Bachelor Thesis

Tiny Machine Learning for Ultrasonic Object Classification
A Naïve Bayes Approach on Red Pitaya

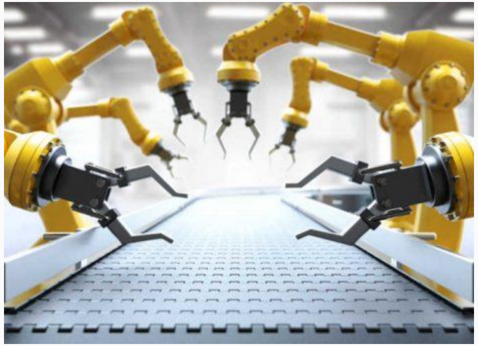
Nguyen Tien Anh Ha
1356973
Supervisor : Prof. Dr.-Ing. Peter Nauth
Co-supervisor: M.Sc. Julian Umansky

FRANKFURT UNIVERSITY OF APPLIED SCIENCES

Table of contents

Motivation

Automotive



Automation

Daily Live

Objective

Human-Machine Interface Prototype

Develop an interface with buttons and LEDs for sensor features, bypassing the dependency on external software.

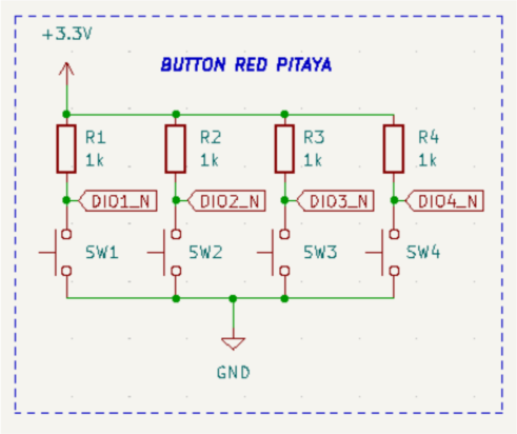
Naive Bayes Classifier Implementation

Utilize Naive Bayes as a probabilistic technique on the Red Pitaya platform for real-time classification.

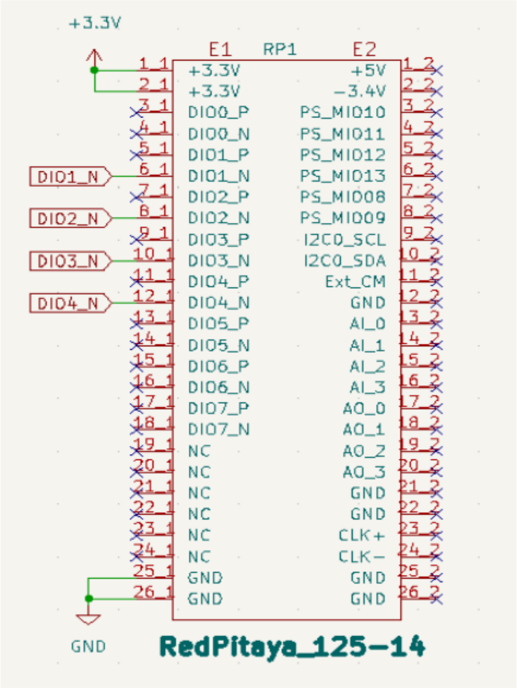
Machine Learning Pipeline

Establish a flexible and updatable machine learning pipeline for effortless model modifications.

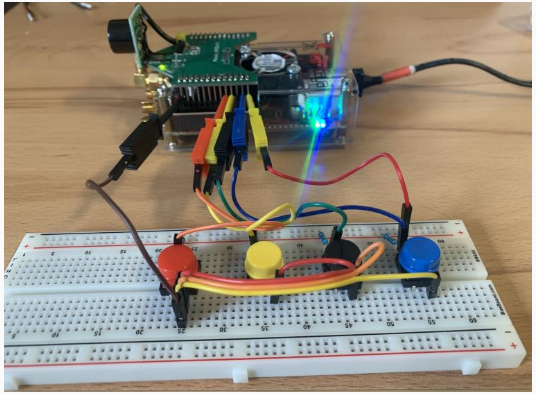
Hardware Setup



(a) On button

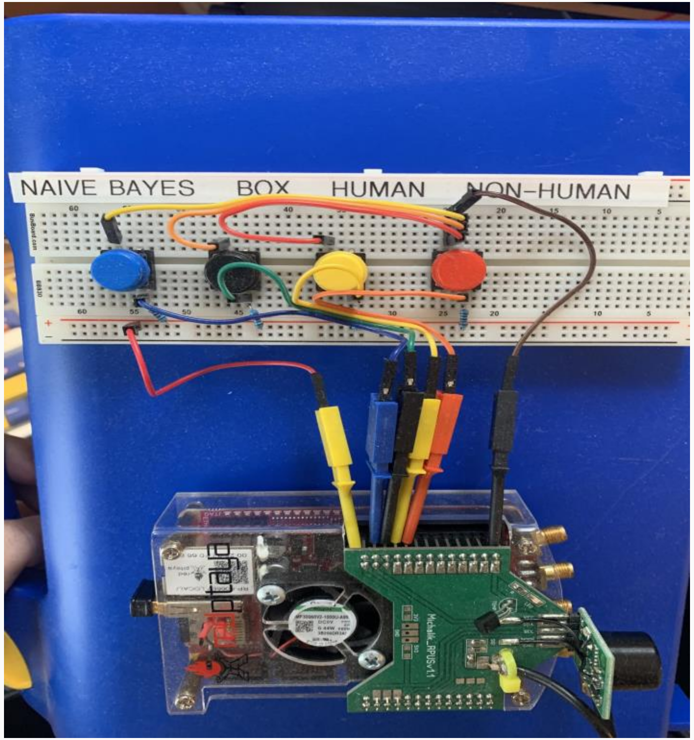


(b) On Red Pitaya



- A pull-up resistor helps to reduce electrical noise on the input pin by providing a definite state
- Without a pull-up, the input pin could pick up random noise

Hardware Setup



- **Button 1:** Saves object 1 sensor data as a binary file
- **Button 2:** Logs object 2 sensor data
- **Button 3:** Activates Box Classifier
- **Button 4:** Activates Naive Bayes Classifier

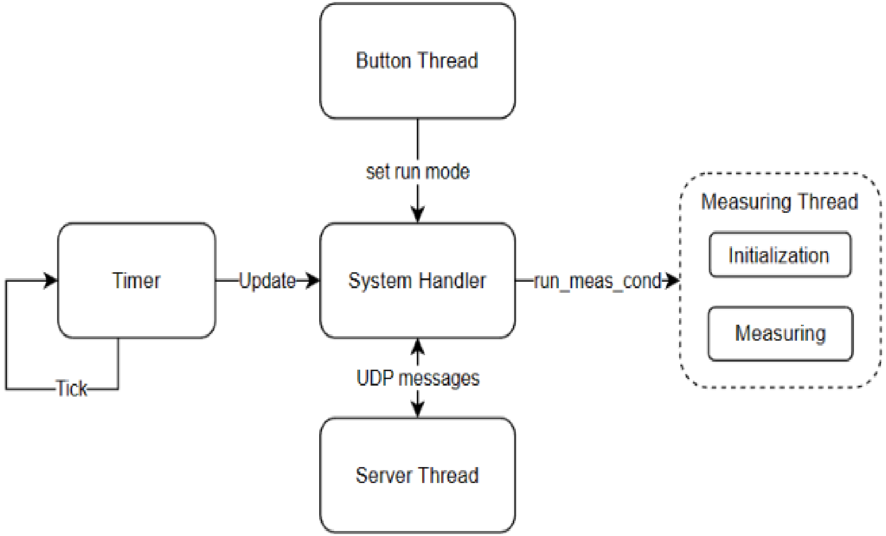
- **LED 1 (Class 0):** Lights up for class 0

- **LED 2 (Class 1):** Lights up for class 1

- **LED 3 (Classifier Mode):** On for Naive Bayes, off for Box Classifier

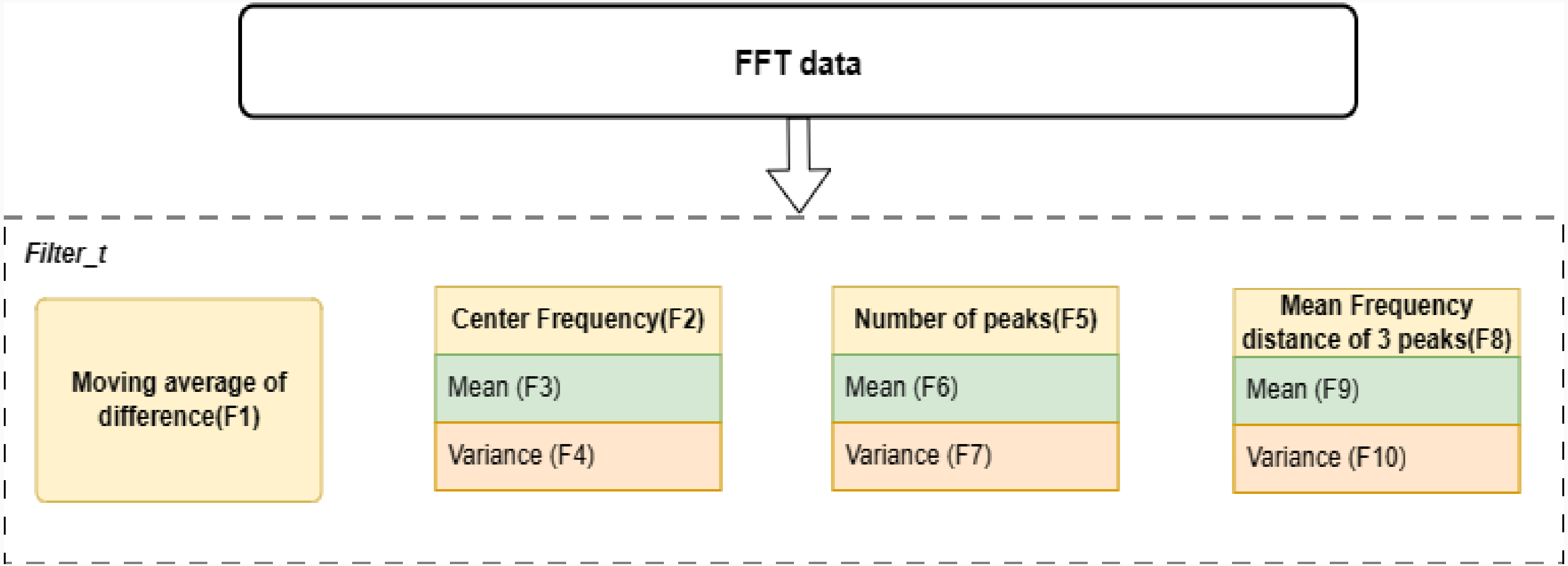
- **LED 6 (Learning Phase):** Indicates when the system is learning and saving data

Red Pitaya Architecture



Button Thread: Manages button interrupts, storage, and prediction controls for standalone sensor operation.
Time Thread: Manages the measuring process with a 100ms cycle for consistent data acquisition.
Server Thread: Listens for UDP messages on port 61231 and parses commands for real-time classification.
Measure Thread: Initializes ADC and FFT process, running feature extraction processes.

Feature Extraction



Naïve Bayes Classifier

The general form of the Naive Bayes theorem for a class variable *C* and a dependent feature vector *O*=(*o*1,*o*2,…,*on*) is given as follows:

Class prior probability

Posterior Probability

Likelihood probability

$$P(H = C_k | O = o) = \frac{P(H = C_k) \prod_{i=1}^{n} P(O_i = o_i | H = C_k)}{P(O_1 = o_1, \ldots, O_n = o_n)}$$

Predictor prior probability

$$\hat{C} = \arg \max_{k \in \{1,\ldots,K\}} \left[\log P(H = C_k) + \sum_{i=1}^{n} \log P(O_i = o_i | H = C_k) \right] \qquad (3.7)$$

Key Principles:

- **Conditional Probability:** The classifier calculates the probability of a data point belonging to each class based on its features.
- **Feature Independence:** Despite its naive assumption, this classifier often performs surprisingly well in practice, even when the independence assumption is violated.
- **Prior Probability:** Represents the overall likelihood of each class in the dataset.
- **Posterior Probability:** The probability of a class given a set of features, which is computed using Bayes' theorem.

Sklearn

```
from sklearn.model_selection import
train_test_split
from sklearn.naive_bayes import
Gaussi

# Spli
and a
X_trai
train_                                    ,
random_

# Trai
gnb = GaussianNB()
model = gnb.fit(X_train, y_train)
```

Metrics:

F1 Score:	0.82
Accuracy(%):	0.82
Recall:	0.85
Precision:	0.82

Data Preparation and Splitting

- Split dataset into features (X) and target (y)
- Use train_test_split to divide data (80% train, 20% test)
- Ensure data is shuffled for randomness

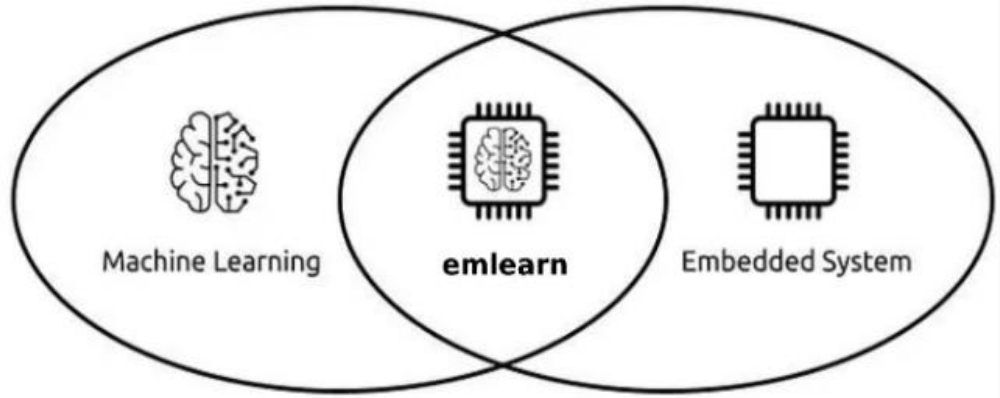
Choosing the Naive Bayes Model

- Select appropriate Naive Bayes variant (Gaussian, Multinomial, Bernoulli)
- Consider feature distribution when choosing

Model Evaluation

- Predict on the test set with model.predict(X_test)
- Calculate accuracy and other metrics

Emlearn



- No libc required
- No dynamic allocations
- Single header file include
- Support fixed-point math
- Support various machine learning algorithm, including Naïve Bayes as well
- Running on AVR Atmega, ESP8266, ESP32, ARM Cortex M (STM32), Linux

Emlearn on Red Pitaya

Model Summaries:
- Mean(**μ)**
- Standard Deviation(σ)
- Natural logarithm of std(ln(σ))

Algorithm 3: Emlearn Gaussian Naive Bayes on Red Pitaya

Input : EmlBayesSummary(mean, std, stdlog2), Input values.
Output : Predicted class: either 0 (non-human) or 1 (human)
Function `EmlearnNaiveBayes()`:
1) Convert all input values to fixed point representation;
2) Calculate ln of the PDF using a quadratic approximation;
3) Compute ln of $N(\mu, \sigma^2)$ based on the standard $N(0, 1)$;
4) Utilize the function `eml_bayes_predict` to:
   - Compute the log-posterior probabilities for each class
   - Identify the class with the highest log-posterior probability

return Predicted class;

Calculating likelihood probabilities

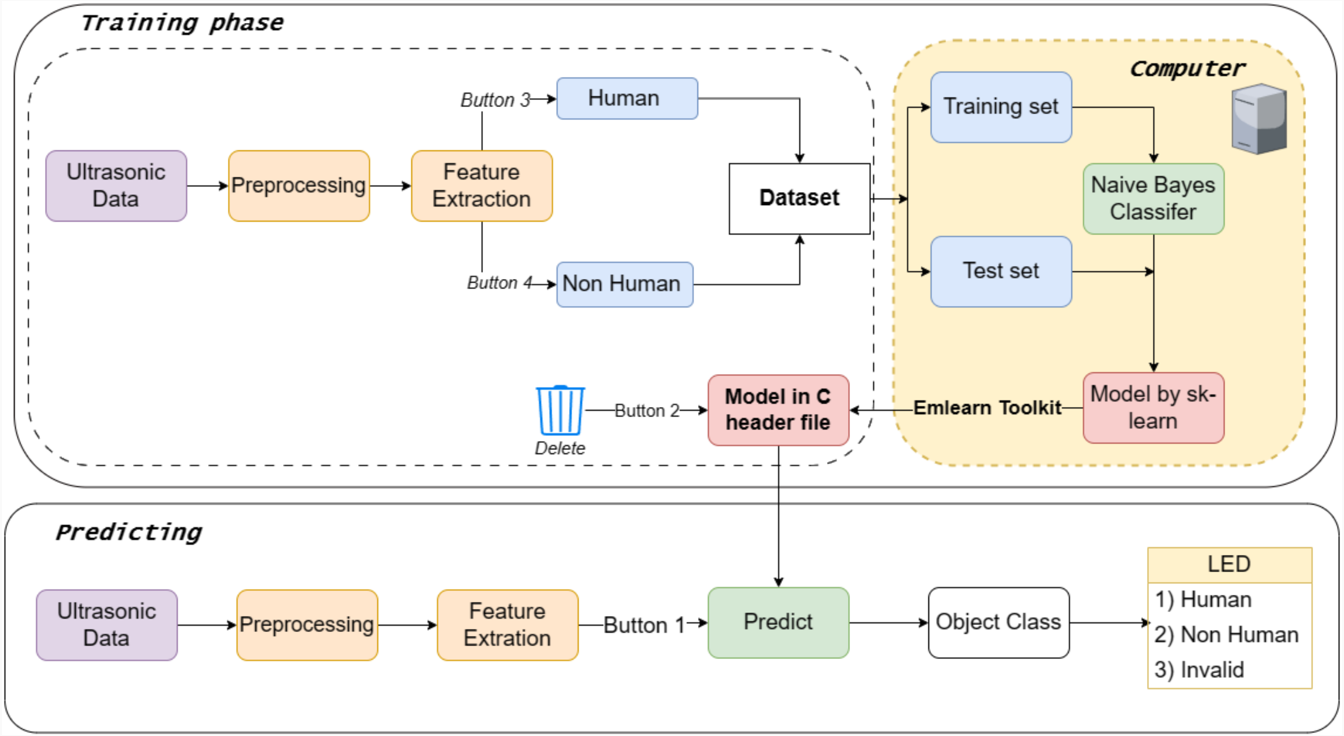
PDF of a Gaussian distribution in terms of the standard nor-mal distribution in logarithmic form, one can derive the following relationship:

$$\ln(f(x|\mu, \sigma 2)) = \ln(f(z)) - \ln(\sigma)$$

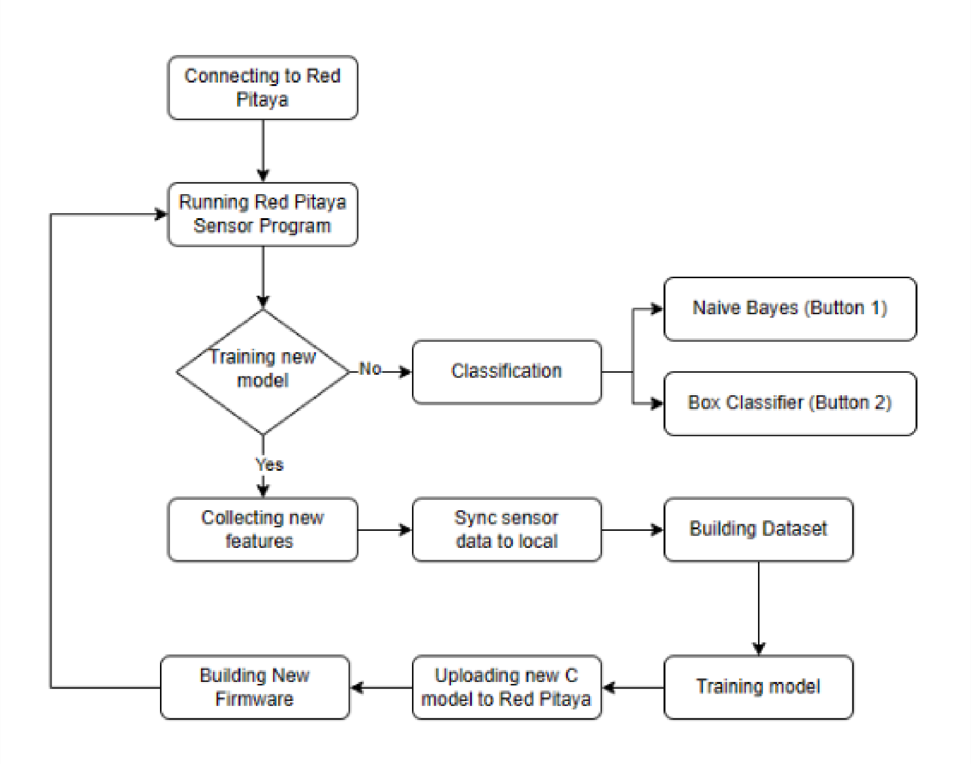
- ln(f(z)): PDF of standard normal distribution using quadratic approximation
- ln(σ): is precomputed by sklearn model

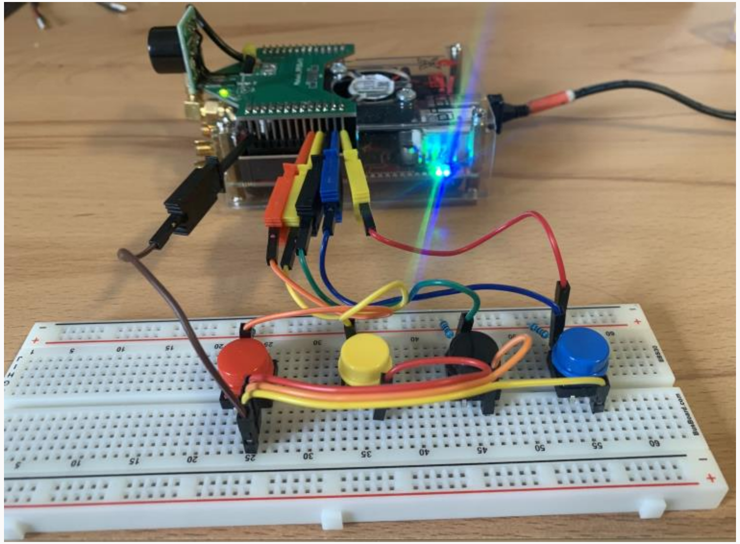
Calculating posterior probabilities

Machine Learning Pipeline

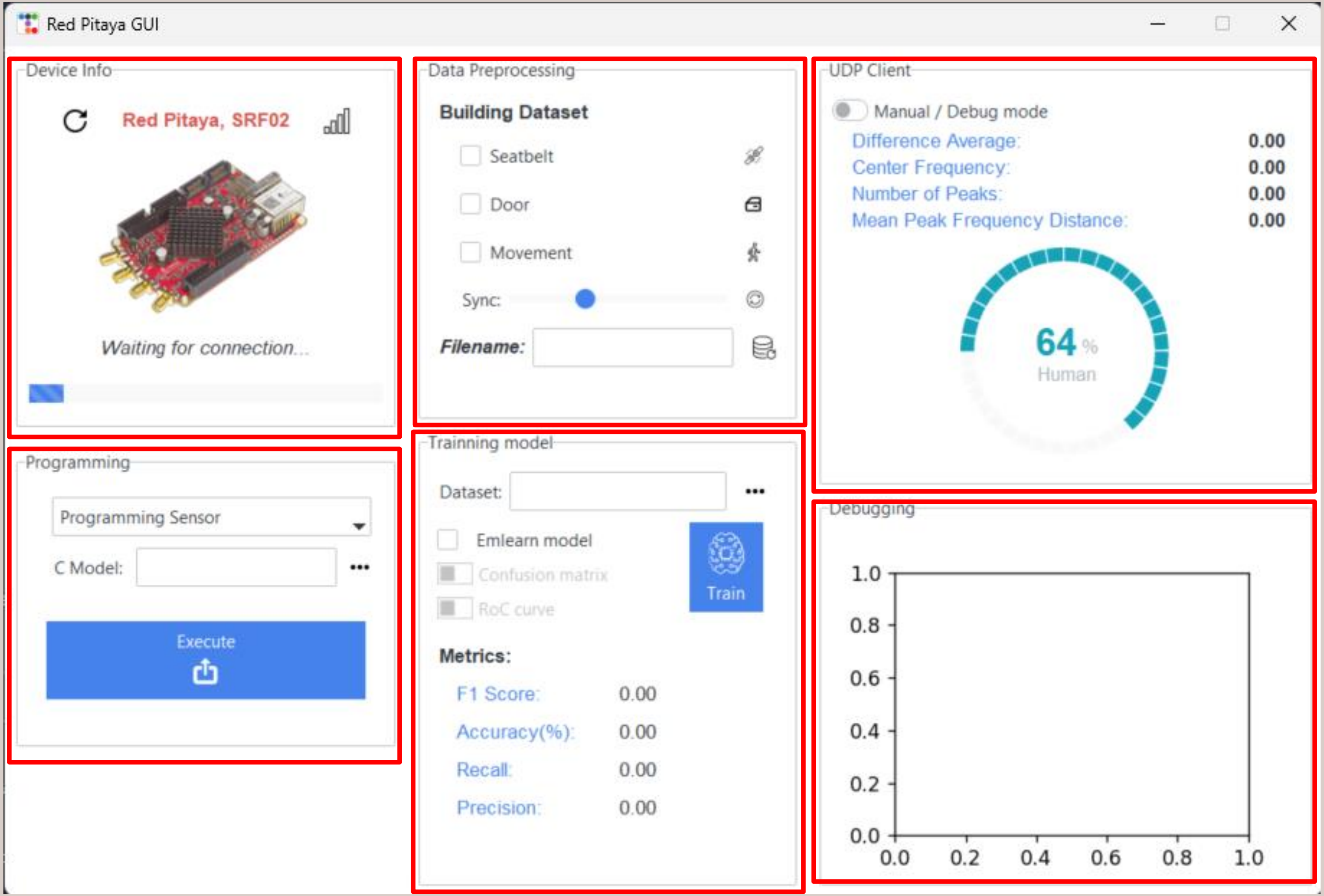


Uploading a model

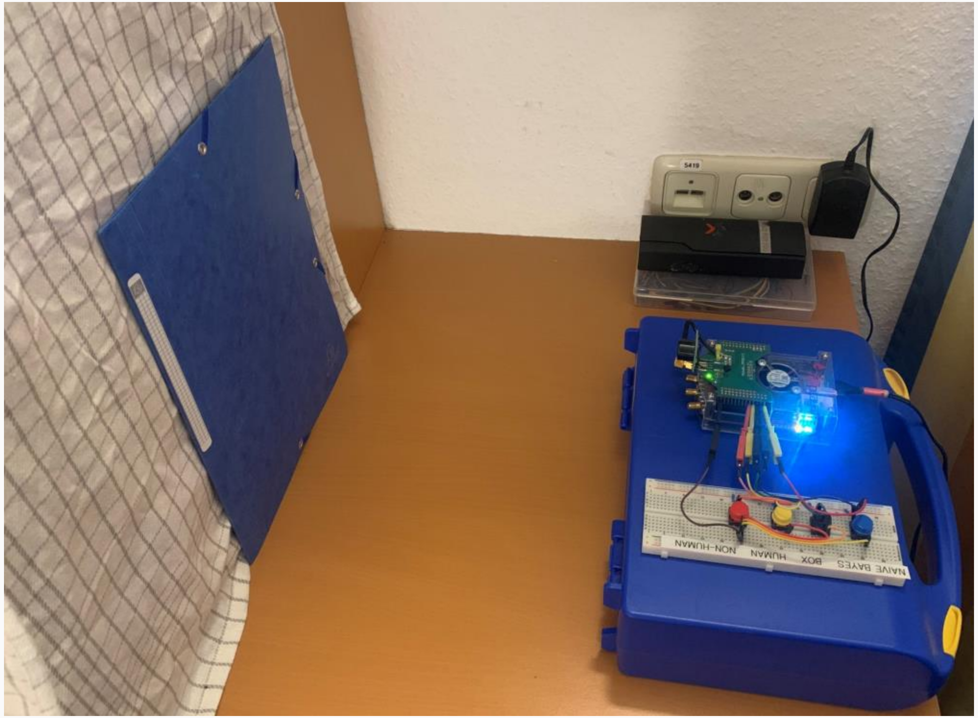




Smart sensor application


Red Pitaya GUI
Device Info
Red Pitaya, SRF02
Waiting for connection...
Programming
Programming Sensor
C Model:
Execute
Data Preprocessing
Building Dataset
Seatbelt
Door
Movement
Sync:
Filename:
Trainning model
Dataset:
Emlearn model
Confusion matrix
RoC curve
Train
Metrics:
F1 Score: 0.00
Accuracy(%): 0.00
Recall: 0.00
Precision: 0.00
UDP Client
Manual / Debug mode
Difference Average: 0.00
Center Frequency: 0.00
Number of Peaks: 0.00
Mean Peak Frequency Distance: 0.00
64 %
Human
Debugging
1.0
0.8
0.6
0.4
0.2
0.0
0.0 0.2 0.4 0.6 0.8 1.0

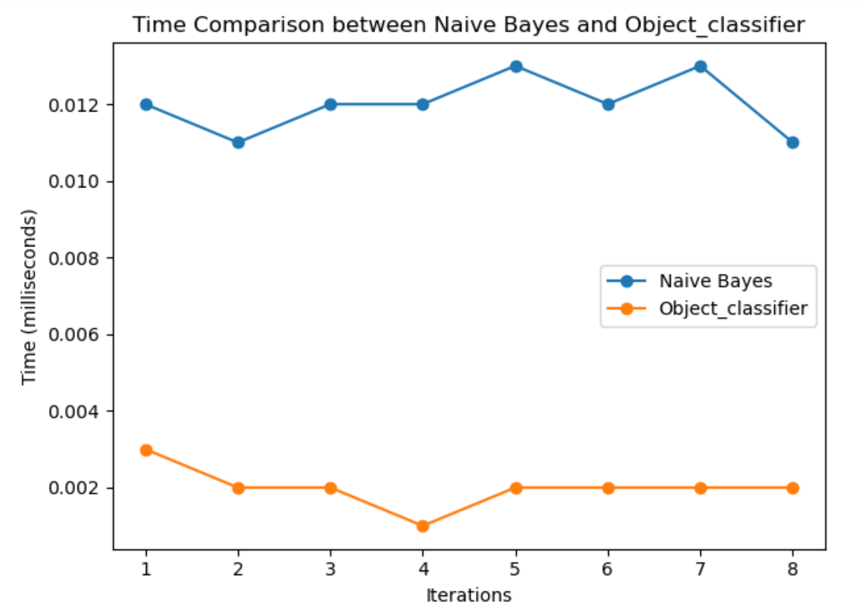
Experimental setup



- Stable measuring condition
- Target Materials: Paper Board and Blanket
- Data Collection: 10 times/ measurement
  - Exploring Sample Sizes
  - Analyzing the Outcomes

04 Results & Discussion

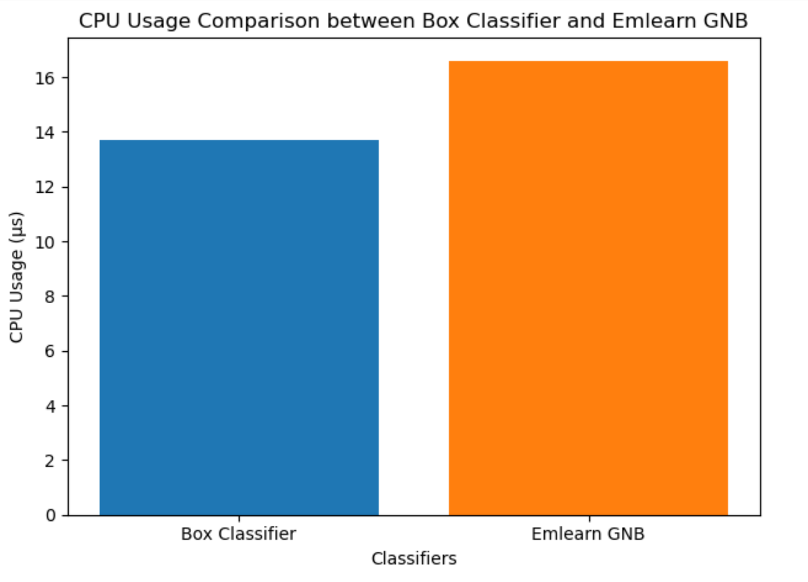
Execution Time



Utilizing time.h library

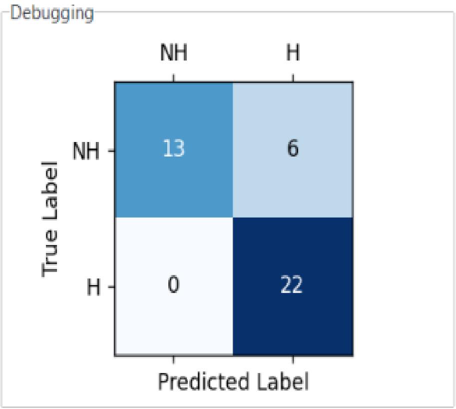
- 0.003(s) – Box Classifier
- 0.012(s) – Naïve Bayes

CPU Usage

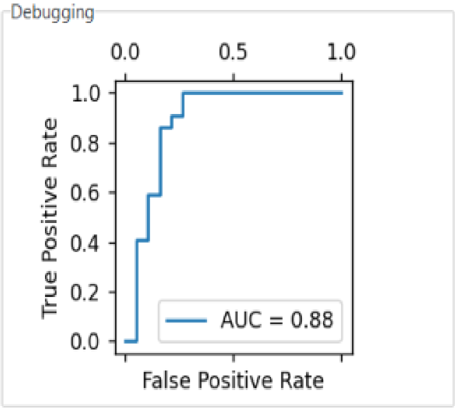


- 13(µs) – Box Classifier
- 17(µs) – Naïve Bayes
  - Emlearn's reliability in differentiating between objects
  - Box classifier's need for calibrations and lower accuracy

Analysis of the results



(a) Confusion Matrix



(b) AUC Graph

Figure 5.8: Confusion Matrix with 200 samples

Confusion Matrix:

- Table showing model's prediction accuracy.
- **True Positives (TP):** Correct positive predictions.
- **True Negatives (TN):** Correct negative predictions.
- **False Positives (FP):** Incorrect positive predictions.
- **False Negatives (FN):** Incorrect negative predictions.
- Used to calculate accuracy, precision, recall, F1-score.

AUC Curve:

- Plots true positive rate vs. false positive rate.
- Measures model's ability to differentiate classes.
- **AUC of 0.5:** no class separation capacity.
- **AUC of 1.0:** perfect class separation.
- Higher AUC indicates better model performance.

Classification

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}$$

Method	Number of Samples		
	200 samples	2,000 samples	20,000 samples
Sk-learn	85%	77%	96%
Real – world	90%	88%	82%

Future Work

- Sensor sensitivity
- Fourier Transform Limitation
- Algorithm Assumption
- Inability to recognize unknown objects

=> Advanced time-frequency based feature analysis techniques, such as the STFT and wavelet transform.

References

- [1]
- "Ultrasonic Sensors Widely Used in a Range of Applications," Arrow.com. Accessed: Nov. 01, 2023. [Online]. Available: https://www.arrow.com/en/research-and-events/articles/cui-ultrasonic-sensors-widely-used-in-a-range-of-applications
- [2]
- "Welcome to emlearn's documentation! — emlearn documentation." Accessed: Nov. 01, 2023. [Online]. Available: https://emlearn.readthedocs.io/en/latest/
- [3]
- "Get To Know About Robotics - Its Applications and Research." Accessed: Oct. 31, 2023. [Online]. Available: https://www.freearcadehall.com/

Thanks!

Do you have any questions?

youremail@freepik.com

+34 654 321 432

yourwebsite.com

CREDITS: This presentation template was created by **Slidesgo**, and includes icons by **Flaticon**, and infographics & images by **Freepik**

Please keep this slide for attribution