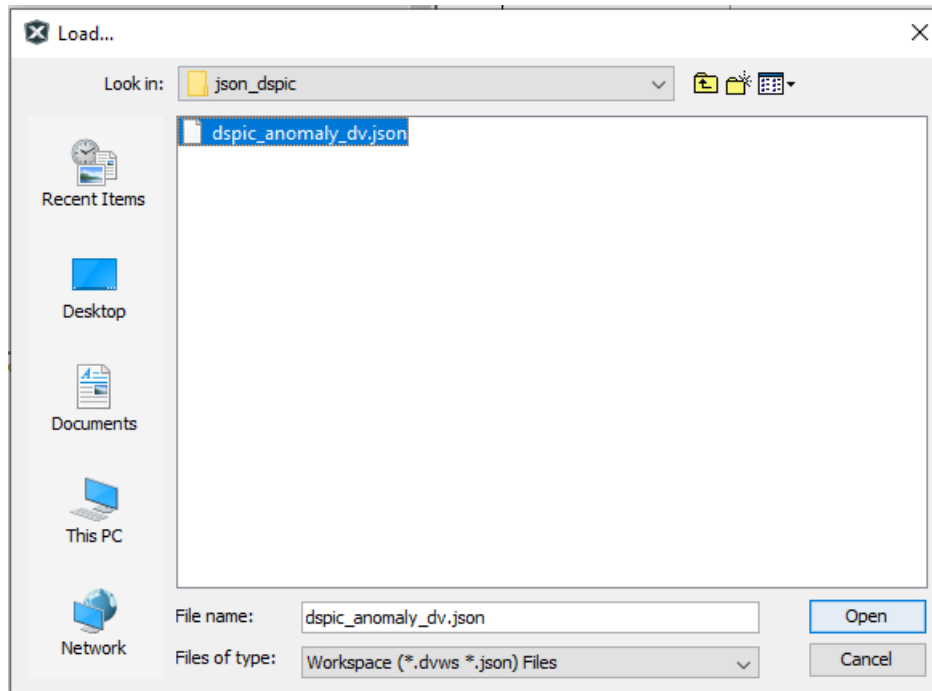
- Use the metadata to segregate data to train the ML model and to test the generated ML model.



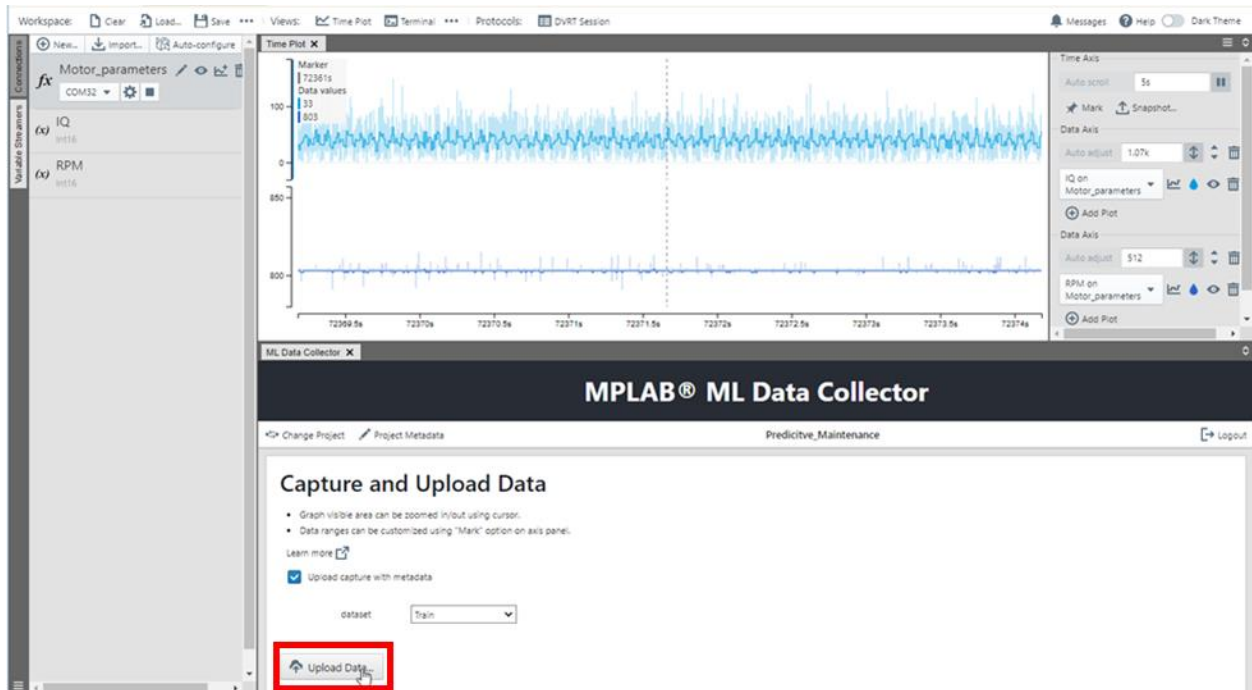
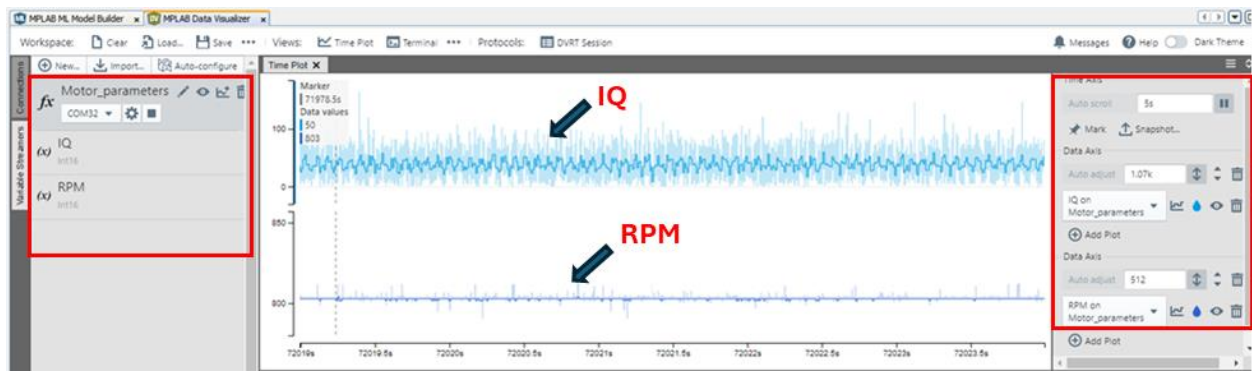
2. Click the Load option at the top of the Data Visualizer windowpane.



- Navigate to the Json_dspic folder, click the "dspic_anomaly_dv.json" file, which consists of 2 variables RPM and IQ. Click Open.



- Next, click “Variable Streamers”, then, under the Motor_parameters section, select the COM port that LVMC Board is using as the source.
- Click the circular ring located next to the source option, then set the baud rate to 115200.
- Click the Show live data option, which appears when the cursor is moved over the signal pane.
- Perform a 5-second streaming operation on the device to collect data for unbalanced load and normal operation. When the streaming is complete, click the Upload Data button in ML Data Collector to proceed.
- The CSV (Comma-Separated Values) file format is selected. Make sure to deselect the “Allow scientific notation” and “Include timestamps” options.
- Name the captured file, then save it with the desired file name.



Save Data Snapshot



Data Capture

Snapshot: Window start (s):
353.183095

Window end (s):
358.183095

Duration (s):
5

Sources to include:

- ☐ All sources
- ☒ IQ on Motor_parameters
- ☒ RPM on Motor_parameters
- ☐ COM14
- ☐ COM32

Previous

Next

Save Data Snapshot



File Format

Preset: ML Upload

File type: ☒ CSV

☐ JSON

CSV options: ☐ Include timestamps

☒ Header row

Data value format: ☐ Allow scientific notation ?

Decimal places:

3

Data layout: ☒ Table ?

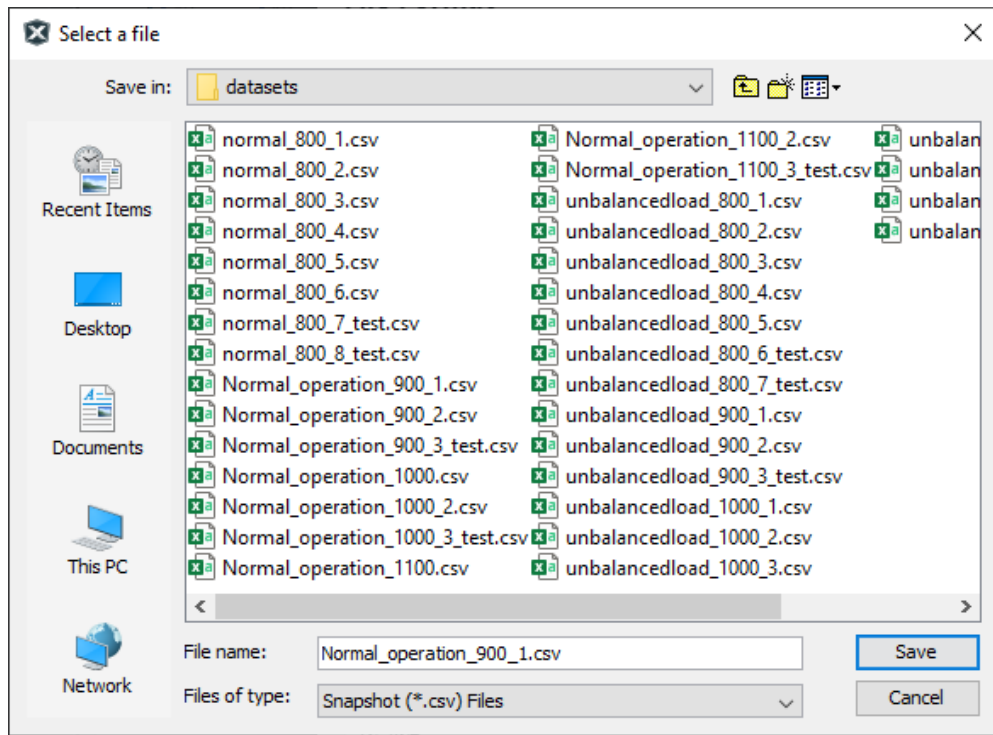
☐ Windowed ?

Preview

```
IQ,RPM
361,804
360,804
366,804
351,804
338,804
353,803
348,803
406,803
367,803
```

Previous

Save



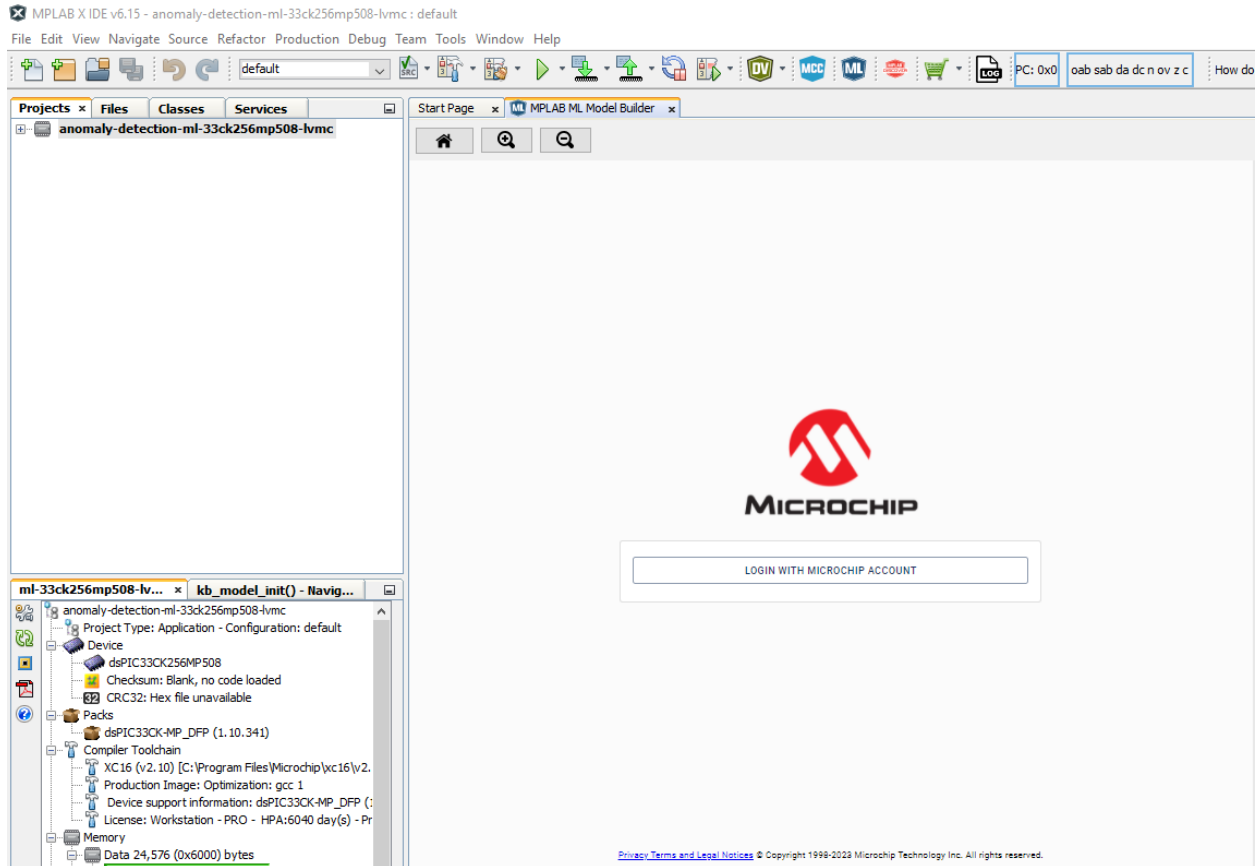
The Dataset:

The dataset that has been collected is provided in the folder named "[dataset](#)". Unzip the folder on your computer to access the data.

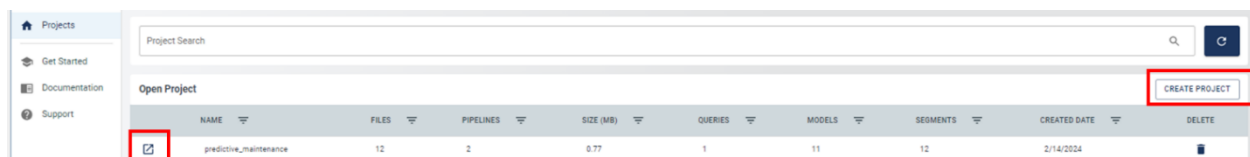
Model Development with MPLAB ML Builder:

To develop the model, follow these steps:

- Launch the MPLAB X IDE software.
- Next, launch the ML Model Builder tool.
- When the ML Model Builder page loads, users are prompted to authenticate their account by entering their myMicrochip login credentials.



- 4. Upon successful login, proceed to initiate a new project by selecting the CREATE PROJECT tab. Specify a name for the new project in the pop-up window.



- 5. Locate the project created earlier in the MPLAB ML Data Collector. Click the Open Project button to open it.
- 6. To create new labels and metadata fields for the project, follow these steps:
 - Go to the Project Summary page.
 - Click Project Settings.

- Under Segment Labels and Metadata, click the ADD LABEL and ADD METADATA buttons to fill in the required information for each new label and metadata field.

Project: Predictive_Maintenance CHANGE PROJECT

OVERVIEW QUERIES PIPELINES MODELS PROJECT SETTINGS

SEGMENT LABELS METADATA

Labels ADD LABEL

NAME	CREATED DATE	UPDATED DATE	EDIT	DELETE
unbalanced_load	1/24/2024 5:49:48 PM	1/24/2024 5:49:48 PM		
normal	1/24/2024 5:48:41 PM	1/24/2024 5:48:41 PM		

Project: Predictive_Maintenance CHANGE PROJECT

OVERVIEW QUERIES PIPELINES MODELS PROJECT SETTINGS

SEGMENT LABELS **METADATA** ADD METADATA

NAME	CREATED DATE	UPDATED DATE	VALUES	EDIT	DELETE
dataset	2/6/2024 11:18:18 AM	2/6/2024 11:18:18 AM			

- Click SAVE to create them.

- Moving to the Data Manager Module, follow these steps to import necessary data files:
 - Navigate to the Data Manager tab.
 - Click IMPORT CAPTURE FILE.
 - Navigate to the "dataset" folder from the downloaded .zip file.

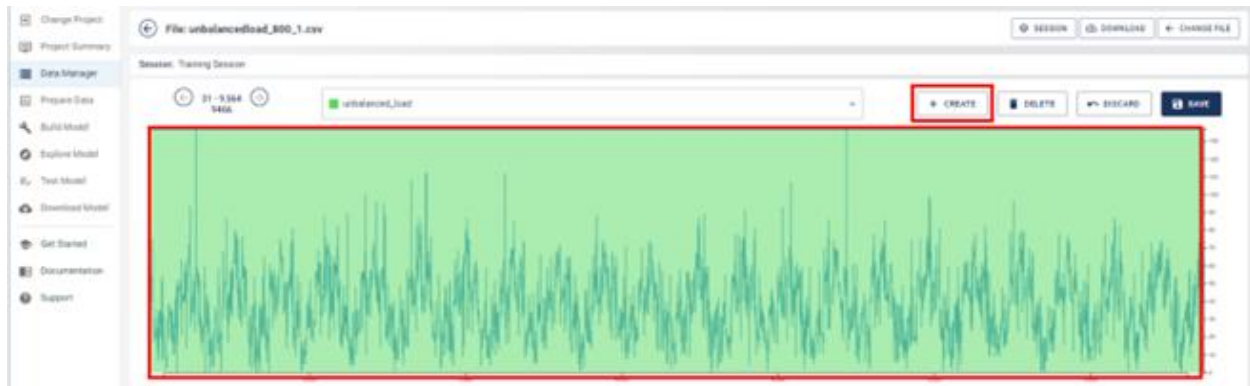
Data Manager Session: Training Session SESSION **IMPORT CAPTURE**

Captures METADATA DELETE

	CAPTURE NAME	SEGMENTS	SIZE (MB)	CREATED DATE	DATASET	OPEN	DOWNLOAD	METADATA	DELETE
<input type="checkbox"/>	normal_11_800.csv	1	0.06	2/8/2024, 10:02 PM	Test				
<input type="checkbox"/>	normal_12_800.csv	1	0.06	2/8/2024, 10:03 PM	Test				
<input type="checkbox"/>	normal_800_1.csv	1	0.06	1/25/2024, 11:31 AM	Train				
<input type="checkbox"/>	normal_800_2.csv	1	0.06	1/25/2024, 11:34 AM	Train				
<input type="checkbox"/>	normal_800_3.csv	1	0.06	1/25/2024, 11:34 AM	Train				

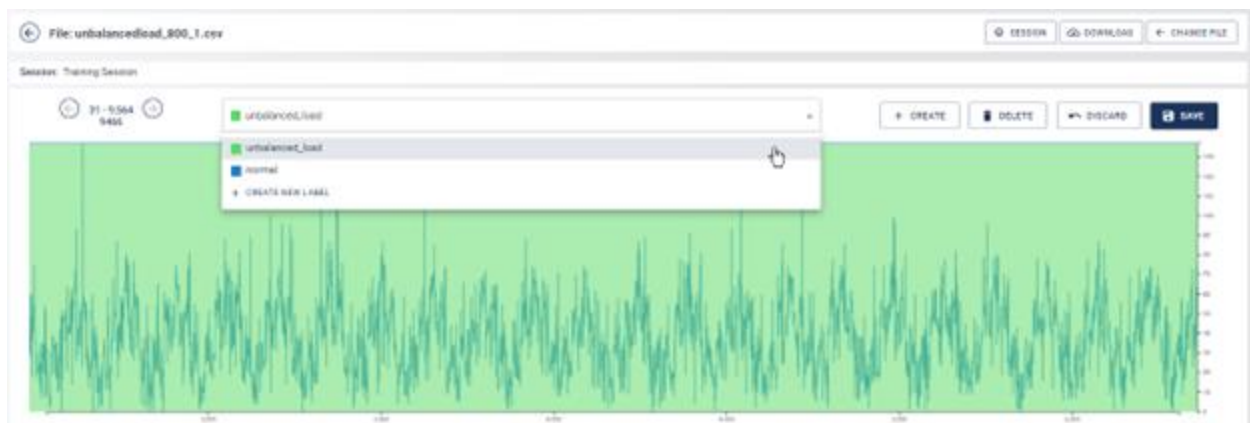
- Select all the required files for the project and begin the import process.

- Access the imported data files by clicking the icons corresponding to the desired files under the Open column. Navigate to the CREATE button to begin segmenting the data.





Note: Since IQ is a continuous signal user can just create 1 segment for the entire region of the signal that represents the state. The labeled segment can be much longer than the pipeline window size because in the Build Model step the segments will be sliced down to the defined window size appropriate for classification.

- Categorize these segments using appropriate labels and metadata fields.



- Proceed to the Prepare Data tab. Click CREATE QUERY to initiate the data preparation process. Query will be used to train machine learning models and determine what data from our dataset will be selected for training. Query parameters will select the samples only from the training fold. We can use this to exclude samples (e.g., our test samples) or exclude RPM data if required.

 Name: query_1

 The query cache is up to date with the project data.

Session

Training Session

Label

Label

Metadata


segment_uuid dataset


Source

IQ

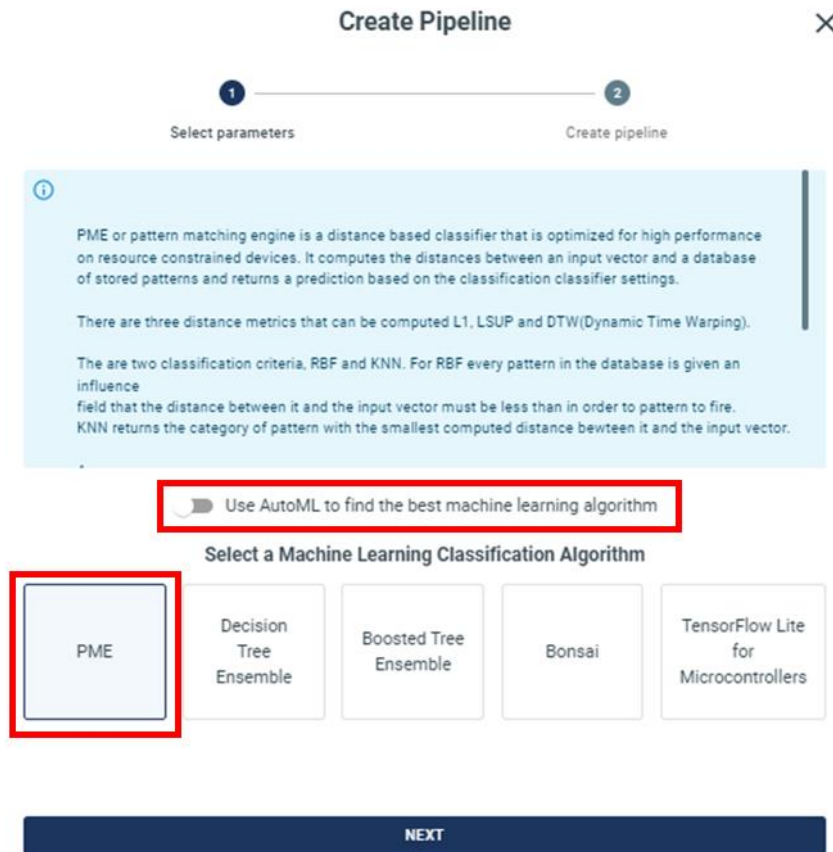
Query Filter

[dataset] IN [Train]

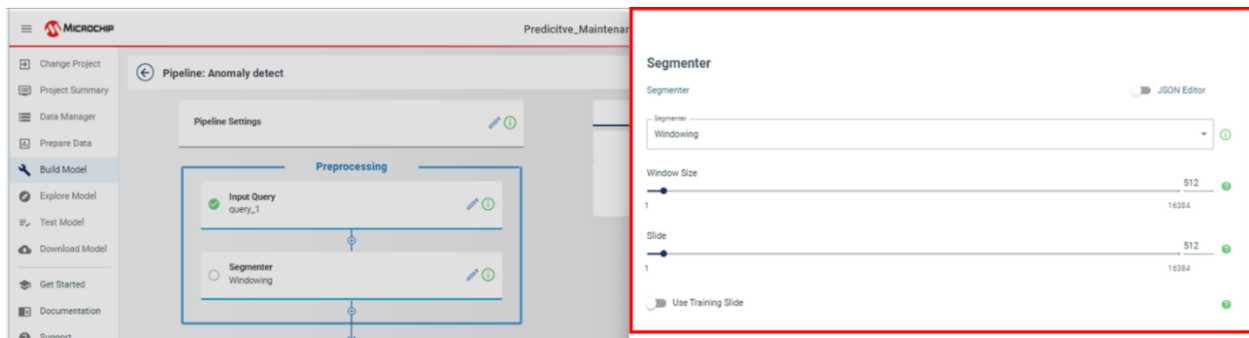
 SAVE CHANGES

 CANCEL

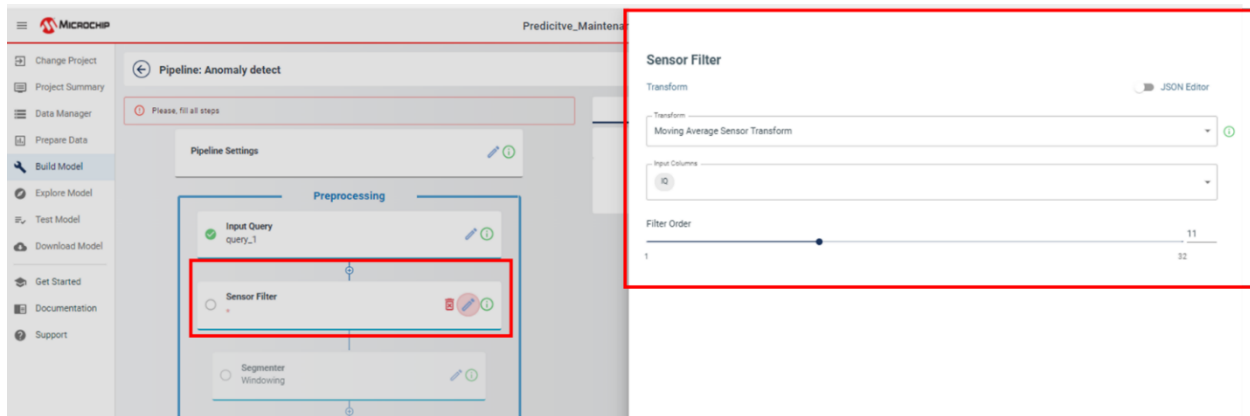
- Click the CACHE button to build out the cache completely.
- Navigate to the Build Model module, click on CREATE NEW PIPELINE. For this application, choose the Pattern Matching Engine (PME) classification algorithm. Turn off Auto ML and select the PME algorithm.



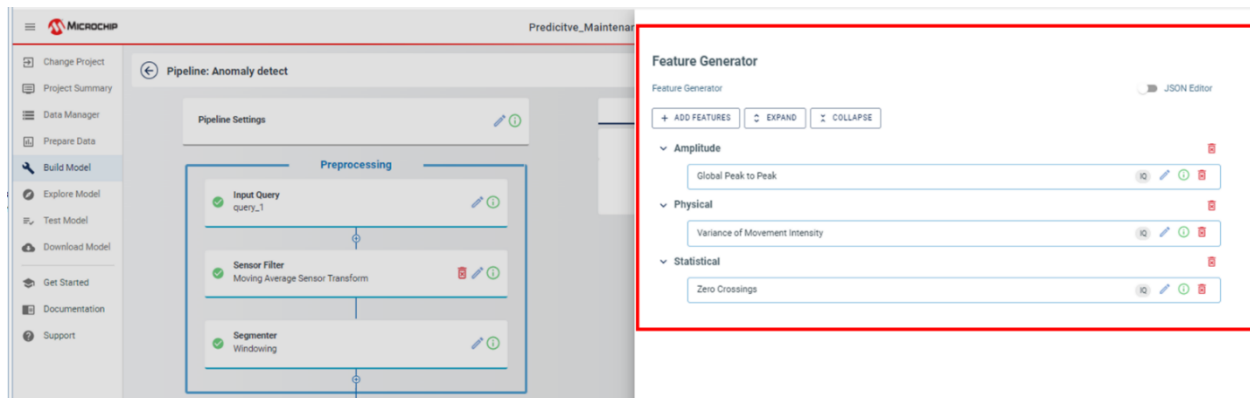
- After naming your pipeline and selecting the query created in the previous module, click **CREATE PIPELINE**. After clicking the **CREATE PIPELINE** button, another pop-up displays with the details regarding the pipeline. Click **NEXT**.
- In the Segmenter module, set the parameters for appropriate segmentation of the data and save the changes.



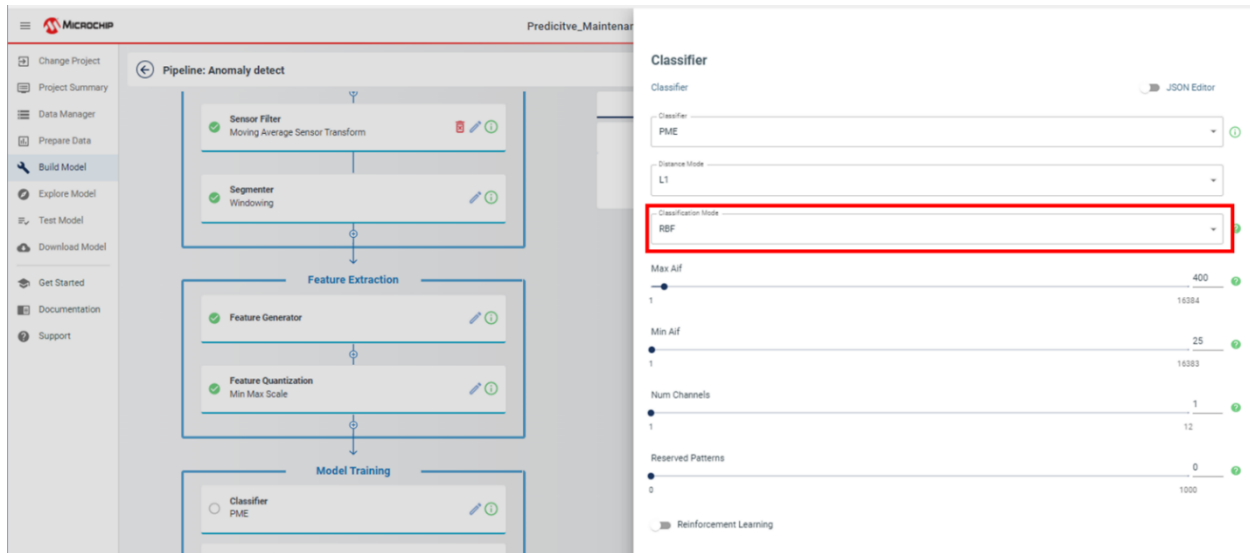
- Add a "Sensor Filter" by clicking on the add/plus button between Input query and Segmenter for noise removal.



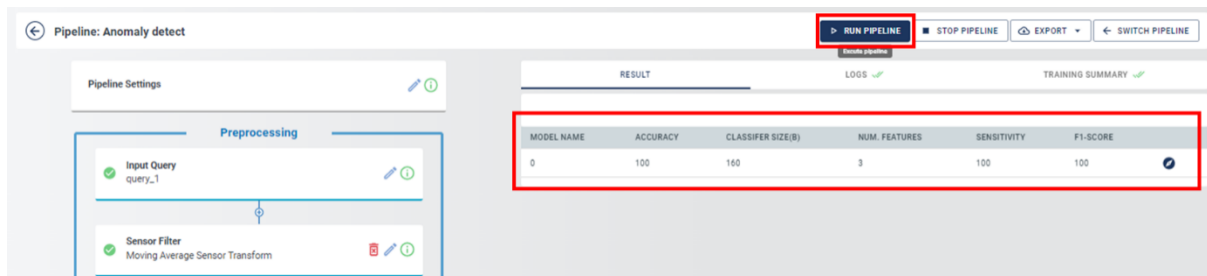
- In the Feature Generator Module, you can specify the types of feature generators, the inputs into the feature generators and any parameters the feature generators require. Verify that all required features are selected and add additional features if desired. Additional features can be added by clicking the **ADD FEATURES** button if desired.



- In the Classifier module, select the PME algorithm and choose Classification Mode as "RBF". Leave other modules to their default settings.



- Click the RUN PIPELINE button to generate the model. It will take couple of minutes to generate Model. ML Model Builder will provide the best accuracy model based on inputs we have provided.



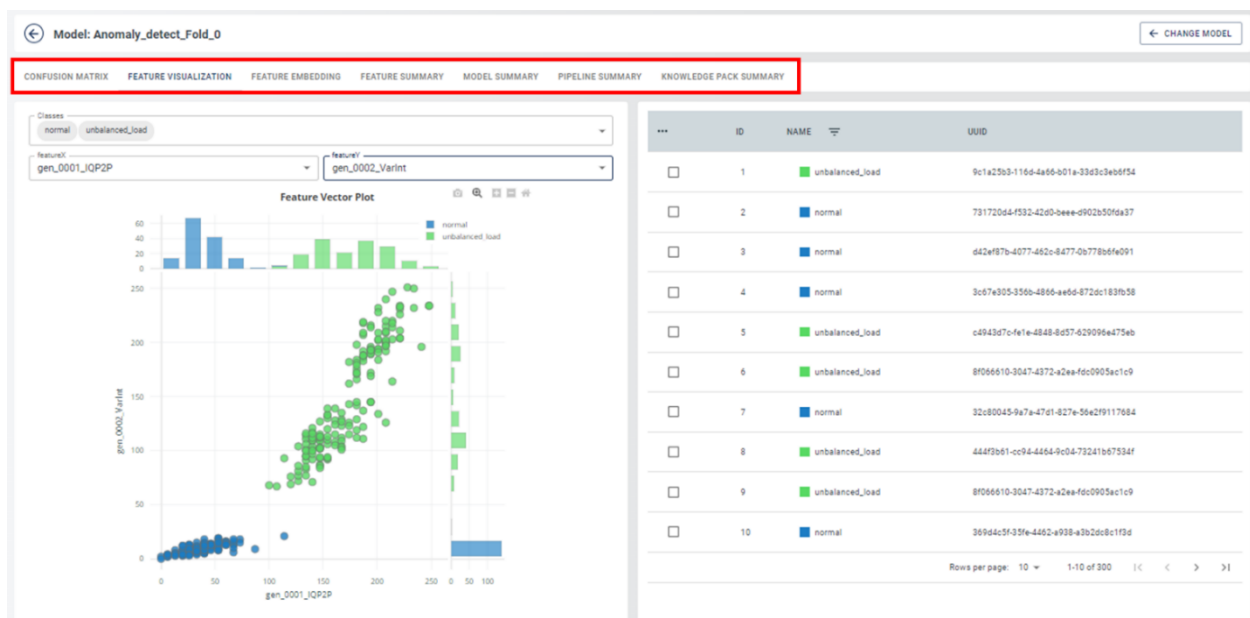
- After completing the Model Optimization process, proceed to the Explore Model tab to access additional information regarding each individual model.

Model: Anomaly_detect_Fold_0

CONFUSION MATRIX FEATURE VISUALIZATION FEATURE EMBEDDING FEATURE SUMMARY MODEL SUMMARY PIPELINE SUMMARY KNOWLEDGE PACK SUMMARY

Validation						Training					
	normal	unbalanced_load	UNK	Support	Sensitivity(%)		normal	unbalanced_load	UNK	Support	Sensitivity(%)
normal	58	0	0	58	100.00	normal	230	0	0	230	100.00
unbalanced_load	0	65	0	65	100.00	unbalanced_load	0	259	0	259	100.00
Predicted	58	65	0	123		Predicted	230	259	0	489	
Pos_Predic(%)	100.00	100.00		Acc(%)	100.00	Pos_Predic(%)	100.00	100.00		Acc(%)	100.00

- Various output results can be viewed by accessing each of these options. FEATURE VISUALIZATION option can be used to analyze selected feature output.



- Navigate to the Test Model tab to filter the data, compute accuracy, and generate a confusion matrix for the test samples. To filter the data and to select only the “test” samples and to compute accuracy and generate a confusion matrix for them:
 - Select the Session you want to work with.
 - Locate and click the upside-down triangle icon in the Dataset column.

- From the dropdown menu, select “test” to filter the data and display only the test samples.
- Locate and click the ellipsis (...) icon located at the leftmost column of the table.
- From the dropdown menu, select “Select All” to include all test samples in the analysis.
- Click the **RECOGNIZE** button to begin computing accuracy metrics for the selected samples.

Captures

► RECOGNIZE ► SUMMARIZE ■ STOP

...	NAME	ACCURACY	SEGMENTS	SIZE(MB)	RESULTS	CREATED	CAPTURE_UUID	CAPTURE_CONFIGURATION_UUID	DATASET
									Test
<input checked="" type="checkbox"/>	normal_11_800.csv	100	1	0.06	RESULTS	2024-02-08T16:32:57.762Z	e9ef94cf-3191-440f-b616-90a57c1454de	eeee2b52-c3d4-41c5-abfa-1e7718c702ab	Test
<input checked="" type="checkbox"/>	normal_12_800.csv	100	1	0.06	RESULTS	2024-02-08T16:33:28.735Z	dffa35c3-a1e9-4be1-a4d9-ac8792c9aeeef	eeee2b52-c3d4-41c5-abfa-1e7718c702ab	Test
<input checked="" type="checkbox"/>	Normal_operation_1000_3_test.csv	100	1	0.07	RESULTS	2024-02-21T04:54:11.953Z	4d83232d-5d4a-4569-8317-aa0f0ef3e54	eeee2b52-c3d4-41c5-abfa-1e7718c702ab	Test
<input checked="" type="checkbox"/>	Normal_operation_1100_3_test.csv	100	1	0.07	RESULTS	2024-02-21T05:36:27.638Z	8371507c-b6d6-4c67-ae10-4220e0fb7bd	eeee2b52-c3d4-41c5-abfa-1e7718c702ab	Test
<input checked="" type="checkbox"/>	Normal_operation_1200_3_test.csv	100	1	0.07	RESULTS	2024-02-21T04:11:38.840Z	a8003466-145f-485e-befb-382a5953afc7	eeee2b52-c3d4-41c5-abfa-1e7718c702ab	Test
<input checked="" type="checkbox"/>	Normal_operation_900_3_test.csv	100	1	0.06	RESULTS	2024-02-21T05:34:38.175Z	77c2138f-ee45-4fc6-9206-78d9747dc01e	eeee2b52-c3d4-41c5-abfa-1e7718c702ab	Test
<input checked="" type="checkbox"/>	unbalanced_11_800.csv	100	1	0.06	RESULTS	2024-02-08T16:34:43.475Z	f2a153c1-d0a1-45fa-b3b7-bf41639f188d	eeee2b52-c3d4-41c5-abfa-1e7718c702ab	Test
<input checked="" type="checkbox"/>	unbalanced_12_800.csv	100	1	0.06	RESULTS	2024-02-08T16:35:09.838Z	d2076566-0d46-47c9-8bce-f2c8c528d21e	eeee2b52-c3d4-41c5-abfa-1e7718c702ab	Test
<input checked="" type="checkbox"/>	unbalancedload_1000_3_test.csv	100	1	0.07	RESULTS	2024-02-21T04:27:25.323Z	d25ec7e5-4b14-4e1e-bc6d-9fa07002f815	eeee2b52-c3d4-41c5-abfa-1e7718c702ab	Test
<input checked="" type="checkbox"/>	unbalancedload_1100_3_test.csv	100	1	0.07	RESULTS	2024-02-21T05:39:53.794Z	332e4ce6-df95-4de4-8928-59a7b6ab3196	eeee2b52-c3d4-41c5-abfa-1e7718c702ab	Test

- When the accuracy is computed, click the **SUMMARIZE** button to generate the confusion matrix for the test samples.

Note: that this process may take a few minutes to complete. Once finished, you will be presented with a table summarizing the classification results.

	Unknown	normal	unbalanced_load	UNK	Ground Truth	Support	Sensitivity(%)
Unknown	0	0	0		0	0	
normal	0	108	0		6	108	100.00
unbalanced_load	0	0	90		5	90	100.00
Predicted	0	108	90		11	198	
Pos_Predic(%)		100.00	100.00			Acc(%)	100.00

- Once finished, deploy the model by navigating to the Download Model tab, selecting the appropriate compiler and settings, and clicking DOWNLOAD.

Model: Anomaly_detect_Fold_0 CHANGE MODEL

Select a Target

Compilers

MPLAB XC8

MPLAB XC16
SELECT

MPLAB XC32

Windows x86_64

x86 GCC Generic

Platform

Name: MPLAB XC16
Manufacturer: Microchip
Description: Compile libraries for Microchip 16 bit processors.
Resources: [Firmware Documentation](#)

Class Map:

1 - normal 2 - unbalanced_load

Knowledge Pack Resource Estimates

Estimated Memory Usage		
SRAM Used:	2400 Bytes	✓
Stack Size:	856 Bytes	✓
Flash Used:	3184 Bytes	✓

Estimated Latency		
Feature Extraction Latency:	2.061 ms (164864)	✓
Total Latency:	2.061 ms (164864)	✓

- Fill out the Knowledge Pack settings using the Pipeline, Model and Data Source you created in the previous steps.
- Select the library output format.
- Select device dsPIC33CK256MP508.
- Click the **DOWNLOAD** button.

Download Knowledge Pack

Platform - MPLAB XC16

←

Format

Library

Processor

dsPIC33CK256MP508

×

Compiler

MPLAB XC16 2.10

Data Source

dspic_motor_control

Application

AI Model Runner

Output

Serial

Debug/Profiling Settings

Debug/Profiling Settings:

Test Data

?

Debug

False

?

Extra build flags

DOWNLOAD

By completing these steps, you will be able to deploy your model and make it available for use. It is important to ensure that all necessary settings are correctly entered before proceeding with the download.

Knowledge Pack Integration:

Integrate the Knowledge Pack APIs into your embedded device firmware by following the [Knowledge Pack Integration](#) guidelines.