Lina BEN-YOUNES

Supervised by Dr Renson

**Project Final Report**

**Automatic Detection of Mode Interactions Using Machine Learning**

**ME4 Individual Project**

Imperial College London

Department of Mechanical Engineering

Word count:

Word limit: 10,000 June 8th 2023

# Abstract

This report examines how coupled vibration modes could be automatically detected using well-established Machine Learning techniques. Combining the use of wavelet transforms and support vector machines, the presented method detects interactions between the fundamental and harmonics up to five, with an accuracy of

In order to train the support vector machines, scalogram databases were computed using the Morse wavelet transform from the MATLAB Wavelet Toolbox. The time series analysed were obtained by integrating beam and duffing equations using Python’s scipy.integrate module.

The report also presents a method which does not rely on Machine Learning techniques but rather on hard-coded logic intended to mimic the reasoning physicists would use to discern interactions. That method is less computationally demanding and allows for more flexibility in defining the threshold for detecting interaction. Still, it fails to exploit the large amount of information the scalogram gives.

Table des matières

[Abstract 2](#_Toc136873645)

[Introduction and Objectives 5](#_Toc136873646)

[Context 5](#_Toc136873647)

[Objectives 5](#_Toc136873648)

[Literature Review and General Concepts 7](#_Toc136873649)

[Nonlinear Normal Modes and Mode Interaction 7](#_Toc136873650)

[Continuous Wavelet Transform & Scalograms. 9](#_Toc136873651)

[Support vector machines 10](#_Toc136873652)

[Linear SVM 12](#_Toc136873653)

[Kernels 12](#_Toc136873654)

[Soft/Hard Margin 13](#_Toc136873655)

[Performance evaluation 15](#_Toc136873656)

[K-fold cross-validation 15](#_Toc136873657)

[Class Imbalance 16](#_Toc136873658)

[Multi-class ROC analysis 17](#_Toc136873659)

[Implementation 18](#_Toc136873660)

[The Python Package. 18](#_Toc136873661)

[First Package: No\_ML 18](#_Toc136873662)

[Second Package: ML\_method 18](#_Toc136873663)

[Time series: Beam equation 19](#_Toc136873664)

[Producing clean data: format and pre-processing 20](#_Toc136873665)

[Class attribution for the training dataset 22](#_Toc136873666)

[Class attribution 22](#_Toc136873667)

[Class imbalance 23](#_Toc136873668)

[Format of the results 23](#_Toc136873669)

[Support Vector Machine Parameters 24](#_Toc136873670)

[Results and Discussion. 25](#_Toc136873671)

[C and γ 25](#_Toc136873672)

[Choice of kernel 27](#_Toc136873673)

[Choice of threshold 28](#_Toc136873674)

[Threshold on the training data 28](#_Toc136873675)

[Threshold on probabilities 28](#_Toc136873676)

[Under-sampling majority classes 29](#_Toc136873677)

[Experimental data & Comparison between the ML and No\_ML methods 30](#_Toc136873678)

[Conclusions and Future Work 34](#_Toc136873679)

[References 35](#_Toc136873680)

[Appendices 36](#_Toc136873681)

[Appendix 1: performance per class for C and γ 36](#_Toc136873682)

[Appendix 2: Pre-processed Scalogram of Wing Engine 39](#_Toc136873683)

[Appendix 3: and algorithm results of Wing Engine 41](#_Toc136873684)

# Introduction and Objectives

## Context

See to add references.

Confronted with complex nonlinear structures, engineers analyse various nonlinear responses, one being coupled vibration modes. In a nonlinear system, vibrational modes may influence each other, affecting individual behaviour and characteristics. These mode interactions can lead to changes in modal frequencies and energy transfer between modes.

They pose a challenging threat to the integrity of the concerned structures. Indeed, energy transfers from a local mode of low effective mass components to a global mode with high effective mass might critically jeopardise the structure.[1]

Nowadays, these interactions are detected by trained-eyed physicists and engineers. They conduct frequency sweeps and inspect scalograms of the structure’s response.

Method

The algorithms must be trained with those scalograms, which entails creating scalogram datasets and classifying them according to whether they present modal interactions. We will also have to write scaling and pre-processing algorithms to obtain clean data and increase the accuracy and efficiency of the machine-learning models.

We will then train and test different Machine Learning models and techniques in order to select those best suited for this classification problem. Finally, the algorithms will be tested on real-life data and examples.

## Objectives

The project goals are:

- Identifying the patterns, parameters, and favourable conditions of modal interactions.

- Creating scalogram datasets of nonlinear models with and without model interactions.

- Creating a procedure to pre-process and standardise the data format on which the support vector machines (SVM) are trained and used.

- Training and adjusting the parameters of the SVM.

- Testing them on real-life data.

# Literature Review and General Concepts

## Nonlinear Normal Modes and Mode Interaction

Mode interactions are a prominent feature of Nonlinear Normal Modes (NNMs). If the system’s natural frequencies are commensurate, an energy exchange between the different modes may be observed during the internal resonance.

Kerschen, G., Peeters, M., Golinval, J., & Vakakis, A. (2009). Nonlinear normal modes, Part I: A useful framework for the structural dynamicist. *Mechanical Systems and Signal Processing, 23*(1), 170-194. <https://doi.org/10.1016/j.ymssp.2008.04.002>

When exciting one mode, one may produce a large-amplitude response on another. Such interactions are identified as n:m interactions, meaning the n-th harmonic of a mode matches the m-th of another.

Such interaction may therefore be observed when the system:

* Is non-linear: The resonances are decoupled if the system is linear.
* Is subject to high forcing: for low forcing, the equations governing the system may be approximated by their linearisation.
* Its frequencies fi and fj are approximately such that: k.fi = p.fj, for low integer values of k and p. We may observe k:p interactions.

In such cases, mode interaction would be identified by high amplitude coefficients for frequency fj when the system is subject to excitation of frequency fi. So in the case of interactions, the scalogram would present high amplitude coefficients for the -th harmonic.

An example can be found in [?].

|  |
| --- |
| Une image contenant texte, Tracé, capture d’écran, ligne  Description générée automatiquement |
| Figure [?] from [?] |

The scalogram in Figure [?] displays the response of a small satellite to a frequency sweep. One identifies the 2:1 interaction, hereby circled, between two internally resonant modes of the structure. The second harmonic of mode 3 matches the frequency of mode 7. Mode 3 involves an out-of-phase motion of the inertia wheel, and mode 7 consists of an axial motion of the telescope supporting panel.

Noël, J., Renson, L., & Kerschen, G. (2014). Complex dynamics of a nonlinear aerospace structure: Experimental identification and modal interactions. *Journal of Sound and Vibration, 333*(12), 2588-2607. <https://doi.org/10.1016/j.jsv.2014.01.024>

## Continuous Wavelet Transform & Scalograms.

Wavelet transforms are mathematical tools used to analyse changing features in data. As the response to a frequency sweep is studied, the feature analysed is frequency.

Wavelet transforms address the limitations of the Fourier transform. While Fourier analysis quantifies the contributions of sine waves of specific frequencies over the full length of the signal, wavelet analysis decomposes signals into wavelets shifted in time and scaled in frequency. A wavelet is a rapidly decaying, wave-like oscillation. This enables wavelets to represent data across multiple scales.

Inc. The MathWorks. (nd) *Wavelet Transforms in MATLAB.* <https://fr.mathworks.com/discovery/wavelet-transforms.html4> [Accessed 20th June 2023]

Wavelet transforms can be classified into continuous wavelet transforms (CWT) and discrete wavelet transforms (DWT). Unlike DWT, discrete approximations of CWT are redundant in information. However, DTW only allows for a restrictive choice of wavelets and is not shift-invariant, the wavelet analysis method used throughout the project is CWT and therefore, this section will cover continuous wavelet transforms.

Inc. The MathWorks. (nd) *Continuous and Discrete Wavelet Transforms* <https://uk.mathworks.com/help/wavelet/gs/continuous-and-discrete-wavelet-transforms.html> [Accessed 22nd June 2023]

We should think of the graph of as a single ‘cycle’, the wavelet is assumed such that:

* It is compactly supported.

But it may be chosen such that it has additional properties such as smoothness. The mother wavelet is dilated by the scale and shifted to the position . We define:

The scale parameter a is inversely proportional to the instantaneous frequency at time b. We shall therefore define the continuous wavelet transform such that we get the relation:

Where is a constant.

The continuous wavelet transform of f corresponding to a choice of wavelet is:

Meyer, Y. (1992), *Wavelets and Operators*, Cambridge, UK: Cambridge University Press

As mentioned in the abstract, physicists use scalograms to characterise mode interaction. Scalograms are graphs displaying the absolute value of the CWT of a signal, plotted as a function of time and frequency, as seen in Fig [?].

## Support vector machines

The scope of the project was limited to interactions for values of < 6. The detection of mode interactions corresponds to a classification problem. The different classes are:

* No interaction
* 1:2 interaction
* 1:3 interaction
* 1:4 interaction
* 1:5 interaction

The common Machine Learning classification techniques are Naïve Bayes, Random forests, Support Vector Machines (SVMs), and K-Nearest-Neighbours (KNN). The choice of SVMs for this project is justified by the format of the data used in this project.

Scalograms were transformed into a set of vectors each corresponding to a time sample of the scalogram, each of these vectors must be attributed a class. The dimension of the vectorial space is 120.

* Naïve Bayes relies on strong independence assumptions between the features of the data, which are not valid on the constructed dataset.
* K-Nearest-Neighbours: KNN is very computationally expensive, not only when trained but also when used to predict a class. Therefore, it would not be suited to classify data in a vectorial space of dimension 120.
* Random forests are a collection of decision trees. This method is best suited for data that are a mixture of numerical and categorical features, which is not the case in this project.

This section will therefore cover SVM algorithms only. The aim of Support Vector classification is to find optimal separating hyperplanes in a high-dimensional feature space. There are several ways to define the optimality of the hyperplane, in this project, the maximal margin classifier was used.

Cristianini, N., Shawe-Taylor, J. (2005). *Support Vector Machines*. In Cambridge University Press eBooks (pp. 93–124). <https://doi.org/10.1017/cbo9780511801389.008>

This means that SVM considers the training points at the extreme of their classes, i.e., the support vectors, and places the decision boundary, or separating hyperplanes, as far from these as possible. The margin is the shortest distance between the support vectors and the boundary. The SVM algorithm searches for a decision boundary that would maximise the margin.

|  |
| --- |
| Figure 1 : Définir la « marge » entre les classes : le critère que les SVM cherchent à optimiser. |
| Figure [?] 2D example of Support Vector Machine  Inc. The MathWorks. (nd) *Support Vector Machines* <https://fr.mathworks.com/discovery/support-vector-machine.html>  [Accessed 28th June 2023] |

### Linear SVM

Let us first focus on linear decision boundaries. The decision hyperplane is such that the linear discriminant function cancels. The linear discriminant functions is:

Where is the weight vector and is the bias or threshold weight. The weight vector is orthogonal to the decision hyperplane. The class of point is determined by the sign of . And the margin is the smallest value of .

In the case of multi-category classification, the usual approach is to define one linear discriminant function for each class . The class of a point is then such that .

### Kernels

Kernels are used to learn non-linear relations with a linear machine. Non-linear machines are built in two steps: a fixed non-linear mapping transforms the data into a feature space F, and a linear machine classifies them in F.

Cristianini, N., & Shawe-Taylor, J. (2000). Kernel-Induced Feature Spaces. In *Support Vector Machines*. <https://doi.org/10.1017/cbo9780511801389.005>

|  |
| --- |
| Une image contenant texte, diagramme, ligne, Police  Description générée automatiquement |
| Figure [?] A feature map for kernel [same ref] |

The common kernel functions are polynomial, Radial Basis Function (RBF) and sigmoid. Those kernels necessitate the introduction of parameter is γ, which defines how much influence a training sample has on the hyper-plane position. Parameter C influences how close the data is fitted if soft margin SVMs.

|  |
| --- |
| gamma=10^-1, C=10^-2, gamma=10^0, C=10^-2, gamma=10^1, C=10^-2, gamma=10^-1, C=10^0, gamma=10^0, C=10^0, gamma=10^1, C=10^0, gamma=10^-1, C=10^2, gamma=10^0, C=10^2, gamma=10^1, C=10^2 |
| Figure [?] Influence of parameters γ and C on an RBF model  *RBF SVM parameters*. (n.d.). Scikit-learn. <https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html> [Accessed June 28th 2023] |

### Soft/Hard Margin

In the case of a hard margin, no support vector can be on the wrong side of the boundary, this is at the expense of a more complex decision boundary. A soft-margin SVM gives a smoother decision boundary by misclassifying some training points.

Cristianini, N., Shawe-Taylor, J. (2005). *Support Vector Machines*. In Cambridge University Press eBooks (pp. 93–124). https://doi.org/10.1017/CBO9780511801389.008

|  |  |
| --- | --- |
| Une image contenant art, peinture, Caractère coloré, capture d’écran  Description générée automatiquement | Une image contenant art, peinture, Caractère coloré, capture d’écran  Description générée automatiquement |
| 1. Hard Margin SVM | 1. Soft margin SVM |
| Figure [?] Comparison between Hard and Soft Margin SVMs [same ref] | |

Soft margin consists in introducing a parameter C called cost or penalty, which controls the trade-off between the size of the margin and the slack variable penalty. A set of values are introduced. They correspond to the error being made of sample . is the sampled point and is the class.

means the point lies on the correct side of the margin. means the point lies on the wrong side of the margin, and means the point lies on the wrong side of the classification hyperplane.

Finding the parameters of the SVM is then an optimisation problem:

Increasing the value of C will therefore increase the accuracy as well as the complexity of the boundary hyperplane, which might result in overfitting the data.

|  |
| --- |
|  |
| Figure [?] Model complexity and accuracy |

Tharwat, A. Parameter investigation of support vector machine classifier with kernel functions. *Knowl Inf Syst* **61**, 1269–1302 (2019). https://doi.org/10.1007/s10115-019-01335-4

## Performance evaluation

### K-fold cross-validation

K-fold cross-validation is used to assess performance by averaging across several divisions of a training set. The steps are:

1. Split all the training data in K equally sized random subsets.
2. Use K-1 of the subsets to train the model.
3. Use the last remaining subset to check the performance of the model.
4. Discard the model and repeat the process another K-1 times, until the model has been tested on every subset.

K-fold is used to confirm that the model is suitable, in particular, to check that it avoids overfitting, which would be noticeable by a significant gap in performance between the training dataset and the testing set.

Receiver operating characteristic

Receiver operating characteristics (ROC) graphs are useful for organizing classifiers and visualizing their performance. Let us first introduce a few metrics, in the case of a binary classification one may use the confusion matrix:

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **True class** | |
| **Positive** | **Negative** |
| **Hypothesized class** | **Positive** | True Positive (TP) | False Positive (FP)  Type I error |
| **Negative** | False Negative (FN)  Type II error | True Negative (TN) |
| **Column Totals** | | P | N |

The following performance metrics are then defined:

* Precision quantifies the ability to correctly identify positive cases without including false positives. *Precision =*
* Sensitivity quantifies the ability to identify the true positives while minimizing false negatives. *Sensitivity =*
* Specificity measures the ability to identify negative cases correctly out of all the actual negative cases. *Specificity =*
* Accuracy measures the proportion of correct predictions or classifications out of all the cases. *Accuracy =*

The ROC graph then illustrates the trade-off between sensitivity (true positive rate) and specificity (false positive rate) at different classification thresholds.

|  |
| --- |
| Une image contenant texte, ligne, diagramme, Tracé  Description générée automatiquement |
| Figure [?] Example of a ROC graph  Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, *27*(8), 861–874. https://doi.org/10.1016/j.patrec.2005.10.010 |

### Class Imbalance

Class imbalance refers to a situation where the distribution of classes in a dataset is significantly skewed, meaning that the number of instances belonging to one class is much higher or lower than the number of instances belonging to the other class.

In class-imbalanced scenarios, accuracy alone may not accurately assess a model's performance. A classifier that predicts only the majority class can achieve high accuracy, even if it fails to identify instances from the minority class correctly. In such cases, one uses the F-measure to assess the performance of a model.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, *27*(8), 861–874. https://doi.org/10.1016/j.patrec.2005.10.010

*F-measure =*

The F-measure accounts for the trade-off between precision and sensitivity, making it suitable for evaluating models in imbalanced datasets. However, the selection of that appropriate trade-off depends on the specific task and the relative importance of false positives and false negatives.

In some cases, a type II error might have worse consequences than a type I error. Typically, when it comes to the detection of mode interactions, which could have security consequences, it might be preferable to have a higher rate of False Positives than False Negatives, meaning sensitivity is more important than precision, and maximizing the F-measure is not always desirable.

### Multi-class ROC analysis

Managing the entire space becomes more complex when dealing with more than two classes in classification problems. Instead of being a 2x2 matrix, the confusion matrix becomes a nxn matrix, where n represents the number of classes. This matrix contains n correct classifications along the major diagonal and n²-n possible errors in the off-diagonal entries.

One of the common approaches to handling n classes is to generate n different ROC graphs, one for each class. This approach is known as the class reference formulation. ROC graph plots the classification performance using class as the positive class and all other classes as the negative class.

However, this formulation has a limitation, it compromises the insensitivity of ROC to class imbalance. For example, if a certain class becomes more prevalent, it might influence the performance of the classifier for another class. Despite this caveat, the class reference formulation can still be practical and provide reasonable flexibility in evaluation.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, *27*(8), 861–874. https://doi.org/10.1016/j.patrec.2005.10.010

# Implementation

To automatise the detection of mode interactions, a Python package was created. Like most ML projects, the package was implemented in 3 phases: dataset building, pre-processing, and testing.

## The Python Package.

The goal was to create a package containing 6 functions in two Python modules, a demo file and a dataset.

### First Package: No\_ML

The No\_ML package was meant to address the issue of automation with no Machine Learning techniques. It would mimic the reasoning physicists would use to discern interactions. Its functions were:

*Preprocessing(scalogram, frequencies, f0, fend)* which returns the pre-processed scalogram and takes the following arguments:

* *scalogram* is the scalogram (matrix of size [time array size] x [frequency array size])
* *frequencies* is the list of corresponding frequencies.
* *f0* and *fend* are the initial and final frequencies in the conducted frequency sweep.

*interaction2(scalogram, preprocessed =True, threshold=2, frequencies=None, f0=None, fend=None)* returns a dictionary, the keys correspond to the interaction and the values are the interaction score for each interaction. Its parameters are:

* *scalogram* is the scalogram to analyse, it is assumed to be pre-processed, unless one specifies *preprocessed =False* and therefore must give values to the pre-processing parameters frequencies, *f0* and *fend*.

*norm(scalogram, i, preprocessed =True, frequencies=None, f0=None, fend=None)* returns the relative amplitude for the i-th harmonic, with the same arguments as previously.

### Second Package: ML\_method

*interaction(scalogram, model=default, preprocessed =True, frequencies=None, f0=None, fend=None)* has the same purpose as *interaction2* but relies on SVM.

*model(threshold=2, Xy='default', chosen\_kernel='linear', chosen\_C=100, chosen\_gamma='auto')* creates and trains the SVM for the wished threshold. One can specify different kernels but the default is linear.

*Cross\_validate(model, threshold=2, k=5)* returns a Python dictionary containing the accuracy as well as the confusion matrix, precision, specificity and sensitivity for each class.

## Time series: Beam equation

The datasets were created using equations adapted from that of a cantilever beam with a nonlinear spring at the tip in [paper].

With the following notations:

* is the parameter of interest, it is the static force–displacement of the beam resulting from the excitation computed at three different axial coordinates, and therefore a 3x1 matrix.
* is a 3x1 matrix where is the i-th modal response of the system.
* is the mode shape matrix. It is a 3x3 matrix such that is the mode shape of the i-th mode at axial coordinate
* is the mode shape vector at the tip of the cantilever.
* , where is the resonant response frequency of the n-th mode.
* is the damping coefficient.
* was a frequency sweep.
* is the non-linear function that introduces the mode interaction.

Shaw, A. D., Hill, T. R., Neild, S. A., & Friswell, M. I. (2016). Periodic responses of a structure with 3:1 internal resonance. *Mechanical Systems and Signal Processing*, 81, 19–34. <https://doi.org/10.1016/j.ymssp.2016.03.008>

Various values of and were used to create the database. The resonant response frequencies were chosen such that , for .

The time series were computed using the fourth-order Runge-Kutta function of Python’s SciPy library. The wavelet transforms were computed using MATLAB’s cwt function in the Wavelet Toolbox.

## Producing clean data: format and pre-processing

A scalogram corresponds to a list of three-dimensional data, a time value, a frequency, and an amplitude. The format of the scalograms computed by MATLAB is an array containing the values of frequency, and a matrix containing (ai,j), where ai,j is the amplitude of frequency indexed in the frequency array, at time tj.

It makes no difference that the scalogram displays an interaction at time t1 rather than t2. We are interested in whether the scalogram presents interaction at all. The time parameter is, therefore, irrelevant to the identification problem. If the time array was of length n, the scalogram is split into n data points. Each data point is a frequency array and an amplitude array. Figure [?] (a) presents three of these data points.

The scalogram is then scaled along its frequency axis. The amplitude of the excitation frequency should be the same coefficient in the amplitude array. This is generalised to all the amplitude coefficients. The n-th coefficient in the amplitude array is the amplitude of frequency , for . The standard frequency array is now

We chose to stop at because we limited the scope of the project to detecting interactions up to 1:5.

Using the frequency array (fi) and the amplitude array (ai,j), the data points are created by dividing (fi) by the excitation frequency at time tj. We then get and the corresponding amplitudes. Using interpolation, we compute the amplitudes corresponding to , a default value can be used to avoid extrapolation, as seen in Figure [?]. This is essentially what the function *preprocessing(scalogram, frequencies, f0, fend)* does in the Python package.

|  |  |
| --- | --- |
| Une image contenant texte, diagramme, capture d’écran, Tracé  Description générée automatiquement | 1. data points before pre-processing |
| Une image contenant texte, diagramme, capture d’écran, Tracé  Description générée automatiquement | 1. Normalised frequency |
| Une image contenant texte, diagramme, capture d’écran, Tracé  Description générée automatiquement | 1. Interpolated |
| Figure [?] 3 data points through the steps of pre-processing. | |

The last part of the pre-processing was to normalise the amplitudes, such that the amplitude of the excitation frequency would always be 1. The integral of amplitude over the excitation frequency was computed and the amplitude array was then divided by it.

An example of a pre-processed scalograms is presented in Figure [?]

|  |  |
| --- | --- |
| Une image contenant capture d’écran, texte, ligne, Caractère coloré  Description générée automatiquement | Une image contenant texte, capture d’écran, Caractère coloré, ligne  Description générée automatiquement |
| 1. Scalogram | 1. Pre-processed scalogram |
| Fig [?] Dataset presenting 1:2 interactions.  The top right part of (b) correspond to information that is not present in the original scalogram and is therefore replaced with a default value. That value is chosen to be the 25th percentile of the coefficient in the data point. | |

## Class attribution for the training dataset

### Class attribution

In the vectorial space of possible inputs, we define five functions:

* , for

is the amplitude of the i-th harmonic. For each vector of the scalogram, was computed, as seen in Figure [?].

|  |  |  |
| --- | --- | --- |
| Une image contenant capture d’écran, texte, Rectangle, ligne  Description générée automatiquement | Une image contenant texte, capture d’écran, Caractère coloré  Description générée automatiquement | Une image contenant texte, capture d’écran, Tracé, diagramme  Description générée automatiquement |
| 1. Scalogram | 1. Pre-processed scalogram | 1. Amplitudes |
| Fig [?] Dataset presenting 1:3 interactions | | |

The set of possible classes is {0,2,3,4,5}. 0 if there is no interaction, 2 if there is a 1:2 interaction, 3 if there is a 1:3 interaction, etc. However, when forming the dataset, if the time series was obtained integrating a beam equation with , the possible classes for the vectors of the computed scalogram are 0 or 2.

So when forming the training set, if a scalogram should display a interaction, the vectors were classified with the following criteria:

* If , the class of the vector is i.
* Otherwise, the class of the vector is 0.

The choice of the threshold will be a parameter of interest in the next section, and it is a parameter one can choose in the Python package. The No\_ML module uses a similar logic to determine whether a scalogram displays an interaction, it checks if for all , which is unnecessary when it come to the training dataset as we already know which interaction the scalogram should be displaying.

The structure of the dataset is therefore:

* A file *X.csv* which contains 275077 vectors. Each vector corresponds to a time sample from the computed scalogram. The dimension of each vector is 120, accordingly with the pre-processing method.
* A *y.csv* file which contains information on the class of each vector. For each of the 275077 vectors from X, *y.csv* stores , and , with the previous notations. The shape of the y matrix is therefore 275077 x 2.

The dataset is structured that way so that the class of each vector is determined when the user chooses the threshold, meaning when the user defines what constitutes mode interaction.

### Class imbalance

The last issue concerning the training set is class imbalance. As the class of the vectors from the training dataset is only computed once the user chooses a threshold, it is impossible to make the representation of each class in the set balanced prior to choosing said threshold.

Therefore, the function *model(threshold=2, Xy='default', chosen\_kernel='linear', chosen\_C=100, chosen\_gamma='auto')* computes a new X, y such that all classes are represented equally, by getting rid of random samples in overrepresented classes.

Another approach could have been to use Cost-Sensitive Learning. In Machine Learning, the common cost-insensitive approach aims to minimize the probability of error, assuming all misclassifications have the same cost. In cases of significant class imbalance, assigning higher misclassification costs for underrepresented classes makes the SVM's optimisation process biased towards reducing misclassifications on the minority class.

Iranmehr, A., Masnadi-Shirazi, H., & Vasconcelos, N. (2019). Cost-sensitive support vector machines. *Neurocomputing*, *343*, 50-64. <https://doi.org/10.1016/j.neucom.2018.11.099>

Although the cost-sensitive learning approach doesn’t include a loss of information, unlike under-sampling majority classes, it requires more computational resources, which is why I chose the former.

## Format of the results

Two functions are used to assess the scalogram are *interaction(scalogram)* and *interaction2(scalogram, threshold)*. They both return a dictionary, with the interaction number as the key and the corresponding interaction score as the value.

The first function takes a scalogram as input and predicts the probabilities of different interaction classes using a Support Vector Machines. It then computes the interaction scores by summing the probabilities.

*interaction2(scalogram, threshold)* calculates interaction scores based on a given threshold and does not use Support Vector Machines. For each time sample, the amplitude of each harmonic is compared to the threshold. The interaction score for each harmonic is computed by summing the instances where the amplitude is superior to the threshold.

For both functions, the scores are then scaled by the number of time samples. The scores do not correspond to a probability of presence of interaction in the scalogram, but rather to the fraction of the scalogram presenting interaction.

It is moreover likely that the second function will give a higher score than the former as for each time sample, *interaction2* will add 1 to the score if it meets the threshold while *interaction* will return a probability inferior or equal to 1.

## Support Vector Machine Parameters

The last step of implementation was to test the differences in accuracy, precision, sensitivity, and specificity for various parameters:

* Parameters C and γ.
* Choice of the kernel.
* Choice of the threshold.
* Does the automation of detection without ML techniques provide sufficient results.
* Does under-sampling majority classes improve performance metrics significantly?

And adjust the functions and parameters of the Python package accordingly.

# Results and Discussion.

## C and γ

For different values of C and γ, a 5-Fold cross-validation was run. The chosen threshold was 2, and the kernel was RBF. The cross-validation function coded for this project returns the accuracy, as well as the precision, sensitivity, and specificity for each class. The per-class results can be found in Appendix [1].

**Accuracy results**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **C** | 0.1 | 1 | 10 | 100 |
| **γ** |  |  |  |  |  |
| 0.1 |  | 98.76% | 99.52% | 99.72% | 99.87% |
| 1 |  | 97.35% | 99.43% | 99.69% | 99.75% |
| 10 |  | 86.29% | 98.21% | 98.02% | 98.38% |
| 100 |  | 66.95% | 92.40% | 92.83% | 93.23% |

Augmenting parameter C slightly improves the accuracy results, which is to be expected as a higher value of C increases the penalty for misclassifying samples. However, it does not seem to have a big influence on the performance of the algorithm, this usually indicates that the data is well-separated and easily classified, which is confirmed by the fact that the model performs better for low values of γ.

Lower values of γ allow for a smoother decision boundary that is less affected by individual data points. It indicates that the data points are well-separated or have distinct patterns that can be captured by a simpler decision boundary.

Wainer, J., Fonseca, P. How to tune the RBF SVM hyperparameters? An empirical evaluation of 18 search algorithms. *Artif Intell Rev* **54**, 4771–4797 (2021). <https://doi.org/10.1007/s10462-021-10011-5>

In order to avoid overfitting, the data when choosing the parameter C, we will look at the variance and bias trade-off for different values of C. The results were very surprising. I expected the variance to increase with the hyperparameter C. [?] As a higher value of C improves the accuracy, and therefore reduces biasThe fact that it didn’t indicate that the

Mehta, P., Bukov, M., Wang, C., Day, A. G., Richardson, C., Fisher, C. K., & Schwab, D. J. (2019). A high-bias, low-variance introduction to Machine Learning for physicists. *Physics Reports*, *810*, 1-124. https://doi.org/10.1016/j.physrep.2019.03.001

|  |
| --- |
| Une image contenant texte, capture d’écran, diagramme, ligne  Description générée automatiquement |
| Figure [?] Bias variance trade-off for different values of C. |

The rest of the testing will be conducted using C=100 and γ=0.1

ecific data distributions: Certain data distributions may exhibit peculiar characteristics that cause the variance of the SVM to decrease with increasing C. For example, if the data is linearly separable with a large margin, increasing C may have little effect on the decision boundary and the variance.

The rest of the testing will be conducted using C=100 γ=0.1.

## Choice of kernel

The accuracy results for the RBF, linear and polynomial kernels are very similar.

|  |  |
| --- | --- |
| **Kernels** | **Accuracy** |
| Linear | 99.87% |
| Polynomial | 99.46% |
| RBF | 99.79% |
| Sigmoid | 72.60% |

It makes sense that the linear kernel would have a good accuracy as each sub-section of the training set was classified using the following criteria:

Where and are linear functions.

The precision, sensitivity and specificity were computed for the RBF and linear kernels. They do not significantly differ. The RBF has a slightly better sensitivity for interaction classes. The RBF kernel was the one used for further testing.

|  |  |  |
| --- | --- | --- |
| **Kernel** | **Linear** | **RBF** |
| Class 0:  No interaction | precision: 99.93%  sensitivity: 99.49%  specificity: 99.98% | precision: 100%  sensitivity: 98.98%  specificity: 100% |
| Class 2:  1:2 interaction | precision: 99.93%  sensitivity: 100%  specificity: 99.98% | precision: 99.96%  sensitivity: 100%  specificity: 99.99% |
| Class 3:  1:3 interaction | precision: 99.78%  sensitivity: 99.96%  specificity: 99.95% | precision: 99.82%  sensitivity: 100%  specificity: 99.95% |
| Class 4:  1:4 interaction | precision: 99.85%  sensitivity: 100%  specificity: 99.96% | precision': 99.53%  sensitivity': 100%  specificity': 99.88% |
| Class 5:  1:5 interaction | precision: 99.93%  sensitivity: 99.96%  specificity: 99.98% | precision: 99.67%  sensitivity: 100%  specificity: 99.92% |

## Choice of threshold

### Threshold on the training data

One must remark that the following graphs are an analogy to ROC graphs and not exactly ROC graphs. In actual ROC testing, X and y are known and do not change with the change of threshold. The same test is conducted on X and predicted class ypred depends on whether the test meets the threshold.

In this case, the Support Vector Machines do not produce a result that must be compared with the threshold but rather they produce ypred. The threshold intervenes in classifying the training data, i.e., in computing y.

Therefore, the choice of threshold should not significantly impact the overall performance of the model. The interest is in comparing the relative impact of the threshold on the classes, as choosing a lower threshold means increasing the size of the dataset for interaction classes.

|  |
| --- |
| Une image contenant texte, capture d’écran, affichage, logiciel  Description générée automatiquement |
| Figure [?] False Positives and True Positives for various thresholds and classes. |

As expected, the choice of threshold does not significantly change the

### Threshold on probabilities

One way to implement actual ROC testing with Support Vector Machines, or other classification algorithms, is to have it compute, for each sample, the probability of it belonging to each class rather than simply give the class with the highest probability. The threshold would then be 0<p<1 such that if the probability of the sample belonging to one class is superior to p, the sample belongs to said class.

Choosing a low threshold for p could be interesting in a detection problem as it could have more cost to miss interactions than it would be to have a ‘No interaction’ sample misclassified.

Une image contenant texte, ligne, diagramme, Tracé

Description générée automatiquement

|  |  |
| --- | --- |
| Une image contenant texte, capture d’écran, ligne, affichage  Description générée automatiquement   1. Class: No Interaction | Une image contenant texte, capture d’écran, diagramme, Tracé  Description générée automatiquement(b) Class: 1:2 interaction |
| Une image contenant texte, capture d’écran, diagramme, Rectangle  Description générée automatiquement (c) Class: 1:3 interaction | Une image contenant texte, capture d’écran, diagramme, Tracé  Description générée automatiquement(d) Class: 1:4 interaction |
| Une image contenant texte, capture d’écran, ligne, Tracé  Description générée automatiquement  (e) Class: 1:5 interaction | |
| Figure [?] ROC graph. Threshold on probabilities | |

Blah

## Under-sampling majority classes

Two cross-validations were run for the same model as described previously, the kernel is RBF with C=100 and γ=0.1.

|  |  |  |  |
| --- | --- | --- | --- |
| **Performance metrics** | | **Under-sampling** | **Class Imbalance** |
| **Accuracy** | | 99.80% | 99.95% |
| **Class 0**  **No Interaction** | Precision | 100% | 99.96% |
| Sensitivity | 98.98% | 99.96% |
| Specificity | 100% | 99.90% |
| **Class 2**  **1:2 interaction** | Precision | 99.89% | 99.64% |
| Sensitivity | 100% | 99.93% |
| Specificity | 99.97% | 100% |
| **Class 3**  **1:3 interaction** | Precision | 99.75% | 99.85% |
| Sensitivity | 100% | 99.87% |
| Specificity | 99.94% | 99.99% |
| **Class 4**  **1:4 interaction** | Precision | 99.64% | 99.93% |
| Sensitivity | 100% | 99.85% |
| Specificity | 99.91% | 99.99% |
| **Class 5**  **1:5 interaction** | Precision | 99.71% | 99.95% |
| Sensitivity | 100% | 99.94% |
| Specificity | 99.93% | 99.99% |

Under-sampling does not improve the model’s accuracy. However, in class-imbalanced scenarios, accuracy alone does not accurately assess the model's performance. A classifier that predicts only the majority class can achieve high accuracy, even if it fails to identify instances from the minority class correctly.

Under-sampling did slightly improve the sensitivity to interactions, which is the aim of this project. Moreover, it has significantly reduced the time necessary to run the training and testing algorithms used throughout this project.

## Experimental data & Comparison between the ML and No\_ML methods

The models were used on experimental data. The proposed method is applied to a wing-engine structure with nonlinear connections. The time series exploited are that from [?]. They are provided by accelerometers mounted on the top of the wing plate measuring vertical acceleration and on two pylons measuring horizontal acceleration.

|  |
| --- |
| Une image contenant capture d’écran  Description générée automatiquement |
| Figure [?] Wing-engine structure from [?]. |

Song, M., Renson, L., Moaveni, B., & Kerschen, G. (2022). Bayesian model updating and class selection of a wing-engine structure with nonlinear connections using nonlinear normal modes. *Mechanical Systems and Signal Processing*, *165*, 108337. <https://doi.org/10.1016/j.ymssp.2021.108337>

All the results and graphs can be found in Appendix [?]. The ones exploited here correspond to the sensors:

* Sensor 4 measures acceleration in the Z direction.
* Sensor 7 measures acceleration in the Z direction where the single point shaker is located.
* Sensor 11 measures acceleration in the X direction.

|  |  |  |
| --- | --- | --- |
|  |  | Une image contenant texte, capture d’écran, conception  Description générée automatiquement |
| Sensor 4 | Sensor 7 | Sensor 11 |
| Figure [?] Pre-processed scalograms | | |

It seems like Sensor 4 exhibits signs of 1:2 and 1:3 interactions, Sensor 7 exhibits signs of 1:5 interactions and Sensor 11 exhibits signs of 1:2 and 1:5 interactions. For each of these scalograms, was plotted throughout time as well as the results provided by the No\_ML and ML methods, they can be found in Appendix [?].

|  |  |
| --- | --- |
| Une image contenant texte, capture d’écran, diagramme, Tracé  Description générée automatiquement | Une image contenant texte, capture d’écran, diagramme, Tracé  Description générée automatiquement |
|  | Comparison between ML and No\_ML methods |
| Figure [?] Sensor 4 results | |

Both the ML and No\_ML methods detect the presence of 1:2 and 1:3 interactions. However, there is a discrepancy in the scores of the ML method and No\_ML methods. It can be attributed to the different computational approaches employed for calculating these scores, as explained in the section "Format of the results."

|  |  |
| --- | --- |
| Une image contenant texte, capture d’écran, diagramme, Tracé  Description générée automatiquement | Une image contenant texte, capture d’écran, diagramme, Tracé  Description générée automatiquement |
|  | Comparison between ML and No\_ML methods |
| Figure [?] Sensor 7 results | |

Both the ML and No\_ML methods identify the 1:5 interaction. The relative low values of those scores (around 8%) does not correspond to a probability of presence of interaction, but rather to the fraction of the scalogram presenting interaction. As seen in the scalogram, most of the samples only shows the presence of harmonics (before sample 50,000 and after sample 150,000).

|  |  |
| --- | --- |
| Une image contenant texte, capture d’écran, diagramme, Tracé  Description générée automatiquement | Une image contenant texte, capture d’écran, diagramme, Tracé  Description générée automatiquement |
|  | Comparison between ML and No\_ML methods |
| Figure [?] Sensor 11 results | |

The ML and No\_ML methods give different results for Sensor 11. The interactions displayed on the scalogram in Figure [?] are probably 1:2 and 1:5. As identified by the ML\_method. The No\_ML method identifies a 1:3 interaction due to the high amplitude of the third harmonic. However a high third harmonic is also to be expected in a 1:5 interaction scalogram.

As explained in the section "Format of the results", the logic used in No\_ML algorithm is quite simplified in that it considers the amplitudes of the different harmonics independently. Therefore, it interprets a high amplitude of the third harmonic as a 1:3 interaction. For comparison, Figure [?] displays one of the 1:5 scalograms used to train the model.

|  |  |
| --- | --- |
| Une image contenant texte, capture d’écran, jaune, Caractère coloré  Description générée automatiquement | Une image contenant capture d’écran, texte, Caractère coloré, jaune  Description générée automatiquement |
| 1. Scalogram | 1. Pre-processed scalogram |
|  | |
|  | |
| Figure [?] 1:5 interaction | |

# Conclusions and Future Work

The methodology used in the project was designed to be adaptable to different datasets, different thresholds of what constitutes modal interaction, and different types of interaction. The test\_parameters file enables the user to finetune the parameters for the chosen dataset.

The good performance of the models is due to a pre-processing that simplified the data to an extent that a non-machine learning approach gave satisfactory results as well as a lack of variety in the training dataset. Although different parameters were used (damping, forcing, polynomial coefficients of the non-linear function), it would be interesting in future work to derive time series from a more diverse set of equations.

Surprisingly, the model's performance did not improve significantly when addressing class imbalance. However, it revealed that the model was sensitive to the size of the training data set. The project would have benefitted of a larger training data set l, although it would require more computational power.

The No\_ML module seems to give good results as well. It was able to flag that there is an interaction but was less good at discriminating which interaction occurs. The logic used in this algorithm is quite simplified in that it considers the amplitudes of the different harmonics independently. The algorithm could be refined so as to give results as satisfactory as that of the Support Vector Machines.

Future work could involve:

1. Creating a larger and more diverse training dataset.
2. Enlarging the scope of the project to incorporate more interactions, such as 2:3, 2:5, 3:4, etc.
3. Adapting the scaling function to be able to handle different frequency sweeps, such as logarithmic, step and chirp sweeps.
4. Reducing the dimension of the dataset using unsupervised learning techniques.
5. Complexifying the No\_ML method to enable it to discriminate the interactions. It should compare harmonic amplitudes, not only with that of the excitation frequency but also with each other.

# References

(see Appendix D).

# Appendices

## Appendix 1: performance per class for C and γ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Class 0 | **C** | 0.1 | 1 | 10 | 100 |
| **γ** |  |  |  |  |  |
| 0.1 |  | precision = 100%  specificity = 100%  sensitivity = 93.80% | precision = 100%  specificity = 100%  sensitivity = 97.59% | precision = 100%  specificity = 100%  sensitivity = 98.61% | precision = 99.93%  specificity = 99.98%  sensitivity = 99.42% |
| 1 |  | precision = 100%  specificity = 100%  sensitivity = 94.46% | precision = 99.93%  specificity = 99.98%  sensitivity = 97.34% | precision = 100%  specificity = 100%  sensitivity = 98.54% | precision = 99.96%  specificity = 100%  sensitivity = 98.83% |
| 10 |  | precision = 100%  specificity = 100%  sensitivity = 88.44% | precision = 100%  specificity = 100%  sensitivity = 95.84% | precision = 99.92%  specificity = 99.98%  sensitivity = 96.14% | precision = 99.89%  specificity = 99.97%  sensitivity =96.35% |
| 100 |  | precision = 100%  specificity = 100%  sensitivity = 40.17% | precision = 100%  specificity = 100%  sensitivity = 85.93% | precision = 99.92%  specificity = 99.98%  sensitivity = 88.44% | precision = 100%  specificity = 100%  sensitivity = 88.30% |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Class 2 | **C** | 0.1 | 1 | 10 | 100 |
| **γ** |  |  |  |  |  |
| 0.1 |  | precision = 99.17%  specificity = 99.79%  sensitivity = 100% | precision = 99.82%  specificity = 99.95%  sensitivity = 100% | precision = 99.96%  specificity = 99.99%  sensitivity = 100% | precision = 99.93%  specificity = 99.98%  sensitivity = 100% |
| 1 |  | precision = 99.71%  specificity = 99.93%  sensitivity = 100% | precision = 99.82%  specificity = 99.95%  sensitivity = 100% | precision = 99.96%  specificity = 99.99%  sensitivity = 100% | precision = 99.96%  specificity = 99.99%  sensitivity = 100% |
| 10 |  | precision = 99.96%  specificity = 99.99%  sensitivity = 98.10% | precision = 100%  specificity = 100%  sensitivity = 100% | precision = 99.89%  specificity = 99.97%  sensitivity = 100% | precision = 100%  specificity = 100%  sensitivity = 100% |
| 100 |  | precision = 100%  specificity = 100%  sensitivity = 74.81% | precision = 100%  specificity = 100%  sensitivity = 99.93% | precision = 99.96%  specificity = 99.99%  sensitivity = 100% | precision = 99.93%  specificity = 99.98%  sensitivity = 99.93% |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Class 3 | **C** | 0.1 | 1 | 10 | 100 |
| **γ** |  |  |  |  |  |
| 0.1 |  | precision = 98.67%  specificity = 99.66%  sensitivity = 100% | precision = 99.53%  specificity = 99.88%  sensitivity = 100% | precision = 99.49%  specificity = 99.87%  sensitivity = 100% | precision = 99.71%  specificity = 99.27%  sensitivity = 99.93% |
| 1 |  | precision = 99.71%  specificity = 99.93%  sensitivity = 100% | precision = 99.20%  specificity = 99.80%  sensitivity = 99.82% | precision = 99.71%  specificity = 99.93%  sensitivity = 99.93% | precision = 99.82%  specificity = 99.95%  sensitivity = 99.93% |
| 10 |  | precision = 59.63%  specificity = 83.08%  sensitivity = 100% | precision = 99.92%  specificity = 99.98%  sensitivity = 95.77% | precision = 99.88%  specificity = 99.97%  sensitivity = 94.35% | precision = 99.92%  specificity = 99.98%  sensitivity = 95.81% |
| 100 |  | precision = 37.73%  specificity = 58.74%  sensitivity = 100% | precision = 99.91%  specificity = 99.98%  sensitivity = 78.27% | precision = 99.86%  specificity = 99.97%  sensitivity = 77.73% | precision = 100%  specificity = 100%  sensitivity = 79.88% |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Class 4 | **C** | 0.1 | 1 | 10 | 100 |
| **γ** |  |  |  |  |  |
| 0.1 |  | precision = 97.48%  specificity = 99.35%  sensitivity = 100% | precision = 98.74%  specificity = 99.68%  sensitivity = 100% | precision = 99.53%  specificity = 99.88%  sensitivity = 100% | precision = 99.82%  specificity = 99.95%  sensitivity = 100% |
| 1 |  | precision = 98.79%  specificity = 99.70%  sensitivity = 98.29% | precision = 99.49%  specificity = 99.87%  sensitivity = 100 | precision = 99.64%  specificity = 99.91%  sensitivity = 100% | precision = 99.75%  specificity = 99.94%  sensitivity = 100% |
| 10 |  | precision = 99.58%  specificity = 99.90%  sensitivity = 96.10% | precision = 99.71%  specificity = 99.93%  sensitivity = 99.45% | precision = 997.81%  specificity = 99.95%  sensitivity = 99.60% | precision = 99.89%  specificity = 99.97%  sensitivity = 99.78% |
| 100 |  | precision = 99.92%  specificity = 99.98%  sensitivity = 90.52% | precision = 99.93%  specificity = 99.98%  sensitivity = 97.89% | precision = 99.93%  specificity = 99.98%  sensitivity = 97.99% | precision = 99.96%  specificity = 99.99%  sensitivity = 98.07% |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Class 5 | **C** | 0.1 | 1 | 10 | 100 |
| **γ** |  |  |  |  |  |
| 0.1 |  | precision = 98.60%  specificity = 99.64%  sensitivity = 100% | precision = 99.53%  specificity = 99.88%  sensitivity = 100% | precision = 99.64%  specificity = 99.91%  sensitivity = 100% | precision = 99.96%  specificity = 99.99%  sensitivity = 100% |
| 1 |  | precision = 99.35%  specificity = 99.85%  sensitivity = 94.02% | precision = 98.74%  specificity = 99.68%  sensitivity = 100% | precision = 99.17%  specificity = 99.79%  sensitivity = 100% | precision = 99.28%  specificity = 99.82%  sensitivity = 100% |
| 10 |  | precision = 99.19%  specificity = 99.90%  sensitivity = 48.82% | precision = 92.11%  specificity = 97.86%  sensitivity = 100% | precision = 91.40%  specificity = 97.65%  sensitivity = 100% | precision = 92.76%  specificity = 98.05%  sensitivity = 99.96% |
| 100 |  | precision = 99.50%  specificity = 99.96%  sensitivity = 29.24% | precision = 72.55%  specificity = 90.54%  sensitivity = 100% | precision = 73.78%  specificity = 91.11%  sensitivity = 100% | precision = 74.78%  specificity = 91.56%  sensitivity = 100% |

## Appendix 2: Pre-processed Scalogram of Wing Engine

|  |  |  |  |
| --- | --- | --- | --- |
| Une image contenant texte, capture d’écran, Police  Description générée automatiquement  Sensor 1 | Sensor 2 | | Une image contenant texte, capture d’écran, ligne, Rectangle  Description générée automatiquement Sensor 3 |
|  |  | |  |
| Une image contenant texte, capture d’écran, Rectangle, Caractère coloré  Description générée automatiquement Sensor 4 | Une image contenant texte, capture d’écran, Rectangle, Police  Description générée automatiquement Sensor 5 | | Une image contenant texte, capture d’écran, Police, Rectangle  Description générée automatiquement Sensor 6 |
|  |  | |  |
| Une image contenant texte, capture d’écran, Rectangle  Description générée automatiquement  Sensor 7 | Une image contenant texte, capture d’écran, diagramme  Description générée automatiquement Sensor 8 | | Une image contenant texte, capture d’écran  Description générée automatiquement Sensor 9 |
|  |  | |  |
| Une image contenant texte, capture d’écran, Police, conception  Description générée automatiquement  Sensor 10 | | Une image contenant texte, capture d’écran, conception  Description générée automatiquement  Sensor 11 | |
|  | |  | |

## Appendix 3: and algorithm results of Wing Engine

|  |  |
| --- | --- |
|  |  |
|  | Comparison between ML and No\_ML methods |
| **Sensor 1** | |

|  |  |
| --- | --- |
| Une image contenant capture d’écran, texte, Tracé, diagramme  Description générée automatiquement | Une image contenant capture d’écran, texte, Tracé, diagramme  Description générée automatiquement |
|  | Comparison between ML and No\_ML methods |
| **Sensor 2** | |

|  |  |
| --- | --- |
|  | Une image contenant capture d’écran, texte, Tracé, diagramme  Description générée automatiquement |
|  | Comparison between ML and No\_ML methods |
| **Sensor 3** | |

|  |  |
| --- | --- |
| Une image contenant texte, capture d’écran, diagramme, Tracé  Description générée automatiquement |  |
|  | Comparison between ML and No\_ML methods |
| **Sensor 4** | |

|  |  |
| --- | --- |
|  |  |
|  | Comparison between ML and No\_ML methods |
| **Sensor 5** | |

|  |  |
| --- | --- |
|  | Une image contenant texte, capture d’écran, diagramme, Tracé  Description générée automatiquement |
|  | Comparison between ML and No\_ML methods |
| **Sensor 6** | |

|  |  |
| --- | --- |
| Une image contenant texte, capture d’écran, diagramme, Tracé  Description générée automatiquement | Une image contenant texte, capture d’écran, diagramme, Tracé  Description générée automatiquement |
|  | Comparison between ML and No\_ML methods |
| **Sensor 7** | |

|  |  |
| --- | --- |
| Une image contenant texte, capture d’écran, diagramme, Tracé  Description générée automatiquement | Une image contenant texte, capture d’écran, diagramme, Tracé  Description générée automatiquement |
|  | Comparison between ML and No\_ML methods |
| **Sensor 8** | |

|  |  |
| --- | --- |
| Une image contenant texte, capture d’écran, diagramme, Tracé  Description générée automatiquement | Une image contenant texte, capture d’écran, diagramme, Tracé  Description générée automatiquement |
|  | Comparison between ML and No\_ML methods |
| **Sensor 9** | |

|  |  |
| --- | --- |
| Une image contenant texte, capture d’écran, diagramme, Tracé  Description générée automatiquement | Une image contenant texte, capture d’écran, diagramme, Tracé  Description générée automatiquement |
|  | Comparison between ML and No\_ML methods |
| **Sensor 10** | |

|  |  |
| --- | --- |
| Une image contenant texte, capture d’écran, diagramme, Tracé  Description générée automatiquement | Une image contenant texte, capture d’écran, diagramme, Tracé  Description générée automatiquement |
|  | Comparison between ML and No\_ML methods |
| **Sensor 11** | |

## Appendix 4: ROC graph threshold on probabilities

|  |  |
| --- | --- |
| Une image contenant texte, capture d’écran, affichage, nombre  Description générée automatiquement  Threshold = 1.5 | Une image contenant texte, capture d’écran, affichage, nombre  Description générée automatiquement  Threshold = 1.75 |
|  |  |
| Une image contenant texte, capture d’écran, affichage, nombre  Description générée automatiquementThreshold = 2 | Une image contenant texte, capture d’écran, affichage, nombre  Description générée automatiquementThreshold = 2.25 |
|  |  |
| Une image contenant texte, capture d’écran, affichage, diagramme  Description générée automatiquementThreshold = 2.5 | Une image contenant texte, capture d’écran, affichage, nombre  Description générée automatiquementThreshold = 2.75 |
|  |  |
| Une image contenant texte, capture d’écran, affichage, nombre  Description générée automatiquementThreshold = 3 |  |