

AN5392

Application note

ISM330DHCX: Machine Learning Core

Introduction

This document is intended to provide information on the Machine Learning Core feature available in the ISM330DHCX. The Machine Learning processing capability allows moving some algorithms from the application processor to the MEMS sensor, enabling consistent reduction of power consumption.

The Machine Learning processing capability is obtained through decision-tree logic. A decision tree is a mathematical tool composed of a series of configurable nodes. Each node is characterized by an "if-then-else" condition, where an input signal (represented by statistical parameters calculated from the sensor data) is evaluated against a threshold.

The ISM330DHCX can be configured to run up to 8 decision trees simultaneously and independently. The decision trees are stored in the device and generate results in the dedicated output registers.

The results of the decision tree can be read from the application processor at any time. Furthermore, there is the possibility to generate an interrupt for every change in the result in the decision tree.

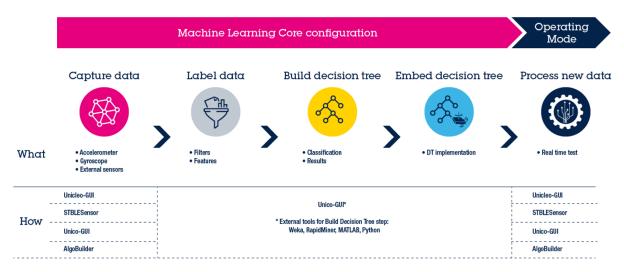


Figure 1. Machine Learning Core supervised approach

The Machine Learning Core (together with the Finite State Machine) is one of the main embedded features available in the ISM330DHCX. It is composed of a set of configurable parameters and decision trees able to implement algorithms in the sensor itself.

The kind of algorithms suitable for the Machine Learning Core are those which can be implemented by following an inductive approach, which involves searching patterns from observations. Some examples of algorithms which follows this approach are: activity recognition, fitness activity recognition, motion intensity detection, vibration intensity detection, carrying position recognition, context awareness, false positive rejection, etc.

The idea behind the Machine Learning Core is to use the accelerometer, gyroscope and external sensor data (readable through the I²C master interface) to compute a set of statistical parameters selectable by the user (such as mean, variance, energy, peak, zero crossing, etc.) in a defined time window. In addition to the sensor input data, some new inputs can be defined by applying some configurable filters available in the device.

The Machine Learning Core parameters are called "Features" and can be used as input for a configurable decision tree which can be stored in the device.

The decision tree which can be stored in the ISM330DHCX is a binary tree composed of a series of nodes. In each node, a statistical parameter (feature) is evaluated against a threshold to establish the evolution in the next node. When a leaf (one of the last nodes of the tree) is reached, the decision tree generates a result which is readable through a dedicated device register.

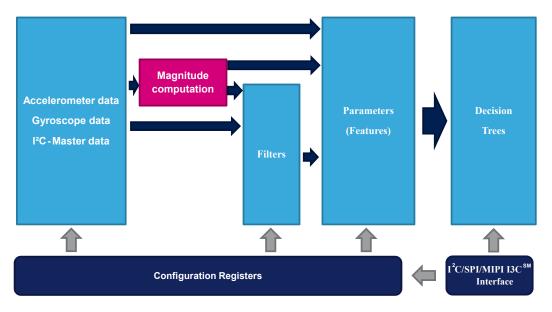


Figure 2. Machine Learning Core in the ISM330DHCX

The Machine Learning Core output data rate can be configured among one of the four available rates from 12.5 to 104 Hz. The bits MLC_ODR in the embedded function register EMB_FUNC_ODR_CFG_C (60h) allow selecting one of the four available rates as shown in the following table.

MLC_ODR bits in EMB_FUNC_ODR_CFG_C (60h)	Machine Learning Core output data rate
00	12.5 Hz
01	26 Hz (default)
10	52 Hz
11	104 Hz

Table 1. Machine Learning Core output data rates

In order to implement the machine learning processing capability of the ISM330DHCX, it is necessary to use a "supervised learning" approach which consists of:

- identifying some classes to be recognized;
- collecting multiple data logs for each class;
- performing some data analysis from the collected logs to learn a generic rule which allows mapping inputs (data logs) to ouputs (classes to be recognized).

In an activity recognition algorithm, for instance, the classes to be recognized might be: stationary, walking, jogging, biking, driving, etc. Multiple data logs have to be acquired for every class, e.g. multiple people performing the same activity.

The analysis on the collected data logs has the purpose of:

- defining the features to be used to correctly classify the different classes;
- defining the filters to be applied to the input data to improve the performance using the selected features;
- generating a dedicated decision tree able to recognize one of the different classes (mapping inputs to outputs).

Once a decision tree has been defined, a configuration for the device can be generated by the software tool provided by STMicroelectronics (described in Section 2 Machine Learning Core tools). The decision tree will run on the device, minimizing the power consumption.

Going deeper in detail on the Machine Learning Core feature inside the ISM330DHCX, it can be thought of as three main blocks (Figure 3):

- 1. Sensor data
- 2. Computation block
- 3. Decision tree

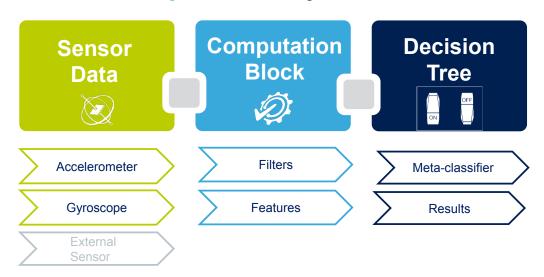


Figure 3. Machine Learning Core blocks

The first block, called "Sensor Data", is composed of data coming from the accelerometer and gyroscope which are built in the device, or from an additional external sensor which might be connected to the ISM330DHCX through the I²C master interface (sensor hub).

The Machine Learning Core inputs defined in the first block are used in the second block, the "Computation Block", where filters and features can be applied. The features are statistical parameters computed from the input data (or from the filtered data) in a defined time window, selectable by the user.

The features computed in the computation block will be used as input for the third block of the Machine Learning Core. This block, called "Decision Tree", includes the binary tree which evaluates the statistical parameters computed from the input data. In the binary tree the statistical parameters are compared against certain thresholds to generate results (in the example of the activity recognition described above, the results were: stationary, walking, jogging, biking, etc.). The decision tree results might also be filtered by an optional filter called "Meta-classifier". The Machine Learning Core results will be the decision tree results which include the optional meta-classifier.

The Machine Learning Core memory is organized in a "dynamic" or "modular" way, in order to maximize the number of computation blocks which can be configured in the device (filters, features, etc.). A dedicated tool has been designed to generate the configuration of the ISM330DHCX, in order to automatically manage memory usage. The tool is available in the Unico GUI and it is described later in Section 2 Machine Learning Core tools. The following sections explain in detail the three main blocks of the Machine Learning Core in the ISM330DHCX described in Figure 3.

1.1 Inputs

The ISM330DHCX works as a combo (accelerometer + gyroscope) sensor, generating acceleration and angular rate output data. The 3-axis data of the acceleration and angular rate can be used as input for the Machine Learning Core. Figure 4 and Figure 5 show the inputs of the Machine Learning Core block in the accelerometer and gyroscope digital chains. The position of the Machine Learning Core (MLC) block in the two digital chains is the same for all four connection modes available in the ISM330DHCX.

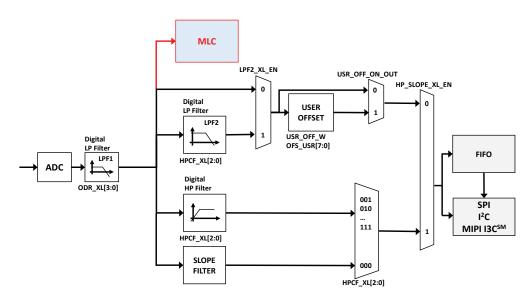
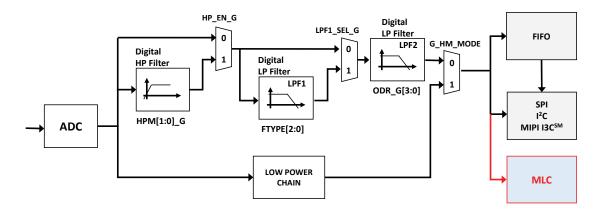


Figure 4. MLC inputs (accelerometer)

Figure 5. MLC inputs (gyroscope)



The rate of the input data must be equal to or higher than the Machine Learning Core data rate configurable through the embedded function register EMB_FUNC_ODR_CFG_C (60h), as described in Table 1.

Example: In an activity recognition algorithm running at 26 Hz, the Machine Learning Core ODR must be selected at 26 Hz, while the sensor ODRs must be equal to or higher than 26 Hz.

The Machine Learning Core uses the following unit conventions:

- Accelerometer data in [g]
- Gyroscope data in [rad/sec]
- External sensor data in [Gauss] for a magnetometer, [Bar] for a pressure sensor

Since it is possible to connect an external sensor (e.g. magnetometer) to the ISM330DHCX through the Sensor Hub feature (Mode 2), the data coming from an external sensor can also be used as input for machine learning processing. In this case, the first six sensor hub bytes (two bytes per axis) are considered as input for the MLC. In other words, the MLC will directly take as input the content of the ISM330DHCX registers from SENSOR_HUB_1 (02h) to SENSOR_HUB_6 (07h).

When using an external sensor, the sensitivity of the external sensor has to be set through registers MLC_MAG_SENSITIVITY_L (E8h) and MLC_MAG_SENSITIVITY_H (E9h).

Example: For a magnetometer like the LIS2MDL, the sensitivity is 1.5 mG/LSB, which becomes 0.0015 G/LSB after converting it to Gauss, and becomes 1624h converted as HFP (half-precision floating point value for the ISM330DHCX sensitivity registers).

Sensitivity [mG/LSB]	Sensitivity [G/LSB]	Sensitivity HFP
1.5 mG/LSB	0.0015 G/LSB	1624h

Note: The half-precision floating-point format is expressed as:

The following procedure allows changing the conversion factor for the external magnetometer data:

- 1. Write 80h to register 01h // Enable access to the embedded function registers
- Write 40h to register 17h // PAGE_RW (17h) = '40h': enable write transaction
- 3. Write 11h to register 02h // PAGE_SEL (02h) = '11h'
- Write E8h to register 08h // PAGE_ADDRESS (08h) = 'E8h'
- 5. Write [LSB] conversion factor (LIS2MDL example, 24h) to register 09h
- 6. Write 11h to register 02h // PAGE_SEL (02h) = '11h'
- 7. Write E9h to register 08h // PAGE_ADDRESS (08h) = 'E9h'
- 8. Write [MSB] conversion factor (LIS2MDL example, 16h) to register 09h
- 9. Write 00h to register 17h // PAGE_RW (17h) = '00h': disable read / write transaction
- 10. Write 00h to register 01h // Disable access to the embedded function registers

The example of the procedure above to change the sensitivity for the external sensor is included in the configuration generated by the Machine Learning Core tool (described in Section 2 Machine Learning Core tools), so the user just needs to set a sensitivity value in the GUI, which will be translated in the register setting by the software.

To summarize the Machine Learning Core inputs:

- Accelerometer data conversion factor is automatically handled by the device;
- Gyroscope data conversion factor is automatically handled by the device;
- External sensor data conversion factor is not automatically handled by the device. A conversion factor must be set by the user in order to make the Machine Learning Core work with the correct unit of measurement.

An additional input available for all sensor data (accelerometer, gyroscope, and external sensor) is the norm. From the 3-axis data the Machine Learning Core (in the ISM330DHCX) internally computes the norm and the norm squared. These two additional signals can be used as inputs for machine learning processing.

The norm and the norm squared of the input data are computed with the following formulas:

$$V = \sqrt{x^2 + y^2 + z^2}$$
$$V^2 = x^2 + y^2 + z^2$$

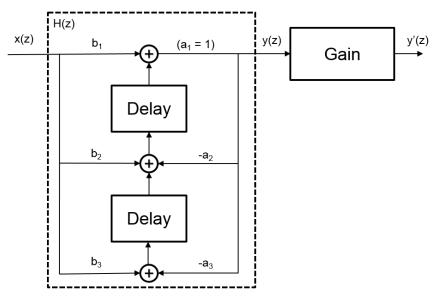
Norm and norm squared data can be used in the decision trees in order to guarantee a high level of program customization for the user.

1.2 Filters

The input data seen in the previous section can be filtered by different kinds of filters available in the Machine Learning Core logic. The basic element of the Machine Learning Core filtering is a second order IIR filter, as shown in Figure 6.

Note: The filters available in the MLC block are independent of any other filter available on the device (the filters described in this section are illustrated in the MLC block of Figure 1 and Figure 2).





The transfer function of the generic IIR 2nd order filter is the following:

ŀ

$$I\left(z\right) = \frac{b_1 + b_2 z^{-1} + b_3 z^{-2}}{1 + a_2 z^{-1} + a_3 z^{-2}}$$

From Figure 6, the outputs can be defined as:

$$y(z) = H(z) \cdot x(z)$$

$$y'(z) = y(z) \cdot Gain$$

To optimize memory usage, the Machine Learning Core has default coefficients for the different kinds of filters (high-pass, band-pass, IIR1, IIR2). The Machine Learning Core tool will help in configuring the filter by asking for the filter coefficients needed after selecting the kind of filter. The following table shows the default values and the configurable values for the coefficients, depending on the filter type choosen. By setting different coefficients it is possibile to tune the filter for the specific application.

Table 2	2. Filter	coefficients
---------	-----------	--------------

Filter type / Coefficients	b ₁	b ₂	b ₃	a ₂	a ₃	Gain
High-pass filter	0.5	-0.5	0	0	0	1
Band-pass filter	1	0	-1	Configurable	Configurable	Configurable
IIR1 filter	Configurable	Configurable	0	Configurable	0	1
IIR2 filter	Configurable	Configurable	Configurable	Configurable	Configurable	1

1.2.1 Filter coefficients

The IIR filter coefficients can be computed with different tools, including Matlab, Octave and Python. In Matlab, for instance, the following function can be used to generate coefficients for a low-pass filter:

[b, a] = butter(N,f_cut / (ODR/2), 'low')

Where:

- N is the order of the IIR filter (1 for IIR1, 2 for IIR2)
- f_cut is the cutoff frequency [Hz] of the filter
- ODR is the Machine Learning Core data rate [Hz]
- 'low' (or 'high') is the kind of filter to be implemented (low-pass or high-pass)

Note: It is possible to configure a high-pass filter with the cutoff at half of the bandwidth (ODR/4) without inserting the coefficients. The Machine Learning Core has some pre-defined coefficient for this configuration.

The following function instead allows generating band-pass filter coefficients through Matlab:

[b,a] = butter(1,[f1 f2]/(ODR/2), 'bandpass')

Note: Since only a2, a3 and Gain are configurable for a band-pass filter, the b vector should be normalized by setting Gain = b(1).

Example:

b = [0.2929 0 -0.2929]; a = [1.0 -0.5858 0.4142];

can be written as $b = [1 \ 0 \ -1]$ and Gain = 0.2929.

So the band-pass filter coefficients will be:

a2 = -0.5858; a3 = 0.4142; Gain = 0.2929.

The following table shows some examples of filter coefficients (most of them considering an ODR of 26 Hz).

Filter type / Coefficients	b ₁	b ₂	b ₃	a ₂	a ₃	Gain
High-pass IIR1, f_cut = 1 Hz, ODR = 26 Hz	0.891725	-0.891725	-	-0.783450	-	1
High-pass IIR1, f_cut = 2 Hz, ODR = 26 Hz	0.802261	-0.802261	-	-0.604521	-	1
High-pass IIR1, f_cut = 5 Hz, ODR = 26 Hz	0.591628	-0.591628	-	-0.183257	-	1
High-pass IIR1, f_cut = 10 Hz, ODR = 26 Hz	0.274968	-0.274968	-	0.450063	-	1
High-pass IIR2, f_cut = 1 Hz, ODR = 26 Hz	0.8428435	-1.685687	0.8428435	-1.6608344	0.710540	1
High-pass IIR2, f_cut = 2 Hz, ODR = 26 Hz	0.709560	-1.419120	0.709560	-1.332907	0.505334	1
High-pass IIR2, f_cut = 5 Hz, ODR = 26 Hz	0.4077295	-0.815459	0.407730	-0.426937	0.203981	1
High-pass IIR2, f_cut = 10 Hz, ODR = 26 Hz	0.085605	-0.171209	0.085605	1.019146	0.361564	1
Low-pass IIR1, f_cut = 1 Hz, ODR = 26 Hz	0.108275	0.108275	-	-0.783450	-	1
Low-pass IIR1, f_cut = 2 Hz, ODR = 26 Hz	0.197739	0.197739	-	-0.604521	-	1
Low-pass IIR1, f_cut = 5 Hz, ODR = 26 Hz	0.408372	0.408372	-	-0.183257	-	1
Low-pass IIR1, f_cut = 10 Hz, ODR = 26 Hz	0.725032	0.725032	-	0.450063	-	1
Low-pass IIR2, f_cut = 1 Hz, ODR = 26 Hz	0.012426	0.024853	0.012426	-1.660834	0.710540	1
Low-pass IIR2, f_cut = 2 Hz, ODR = 26 Hz	0.043107	0.086213	0.043107	-1.332907	0.505333	1
Low-pass IIR2, f_cut = 5 Hz, ODR = 26 Hz	0.194261	0.388522	0.194261	-0.426937	0.203981	1
Low-pass IIR2, f_cut = 10 Hz, ODR = 26 Hz	0.595178	1.190355	0.595178	1.019146	0.361564	1
Band-pass IIR2, f1 = 1.5 Hz, f2 = 5 Hz, ODR = 26 Hz	0.310375	0	-0.310375	-1.069500	0.379250	1
Band-pass IIR2, f1 = 0.2 Hz, f2 = 1 Hz, ODR = 100 Hz	0.0236	0	-0.0236	-1.9521	0.9528	1

Table 3. Examples of filter coefficients

1.3 Features

The features are the statistical parameters computed from the Machine Learning Core inputs. The Machine Learning Core inputs which can be used for features computation are:

- the sensor input data which includes
 - sensor data from the X, Y, Z axes (e.g. Acc_X, Acc_Y, Acc_Z, Gyro_X, Gyro_Y, Gyro_Z);
 - external sensor data (e.g. ExtSens_X, ExtSens_Y, ExtSens_Z);
 - norm and norm squared signals of sensor / external sensor data (Acc_V, Acc_V2, Gyro_V, Gyro_V2, ExtSens_V, Ext_Sens_V2);
- the filtered data (e.g. high-pass on Acc_Z, band-pass on Acc_V2, etc.)

All the features are computed within a defined time window, which is also called "window length" since it is expressed as the number of samples. The size of the window has to be determined by the user and is very important for the machine learning processing, since all the statistical parameters in the decision tree will be evaluated in this time window. It is not a moving window, features are computed just once for every WL sample (where WL is the size of the window).

The window length can have values from 1 to 255 samples. The choice of the window length value depends on the sensor data rate (ODR), which introduces a latency for the generation of the Machine Learning Core result, and on the specific application or algorithm. In an activity recognition algorithm for instance, it can be decided to compute the features every 2 or 3 seconds, which means that considering sensors running at 26 Hz, the window length should be around 50 or 75 samples respectively.

Some of the feaures in the Machine Learning Core require some additional parameters for the evaluation (e.g. an additional threshold). The following table shows all the features available in the Machine Learning Core including additional parameters.

Note: The maximum number of features which can be configured in the MLC is 63.

Feature	Additional parameter
MEAN	-
VARIANCE	-
ENERGY	-
PEAK TO PEAK	-
ZERO CROSSING	Threshold
POSITIVE ZERO CROSSING	Threshold
NEGATIVE ZERO CROSSING	Threshold
PEAK DETECTOR	Threshold
POSITIVE PEAK DETECTOR	Threshold
NEGATIVE PEAK DETECTOR	Threshold
MINIMUM	-
MAXIMUM	-

Table 4. Features

1.3.1 Mean

The feature "*Mean*" computes the average of the selected input (*I*) in the defined time window (*WL*) with the following formula:

$$Mean = \frac{1}{WL} \sum_{k=0}^{WL-1} I_k$$

1.3.2 Variance

The feature "*Variance*" computes the variance of the selected input (*I*) in the defined time window (*WL*) with the following formula:

$$Variance = \left(\frac{\sum_{k=0}^{WL-1} {I_k}^2}{WL}\right) - \left(\frac{\sum_{k=0}^{WL-1} {I_k}}{WL}\right)^2$$

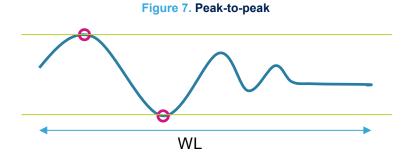
1.3.3 Energy

The feature "*Energy*" computes the energy of the selected input (*I*) in the defined time window (*WL*) with the following formula:

$$Energy = \sum_{k=0}^{WL-1} {I_k}^2$$

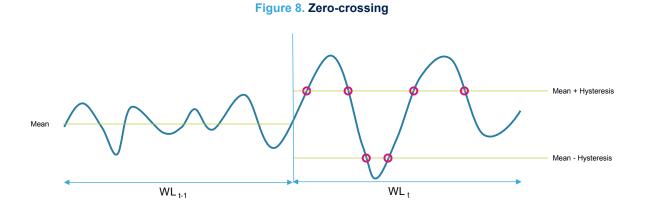
1.3.4 Peak-to-peak

The feature "*Peak to peak*" computes the maximum peak-to-peak value of the selected input in the defined time window.



1.3.5 Zero-crossing

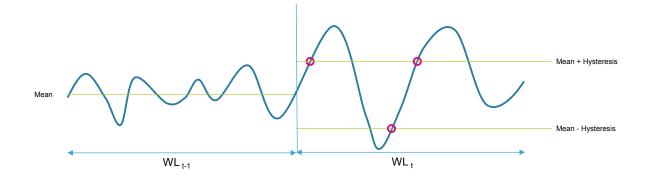
The feature "*Zero-crossing*" computes the number of times the selected input crosses a certain threshold. This internal threshold is defined as the sum between the average value computed in the previous window (feature "Mean") and hysteresis defined by the user.



1.3.6 Positive zero-crossing

The feature "*Positive zero-crossing*" computes the number of times the selected input crosses a certain threshold. This internal threshold is defined as the sum between the average value computed in the previous window (feature "Mean") and hysteresis defined by the user. Only the transitions with positive slopes are considered for this feature.

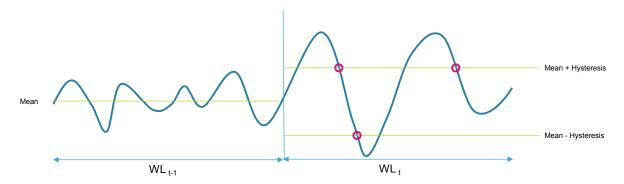
Figure 9. Positive zero-crossing



1.3.7 Negative zero-crossing

The feature "*Negative zero-crossing*" computes the number of times the selected input crosses a certain threshold. This internal threshold is defined as the sum between the average value computed in the previous window (feature "Mean") and hysteresis defined by the user. Only the transitions with negative slopes are considered for this feature.

Figure 10. Negative zero-crossing



1.3.8 Peak detector

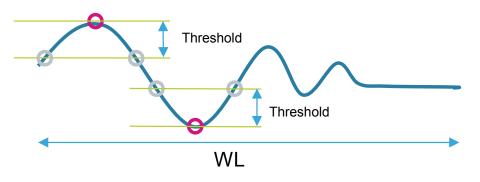
The feature "*Peak detector*" counts the number of peaks (positive and negative) of the selected input in the defined time window.

A threshold has to be defined by the user for this feature, and a buffer of three values is considered for the evaluation. If the second value of the three values buffer is higher (or lower) than the other two values of a selected threshold, the number of peaks is increased.

The buffer of three values considered for the computation of this feature is a moving buffer inside the time window.

The following figure shows an example of the computation of this feature, where two peaks (one positive and negative) have been detected in the time window.





1.3.9 Positive peak detector

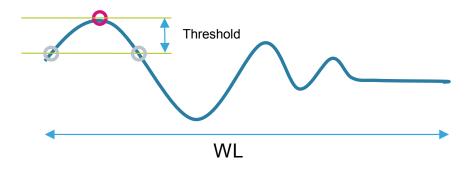
The feature "Positive peak detector" counts the number of positive peaks of the selected input in the defined time window.

A threshold has to be defined by the user for this feature, and a buffer of three values is considered for the evaluation. If the second value of the three values buffer is higher than the other two values of a selected threshold, the number of peaks is increased.

The buffer of three values considered for the computation of this feature is a moving buffer inside the time window.

The following figure shows an example of the computation of this feature, where just one peak (positive) has been detected in the time window.





1.3.10 Negative peak detector

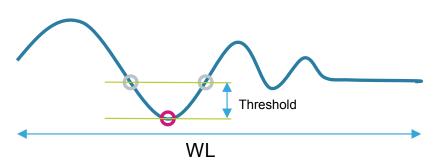
The feature "*Negative peak detector*" counts the number of negative peaks of the selected input in the defined time window.

A threshold has to be defined by the user for this feature, and a buffer of three values is considered for the evaluation. If the second value of the three values buffer is lower than the other two values of a selected threshold, the number of peaks is increased.

The buffer of three values considered for the computation of this feature is a moving buffer inside the time window.

The following figure shows an example of the computation of this feature, where just one peak (negative) has been detected in the time window.





1.3.11 Minimum

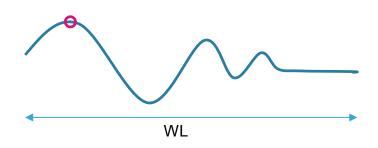
The feature *"Minimum"* computes the minimum value of the selected input in the defined time window. The following figure shows an example of minimum in the time window.



1.3.12 Maximum

The feature "*Maximum*" computes the maximum value of the selected input in the defined time window. The following figure shows an example of maximum in the time window.

Figure 15. Maximum



1.3.13 Selection of features

The selection of the features to be used for the Machine Learning Core configuration depends on the specific application.

Considering that the use of too many features may lead to overfitting and too large decision trees, it is recommended to start first by selecting the four most common features:

- Mean
- Variance
- Energy
- Peak-to-peak

If the performance is not good with these features, and in order to improve the accuracy, other features can be considered to better separate the classes.

Input data for the features calculation (from the accelerometer, gyroscope) and axes (e.g. X, Y, Z, V) have to be chosen according to the specific application as well. Some classes are strongly correlated with sensor orientation (i.e. applications which use the device carry position), so it is better to use individual axis (X, Y, Z). Other classes (like walking) are independent of orientation, so it is better to use the norm (V or V2).

Sometimes the basic features (mean, variance, energy, etc.) might not help in distinguishing the dominating frequency, so embedded digital filters can be enabled to select a specific region of frequency. Using the filtered signal, certain classes may be distinguished more precisely. For instance, if the user is walking, the typical signal is around 1-2 Hz, while if the user is jogging, the typical signal is around 2.5-4 Hz.

The information contribution from a single feature can be evaluated by a measure of how much different classes are separated (from one another). This analysis can be done in a graphical way, by plotting 1D/2D graphs as described in the following examples.

1.3.13.1 Histogram of a single feature (1D plot)

The following figure shows a histogram of the computed values of a single feature for three different classes. These three classes are reasonably separated, so an important level of information is expected with this feature. For reference, the computed classification accuracy with this single feature is around 75%.

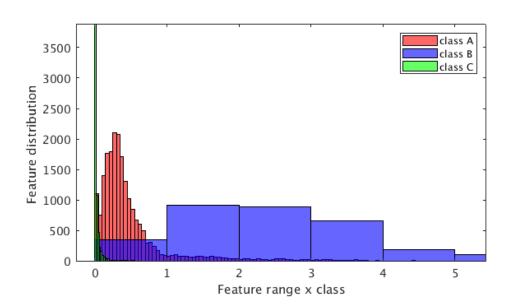


Figure 16. Distribution of single feature for three different classes

1.3.13.2 Visualization of two features (2D plot)

The following figure shows a 2D plot related to a 2-class classification problem with the selection of two features:

- Feature 1 on the graph vertical axis
- Feature 2 on the graph horizontal axis

In this case, the strict separation between the two classes is evident:

- Class A in red
- Class B in blue

A good information contribution can be obtained by combining the two features. For reference, the classification accuracy obtained with this example is more than 95%.

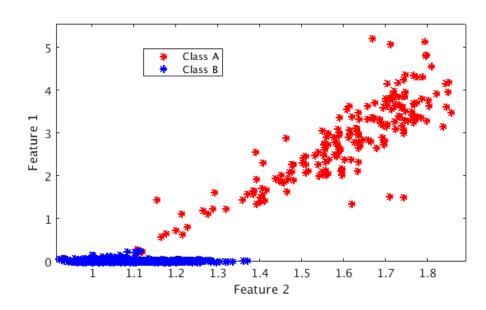


Figure 17. Visualization of two features and two classes



1.3.13.3 Ranking of features

Different machine learning tools offer automated methods to order features in terms of the information contribution. This form of output ranking is based on criteria/metrics such as correlation, information gain, probabilistic distance, entropy and more. An example is given by Weka, which automatically handles the calculations needed to generate optimal decision trees as indicated in the figure below.



	Metric
Attribute Evaluator	
Choose CorrelationAttributeEval	
Search Method	
Choose Ranker -T -1.79769313486	23157E308 -N -1
Attribute Selection Mode	Attribute selection output
Use full training set Cross-validation Folds Seed	Attribute Evaluator (supervised, Class (nominal): 13 class): Correlation Ranking Filter Ranked attributes: 0.506 5 feat_5 0.497 10 feat_10
(Nom) class	0.479 3 feat_3 0.454 9 feat_9
Start Stop	0.447 7 feat_7 0.428 4 feat 4
Result list (right-click for options) 15:49:03 - Ranker + CorrelationAttribut	0.428 4 feat_4 0.359 1 feat_1 0.349 6 feat_6 0.345 11 feat_11 0.326 12 feat_12 0.21 2 feat_2 0.142 8 feat_8

Note that different features could share the same information contribution. This can be evaluated again by visualizing the single feature or by checking the accuracy obtained with the subset of features taken one-by-one, and together, as explained in previous sections.

A final consideration can be done on the number of features which have been selected. In general, the higher the number of features selected:

- the higher the risk of overfitting
- the larger the size of the resulting decision tree

1.4 Decision tree

The decision tree is the predictive model built from the training data which can be stored in the ISM330DHCX. The training data are the data logs acquired for each class to be recognized (in the activity recognition example the classes might be walking, jogging, driving, etc.).

The outputs of the computation blocks described in the previous sections are the inputs of the decision tree. Each node of the decision tree contains a condition, where a feature is evaluated with a certain threshold. If the condition is true, the next node in the true path is evaluated. If the condition is false, the next node in the false path is evaluated. The status of the decision tree will evolve node by node until a result is found. The result of the decision tree is one of the classes defined at the beginning of the data collection.

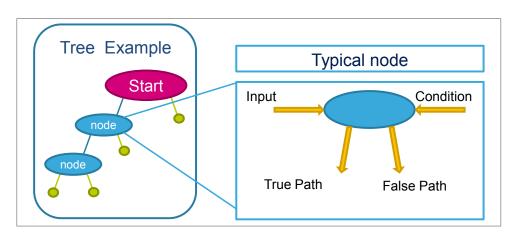


Figure 19. Decision tree node

The decision tree generates a new result every time window (the parameter "window length" set by the user for the computation of the features). The decision tree results can also be filtered by an additional (optional) filter called "Meta-classifier", which is described in Section 1.5 Meta-classifier.

The Machine Learning Core results (decision tree results filtered or not filtered) are accessible through dedicated registers in the embedded advanced features page 1 of the ISM330DHCX registers (as shown in Table 5). These registers can be countinuously read (polled) to check the decision tree outputs. The register MLC_STATUS_MAINPAGE (38h) contains the interrupt status bits of the 8 possible decision trees. These bits will be automatically set to 1 when the corresponding decision tree value changes. Furthermore, the interrupt status signal generated using these bits can also be driven to the INT1 pin by setting the MLC_INT1 (0Dh) register, or to the INT2 pin by setting the MLC_INT2 (11h) register (Table 6). Using the interrupt signals, an MCU performing other tasks or sleeping (to save power), can be awakened when the Machine Learning Core result has changed. The Machine Learning Core interrupt signal is pulsed by default. The duration of the pulsed interrupt is defined by the fastest ODR among the Machine Learning Core, Finite State Machine and sensor ODRs:

interrupt pulse duration = 1 / max(MLC_ODR, FSM_ODR, XL_ODR, GYRO_ODR)

The Machine Learning Core interrupt signal can also be set latched through the bit EMB_FUNC_LIR in the embedded function register PAGE_RW (17h).

Table 5. Decision tree results

Register	Content
MLC0_SRC (70h)	Result of decision tree 1
MLC1_SRC (71h)	Result of decision tree 2
MLC2_SRC (72h)	Result of decision tree 3
MLC3_SRC (73h)	Result of decision tree 4
MLC4_SRC (74h)	Result of decision tree 5
MLC5_SRC (75h)	Result of decision tree 6
MLC6_SRC (76h)	Result of decision tree 7
MLC7_SRC (77h)	Result of decision tree 8

Table 6. Decision tree interrupts

Register	Content
MLC_STATUS_MAINPAGE (38h)	Contains interrupt status bits for changes in the decision tree result
MLC_STATUS (15h)	Contains interrupt status bits for changes in the decision tree result
MLC_INT1 (0Dh)	Allows routing of interrupt status bits for decision trees to INT1 $\ensuremath{\text{pin}}^{(1)}$
MLC_INT2 (11h)	Allows routing of interrupt status bits for decision trees to INT2 $\ensuremath{\text{pin}}^{(2)}$

1. Routing is established if the INT1_EMB_FUNC bit of MD1_CFG (5Eh) is set to '1'.

2. Routing is established if the INT2_EMB_FUNC bit of MD2_CFG (5Fh) is set to '1'.

1.4.1 Decision tree limitations in the ISM330DHCX

The ISM330DHCX has limited resources for the Machine Learning Core in terms of number of decision trees, size of the trees, and number of decision tree results.

Up to 8 different decision trees can be stored in the ISM330DHCX, but the sum of the number of nodes for all the decision trees must not exceed 512 (*). Every decision tree can have up to 256 results in the ISM330DHCX.

(*) This number might also be limited by the number of features and filters configured. In general, if using few filters and features, there is no further limitation on the size of the decision tree. However, when using many filters and features, the maximum number of nodes for the decision trees is slightly limited. For instance, if the number of filters configured is 10 and the number of features configured is 50, the maximum number of nodes might be reduced by 100. The tool will inform the user of the available nodes for the decision tree.

The table below summarizes the limitations of the ISM330DHCX.

Table 7. Decision tree limitations in the ISM330DHCX

	ISM330DHCX
Maximum number of decision trees	8
Maximum number of nodes (Total number for all the decision trees)	512 (*)
Maximum number of results per decision tree	256

Note: when using multiple decision trees, all the parameters described in the previous sections (inputs, filters, features computed in the time window, the time window itself, and also the data rates) are common for all the decision trees.

1.5 Meta-classifier

A meta-classifier is a filter on the outputs of the decision tree. The meta-classifier uses some internal counters in order to filter the decision tree outputs.

Decision tree outputs can be divided in subgroups (e.g. similar classes can be managed in the same subgroup). An internal counter is available for all the subgroups of the decision tree outputs. The counter for the specific subgroup is increased when the result of the decision tree is one of the classes in the subgroup and it is decreased otherwise. When the counter reaches a defined value, which is called "end counter" (set by the user), the output of the Machine Learning Core is updated. Values allowed for end counters are from 0 to 14.

Decision tree result	A	A	Α	в	Α	в	в	в	Α	в	в	в	Α	Α	Α
Counter A	1	2	3	2	3	2	1	0	1	0	0	0	1	2	3
(End counter = 3)	1	2	3	2	3	2	1	0	1	0	0	0	1	2	3
Counter B	_	0	0	1	0	1	2	3	2	3	4	5	4	3	2
(End counter = 4)	0	0	0	1	0		2	3	2	3	4	Э	4	3	2
Machine Learning Core result (including meta-classifier)	x	x	Α	Α	Α	Α	Α	Α	Α	Α	в	в	в	в	Α

Table 8. Meta-classifier example

The previous table shows the effect of filtering the decision tree outputs through a meta-classifier. The first line of the table contains the outputs of the decision tree before the meta-classifier. Counter A and Counter B are the internal counters for the two decision tree results ("A" and "B"). In the activity recognition example, the result "A" might be walking and the result "B" jogging. When the internal counter "A" reaches the value 3 (which is the End Counter for counter "A"), there is a transition to result "A". When the internal counter "B" reaches value 4, there is a transition to result "B".

The purpose of the meta-classifier is to reduce the false positives, in order to avoid generating an output which is still not stable, and to reduce the transitions on the decision tree result.

1.5.1 Meta-classifier limitations in the ISM330DHCX

The meta-classifier has a limited number of sub-groups, 8 sub-groups can be used in the ISM330DHCX. Similar classes may need to be grouped in the same subgroup to use the meta-classifier.

Table 9. Meta-classifier limitations in the ISM330DHCX

	ISM330DHCX
Maximum number of results per decision tree	256
Result sub-groups for meta-classifier per decision tree	8

Note: multiple meta-classifiers can be configured. One meta-classifier is available for any decision tree configured in the Machine Learning Core.

1.6 Finite State Machine interface

The ISM330DHCX also provides a configurable Finite State Machine which is suitable for deductive algorithms and in particular gesture recognition.

Finite state machines and decision trees can be combined to work together in order to enhance the accuracy of motion detection.

The decision tree results generated by the Machine Learning Core can be checked by the Finite State Machine available in the ISM330DHCX; this is possible through the condition CHKDT (described in the application note AN5388 ISM330DHCX: Finite State Machine).

2 Machine Learning Core tools

The Machine Learning Core programmability in the device is allowed through a dedicated tool, available as an extension of the Unico GUI.

2.1 Unico GUI

Unico is the Graphical User Interface for all the MEMS sensor demonstration boards available in the STMicroelectronics portfolio. It has the possibility to interact with a motherboard based on the STM32 microcontroller (Professional MEMS Tool), which enables the communication between the MEMS sensor and the PC GUI. Unico also has the possibility to run offline, without a motherboard connected to the PC.

Details of the Professional MEMS tool board can be found on www.st.com at STEVAL-MKI109V3.

Unico GUI is available in three software packages for the three operating systems supported.

- Windows
 - STSW-MKI109W
- Linux
 - STSW-MKI109L
 - Mac OS X
 - STSW-MKI109M

Unico GUI allows visualization of sensor outputs in both graphical and numerical format and allows the user to save or generally manage data coming from the device.

Unico allows access to the MEMS sensor registers, enabling fast prototyping of register setup and easy testing of the configuration directly on the device. It is possible to save the current register configuration in a text file (with .ucf extension) and load a configuration from an existing file. In this way, the sensor can be re-programmed in few seconds.

The Machine Learning Core tool available in the Unico GUI abstracts the process of register configuration by automatically generating configuration files for the device. The user just needs to set some parameters in the GUI and by clicking a few buttons, the configuration file is already available. From these configuration files, the user can create his own library of configurations for the device.

Since the machine learning approach requires the collection of data logs, they can be acquired through the load/save tab of Unico (Figure 20). For the accelerometer, the checkbox "Acceleration" allows saving data in [mg]. For the gyroscope, the checkbox "Angular rate" allows saving data in [dps].

Note: When logging data, "Start" and "Stop" buttons (in the Load/Save tab of Unico) must be used properly in order to avoid logging incorrect data at the beginning or at the end of the acquisition. For instance, when logging a data pattern for the class "walking", the user should start walking before pressing the button "Start" and stop walking after pressing the button "Stop". It is important to select the correct ODR for data logging (ODR 26 Hz). If the MLC ODR (e.g. 26 Hz) is already defined, it is recommended to use the same ODR for data logging (ODR 26 Hz). If the MLC ODR is not defined, it is recommended to log the data at ODR 104 Hz (which is the maximum ODR for MLC), and then downsample the data if needed. Depending on the algorithm to be implemented, different data logs are needed (at least one per class to use the supervised machine learning approach). It is recommended to have different data logs for each class (e.g. 30 data logs per class) in order to capture some diversity or variation which can be expected in the final application (e.g. different users, different tests, or different conditions).

If using Unico GUI offline (without connecting the motherboard to the PC), the user, who has already acquired the data logs, can directly upload them to generate a Machine Learning Core configuration.

Figure 20. Unico GUI

🔀 Unico													
🚺 Back	Select F	Port: C	COM3	~	🔎 Connect	🎤 Disconnect	🕨 Start 🥥 Stop 🔀	🕻 Exit					
	Info	Options	Registers	Registers 2	Embedded Reg	1 Embedded Reg	2 Embedded Adv Page 0	Embedded Adv Page 1	Sensor Hub	Registers	Load/Save		
Bars													
Rot Plot		Save	Data										
Data 4			- Choose the	e file name:							Browse		
Interrupt						(Accelerometer)	LSB		🗹 Angle	V In	terrupt		
FSM			- Check the	data you want to	save:	(Gyroscope)	LSB	Angular Rate					
(<u>)</u>							Temperature LSB	3 Temperature [C]		🗹 De	ecisionTree Resu	ts	
FIFO			- Press Start	for logging:				Start		Stop)	
S FIFO												,	
6		Load	/Save Configu	uration									
6D		Loud	, save comige										
Indin.									Loa)	
EFT 1			- Click Load f	for loading a conf	iguration from a l	ext file:			LOa	a		J	
1			- Click Save	for saving the cu	rrent configuratio	n to a text file:			Sav	/e)	
FET													
MLC)												
A Pedomet.													
	roelectroni	cs	Demo	nstration kit = ST	EVAL-MKI 197V1	LSM6DSOX)	Board = ProfiMEMSTool	Firmware Version	= V3.6.36	Unico Ve	ersion = 9.8.0.0		

The collected data logs can then be loaded in the Machine Learning Core tool of Unico, available on the left side of the GUI, by using the Data Patterns tab (Figure 21). An expected result must be assigned to each data pattern loaded (for instance, in the activity recognition algorithm, the results might be: still, walking, jogging, etc.). This assignment is also called "data labeling". The label has to be a set of characters including only letters and numbers (no special characters and no spaces). It is also possible to load a group of data patterns (by multiple selection in the folder where the files are located) and assign the label just once for all the files selected.

hine Learning Core		_	_	_	_	_	_	_	?
atterns Configu	ration								
ad Data Pattern									
C:/logs/running_2.	bit					Browse		Advanc	ed Options
rrent Data Patter									
rrent Data Pattei	m								
Sample [#] 879	AccX -86	AccY -220	AccZ 1476	GyrX -43.1988	GyrY 19.95	GyrZ -11.9088	ExtX 0	ExtY 0	ExtZ <u> </u>
879	-80	-220	1476	24.535	7.6125	0.41125	0	0	0
881	67	-216	882	39.9788	-9.03	10.955	0	0	0
882	-28	-141	869	23.0738	-8.7675	13.4225	0	0	0
883	-40	-66	807	23.2313	-0.42	14.1488	0	0	0
884	-24	-4	707	32.1125	8.54	11.9	0	0	0 -
1									<u> </u>
Set Class (L		running		LOAD					
Set Class (L	abel):			Lond					
ta Patterns Load	ed								CLEAR
Pattern [#]	Samples [#]	Result				Location			
0	910	still				C:/logs/still_1.txt			
1	429	still				C:/logs/still_2.txt			
2	1303	walking				:/logs/walking_1.txt			
3	1265	walking				/logs/walking_2.txt			
4	1284	walking				:/logs/walking_3.txt			
5	933	running				/logs/running_1.txt			

Figure 21.	Machine	Learning	Core	tool	- Data	Patterns
------------	---------	----------	------	------	--------	----------

The unit of measurement for the data expected in the data patterns tab of the Machine Learning Core tool are:

- [mg] (or alternatively [g]) for the accelerometer
- [dps] (or alternatively [mdps]) for the gyroscope

The conversion from [mg] to [g] for the accelerometer, and [dps] to [rad/s] for the gyroscope, is automatically managed internally by the Machine Learning Core tool, to allow the Machine Learning Core logic to work with the correct data ([g] and [rad/s]). For the external sensor data, the user will be required at a later stage in the configuration to set the proper sensitivity.

In the "Configuration" tab of the Machine Learning Core tool (Figure 22), all the parameters of the Machine Learning Core (such as ODR, full scales, window length, filters, features, meta-classifier) can be configured. The tool allows selecting multiple filters which can be applied to the raw data, and multiple features to be computed from the input data or from the filtered data. The features computed will be the attributes of the decision tree. When the board is connected and the device already configured, the tool automatically suggests ODRs and full scales (for accelerometer and gyroscope) according to the current device configuration.

Machine	Learning (mber of sar	Core configuration nples for the window of interest	:	52			1
	ter configu			End filt	ers configuration		
L							
Fe	atures Feature:	Variance					-
	Peature:		Input: ACC_V2	Enab		^	4
	Feature:	Energy	Input: ACC_X	Enab	iled 🗹 Signed	(+)	
	Feature:	Energy	Input: ACC_Y	Enab	led Signed	(+)	
	Feature:	Energy	Input: ACC_Z	Enab	led 🛛 Signed	(+)	4
	Feature:	Energy	Input: ACC_V	🗹 Enab	led		
	Feature:	Energy	Input: ACC_V2	Enab	led		
	Feature:	Peak to Peak	Input: ACC_X	Enab	led Signed	(+)	
	Feature:	Peak to Peak	Input: ACC_Y	Enab	oled Signed	(+)	
	Feature:	Peak to Peak	Input: ACC_Z	Enab	oled Signed	(+)	
	Feature:	Peak to Peak	Input: ACC_V	🗹 Enab	led		
		· ·	r	(<u> </u>	f.		-
							•

Figure 22. Machine Learning Core tool - Configuration

The "Configuration" tab of the Machine Learning Core tool generates an Attribute-Relation File (ARFF), which is the starting point for the decision tree generation process. The decision tree can be generated by different machine learning tools (Section 2.2).

Once the decision tree has been generated, it can be uploaded to the Machine Learning Core tool in Unico to complete the generation of the register configuration for the ISM330DHCX.

The Unico GUI, by accessing the sensor registers, can read the status of the decision tree ouputs, visualize them together with sensor data, and make it possibile to log all the data (sensor ouputs and decision tree outputs) together in the same text file.

2.2 Decision tree generation

57/

Unico (starting from version 9.8) is able to automatically generate the decision tree, as shown in the following figure.

Two parameters can be fine-tuned when starting the decision tree generation in Unico:

- the maximum number of nodes, which can be set to make sure the generated tree can fit in the MLC configuration;
- the confidence factor, for controlling decision tree pruning (by lowering this number, overfitting can be reduced).

🔀 Machine Learning Core		?	×
Data Patterns Configuration			MLC Tool
Machine Learning Core configuration			ಠ
		-	
Decision Tree #1 File - [Available nodes = 245]			
	Max number of nodes:		
 Click "GENERATE" to generate a new decision tree Click "Browse" to import a different decision tree file (e.g. generated by external tools) 	15		
	Confidence factor:		
GENERATE	Decision tree name:		
Browse C:/logs/ST_decision_tree_20200430_175930_003.txt	DT1		
Decision tree info			
F1_VAR_on_ACC_V <= 0.0038125	- 	¥	
Reset 100%	Next		

Figure 23. Decision tree generation in Unico

Besides Unico, there are other external machine learning tools able to generate decision trees and some of them are supported by Unico.

One of the most frequently used tool is Weka, software developed by the University of Waikato (more details about this software can be found in Appendix A). Other alternative tools are: RapidMiner (Appendix B), Matlab (Appendix C), Python (Appendix D).

Weka is able to generate a decision tree starting from an Attribute-Relation File (ARFF). Through Weka it is possible to evaluate which attributes are good for the decision tree, and different decision tree configurations can be implemented by changing all the parameters available in Weka. Figure 24 and Figure 25 show the "Preprocess" and "Classify" tabs of Weka which allow evaluating the attributes and generating the decision tree.

Figure 24.	Weka	preprocess
------------	------	------------

🕢 Weka Explorer	
Preprocess Classify Cluster Associate Select attributes	Visualize
Open file Open URL Open DB Gene	erate Undo Edit Save
Choose None	Apply
Current relation	Selected attribute
Relation: ActivityDataForProgram Attributes: 15 Instances: 6217 Sum of weights: 6217	Name: MEAN_on_ACC_X Type: Numeric Missing: 0 (0%) Distinct: 3243 Unique: 1894 (30%)
Attributes	Statistic Value
All None Invert Pattern	Minimum 0 Maximum 1.487 Mean 0.47 StdDev 0.384
1 MEAN_on_ACC_X 2 MEAN_on_ACC_Y 3 MEAN_on_ACC_Z 4 MEAN_on_ACC_V2 5 VAR_on_ACC_V2 6 VAR_on_ACC_Y	Class: class (Nom)
7 VAR_on_ACC_Z	789 342 157 212 154 119 154 119 100 00 34 32 11 2 0 0.74 149
ок	Log 🛷 x0

Figure 25. Weka classify

assifier	
Choose J48 -C 0.25 -M 2	
st options	Classifier output
Use training set Supplied test set Set Cross-validation Folds 10 Percentage split % 66 More options	VAR_on_ACC_V^2 <= 0.018555: still (62.0) VAR_on_ACC_V^2 > 0.018555 VAR_on_ACC_V^2 <= 0.078125 VAR_on_ACC_X <= 0.000696: still (5.0) VAR_on_ACC_X > 0.000696 VAR_on_ACC_Y <= 0.014343: walking (43.0/1.0) VAR_on_ACC_Y > 0.014343: still (4.0) VAR_on_ACC_V^2 > 0.078125
Iom) class Start Stop sult list (right-click for options)	<pre>I = ENERGY_on_ACC_V^2 <= 86.5 I = ABS(MEAN)_on_ACC_X <= 0.049011 I = VAR_on_ACC_X <= 0.003302 I = ABS(MEAN)_on_ACC_X <= 0.01387: fastwalking (7.0/1.0) I = ABS(MEAN)_on_ACC_X > 0.01387: walking (12.0) I = VAR_on_ACC_X > 0.003302: fastwalking (31.0/2.0)</pre>
16:47:04 - trees.J48	<pre>1 ABS(MEAN)_on_ACC_X > 0.049011 1 ABS(MEAN)_on_ACC_Z <= 0.937988 1 ABS(MEAN)_on_ACC_Z <= 0.937988 1 PeakToPeak_on_ACC_V <= 0.731934: walking (2.0) 1 PeakToPeak_on_ACC_V > 0.731934: fastwalking (5.0) 1 ABS(MEAN)_on_ACC_Z > 0.937988: walking (29.0) 1 ENERGY_on_ACC_V^2 > 86.5: fastwalking (17.0/1.0) </pre>
	Number of Leaves : 11 Size of the tree : 21

Once the decision tree has been generated, it can be uploaded to the Machine Learning Core tool in Unico, to complete the generation of the register configuration for the ISM330DHCX.

The Machine Learning Core tool in Unico accepts as input the decision tree files in a textual format (.txt). The textual file must contain the decision tree in the Weka J48 format (an example of a decision tree is shown in Figure 26). From the Weka classifier output (Figure 25), the decision tree has to be selected starting from the first line (first node) or in the RapidMiner format (Appendix B). The last two rows (number of leaves and size of the tree) are optional. The selected ouput from Weka has to be copied to a text file.

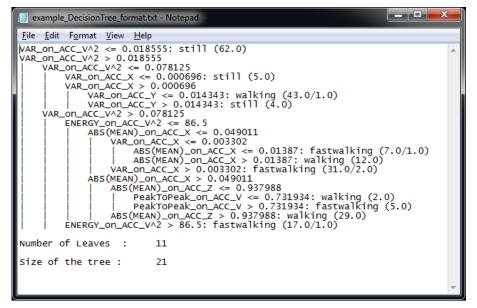


Figure 26. Decision tree format

If the decision tree has been generated from a different tool, the format must be converted to the Weka J48 format (or to the RapidMiner format) in order to allow the Machine Learning Core tool in Unico to read the decision tree correctly.

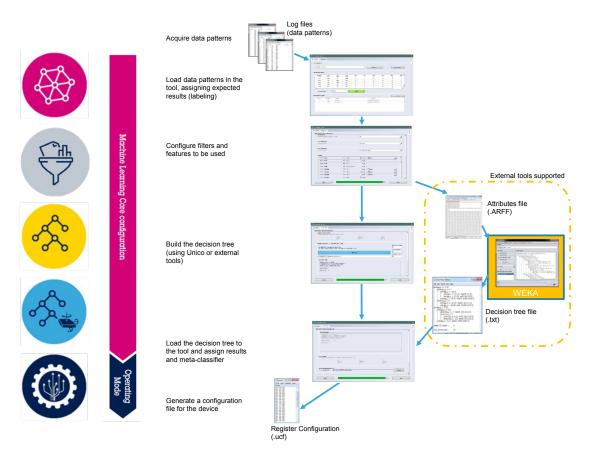
2.3 Configuration procedure

Figure 27 shows the whole procedure of the machine learning processing, from the data patterns to the generation of a register setting for the device (ISM330DHCX).

As seen in Section 2.1 Unico GUI, the data patterns can be acquired in the "Load/Save" tab of the Unico GUI. If this is not possible or if the user wants to use some different data patterns, they can still be uploaded in the Machine Learning Core tool of Unico, with a few limitations:

- Every data patterns has to start with a header line, containing the unit of measurement of the data
 A_X [mg] A_Y [mg] A_Z [mg] G_X [dps] G_Y [dps] G_Z [dps]
 - The data after the header line must be separated by "tab" or "space".
- The order of sensors in the file columns must be accelerometer data (if available), gyroscope data (if available), external sensor data (if available).
- The order of the axes in the columns of any sensor is X, Y, Z.





Opening the Machine Learning Core tool available in Unico, the data patterns, acquired in the format described above, can be loaded assigning the expected result for each data log (as shown in the following figure).

Figure 28.	Assigning a	result to a	data pattern
------------	-------------	-------------	--------------

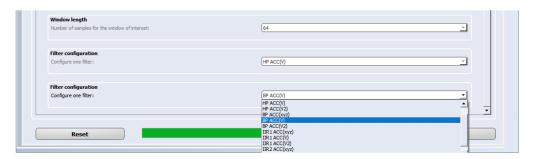
_	2		_	_	_	_	_	_	
terns Configu	uration								
Data Pattern									
/logs/running 2.	byt .					Browse			ed Options
nogon anning_er						browse		Advanc	ted Obtions
ent Data Patte	m								
Sample [#]	AccX	AccY	AccZ	GyrX	GyrY	GyrZ	ExtX	ExtY	ExtZ 🔺
879	-86	-220	1476	-43.1988	19.95	-11.9088	0	0	0
880	16	-186	1094	24.535	7.6125	0.41125	0	0	0
881	67	-216	882	39.9788	-9.03	10.955	0	0	0
882	-28	-141	869	23.0738	-8.7675	13.4225	0	0	0
883	-40	-66	807	23.2313	-0.42	14.1488	0	0	0
884	-24	-4	707	32.1125	8.54	11.9	0	0	0 -
									<u> </u>
Set Class (L	abel):	running		LOAD					
	(
Patterns Load	ed								CLEAR
Pattern [#]	Samples [#]	Result				Location			
0	910	still				C:/logs/still_1.txt			
1	429	still				C:/logs/still_2.txt			
2	1303	walking			C:	/logs/walking_1.td	t		
3	1265	walking			C:	/logs/walking_2.td	t		
4	1284	walking			C:	/logs/walking_3.td	t		
	933	running			C:	/logs/running_1.td	t		
3	1265 1284	walking walking			C: C:	/logs/walking_2.td /logs/walking_3.td	t t		

When all the data patterns have been loaded, the Machine Learning Core parameters can be configured through the configuration tab. These parameters are ODR, full scales, number of decision trees, window length, filters, features, etc. (as shown in Figure 29, Figure 30, Figure 31, Figure 32).

🔛 Machine Learning Core		
Data Patterns Configuration		MLC Tool
Machine Learning Core configuration		
Select the device:	<u> </u>	
Machine Learning Core ODR		
Select the internal data rate for the Machine Learning Core: 26 Hz	-	
Inputs		
Select the Machine Learning Core inputs: Accelerometer only	9	
Accelerometer		
Full scale: 2 g		
ODR: 26 Hz		
Decision trees		
Number of decision trees:	- -	
Reset 100% Next		

Figure 29. Configuration of Machine Learning Core







Feature:	Variance	Input: ACC_V	Enabled	<u> </u>
Feature:	Variance	Input: ACC_V2	Enabled	
Feature:	Variance	Input: filter DIFF on ACC V	Enabled	
Feature:	Energy	Input: ACC_X	Enabled Signed	(+)
Feature:	Energy	Input: ACC_Y	Enabled Signed	(+)
Feature:	Energy	Input: ACC_Z	Enabled Signed	(+)
Feature:	Energy	Input: ACC_V	Enabled	
Feature:	Energy	Input: ACC_V2	Enabled	
Feature:	Energy	Input: filter DIFF on ACC V	Enabled	
Feature:	Peak to Peak	Input: ACC_X	Enabled Signed	(+)

Feature: Peak to Peak	Input: ACC_X	Enabled Signed		(+)
Save ARFF file ARFF file:				Browse
Reset			100%	Next

Multiple filters and multiple features can be chosen. The GUI will iteratively ask for another filter until the parameter "End filter configuration" is chosen (Figure 30). All the available features can be easily selected using the checkboxes (Figure 31).

Once all the features have been configured, the Machine Learning Core tool in Unico will generate an ARFF file (Figure 32), which is the file containing all the features computed from the training data. Figure 33 shows an example of an ARFF file generated by the Machine Learning Core tool in Unico.

Unico has a built-in tool for decision tree generation which internally uses the ARFF file to generate a decision tree, however, the ARFF file can also be used in external tools (for instance Weka). The decision tree generated with the external tool can be imported in Unico.

In some cases, the user must adapt the ARFF file to the file format required by other external tools for decision tree generation, and also the decision tree must be compatible with the format described in Section 2.2 Decision tree generation. For more information about the external tools supported, please see the appendix sections.

Comparamental Mutanguistic Laboratory Compare Service Laboratory Compare Laborato

Figure 33. ARFF file

Before generating or loading the decision tree, Unico also asks for the result values associated to each class recognized by the decision tree. These values will be used as possible values for the MLCx_SRC registers.



Nachine Learning Core	
Data Patterns Configuration	MC Tool
Machine Learning Core configuration	
Decision Tree #1 Results Insert the result values [from 0 to 15] for decision tree #1: stil walking running 0 4 8	- -
Decision Tree #1 File - [Available nodes = 245]	
	Max number of nodes:
 - Click "GENERATE" to generate a new decision tree - Click "Browse" to import a different decision tree file (e.g. generated by external tools) 	15
	Confidence factor:
GENERATE	0.9
Browse	Decision tree name:
Di Olise	
Decision tree info	
Reset	100% Next

Figure 34. Configuration of results and decision tree

The last step of the configuration process is to configure the meta classifier, which is the optional filter for the generation of the decision tree results. After that, the tool is ready to generate a configuration for the device (Figure 35).

hine Learning Core configuratio	n			
Decision tree info				
	$\begin{array}{l} 100341377\\ < < 64.25\\ y < < 64.25\\ y < < 64.125\\ v < 100390625: still (1.0)\\ v > 0.00390625: still (1.0)\\ v > 0.00390625: still (1.0)\\ y > 64.125: still (3.0)\\ > 64.25: still (3.0)\\ > 64.25: still (3.0)\\ < 86.4375: walking (51.0/6.0)\\ < 86.4375: walking (51.0/6.0)\\ < 86.4375: walking (2.0)\\ \end{array}$			×.
Meta-classifier Insert the end counter values for	the decision tree (allowed values from #1: still 0	0 to 14): #2: walking	#3: running	
Save Configuration File (.ucf) Configuration File Path:				Browse

Figure 35. Meta classifier and device configuration

Once the MLC configuration has been completed, Unico allows loading the .ucf file generated to directly program the device. The loading feature is available in the "Load/Save" tab of the Unico main window (Figure 37). Alternatively, at the end of the MLC configuration a check box allows directly loading the configuration created on the device as shown in Figure 36.

Figure 36. Creation of configuration file

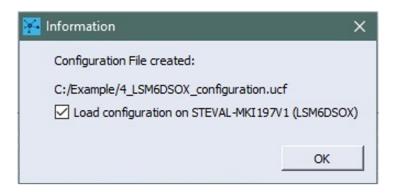


Figure 37. Unico load configuration

Load/Save Configuration	
- Click Load for loading a configuration from a text file:	Load
- Click Save for saving the current configuration to a text file:	Save

When the device is programmed, the Machine Learning Core results can be monitored in the Data window of Unico (Figure 38) or in one of the registers tabs containing the Machine Learning Core source registers (Figure 39).

The .ucf file generated by Unico can also be used for integrating the generated MLC configuration in other platforms and software (e.g. AlgoBuilder, Unicleo, SensorTile.box, etc.).

From the .ucf file it is also possible to convert the sequence of values to a header file (.h) to be imported in any C project (e.g. driver, firmware, etc.): Unico allows .h file generation (from .ucf files) through the "C code generation" dedicated tool in the "Options" tab of the Unico main window. An example of using the generated .h file in a standard C driver is available in [STMems_Standard_C_drivers repository] on GitHub.

Figure	38.	Unico	Data	window
···gaio		0		

🔲 Data							
Accelerometer:							
Data read from the Accelerometer	X:	1258	LSB	153	mg	8	Deg
	Y:	-320	LSB	-39	mg	-2	Deg
	Z:	8879	LSB	1083	mg	82	Deg
Gyroscope:							
Data read from the Gyroscope	X:		LSB	0.00	dps		
	Y:		LSB	0.00	dps		
	Z:	0	LSB	0.00	dps		
Decision Tree results:							
Accelerometer: Data read from the Accelerometer Gyroscope: Data read from the Gyroscope Decision Tree results: 1. 8 2. 0 3.	0 4.	0 5.	0 6.	0 7.	0	8. 0	

Figure 39. Unico - Machine Learning Core source registers

MLC0 SRC	(70h)	01	Read	Write	Default
MLC1 SRC	(71h)	00	Read	Write	Default
MLC2 SRC	(72h)	00	Read	Write	Default
MLC3 SRC	(73h)	00	Read	Write	Default
MLC4 SRC	(74h)	00	Read	Write	Default
MLC5 SRC	(75h)	00	Read	Write	Default
MLC6 SRC	(76h)	00	Read	Write	Default
_		\rightarrow	Read	Write	Default
MLC7_SRC	(77h)	_00 _	Read	Write	Default

3 Decision tree examples

This section describes some examples of a decision tree which can be loaded in the ISM330DHCX.

3.1 Vibration monitoring

The decision tree in the following figure shows a simple example of vibration monitoring. Three different levels of vibrations are recognized (vibration1, vibration2, vibration3) using a simple decision tree with just one feature, the peak-to-peak feature in the accelerometer norm squared input (p2p accNorm2).

The vibration monitoring example runs at 26 Hz, computing features in a window of 16 samples. The current consumption of the ISM330DHCX is around 171 μ A at 1.8 V. Turning off the Machine Learning Core, the current consumption of the ISM330DHCX would be around 170 μ A, so the additional current consumption of the Machine Learning Core is just 1 μ A.

DecisionTreeVibration.txt - Notepad 💷 💷 💻 🛛	
<u>File E</u> dit F <u>o</u> rmat <u>V</u> iew <u>H</u> elp	
<pre>p2p_accNorm2 <= 0.03: vibration1 p2p_accNorm2 > 0.03</pre>	*
Number of Leaves : 3	
Size of the tree : 5	
	Ŧ

Figure 40. Vibration monitoring decision tree

3.2 Motion intensity

The decision tree in the following figure shows a simple example of motion intensity implemented using just the feature "variance" in the accelerometer norm. Eight different intensity levels are recognized by this decision tree. The configuration for motion intensity described in this example runs at 12.5 Hz, computing features in a window of 39 samples. The current consumption of the ISM330DHCX is around 171 μ A at 1.8 V. Turning off the programmable sensor, the current consumption of the ISM330DHCX would be around 170 μ A, so the additional current consumption of the Machine Learning Core is just 1 μ A.

Figure 41.	Motion	intensity	decision	tree
------------	--------	-----------	----------	------

dec_tree.txt - Notepad	
<u>F</u> ile <u>E</u> dit F <u>o</u> rmat <u>V</u> iew <u>H</u> elp	
<pre>module_variance <= 0.009: Intensity_0 module_variance > 0.009 module_variance <= 0.013671875: Intensity_1 module_variance > 0.013671875: Intensity_2 module_variance <= 0.0234375: Intensity_2 module_variance > 0.0234375 module_variance <= 0.033203125: Intensity_3 module_variance > 0.033203125: Intensity_4 module_variance <= 0.078125: Intensity_4 module_variance > 0.078125: Intensity_5 module_variance > 0.1640625: Intensity_5 module_variance > 0.1640625: Intensity_6 module_variance > 0.3125: Intensity_7</pre>	*
Number of Leaves : 8	
Size of the tree : 15	Ŧ

3.3 6D position recognition

The ISM330DHCX already has a 6D position recognition algorithm embedded in the device. The example described in this section shows just a different implementation using a decision tree.

The six different positions (Figure 42) can be easily recognized by a simple decision tree (Figure 43) using the following features:

- meanx_abs: Mean of the accelerometer X axis (unsigned)
- *meany_abs*: Mean of the accelerometer Y axis (unsigned)
- meanz_abs: Mean of the accelerometer Z axis (unsigned)
- meanx_s: Mean of the accelerometer X axis (signed)
- meany_s: Mean of the accelerometer Y axis (signed)
- meanz_s: Mean of the accelerometer Z axis (signed)

The configuration for 6D position recognition described in this example runs at 26 Hz, computing features in a window of 16 samples. The current consumption of the ISM330DHCX is around 172 μ A at 1.8 V. Turning off the Machine Learning Core, the current consumption of the ISM330DHCX would be around 170 μ A, so just 2 μ A is the additional current consumption of the Machine Learning Core.

Figure 42. 6D positions

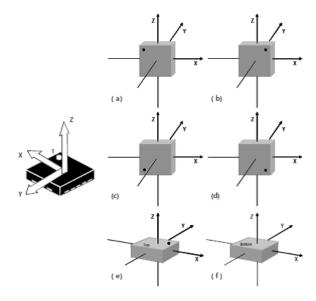


Figure 43. 6D decision tree

🗐 six_d.txt - Notepad	
<u>File Edit Format View H</u> elp	
<pre>meanx_abs <= 0.3 meany_abs <= 0.3 meany_abs <= 0.3: zdw meany_abs > 0.3: zup meany_abs > 0.3: zup meany_abs > 0.3 meany_abs <= 0.3 meany_s <= 0.3: ydw meany_abs > 0.3: others meanx_abs > 0.3 meany_abs <= 0.3 meany_abs <= 0.3 meany_abs <= 0.3 meany_abs > 0.3: xdw meany_abs > 0.3: others Number of Leaves : 9 Size of the tree : 17</pre>	*
	Ŧ

3.4 Summary of examples

The following table shows a summary of all the examples described in this document in order to show the typical current consumption of the Machine Learning Core in the ISM330DHCX for different configurations. The main contributors are the Machine Learning Core ODR (which might be different from the sensor ODRs), the number of decision trees configured, and the number of nodes (of each decision tree).

Table	10.	Summary	of	examples
--------------	-----	---------	----	----------

Algorithm	MLC_ODR	Number of decision trees	Number of nodes	MLC additional current consumption
Vibration monitoring	26 Hz	1	2	1 µA
Motion intensity	12.5 Hz	1	7	1 µA
6D position recognition	26 Hz	1	8	2 μΑ

Appendix A Weka

Weka is free software developed at the University of Waikato, New Zealand. It cointains a collection of visualization tools and algorithms for data analysis and predictive modeling, together with graphical user interfaces for easy access to these functions.

Weka is one of the most popular machine learning tools for decision tree generation. This section contains some details about this external software, additional details can be found at the links below:

- Weka download
- Weka website
- Weka user guide

All of Weka's techniques are predicated on the assumption that the data is available as one flat file or relation, where each data point is described by a fixed number of attributes.

An ARFF (Attribute-Relation File Format) file is an ASCII text file that describes a list of instances sharing a set of attributes. The ARFF files have two distinct sections, as shown in Figure 44: a header section containing the attributes (features, classes), and a data section containing all the feature values together with the corresponding class to be associated to that set of features.

Figure 44. ARFF example

Bit End Sector Yee Description Description Image: Sector Image: Sector <th>igs\ex</th> <th>xample and + Notepad + + [Administrator]</th> <th>- 0 - X</th>	igs\ex	xample and + Notepad + + [Administrator]	- 0 - X
Image: No. Internet and Image: Internet and Image: Image	dit 1	Search View Encoding Language Settings Tools Macro Run Plugins Window ?	X
<pre>FileWork file FileWork file FileWork file FileWork file FileWork file FileWork file</pre>			
Prestion 'MCd' Stritzburg ASG (MAX), on, ACC, X numeric Stritzburg ASG (MAX), on, ACC, X numeric Stritzburg ASG (MAX), on, ACC, Y numeric			
<pre> *********************************</pre>	iple.ad	# C	
<pre>6 #strtibute AMS(DEMD)_co_NC_7 sumeric 8 strtibute AMS(DEMD)_co_NC_7 sumeric 6 #strtibute AMS(DEMD)_co_NC_7 sumeric 7 #strtibute AMS(DEMD)_co_NC_7 sumeric 8 #strtibute AMS(DEMD)_co_NC_7 sumeric 8 #strtibute AMS(DEMD)_co_NC_7 sumeric 8 #strtibute AMS(DEMD)_co_NC_7 sumeric 8 #strtibute AMS(DEMD)_co_NC_7 sumeric 10 #strtibute AMS(DEMD)_co_NC_7 sumeric 11 #strtibute AMS(DEMD)_co_NC_7 sumeric 12 #strtibute AMS(DEMD)_co_NC_7 sumeric 13 #strtibute AMS(DEMD)_co_NC_7 sumeric 14 #strtibute AMS(DEMD)_co_NC_7 sumeric 15 #strtibute AMS(DEMD)_co_NC_7 sumeric 16 #strtibute AMS(DEMD)_co_NC_7 sumeric 17 #strtibute AMS(DEMD)_co_NC_7 sumeric 18 #strtibute AMS(DEMD)_co_NC_7 sumeric 18 #strtibute AMS(DEMD)_co_NC_7 sumeric 18 #strtibute AMS(DEMD)_co_NC_7 sumeric 19 #strtibute AMS(DEMD)_co_NC_7 sumeric 19 #strtibute AMS(DEMD)_co_NC_7 sumeric 10 #strtibute AMS(DEMC)_co_NC_7 sumeric 11 #strtibute AMS(DEMC)_co_NC_7 sumeric 12 #strtibute AMS(DEMC)_co_NC_7 sumeric 13 #strtibute AMS(DEMC)_co_NC_7 sumeric 14 #strtibute AMS(DEMC)_co_NC_7 sumeric 15 #strtibute AMS(DEMC)_co_NC_7 sumeric 15 #strtibute AMS(DEMC)_co_NC_7 sumeric 15 #strtibute AMS(DEMC)_co_NC_7 sumeric 16 #strtibute AMS(DEMC)_co_NC_7 sumeric 17 #strtibute AMS(DEMC)_co_NC_7 sumeric 18 #strtibute AMS(DEMC)_co_NC_7 sumeric 19 #strtibute AMS(DEMC)_co_NC_7 sumeric 10 #strtibute AMS(DEMC)_co_NC_7 sumeric</pre>	(rel	lation 'MLd'	*
<pre>6 #strtibute AMS(DEMD)_co_NC_7 sumeric 8 strtibute AMS(DEMD)_co_NC_7 sumeric 6 #strtibute AMS(DEMD)_co_NC_7 sumeric 7 #strtibute AMS(DEMD)_co_NC_7 sumeric 8 #strtibute AMS(DEMD)_co_NC_7 sumeric 8 #strtibute AMS(DEMD)_co_NC_7 sumeric 8 #strtibute AMS(DEMD)_co_NC_7 sumeric 8 #strtibute AMS(DEMD)_co_NC_7 sumeric 10 #strtibute AMS(DEMD)_co_NC_7 sumeric 11 #strtibute AMS(DEMD)_co_NC_7 sumeric 12 #strtibute AMS(DEMD)_co_NC_7 sumeric 13 #strtibute AMS(DEMD)_co_NC_7 sumeric 14 #strtibute AMS(DEMD)_co_NC_7 sumeric 15 #strtibute AMS(DEMD)_co_NC_7 sumeric 16 #strtibute AMS(DEMD)_co_NC_7 sumeric 17 #strtibute AMS(DEMD)_co_NC_7 sumeric 18 #strtibute AMS(DEMD)_co_NC_7 sumeric 18 #strtibute AMS(DEMD)_co_NC_7 sumeric 18 #strtibute AMS(DEMD)_co_NC_7 sumeric 19 #strtibute AMS(DEMD)_co_NC_7 sumeric 19 #strtibute AMS(DEMD)_co_NC_7 sumeric 10 #strtibute AMS(DEMC)_co_NC_7 sumeric 11 #strtibute AMS(DEMC)_co_NC_7 sumeric 12 #strtibute AMS(DEMC)_co_NC_7 sumeric 13 #strtibute AMS(DEMC)_co_NC_7 sumeric 14 #strtibute AMS(DEMC)_co_NC_7 sumeric 15 #strtibute AMS(DEMC)_co_NC_7 sumeric 15 #strtibute AMS(DEMC)_co_NC_7 sumeric 15 #strtibute AMS(DEMC)_co_NC_7 sumeric 16 #strtibute AMS(DEMC)_co_NC_7 sumeric 17 #strtibute AMS(DEMC)_co_NC_7 sumeric 18 #strtibute AMS(DEMC)_co_NC_7 sumeric 19 #strtibute AMS(DEMC)_co_NC_7 sumeric 10 #strtibute AMS(DEMC)_co_NC_7 sumeric</pre>			1
<pre>strtibute ABS(DEW)_co_NCC_V = numeric % strtibute ABS(DEW)_co_NC_V = numeri</pre>	Satt	tribute ABS(MEAN)_on_ACC_X numeric	
6 Bettribute AMS(TAN)_co., Xumeric 9 Settribute AMS(TAN)_co., Xumeric 10 Settribute AMS(TAN)_co., ACC_Y numeric 11 Settribute AMS(TAN)_co., ACC_Y numeric 12 Settribute AMS(TAN)_co., ACC_Y numeric 13 Settribute AMS(TAN)_co., ACC_Y numeric 14 Settribute AMS(TAN)_co., ACC_Y numeric 15 Settribute AMS(TAN)_co., ACC_Y numeric 16 Settribute AMS(TAN)_co., ACC_Y numeric 17 Settribute AMS(TAN)_co., ACC_Y numeric 18 Settribute AMS(TAN)_co., ACC_Y numeric 19 Settribute AMS(TAN)_co., ACC_Y numeric 19 Settribute AMS(TAN)_co., ACC_Y numeric 19 Settribute Class (Stationary, Maiking, Jogging, Niking) 19 Settribute Class (Stationary, Kaiking, Jogging, Niking) 19 Settribute Class (Stationary, Kaiking, Jogging, Niking) 10 CodeStat, CodeStat, Loidet, Loidet, Loidet, Actionary 10 Settribute Class (Stationary, Kaiking, Jogging, Niking) 10 Settribute Class (Stationary, Kaiking, Jogging, Niking) 10 CodeStat, CodeStat, Loidet, Loid	Satt	tribute ABS(MEAN)_on_ACC_Y numeric	
P Strtibute XAD(XAD) = 0.CC Y numeric			
1 Setztibute XBS(XB)_cm_ACC_* numeric 1 Se			
<pre>9 Settribute XB(XB)_cm_ACC_VP anneric 10 Nettribute XB(XB)_cm_ACC_VP anneric 11 Nettribute XB(XB)_cm_ACC_VP anneric 12 Nettribute XB(XB)_cm_ACC_VP anneric 13 Nettribute XB(XB)_cm_ACC_VP anneric 14 Nettribute XB(XB)_cm_ACC_VP anneric 15 Nettribute XB(XB)_cm_ACC_VP anneric 16 Nettribute XB(XB)_cm_ACC_VP anneric 17 Nettribute XB(XB)_cm_ACC_VP anneric 18 Nettribute XB(XB)_cm_ACC_VP anneric 19 Nettribute XB(XB)_cm_ACC_VP anneric 10 Nettribute XB(XB)_cm_ACC_VP anneric 10</pre>			
<pre>10 Settribute XaG(NAD)_ca_ACC_V numeric 12 Settribute XaG(NAD)_ca_ACC_V numeric 13 Settribute XaG(NAD)_ca_ACC_V numeric 14 Settribute XaG(NAD)_ca_ACC_V numeric 15 Settribute XaG(NAD)_ca_ACC_V numeric 16 Settribute XaG(NAD)_ca_ACC_V numeric 17 Settribute XaG(NAD)_ca_ACC_V numeric 18 Settribute XaG(SetCross)_on_filter_IFA_GA_ACC_V numeric 18 Settribute XaG(SetCross)_on_filter_IFA_GA_ACC_V numeric 19 Settribute XaG(SetCross)_on_filter_IFA_GA_ACC_V numeric 10 Settribute XaG(SetCross)_SetTribute_IFA_GA_ACC_V numeric 10 Settribute XaG(SetCross)_SetTribute_IFA_GA_ACC_V numeric 10 SetTribute_IFA_GA_ACC_V numeric 10 SetTribute_IFA_ACC_V Numeric 10 SetTribute_</pre>			
<pre>11 #strtibute AdS(EREDGY)_onECty*2 numeric 12 #strtibute AdS(EREDGY)_on_Eliter_PR_ACCy*2 numeric 13 #strtibute AdS(EREDGY)_on_Eliter_PR_ACCy*2 numeric 14 #strtibute AdS(EREDGY)_on_Eliter_PR_ACCy*2 numeric 15 #strtibute AdS(EreCtoss)_on_filer_PR_ACCy*2 numeric 16 #strtibute AdS(EreCtoss)_on_filer_PR_ACCy*2 numeric 17 #strtibute AdS(EreCtoss)_on_filer_PR_ACCy*2 numeric 18 #strtibute class [Stationary, Making, Jougt*, Numeric 18 #strtibute class [Stationary, Making, Jougt*, Outperform, Stationary 10 codesets, 0.0098ct3, 1.01465, 1.01455, 1.01501, J.02456-c6, J.6149-c6, 0.0068554, 0.00078542, 0.0157, 0.00077152, 0.0, 0.0, 0.0107422, Stationary 10 codesets, 0.0098ct4, 1.01456, 1.0151, J.02458-c6, J.6149-c6, 0.000875452, 0.004821, 0.01, 0.015, 0.0, 0.0, 0.0107423, Stationary 10 codesets, 0.0098ct4, 1.0145, 1.0121, 1.0151, J.0154-c6, 0.000971452, 0.000971452, 0.00077152, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.</pre>			
<pre>11 Seturibute AdS (DEEROT)_on_line(r, BC_on_ACC_Y' 2 numeric Seturibute AdS (DEEROT)_on_line(r, BC_on_ACC_Y' 2 numeric Seturibute AdS (DEEROT)_on_line(r, BC_on_ACC_Y' 2 numeric Seturibute AdS (DeeroCross)_on_line(r, BC_on_ACC_Y' 2 numeric Seturibute AdS (DeeroCross)_Seturi</pre>			
<pre>1 stc::box AdS(DEREGYOnEntryINEntryY numeric 1 stc::box AdS(DEREGY_OnEntryINCAV_Y numeric 1 stc::box AdS(DereCorbsyOnItilerINZ, onACC_Y numeric 1 stc::box AdS(DereCorbsyOn_ItilerINZ, onACC_Y numeric 1 stc::box AdS(DereCorbsyOn_ItilerINZ, on_ACC_Y numeric 1 stc::box AdS(D</pre>			
<pre>14 Settlbuck AdS (LercCores)</pre>			
<pre>1 1 Settline ABS (SetCons)</pre>			
<pre>1 8 trtince ASS (PeakToPeak] on ACC 'V numeric 8 trtince class (Stationary, Walking, Orgging, Bing) 1 8 trtince class (Stationary, Orgging,</pre>			
Bata Constraint Constraint <th></th> <th></th> <th></th>			
Bata Construction Constructio	Batt	tribute class (Stationary, Walking, Jogging, Biking)	
 0.0058403, 0.03870425, 1.01465, 1.00315, 1.02850-66, 3.1142-06, 0.00643549, 0.0017422, 10.128, 73.75, 0.000716748, 73.75, 0.0, 0.0107422, Stationary 0.0058970, 1.01465, 1.00315, 1.02825, 4.643164-06, 4.76377-06, 0.00673548, 0.0018208, 73.75, 0.00071523, 10.04565, 50.00136718, 50.0013718, 50.00136718, 50.0013			
 Coccesser, B. Competent, I. Coletty, 2. 202566-06, 7. 42393-06, 0.00278564, D. Coletty, 79. 5175, 0. 0. 0.0107162, Stationary Coccesser, O. Competent, I. Coletty, J. Coletty, 1. 44392-06, 0.00078562, D. Coletty, 79. 5175, D. Coletty, 0. Coletty, Stationary Coccesser, C. Competent, I. Coletty, J. Coletty, 4. 44382-06, 0.00078562, D. Coletty, 0.0007061, 0.0007061, Stationary Coccesser, C. Competent, J. Coletty, J. Coletty, J. Coletty, J. Coletty, C. Competent, J. Coletty, J. Cole	8dat	TA	
 C. 0009207, 0.039975, 1.01465, 1.0322, 4.49348-05, 4.76378-05, 0.000975623, 0.00048283, 0.01, 0.015, 0.000771523, 0.00017533, 0.00071533, 0.00017153, 0.00017153, 0.00017153, 0.00			
 0.0059235, 0.0400656, 1.0127, 1.03526, 3.03984-06, 0.0071213, 0.0059593, 100, 0.0004273, 80, 0, 0, 0.0124693, Bationary 0.0059278, 0.0400656, 1.01352, 1.01354, 1.03542, 0.0354, 6.85070-06, 0.00078542, 0.013513, 0.00039003, 80.125, 0, 0.0124693, Bationary 0.0059277, 0.0397074, 1.0146, 1.0132, 1.40140-05, 0.00078642, 0.0315, 0.00098642, 0.8755, 0, 0, 0.0124693, Bationary 0.00591074, 0.0398073, 1.0146, 1.0132, 1.01242-06, 1.14144-05, 0.00078642, 0.0315, 0.00078643, 0.0126937, 80.135, 0, 0.0124693, Bationary 0.00591076, 0.0397074, 1.0146, 1.0132, 1.01242-06, 1.14144-05, 0.00078642, 79.375, 0.00078642, 0.0315, 0.0009871, 80, 0, 0.012463, Bationary 0.00591076, 0.0398074, 1.01454, 1.0312, 1.01242-06, 0.00078642, 0.00078642, 0.03155, 0.00084515, 80.105, 0, 0.0107422, Bationary 0.00591076, 0.0398064, 1.01454, 1.0312, 1.01242-06, 0.00078642, 0.00078642, 0.03155, 0.00048515, 80.0425, 0, 0.0107422, Bationary 0.00591076, 0.0398064, 1.01454, 1.0312, 1.00728-06, 0.00078642, 0.00078642, 0.03155, 0.00048515, 80.0425, 0, 0.0107422, Bationary 0.00591076, 0.0398064, 1.01454, 1.0312, 1.00728-06, 0.00078642, 0.00078642, 0.03155, 0.00048515, 80.0425, 0, 0.0017422, Bationary 0.00591076, 0.0398064, 1.01454, 1.0312, 1.00454-05, 0.000786454, 0.00059584, 80.0425, 0.000485154, 80.0425, 0, 0.0017422, Bationary 0.0059109, 0.039809, 1.0127, 1.0312, 1.00454-05, 0.000786454, 0.00059584, 80.0425, 0.000485154, 80.0425, 0, 0.0107423, Bationary 0.00591709, 0.039809, 1.0127, 1.0312, 1.03828-05, 0.000786454, 0.00059584, 80.0425, 0.000485545, 80.0425, 0, 0.0107815, Bationary 0.005917109, 0.039809, 1.0127, 1.0322, 1.40840-07, 0.00039645, 0.000786454, 0.000786454, 0.0175, 0, 0.0.017815, Bationary 0.0058456, 0.0389164, 1.01454, 1.0322, 1.4388-07, 0.000396454, 0.000396454, 0.000786454, 0.017580, 7.0, 0.00137815, Bationary 0.0058457, 0.0187174, 1.0322, 1.40807, 0.0			
 C. 00015744, 0.0392151, 1.01545, 1.03015, 3.45070-06, 2.5007754542, 0.000756424, 0.0.01570, 0.0.0265051, 0.0.00075642, 0.0.01570, 0.0.00165053, Stationary C. 000512776, 0.0392761, 1.01646, 1.0315, 3.47267-06, 0.00075642, 0.015312, 0.000775642, 0.05175, 0.0, 0.0126553, Stationary C. 000512776, 0.0395061, 1.01645, 1.03153, 3.77226-06, 0.00075642, 0.0123504, 0.00075642, 0.05175, 0.000167153, Stationary C. 000512776, 0.0395061, 1.01645, 1.03153, 3.77226-06, 0.00075642, 0.00075642, 0.012350, 0.000167153, Stationary C. 000512776, 0.0395061, 1.01645, 1.03153, 3.77226-06, 0.00075642, 0.00075642, 0.012350, 0.000016127, 0.00015785, Stationary C. 00051276, 0.0395061, 1.01645, 1.01535, 3.77226-06, 0.00075642, 0.00075642, 0.00016427, Stationary C. 00051276, 0.0395061, 1.01645, 1.01535, 3.77226-06, 0.00075642, 0.00075642, 7.9.375, 0.00001625, 0.0, 0.0107422, Stationary C. 00051741, 0.0394271, 1.0127, 1.00213, 1.00046-05, 0.00075642, 0.00055914, 0.00051975, 0.0, 0, 0.0101423, Stationary C. 00051741, 0.039501, 1.0127, 1.00213, 1.00046-05, 0.00075642, 0.00051914, 0.0004605, 0.0, 0.0101423, Stationary C. 00051741, 0.039501, 1.0127, 1.00213, 1.00046-05, 0.00075842, 0.00051914, 0.0004605, 0.0, 0.0101423, Stationary C. 00051744, 0.039501, 1.0127, 1.0021, 1.00314, 1.201461-0.051, 0.00074642, 0.00051914, 0.0100142, Stationary C. 00061750, 0.039701, 1.0124, 1.0145, 1.0132, 1.40346-06, 0.00071840, 0.00051912, 10.135, 0.00014615, 0.0011718, Stationary C. 00061751, 0.01371, 1.0145, 1.0132, 1.40346-06, 0.00051945, 0.00021445, 0.011185, 0.0011718, Stationary C. 00061751, 0.013713, 1.0145, 1.0132, 1.40346-06, 0.00051945, 0.00021445, 0.0002100, 0.01718, Stationary C. 0014110, 1.0145, 1.0145, 1.4014, 1			
 C.0030624, 0.0084582, 1.01646, 1.01324, 4.0332-e-66, 9.03076424, 0.01275, 0.000070642, 0.01275, 0.000005, 00.128, 0, 0.012863, Stationary C.0030127, 0.0302761, 1.0164, 1.01324, 3.64584-e-66, 1.020076542, 0.03175, 0.000070648, 0.03175, 0, 0.0120453, Stationary C.0051276, 0.0309671, 1.0146, 1.01324, 3.46584-e-64, 1.000076542, 0.03175, 0.000070648, 0.03175, 0, 0.0120423, Stationary C.0051276, 0.0309761, 1.0146, 1.01324, 1.00125, 1.001264-e-64, 1.010047642, 78, 3075, 0.000070648, 0.03175, 0, 0.01175, 3.000077648, 0.031764 C.0051276, 0.0109761, 1.01524, 1.01213, 1.00128-e-64, 4.67372-e-64, 0.00076542, 1.03175, 0.00017654, 0.03176, 0.010172, Stationary C.0051776, 0.0109761, 1.01364, 1.01215, 1.010247e-64, 0.000776542, 1.03175, 0.00017654, 0.0, 0.010742, Stationary C.0051776, 0.010963, 1.0127, 1.01615, 1.01215, 1.010247e-64, 0.000778542, 1.0012518, 0.0101768, 0.05, 0, 0.0112653, Stationary C.0051716, 0.019963, 1.0127, 1.01615, 1.0127, 1.01857, 0.000178542, 1.0012548, 0.00151831, 0.0128, 0.0017813, Stationary C.0051716, 0.019963, 1.0127, 1.01645, 1.0322, 1.4084e-64, 0.00078544, 0.000078544, 0.0018543, 0.0018543, 0.0018543, Stationary C.00507643, 0.0087573, 1.01364, 1.0132, 1.4084e-64, 0.407928-64, 0.00027864, 0.0018513, 0.001853, 0.0018753, Stationary C.00507643, 0.0087573, 1.01364, 1.0132, 1.40854e-64, 0.00086284, 0.00396248, 0.001851, 0.01785, 0.0, 0.011781, Stationary C.0060646, 0.0189573, 1.01364, 1.0132, 1.40854e-64, 0.00086284, 0.0008645, 0.00087804, 0.01875, 0.0, 0.011781, Stationary C.0060646, 0.0189564, 1.01464, 1.0132, 1.40856e-64, 0.00086284, 0.00396454, 0.01875, 0.0, 0.011781, Stationary C.0060646, 0.0189564, 1.01464, 1.0132, 1.40856e-64, 0.00087644, 0.00086454, 0.103656, 0.1175, 0.0, 0.011781, Stationary C.0188564, 0.0018564, 1.01855, 1.01857, 1.01856, 1.01856, 1.01856, 0.00087644, 0.00087644, 0.000078744, 0.018756			
 0.00391276, 0.0339763, 1.0186, 1.04315, 3.04324-06, 7.42394-06, 0.000976542, 0, 80.25, 0, 0.00075659, 10.875, 0, 0, 0.0124653, Stationary 0.0051276, 0.0339076, 1.01645, 1.01515, 3.7722-06, 1.11441-05, 0.000976542, 0.91357, 0.00016017, 0.0, 0.0017623, Stationary 0.0051276, 0.0339076, 1.01645, 1.01355, 3.7722-06, 0.000976542, 0.91357, 0.00016017, 0.0, 0.0017623, Stationary 0.0051276, 0.0339076, 1.01645, 1.01355, 3.7722-06, 0.000976542, 0.000976542, 0.000195617, 0.00016175, 50.5, 0, 0, 0.0107422, Stationary 0.0051276, 0.0339076, 1.01645, 1.0135, 3.5722-06, 0.000976542, 0.000976542, 0.00019575, 0.0, 0.0001427, 50.5, 0.0, 0.0107422, Stationary 0.0051276, 0.0339076, 1.0127, 1.0323, 6.13868-06, 1.04640-05, 0.000976542, 0.00019531, 0.0.00017925, 60.5, 0, 0, 0.0107423, Stationary 0.0051769, 0.039903, 1.0127, 1.0323, 2.04680-06, 1.04640-05, 0.000976542, 0.00239548, 0.00024955, 0.0, 0.0, 0.010742, Stationary 0.0051709, 0.039903, 1.0127, 1.0323, 2.14680-06, 0.000976542, 0.00239548, 0.00024955, 0.0, 0.0, 0.0107123, Stationary 0.00517109, 0.039903, 1.0127, 1.0323, 2.14689-06, 0.000976542, 0.0023948, 0.0013512, 0.01004005, 50.0425, 0, 0.0013713, Stationary 0.00517109, 0.039903, 1.0127, 1.0332, 1.4388-05, 0.0023948, 0.0039812, 0.012801, 0.001780, 0.0017135, Stationary 0.00517109, 0.039504, 1.0126, 1.01352, 1.3438-05, 0.00023948, 0.00138512, 0.0138512, 0.013851, 0.0017851, Stationary 0.00585142, 0.01485151, 1.010351, 1.049051, 4.0378-06, 0.000976524, 0.000355445, 0.0003765			
 C.00541306, D.0393677, 1.01455, 1.0125, 3.727264-06, 1.013964-05, 0.000278642, 79.3375, 0.00048137, 80, 0.0, 0.0012873, Battomary C.0054956, 0.0393066, 1.01455, 1.01552, 1.03513, 3.07284-06, 0.00028968, 0.0032868, 0.00048137, 80, 0.0, 0.012742, Stationary C.00549776, 0.0393066, 1.01455, 1.0137, 1.0327, 1.04814, 2.000278642, 0.00028964, 0.0009812, 0.1355, 0.000497586, 0.5, 0.0, 0.0107422, Stationary C.00549776, 0.0393066, 1.01457, 1.0352, 1.04940-05, 0.00028968, 0.0015912, 0.01350, 0.000497586, 0.5, 0.0, 0.0107422, Stationary C.00549776, 0.0393066, 1.01457, 1.0352, 1.04804-05, 0.00278684, 0.00028912, 0.0137, 0.000497586, 0.5, 0.0, 0.0107422, Stationary C.00549787, 0.0387733, 1.0127, 1.03813, 2.04804-05, 0.000278648, 0.00028912, 0.0128, 0.000419725, 0.0, 0.0107422, Stationary C.00549786, 0.0389066, 1.01456, 1.0322, 1.43884-05, 0.00028968, 0.00028912, 0.0128, 0.000419725, 0.0, 0.0112783, Stationary C.005497873, 1.0127, 1.03813, 2.24688-06, 0.00039648, 0.00028948, 0.00028948, 0.0011383, 0.0018931, 0.013, 0.0011831, Stationary C.005497450, 0.0387418, 1.01347, 1.0332, 3.45888-06, 3.53878-07, 0.00039648, 0.00028948, 0.0011831, 0.001831, 0.0173, 0.0, 0.011118, Stationary C.005497450, 0.0387418, 1.01347, 1.0332, 3.45888-06, 0.000378648, 0.00039648, 0.0028700, 0.01738, 0.01738, Stationary C.0054956, 0.0389164, 1.01345, 1.0332, 3.45888-06, 0.000378648, 0.00039648, 0.0028700, 0.01738, 0.012863, Stationary C.0054956, 0.014819, 1.01347, 1.0332, 3.4588-06, 0.000716840, 7.0039648, 0.00287008, 50.173, 0, 0. 0.012863, Stationary C.0054956, 0.0148194, 1.01459, 1.0322, 3.48104-064, 0.000716840, 1.04625, 0.00049208, 51.3, 0.002883, Stationary C.0154568, 0.014819, 1.01385, 1.04004, 7.88718-06, 0.000976848, 0.00083548, 0.00083548, 0.00037068, 0.014853, Stationary C.0154568, 0.015461, 1.01555, 1.04019, 4.43734-06, 0.0000976548, 0.00083548, 0.0			
 C. 00075964, 0. 0389059, 1.01582, 1.03738-06, 1.14441-05, 0.00272496, 0.0272496, 0.00078542, 0.00018612, 0.000186			
 0.0054274, 0.0393064, 1.01455, 1.0153, 3.5742805, 6.67572805, 0.000795424, 3.9375, 0.00019427, 10.001427, 0.0, 0.017422, Stationary 0.0005192, 0.0397353, 1.0157, 1.01525, 1.01525, 1.01525, 0.00029864, 0.0015512, 0.0125, 0.00049725, 0.0, 0.0, 0.0126755, Stationary 0.0005192, 0.0397353, 1.0127, 1.0151, 2.0304805, 1.04904-05, 0.00079864, 0.0171, 0.0153, 0.00014927, Stationary 0.0005192, 0.0397452, 1.0127, 1.0151, 2.0304805, 0.00079864, 0.00079864, 0.00097554, 0.0, 0.0, 0.0127615, Stationary 0.00051169, 0.0394552, 1.0127, 1.0151, 2.841028-05, 0.50978564, 0.00097854, 0.000149725, 0.000149725, 0.0, 0.0107185, Stationary 0.00051169, 0.0394552, 1.01454, 1.0137, 1.0351, 3.9351805, 0.000942549, 0.013512, 0.00014862, 0.027375, 0.0, 0.017185, Stationary 0.0005169, 0.0394552, 1.01454, 1.0137, 1.0351, 3.9351805, 0.000942549, 0.013931, 0.0137, 0.00014815, 10.017185, Stationary 0.0005169, 0.0394552, 1.01454, 1.01451, 1.03951805, 0.00094254, 0.000942549, 0.013931, 0.0175, 0, 0.0127453, Stationary 0.0005451, 0.0145161, 1.01451, 1.01451, 1.03931062, 0.00094254, 0.000942549, 0.013931, 0.0175, 0, 0.0126453, Stationary 0.0014605, 0.0145164, 1.01454, 1.01451, 1.01931-0.0004, 0.000715454, 0.01453, 0.00074250, 5t.1575, 0, 0.00124503, Stationary 0.0154547, 0.014151, 1.01551, 1.00071, 1.8703806, 0.000715454, 0.000784545, 0.0155, 0.00071128, 5tationary 0.0154547, 0.014151, 1.01551, 1.00071, 1.8703807, 0, 0, 0.00715450, 0.0155, 0, 0.00171128, 0.0155, 0, 0.0017128, 0.01550, 0.000715450, 0.01550, 0.000715450, 0.01550, 0.000715450, 0.01550, 0.000715450, 0.01550, 0.000715450, 0.01570, 0.0117188, 5tationary 0.015547, 0.0145151, 1.01551, 1.00071, 1.572208-054, 0.000715454, 0.000715454, 0.017158, 0.01550, 0.000715450, 0.0157556, 0.000715450, 0.0157556, 0.000715450, 0.0157556, 0.000715450, 0.000715450, 0.0117188, 5tationary 0.0157560, 0.0155561, 0.0			
 0.00051742, 0.0394207, 1.01257, 1.01223, 6.19858-05, 1.049304-05, 0.00229584, 0.00125512, 0.0125, 0.00047756, 0.5, 0. 0, 0.0107423, Stationary 0.00051742, 0.039703, 1.0127, 1.01213, 1.01940-05, 0.00976542, 0.00255514, 00.0423, 0.00047756, 0.5, 0.00107423, Stationary 0.00051709, 0.039703, 1.0127, 1.01263, 1.01646-05, 0.00078562, 0.0025554, 00.0423, 0.00046455, 00.00107423, Stationary 0.00051709, 0.039703, 1.0127, 1.01263, 1.21804-05, 0.00078562, 0.00052564, 0.0011253, 0.00046455, 0.0, 0.0107423, Stationary 0.00051709, 0.039703, 1.01456, 1.01323, 1.45856-05, 0.00078562, 0.00052564, 0.0011253, 0.00146453, 500, 0.0, 0.0117183, Stationary 0.00051750, 0.039703, 1.0126, 1.01353, 1.93950-06, 0.00052564, 0.00197664, 0.978350, 0.0024700, 0.0175, 1.512, Stationary 0.000617645, 0.0398064, 1.01465, 1.0322, 3.15900-06, 0.00052564, 0.00197664, 7.8375, 0.000247002, 0.0175, 0, 0.0107183, Stationary 0.000617645, 0.0398064, 1.01465, 1.0322, 3.15900-06, 0.000718644, 0.00396454, 0.01158, 0.001700, 0.0175, 0, 0.0107483, Stationary 0.00061646, 0.0148154, 1.01265, 1.01264, 1.0127, 1.0136, 0.000718644, 0.00396454, 0.00396454, 0.01175, 0, 0.0112685, Stationary 0.0014664, 0.0148154, 1.01265, 1.01264, 1.0127, 1.0136, 0.000718644, 0.00396454, 0.00396454, 0.01175, 0, 0.0112685, Stationary 0.0114654, 0.0148154, 1.01265, 1.0127, 1.51266-66, 0.000716544, 0.00396454, 0.013964, 0.01175, 0, 0.0112685, Stationary 0.0114654, 0.0148154, 1.01554, 1.01554, 1.01564, 4.4838-06, 0.000716544, 0.00396454, 0.01756, 0.00216653, Stationary 0.0114754, 1.01555, 1.0155, 1.0155, 1.0156, 4.4738-045, 0.000716545, 0.000716545, 0.0037057, 0.0146545, Stationary 0.011755, 0.0145555, 1.01555, 1.01555, 1.0155, 1.0155, 1.0155, 0.000716554, 0.000716554, 0.000716554, 0.011155, 0.000716554, 0.000716554, 0.000716554, 0.000716554, 0.00111155, 0.0111755, 0.0111755, 0.0111755, 0.0111155, 0.000716554,			
 0.0005392, 0.0397339, 1.0127, 1.00812, 2.0348-06, 1.04904-05, 0.00978564, 0.00623544, 00.049725, 80, 0, 0, 0.0107422, Stationary 0.00053109, 0.0399054, 1.0127, 1.00812, 2.648108-06, 6.67572-06, 0.00078504, 0.0058314, 0.125, 0.00042462, 0.005, 0, 0.0137185, Stationary 0.00051169, 0.0394502, 1.01245, 1.01325, 2.64988-05, 0.00396254, 0.003922494, 0.12135, 0.00042462, 0.0, 0, 0.0137185, Stationary 0.0005169, 0.0398704, 1.01247, 1.03251, 2.45818-06, 0.00396254, 0.003922494, 0.12135, 0.00024702, 80.175, 0, 0.0037815, stationary 0.0005465, 0.0398704, 1.01247, 1.03251, 3.35936-06, 0.00054624, 0.00396254, 0.0239705, 0.01351, 0.0175, 0, 0.0137815, Stationary 0.0005465, 0.0339704, 1.01454, 1.01247, 1.03251, 3.35936-06, 0.00078642, 0.00396254, 0.0239705, 0.012765, 0, 0.0017212, Stationary 0.0014605, 0.0139504, 1.01454, 1.01247, 1.0351, 1.0499, 4.3738-06, 0.00078642, 0.0039625, 0.25, 0.039703, 0.017185, Stationary 0.015869, 0.0135462, 1.01551, 1.0091, 5.83978-06, 0.00078642, 0.00396425, 0.51, 0.0017105, 0, 0.0126853, Stationary 0.015869, 0.0135462, 1.01551, 1.0091, 5.8238-06, 0.00078542, 0.00078545, 0.00078545, 0.0171128, 0.00171128, 5.84100ary 0.015869, 0.0135462, 1.01551, 1.0091, 5.8238-06, 0.00078545, 0.00078545, 0.00078545, 5.0, 0.0017128, 0.5160578, 5.1, 0.00171128, 5.84100ary 0.0158569, 0.0135462, 1.01551, 1.0091, 5.8238-06, 0.000785450, 0.00078545, 0.00078545, 0.00078545, 0.0017128, 5.81100ary 0.0158569, 0.0135422, 1.01551, 1.0091, 5.8238-06, 0.00978542, 0.00078545, 0.0007854			
 0.00051169, 0.0394552, 1.01455, 1.0323, 2.2449806, 1.2525805, 0.00232269, 0.0015312, 0.25, 0.00024462, 0.25, 0, 0.017128, factomary 0.0005456, 0.0397542, 1.01457, 1.0325, 3.4558805, 0.0095462, 0.02922498, 0.1235, 0.00024462, 0.0, 0.0075152, 0.1235 0.0005457, 0.038774, 1.01267, 1.0325, 3.4558805, 0.00954625, 0.0095465, 0.03758, 0.00024702, 0.017515, 0, 0, 0.0175185, Stationary 0.0005465, 0.0393054, 1.01456, 1.01253, 3.1950505, 0.00075454, 0.00758454, 0.03753, 0.00024702, 0.0175, 0, 0.00075453, 0.00175150, 0.0175150, 0.01075150, 0.01075103, 0.0107510, 0.00075103, 0.0107510, 0.0107510, 0.0107510, 0.0107510, 0.0107510, 0.0107510, 0.00075103, 0.00075120, 0.00075120, 0.00075120, 0.00075120, 0.00075120, 0.00075103, 0.000075103, 0.0007510, 0.00075103, 0.00075103, 0.00075103, 0.0007			
 0.00097465, 0.00397573, 1.01257, 1.0325, 1.03585-06, 3.550748-07, 0.00390425, 0.00292465, 0.012335, 0.0013935, 0.0175761, Stationary 0.0042565, 0.0387573, 1.01257, 1.0325, 1.93555-06, 7.42339-06, 0.00482154, 0.0047360, 7.8375, 0.00237005, 0.017576, 0, 0.00129633, Stationary 0.00450566, 0.0148455, 1.01255, 1.01256, 1.0126, 1.01025, 0.00258654, 0.000390454, 0.014253, 0.00218633, Stationary 0.0146567, 0.0144154, 1.01555, 1.01256, 1.0155, 1.01015, 0.0216576, 0.00055864, 0.00055864, 0.00482356, 0.0148536, Stationary 0.014567, 0.014415, 1.01555, 1.01015, 1.21256-06, 0.0015564, 0.00055864, 0.00055864, 0.00148258, 0.0148535, Stationary 0.014567, 0.014415, 1.01555, 1.01015, 1.21256-06, 0.00156364, 0.00055864, 0.00058644, 0.01481545, Stationary 0.014569, 0.0144154, 1.01555, 1.01015, 1.21256-06, 0.00156364, 0.00055864, 0.00055864, 0.0, 0.0148155, Stationary 0.0117075, 0.0145154, 1.01555, 1.01015, 1.21256-06, 0.00946245, 0.00055844, 0.0005584, 0.0, 0.011165, Stationary 0.0117075, 0.0145154, 1.01555, 1.010154, 1.21256-06, 0.00946245, 0.00055844, 0.000576454, 0.000576454, 0.0017175, 0.0, 0.0146545, Stationary 0.0117075, 0.0145154, 1.01555, 1.01004, 4.4178-05, 3.00946-06, 0.000576544, 0.000576544, 0.000270654, 0.0117154, Stationary 0.0117075, 0.0145154, 1.01555, 1.01554, 1.01554, 1.21256-06, 0.000576544, 0.000576544, 0.000270654, 0.000270654, Stationary 0.0117175, 0.0145154, 1.01555, 1.01554, 1.21256-06, 0.00057654, 0.000576544, 0.000270654, 0.0117154, 5tationary 0.0111715, 0.0145154, 0.101555, 0.1015164, 5tationary 0.0111715, 0.0145154, 0.102554, 1.01255, 0.011185, 5tationary 0.0111715, 0.0145154, 0.101554, 0.101564, 0.1000576545, 0.000576544, 0.000576544, 0.000270654, 0.00171185, 5tationary 0.0111154, 0.0111154, 0.101554, 0.0111154, 0.111154, 0.0111154, 0.01111154, 0.0111154, 0.0111154, 0.0111154, 0.0111154, 0.0111154,	0.00	0537109, 0.039093, 1.0127, 1.03613, 2.86102e-06, 6.67572e-06, 0.00878906, 0.00585938, 80.125, 0.000460625, 80.0625, 0, 0, 0.0136719, Stationary	
 0.0014827, 0.0387184, 1.01387, 1.03831, -058, 7.42338405, 0.00487281, 0.00178964, 79.3783, 0.00247002, 80.175, 0, 0.0127422, Stationary 0.001566, 0.0339046, 1.01456, 1.01253, 3.1950505, 6.58307-056, 0.00976452, 0.00396255, 61.275, 0, 0.00124853, Stationary 0.017460, 0.0148168, 1.02651, 1.01459, 1.02734-05, 2.841024-056, 0.000784542, 0.00394255, 61.4675, 0, 0.00121858, Stationary 0.015469, 0.0135422, 1.01859, 1.04004, 1.09735-05, 0.00078542, 0.00275358, 1.0425, 0, 0.0011188, Stationary 0.015469, 0.0135422, 1.01855, 1.03711, S.722054-05, 0, 0.00078542, 0.00278549, 0.017128, 0, 0.0111188, Stationary 0.015549, 0.0135422, 1.01855, 1.10971, S.722054-05, 0, 0.00078542, 0.00235496, 0.0.475, 0.000270547, 51.275, 0, 0.0011188, Stationary 0.015549, 0.0135422, 1.01855, 1.10971, S.722054-05, 0, 0.00078542, 0.00235496, 0.0.475, 0.000270547, 51.275, 0, 0.0011188, Stationary 0.018755, 0.0143555, 1.01553, 1.10971, S.722054-05, 0.00078542, 0.00235496, 0.0.475, 0.000270547, 51.275, 0, 0.0017188, Stationary 0.018756, 0.013556, 1.01553, 1.03971, 2.24584-05, 0.000376354, 1.00253, 0.00270547, 51.2055, 0, 0.0017188, Stationary 0.018756, 0.013556, 1.10355, 1.10951, 1.2002, 2.2458-05, 0.000376354, 1.0025, 0.00270547, 51.275, 0, 0.0017188, Stationary 0.018756, 0.013556, 1.10355, 1.0315, 1.0371, 2.2458-05, 0.000376354, 1.0025, 0.000370545, 0.000370545, 0.000370545, 0.000370545, 0.000370545, 0.11788, Stationary 0.018756, 0.113561, 1.0355, 1.0355, 1.0371, 2.2458-05, 0.000376354, 0.000376354, 0.1178, 0.000370545, 0	0.00	0451169, 0.0394592, 1.01465, 1.0332, 2.26498e-06, 1.52588e-05, 0.00292969, 0.00195312, 80.25, 0.000424862, 80.25, 0, 0, 0.0117188, Statiomary	
 O.COMSD569, O.CONSD669, J. 101455, J. 102213, J.159050-06, E.58070-06, O.CONSP4624, D.CONSD625, D.C.S. 0.COMST9005, NO.18775, O., O. O.CONSF635, Stationary O.CONSD609, O.CONST612, J.CONSS, J. 104004, T.867E1-06, I.50735-06, O. O.CONSD625, D.C.S. 0.CONSP625, D.C.S. 0.CONSF635, Stationary O.CONSD629, O.CONST622, J.CONSS, J.10871, J.2025-06, A.76837-07, O. O.CONSF635, Stationary O.CONSD629, O.CONST622, J.CONSS, J.CONST620, D.CONST622, D.CONST620, D.CONSF6359, Stationary O.CONST622, J.CONST62, J.CONSS, J.CONST624, D.CONST622, D.CONST626, J.CONST626, J.CONSF635, J.CONST647, O.CONSF635, J.CONSF635, D.CONSF635, D.CONSF6354, D.CONS	0.00	0587463, 0.0387573, 1.01367, 1.0332, 3.63588e-06, 9.53674e-07, 0.00390625, 0.00292969, 80.3125, 0.00118351, 80.875, 0, 0, 0.0175781, Stationary	
 C.017460, C.01248168, 1.02051, 1.04199, 4.64378-06, 2.840128-06, 0.00055388, E1.0425, 0.000482935, E1.6475, 0.0, 0.0124853, Stationary C.015547, 0.014151, 1.01585, 1.04004, 1.89738-06, 0.00071851, 0.610, 0.00178153, 0.0111188, Stationary C.015549, 0.0135422, 1.01855, 1.0311, S.722058-06, 4.76837-07, 0, 0, 20.6275, 0.000784514, 0.012683, Stationary C.015549, 0.0135422, 1.01853, 1.0391, S.43285-06, 1.66838-06, 0.00078542, 0.00232849, 0.0.750, 0.000370847, 8.13675, 0, 0.0017188, Stationary C.0187578, 0.0143555, 1.01855, 1.0185, 1.6393, -65, 3.00944-06, 0.00078542, 0.0063354, E1.0625, 0, 0, 0.017188, Stationary C.0187578, 0.013555, 1.01855, 1.0185, 1.01947, J.5256-06, 2.26488-06, 0.00078542, 0.0063354, E1.0750, 0.000270867, E1.1075, 0, 0.0017188, Stationary C.0187618, 0.0135054, 1.01855, 1.01951, J.26497-06, 0.00096254, 0.0063354, E1.0750, 0.000270867, E1.1075, 0, 0.0117188, Stationary C.0187614, 0.0135565, 1.10351, 1.0491, 4.3738-06, 2.26488-06, 0.00085349, E1.0750, 0.000370367, E1.1075, 0, 0.0117188, Stationary C.0187614, 0.0135565, 1.10351, 1.0491, 4.3738-06, 0.00085349, E1.0.750, 0.000370357, E1.1075, 0, 0, 0.0117188, Stationary C.0187614, 0.0135565, 1.10351, 1.0491, 4.3738-06, 0.00085349, E1.0750, 0.000873454, D.1017188, Stationary C.0187615, 0.013178, J.104510, 4.3738-06, 0.000815349, E1.0750, 0.000812354, E1.175, 0, 0, 0.0117188, Stationary C.0187614, 0.0135568, 1.02051, 1.0351, 4.3738-06, 0.00081534, E1.750, 0.000812345, E1.175, 0, 0, 0.0117188, Stationary C.0187615, 0.0117188, Stationary C.0187615, 0.011718, J.104510, J.40779-06, 0.000815349, E1.750, 0.000812354, E1.175, 0, 0, 0.0117188, Stationary C.0187615, 0.0117188, J.10451, 0.0007614204, 0.000815344, E1.0750, 0.00017188, J.1050, 0.0017188, J.1050,			
 0.0185547, 0.014133, 1.01955, 1.0197, 7.067918-06, 0, 0.0071035, 00.0253, 00.0071845, 0.0071845, Santionary 0.0185547, 0.014133, 1.01855, 1.01155, 1.01155, 1.01155, 1.01155, 1.01155, 0.000718416, 1.01655, 0.000718416, 1.017186, 1.01			
 O.028569, 0.013562, 1.01553, 1.03711, 5.72205-c65, 4.76237-c70, p. 0. 80.4875, 0.00078414, 81.20453, 0. 0. 0.0129383, Dationary O.028510, 0.014556, 1.01553, 1.01396, 5.44356-c6, 1.00976542, 0.002978542, 0.027367, 81.1475, 0. 0. 0.011718, Jationary O.0287378, 0.014355, 1.01553, 1.04004, 6.4373-c6, 3.09944-c6, 0.000976542, 0.00293454, 81.0475, 0.00007057, 81.1475, 0. 0. 0.011718, Jationary O.0287378, 0.0143554, 1.01553, 1.04004, 6.4373-c6, 3.09944-c6, 0.000976542, 0.0029344, 81.0020374, 81.1475, 0. 0, 0.00974542, Jationary O.0287456, 0.0120161, 1.01553, 1.014904, 6.4373-c6, 3.0094524, 0.00423434, 81.0402, 0.0003742, 21.1475, 0. 0, 0.011718, Jationary O.0287456, 0.0120161, 1.01553, 1.014904, 6.4374-c6, 3.00453494, 81.04024, 0.0003742, 0.00032355, 81.1755, 0, 0, 0.0114464, Jationary O.0287456, 0.0120161, 1.01553, 1.014904, 6.4374-c6, 3.00453494, 81.04024, 0.00037323, 0.00032355, 81.1755, 0, 0, 0.011218, Jationary O.0287456, 0.0120161, 1.01553, 1.014904, 5.4374-c6, 3.00453494, 81.04024, 0.0003235, 0.0003235, 81.1755, 0, 0, 0.011218, Jationary O.0287456, 0.0120161, 1.01553, 1.014914, 5.4408-c64, 3.00453494, 81.040243, 0.0003235, 81.1755, 0, 0, 0.011218, Jationary O.0287456, 0.012056, 1.01553, 1.1354, 6.3764, 0.0048734, 81.04253, 0.00048235, 81.1475, 0, 0, 0, 0.011218, Jationary O.0287456, 0.012056, 1.00553, 1.01563, 1.1350-c64, 0.0048734, 81.04253, 0.00048235, 81.1475, 0, 0, 0.011218, Jationary O.0287456, 0.012056, 1.01055, 1.1355, 1.44046, 5.46046-c6, 1.0004874, 0.014874, 0.014874, 0.11478, Jationary O.0287456, 0.012056, 1.0056, Jationary O.0287456, 0.014644, Jationary 			
 0.0183105, 0.018558, 1.01855, 1.01856, 5.48358-05, 1.668938-05, 0.003976552, 0.00329459, 00.375, 0.00405079, 81.15625, 0.0, 0.0017188, Stationary 0.0187778, 0.0143555, 1.01855, 1.01855, 1.44738-06, 3.09944-06, 0.009976542, 0.00683584, 81, 0.000070367, 81.1575, 0, 0, 0.00976542, Stationary 0.0189178, 0.0180156, 1.01855, 1.01855, 1.01857, 0.21854-06, 0.0048254, 0.00397652, 0.00683584, 81, 0.000070367, 81.1575, 0, 0, 0.014484, Stationary 0.0189178, 0.0180154, 1.01855, 1.01855, 0.44738-06, 2.26498-06, 0.0048254, 0.0068394, 81, 0.000370355, 81.175, 0, 0, 0.014484, Stationary 0.0189181, 0.018356, 1.02055, 1.03851, 6.4738-06, 2.04498-06, 0.0048254, 0.10068394, 81, 0.00033255, 81.175, 0, 0, 0.011188, Stationary 0.0189181, 0.018351, 1.01855, 1.50855, 6.4738-06, 2.04498-06, 0.0048254, 0.10068394, 81, 0.00033255, 81.175, 0, 0, 0.011188, Stationary 0.0189181, 0.018351, 1.01851, 5.40846, 5.34984-06, 0.0008394, 81, 0.00083948, 81, 0.00033255, 81.175, 0, 0, 0.011188, Stationary 0.0189181, 0.018351, 1.01851, 5.40846, 5.34984-06, 0.0008394, 0.015, 0.00048395, 81.1255, 0, 0, 0.011188, Stationary 0.0189181, 0.018351, 1.01851, 5.40846, 5.34984-06, 0.0008394, 81, 0.00083958, 81.1055, 0, 0, 0.011188, Stationary 			
 0.0187378, 0.018855, 1.01855, 1.04004, 6.4373-e-6, 3.09944-e-6, 0.000976542, 0.00097554, 0.00073057, 81.1875, 0, 0.00978542, Stationary 0.0181456, 0.0180161, 1.01855, 1.01895, 7.1354e-64, 2.2494e-66, 0.0009542, 0.00083948, 81, 0.00037382, 1.1875, 0, 0, 0.0144644, Stationary 0.018141, 0.018569, 1.02053, 1.03613, 4.4378-e-64, 2.24498-e-66, 0.0008234, 81, 0.6203, 0.00032355, 81.1755, 0, 0, 0.011218, Stationary 0.018113, 0.011373, 1.0355, 1.03643, 5.4648e-66, 2.00082349, 81, 0.0008334, 81, 0.6203, 0.00032355, 81.1755, 0, 0, 0.011218, Stationary 0.018131, 0.011313, 1.0355, 1.03643, 5.4648e-66, 2.00082349, 81, 0.620, 0.0003235, 81, 0.000334, 81, 0.01135, 0, 0.011218, Stationary 			
 0.0182495, 0.0138016, 1.01855, 1.04199, 7.15254e-66, 2.24498e-66, 0.00030625, 0.00483594, 81, 0.000507832, 81.1875, 0, 0, 0.014484, Stationary 0.0189141, 0.0148549, 1.02051, 1.00451, 4.473e-66, 2.24498e-66, 0.0048221, 0.0048394, 81.1682, 0.00032385, 81.1875, 0, 0, 0.0111188, Stationary 0.018731, 0.0143731, 1.016531, A.4078e-66, 2.24498e-66, 0.0048234, 0.1068394, 81.04530, 0.0003235, 81.1875, 0, 0, 0.0111188, Stationary 0.018731, 0.0143731, 1.016531, A.4078e-66, 2.24498e-66, 0.00048234, 0.1068394, 0.155, 0.00048234, 0.1275, 0, 0.0111188, Stationary 			
43 0.0188141, 0.018556, 1.02053, 1.03613, 6.4373-06, 2.246488-06, 0.00487281, 0.00683584, 81.0623, 0.00032235, 81.073, 0, 0, 0.011788, Sationary 40.0287381, 0.014373, 1.0359, 1.03604, 5.49648-06, 1.13020-06, 0, 0.00683584, 81.0423, 0.00049239, 81.125, 0, 0, 0.011788, Sationary			
44 0.0187531, 0.0141373, 1.01953, 1.04004, 5.96046e-06, 1.19209e-06, 0, 0.00683594, 80.75, 0.000498295, 81.125, 0, 0, 0.0117188, Stationary			
			-
	_		-
Normal text file length: 4,215,755 lines: 31,380 Ln:1 Col:15 Sel:0 0 Windows (CRLF) UTF-8 I	text fil	ile length: 4,215,755 lines: 31,380 Ln: 1 Col: 15 Sel: 0 0 Windows (CR LF) UTF-8	INS

Figure 45. Weka GUI Chooser



When launching Weka, the Weka GUI Chooser window appears (Figure 45), and the "Explorer" section, selectable through the first button, is the Weka main user interface.

When selecting the Weka Explorer a new interface appears (Figure 46). Several panels are available in the Explorer interface:

- The *Preprocess* panel has facilities for importing data.
- The *Classify* panel allows applying classification and regression algorithms to the dataset in order to estimate accuracy of the resulting predictive model and to visualize erroneous predictions.
- The Cluster panel gives access to the clustering techniques in Weka.
- The Associate panel provides access to association rule learners that attempt to identify all important interrelationships between attributes in the data.
- The Select attributes panel provides algorithms for identifying the most predictive attributes in a dataset.
- The Visualize panel shows a scatter plot matrix.

In this appendix section, only the Preprocess and Classify panels are described.

The Preprocess panel is shown in Figure 46, it allows loading an ARFF file from the "Open file" button.

🕝 Weka Explorer				
Preprocess Classify Clust	ter 🗍 Associate 🗍 Select attributes 🗍 V	/isualize		
Open file Open UF	RL Open DB Gene	undo	Edit	Save
Choose None				Apply
Current relation		Selected attribute		
Relation: None Instances: None	Attributes: None Sum of weights: None	Name: None Missing: None	Distinct: None	Type: None Unique: None
All None	Invert Patterm			Visualize All
Status Welcome to the Weka Explorer	r			Log 💉 x0

Figure 46. Weka Explorer

When the ARFF file has been loaded, the preprocess panel shows all the attributes (features and classes) of the imported ARFF file. The attributes can be visualized in a graphical way and the user can select the attributes to be used for the classification.

🕢 Weka Explorer	
Preprocess Classify Cluster Associate Select attributes	Visualize
Open file Open URL Open DB Gen	erate Undo Edit Save
Choose None	Apply
Current relation	Selected attribute
Relation: ProgrammableSensor Attributes: 3 Instances: 264 Sum of weights: 264	Name: ABS(VAR)_on_ACC_V Type: Numeric Missing: 0 (0%) Distinct: 77 Unique: 60 (23%)
Attributes	Statistic Value
All None Invert Pattern	Minimum0Maximum0.596Mean0.059StdDev0.138
Current relation Relation: ProgrammableSensor Instances: 264 Attributes All No. No. No. No. No. Status OK	Class: class (Nom) Visualize All
Remove	
Status	
ок	Log x0

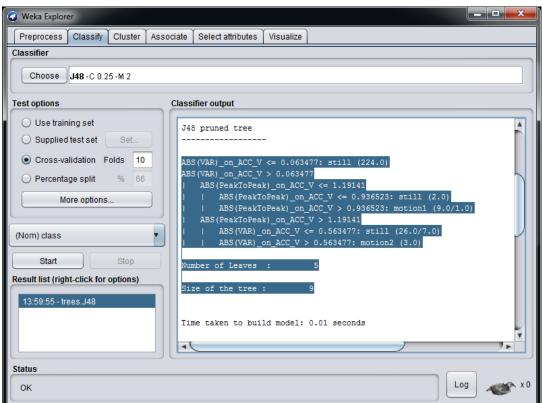
Figure 47. Weka Explorer - Attributes

After choosing the attributes, a classifier can be configured in the *Classify* panel of Weka Explorer (Figure 48). There are many classifiers available in Weka: by choosing the classifier J48 (under "trees") a decision tree can be generated (Figure 49).

📀 Weka Explorer	
Preprocess Classify Cluster Associate Select	attributes Visualize
Classifier	
V 🚔 weka V 🚔 classifiers	
Te bayes	put
▶ È functions	
► 💼 lazy	
▶ 🚞 meta	
► 📄 misc	
► 📄 rules ▼ 🗁 trees	
DecisionStump	
HoeffdingTree	
J48	
() LMT	
M5P	
RandomForest	
Re REPTree	
	· · · · · · · · · · · · · · · · · · ·
Close	
Status	
ок	Log x0
L	

Figure 48. Weka Classify





Many parameters can be changed in the classifier section (Figure 50), and different decision trees can be generated by clicking the "Start" button (see Figure 49).

🕢 weka.gui.GenericObjectEditor		x							
weka.classifiers.trees.J48									
About									
Class for generating a pruned									
	Capabilities								
batchSize	100								
binarySplits	False								
collapseTree	True								
confidenceFactor	0.25								
debug	False								
doNotCheckCapabilities	False								
doNotMakeSplitPointActualValue	False	•							
minNumObj	2								
numDecimalPlaces	2								
numFolds	3								
reducedErrorPruning	False	•							
saveInstanceData	False	•							
seed	1								
subtreeRaising	True	•							
unpruned	False	•							
useLaplace	False	•							
useMDLcorrection	True	•							
Open Save	e OK Cancel								

Figure 50. Weka J48 classifier parameters

All the decision trees generated can be easily compared in terms of:

Number of nodes

Since the decision tree generated by the J48 algorithm in Weka is a binary tree, the number of nodes can be obtained by subtracting one from the parameter "Number of Leaves" which appears in the first row just after the decision tree (see Figure 51. Correctly classified instances). It is also possible to visualize the decision tree graphically by right-clicking on the "Result list" section on the left part of the tool (where all the models created can be easily compared).

Correctly classified instances

It is an estimate of the accuracy of the model created. The result of the model is compared to the result provided by the labels. Figure 51. Correctly classified instances shows the correctly classified instances of an activity recognition model.

Confusion matrix

An NxN table that summarizes how successful the classification model predictions were, i.e. the correlation between the label and the model classification. One axis of a confusion matrix is the label that the model predicted, and the other axis is the actual label.

🥥 Weka Explorer										
Preprocess Clas	sify Cluster	Associate	Select attributes	Visualize						
Classifier										
Choose J48 -	C 0.25 -M 2									
Test options		Class	ifier output						 	
Use training si Supplied test s Oross-validatio Percentage sp More op (Nom) class	et Set on Folds 10 lit % 66	Nu Si	Al ABS (El	ABS (ZeroCros ABS (MEAN ABS (MEAN NERGY)_on_f coss)_on_fi : :	<pre>s)_on_filte)_on_ACC_Z)_on_ACC_Z ilter_BP_or lter_BP_on_ 198 95</pre>	er_BP_on_ACC <= 0.062927 > 0.062927: n_ACC_V^2 > ACC_V^2 > 2	_V^2 > 24 : Walking Jogging 392.25: Jo	(3.0)		
Result list (right-clic			= Stratified cro = Summary === rrectly Classif	ied Instanc	es 30	0822 539		.2813 % 7187 %		
Status OK										Log 💉 x0

Figure 51. Correctly classified instances

Figure 52. Confusion matrix shows an example of a confusion matrix for an activity recognition algorithm with four classes (Stationary, Walking, Jogging, Biking).

Weka Explorer							_ D X	
Preprocess Classify Cluster Ass	ociate Sel	lect attrib	utes	Visualiz	e			
Classifier					_			
Choose J48 - C 0.25 - M 2								
Test options	Classifier	output						
O Use training set	-	nfusion	Matri	×				
Cross-validation Folds 10	a		c	 d		< classified as		
O Percentage split % 66	9852	_	0	144		a = Stationary		
More options	20	17907 19	16 1070	08 0	T T	b = Walking c = Jogging		
(Nom) class	126	111	0	1993	i	d = Biking		
OK						Log) 🛷 x0	

Figure 52. Confusion matrix

Appendix B RapidMiner

RapidMiner is a data science software platform which provides an integrated environment for data preparation, machine learning, deep learning, text mining, and predictive analytics. It is used for business and commercial applications as well as for research, education, training, rapid prototyping, and application development and supports all steps of the machine learning process including data preparation, results visualization, model validation and optimization.

This appendix describes the process to generate a decision tree starting from an ARFF file, using RapidMiner Studio. A simple example of a hand-washing detection algorithm is considered for this purpose. After opening RapidMiner Studio, the main steps are the following:

- 1. Add the "Open File" operator from the "Operators" window on the left, and drag the operator to the blank "Process" window as shown in Figure 53.
- 2. Double click the "Open File" operator to choose the ARFF file to be loaded.
- 3. Find the "Read ARFF" operator and drag it to the "Process" window. Then connect the "Read ARFF" operator to the "Open File" operator as shown in Figure 54.
- 4. Find the "Set Role" operator and drag it to the "Process" window. Then, double click the "Set Role" operator and type the attribute name and target role in the "Parameters" window as shown in Figure 55.
- 5. Find the "Decision Tree" operator and set the corresponding parameters as shown in Figure 56. You also need to connect the "Decision Tree" operator to "res".
- 6. Click the "Run" button (blue triangle icon) in the upper left section of RapidMiner Studio.
- 7. After the "Run" button has been clicked, the "Results" tab will show the decision tree generated, in terms of "Graph" (Figure 57) and "Description".
- 8. In the "Description" section of the decision tree generated (Figure 58) you need to copy the decision tree to a text file, which can be imported in the MLC tool in Unico.

🧶 <new process*=""> – RapidMiner Stu</new>	udio Trial 9.2.001 @ SC	ZCWL6602								
<u>File Edit Process View Connections</u>	Cloud Settings Exten	sions <u>H</u> elp								
	Views:	Design	Results	Turbo Prep	More 🔻]	Find data, oper	atorsetc	2	All Studio 🔻
Repository ×	Process						Paramete	rs	× Help	×
🔁 Import Data 🛛 🖃 🔻	Process		100%	, p p p	🛃 🍒 🍠	[]		Data Edit	or ×	
🕨 🔁 Samples	Process					^		i ijir i i		F IIF 🌩
🕨 📕 DB										
Local Repository (hhunhis)	Dinp					re				
Figure 1 Temporary Repository (hhunhia)										
Cloud Repository (disconnected)										
Operators X Open File X Clility (1) Clility (1) Open File (1) Open File		nen File				-				
	<					>				
	Leverage the Wisdom of design!	Crowds to get ope	rator recommen	dations based on	your process	\$				
Get more operators from the <u>Marketplace</u>		Activate	Wisdom of Crov	vds			Drag&Drop a	n Example	Set from the	repository or

Figure 53. RapidMiner Studio - Open File



<u>Eile Edit Process View Connections Cloud Settings Extensions H</u> elp	
Eine Eine Eine Einen Einen Connections Ciona Settensions Eineh	
Image: Second	🔎 All Studio 🔻
Repository X Process	Help \times
🕒 Import Data 📃 👻 🤍 Process	Parameters × Data Editor ×
Samples Process	
> II DB	t 🗧 📕 🔁
▶ 🜉 Local Repository (hhunhia) Disp res (
F Temporary Repository (hhun)	
Cloud Repository (disconnect	
Operators Read Arf Data Access (1) Read (1) Read ARFF	
Leverage the Wisdom of Crowds to get operator recommendations based on your process design!	
Set more operators from Crowds Image: the Marketplace Dr	rag&Drop an Example

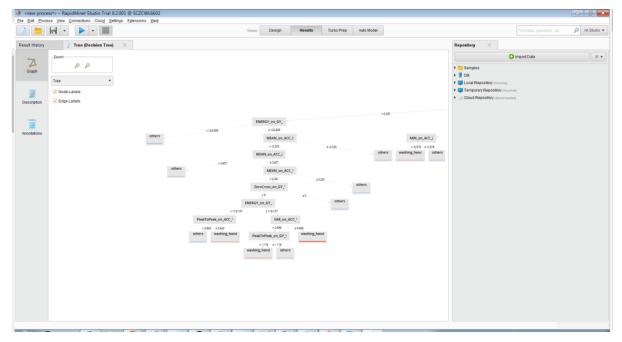


Contractions × Decision × Contractions × Cont	<new process*=""> – RapidMiner Studio Trial 9.2.003</new>					- 0 -
Open File Image: Section for Sectio		Egensions Help	Views: Design Results Turbo Prep	Auto Model		ata, operatorsetc 👂 All Studio 🔻
• Precision • Control Residentity result • Control Residentity result • Open File • Open File <		Process			Data Editor X Parameters	× Help ×
Implemente Implemente </th <th></th> <th></th> <th></th> <th>1404 D D D 📑 Z 💣 🕅</th> <th></th> <th></th>				1404 D D D 📑 Z 💣 🕅		
 • I contraction provide the state of the state o		O FIRCHA			attribute name	dass • 0
 Constrained 	DB				target role	label • 0
 Cloud Repeated y provinces and the provinces deput Cloud Repeated y provinces deput 						
Operation X Document X • booking (8) • operation • Document • operation •	Cloud Repository (deconnected)				set additional roles	D Edit List (0)
Operation X Document X • booking (8) • operation (2000) • operation (2000) X • booking (8) • operation (2000) • operation (2000) X						
Operation X © proteins (%) Production (%) © proteins (%) Proteins (%)			Open File Read ARFF Set Role			
Operation X Postories (%) X * Operation (
Decision x Decision (S) Decision (Free Constant Trees Constant Constant C			on			
Decision x Decision (S) Decision (Free Constant Trees Constant Constant C						
Decision x Decision (S) Decision (Free Constant Free Constant Free Con						
Decision x Decision (S) Decision (Free Constant Trees Constant Constant C						
	Operators ×					
Control to the state of th	Decision X					
Tores (0) Consister Trees Consister Consister Trees Consister Trees Consister Tre	V Hodeling (8)					
Random Tree Random Tr						
Gradeet Boote Trees 0 03 0 Octasion Stamp 0 Decision Tree (Hright Based) 0 Decision Tree (Hright Based) 0 Decision Tree (Hright Based) 1 Random Tree Levrage the Wildom of Crowts to get operator recommendations based on your process design? Levrage the Wildom of Crowts to get operator recommendations based on your process design? Levrage the Wildom of Crowts to get operator recommendations based on your process design? Levrage the Wildom of Crowts to get operator recommendations based on your process design?						
Decision flamp Decision						
Cossistor Tree Bullivary) Cossistor Tree (Interpretational) Cossistor Tre						
Concession Three (Wringeh Based) Concession Three (Wringeh Based) Concession Tree Concession Tree Concession Tree Concession of Concession of Concession of Concession State of your process design? Concession of Concession of Concession State of your process design? Concession of Concession of Concession State of your process design? Concession of Concession of Concession State of your process design? Concession of Concession of Concession State of your process design? Concession of Concession of Concession State of your process design? Concession of Concession of Concession State of your process design? Concession of Concession of Concession State of your process design? Concession of Concession of Concession State of Your Process design? Concession of Concession State of Your Procession State of Your Pro						
Learage the Witsdom of Crowds to per operator recommendations based on your process design?	Decision Tree (Weight-Based)					
Leverage the Wisdom of Crowds to get operator recommendations based on your process design?	💡 Random Tree			~		
			of Crossies to not operator appropriated times instant on your persons during	>*	-	
Get more operations from the Marketplace Volumer Companyone (19.2.001)		Leverage the Wisdom o				
	Get more operators from the Marketplace		Adwate Wisdom of Crowds		· SUBJECT CONTRACTOR (0.2.371)	

<new process*=""> – RapidMiner Studio Trial 9.2.001 @ SCZCWL60 Elle Edit Process View Connections Cloug Settings Extensions H</new>		- 0 ×
Die For Doces Zen Zomennis chañ Semite Stepennis B	Veres: Design Results Turbo Prep Auto Model Find data, convolution data,	🔎 All Studio 🔻
Repository ×	Process Data Editor × Parameters × Help ×	
🕒 Import Data 🛛 = 👻	🕒 Process 150% 🔑 🖉 💆 🖉 💟 🗘 Decision Tree	
> Samples	otherion gain_raso	• 0
B B Local Repository (Internia)	Open File Read ARFF Set Role Decision Tree maximal depth 10	Ø
Temporary Repository (Internal) Cloud Repository (Internal)	n appy pruning	Ø
Coust reputatory (acconnected)	v d v d v d v d v d v d v d v d v d v d	۵
		D
	minimal gain 0.01	Φ
	minimal leaf size 2	Φ
Operator × Decision X Statustical (b) Statustical (c) Statustical (c)		
	Leverage the Wisdom of Crowds to get operator recommendations based on your process design!	
0	Activate Wisdom of Crowds	
Get more operators from the Marketplace	· · · · · · · · · · · · · · · · · · ·	

Figure 56. RapidMiner Studio - Decision Tree operator







ile <u>E</u> dit <u>P</u> roce	ss <u>V</u> iew <u>C</u> onnections Clou <u>d</u> <u>S</u> ettings Extensions <u>H</u> elp	
	Views: Design Results Turbo Prep Au	to Model Find data, operatorsetc 🔎 All Studio 🔻
Result History	🖓 Tree (Decision Tree) 🛛 🗙	Repository ×
Graph	Tree VAR_on_ACC_Z > 0.281 ENERGY_on_GY_Z > 142.688: others {others=98, washing_hand=1} ENERGY on GY Z ≤ 142.688	Import Data Import Data
Description	<pre> MEAN_on_ACC_X > -0.325 MEAN_on_ACC_Z > 0.607: others (others=5, washing_hand=0) MEAN_on_ACC_Z > 0.607 MEAN_on_ACC_Y > 0.261</pre>	Emporary Repository (hhunhia) Cloud Repository (disconnected)
Annotations	<pre> ZeroCross_on_GY_Y > 5 ENERGY_on_GY_Z > 118.187 PeaKTOPeak_on_ACC_Y > 2.642: others (others=2, PeaKTOPeak_on_ACC_Y > 2.642: washing_hand (others=1, PeaKTOPeak_on_GY_Z < 11.178: washing_hand (others=1, PeaKTOPeak_on_GY_V > 1.178: washing_hand (c PeaKTOPeak_on_GY_V > 1.178: others (others=4, v PeaKTOPeak_on_GY_Y < 5: others (others=4, v ZeroCross_on_GY_Y < 5: others (others=2, washing_hand=0) MEAN_on_ACC_Y < 0.261: others (others=3, washing_hand=0) MAX_on_ACC_Y > 2.544 MIN_on_ACC_Y > 2.578: washing_hand (others=0, washing_hand=3) MIN_on_ACC_Z < 0.378: washing_hand (others=2, washing_hand=0)</pre>	

Appendix C Matlab

Decision trees for the Machine Learning Core can be generated with Matlab. Dedicated scripts for Matlab are available at Matlab.

After importing all the scripts in the Matlab workspace, the function "*Generate_DecisionTree()*" should be called, specifying two file names (an *.arff* file containing the features computed by the Machine Learning Core tool in Unico and a *.txt* file which will contain the decision tree generated):

filename_ARFF = 'features.arff';

filename_dectree = 'decision_tree.txt';

Generate_DecisionTree(filename_ARFF, filename_dectree);

More details can be found in the *README.md* file available contained in the *"matlab"* folder of the GitHub repository.

Appendix D Python

Decision trees for the Machine Learning Core can be generated with Python through the the "*scikit*" package. Python scripts are available at Python both as a Jupyter notebook (*.ipynb) and as a common Python script (*.py). More details can be found in the *README.md* file available contained in the "*python*" folder of the GitHub repository.

Appendix E Glossary

This section contains a glossary of terms used in machine learning. Most of the terms have been taken from https://developers.google.com/machine-learning/glossary/.

ARFF	An ARFF (Attribute-Relation File Format) file is an ASCII text file that describes a list of instances sharing a set of attributes. ARFF files were developed by the Machine Learning Project at the Department of Computer Science of The University of Waikato for use with the Weka machine learning software.
Attribute/Feature	An attribute is an aspect of an instance (e.g. temperature, humidity). Attributes are often called features in Machine Learning. A special attribute is the class label that defines the class this instance belongs to (required for supervised learning).
Binary classification	A type of classification task that outputs one of two mutually exclusive classes. For example, a machine learning model that evaluates email messages and outputs either "spam" or "not spam" is a binary classifier.
Class	One of a set of enumerated target values for a label. For example, in a binary classification model that detects spam, the two classes are spam and not spam. In a multi-class classification model that identifies dog breeds, the classes would be poodle, beagle, pug, and so on.
Classification model	A type of machine learning model for distinguishing among two or more discrete classes. For example, a natural language processing classification model could determine whether an input sentence was in French, Spanish, or Italian.
Classification threshold	A scalar-value criterion that is applied to a model's predicted score in order to separate the positive class from the negative class. Used when mapping logistic regression results to binary classification. For example, consider a logistic regression model that determines the probability of a given email message being spam. If the classification threshold is 0.9, then logistic regression values above 0.9 are classified as spam and those below 0.9 are classified as not spam.
Class-imbalanced dataset	A binary classification problem in which the labels for the two classes have significantly different frequencies. For example, a disease dataset in which 0.0001 of examples have positive labels and 0.9999 have negative labels is a class-imbalanced problem, but a football game predictor in which 0.51 of examples label one team winning and 0.49 label the other team winning is not a class-imbalanced problem.
Clipping	A technique for handling outliers. Specifically, reducing feature values that are greater than a set maximum value down to that maximum value. Also, increasing feature values that are less than a specific minimum value up to that minimum value.
Confusion matrix	An NxN table that summarizes how successful the classification model predictions were; that is, the correlation between the label and the model classification. One axis of a confusion matrix is the label that the model predicted, and the other axis is the actual label.
Cross-validation	A mechanism for estimating how well a model will generalize to new data by testing the model against one or more non-overlapping data subsets withheld from the training set.
Data analysis	Obtaining an understanding of data by considering samples, measurement, and visualization. Data analysis can be particularly useful when a dataset is first received, before one builds the first model. It is also crucial in understanding experiments and debugging problems with the system.
Data augmentation	Artificially boosting the range and number of training examples by transforming existing examples to create additional examples. For example, suppose images are one of your features, but your dataset doesn't contain enough image examples for the model to learn useful associations. Ideally, you'd add enough labeled images to your dataset to enable your model to train properly. If that's not possible, data augmentation can rotate, stretch, and reflect each image to produce many variants of the original picture, possibly yielding enough labeled data to enable excellent training.
Data set or dataset	A collection of examples.
Decision boundary	The separator between classes learned by a model in a binary class or multi-class classification problems.
Decision threshold	Synonym for classification threshold.
Decision tree	A model represented as a sequence of branching statements.
Discrete feature	A feature with a finite set of possible values. For example, a feature whose values may only be animal, vegetable, or mineral is a discrete (or categorical) feature. Contrast with continuous feature.

 define the conditional probability of an output given the features and weights. Overloaded term that can mean either of the following: Reducing the amount of information in a feature in order to train a model more efficiently. Training on a disproportionately low percentage of over-represented class examples in order to improve model training on under-represented classes. A model that is trained online in a continuously updating fashion. That is, data is continuously entering the model. One row of a dataset. An example contains one or more features and possibly a label. See also labeled example and unlabeled example. An example in which the model mistakenly predicted the negative class. For example, the model inferred that a particular email message was not spam (the positive class), but that email message actually was spam.
 Training on a disproportionately low percentage of over-represented class examples in order to improve model training on under-represented classes. A model that is trained online in a continuously updating fashion. That is, data is continuously entering the model. One row of a dataset. An example contains one or more features and possibly a label. See also labeled example and unlabeled example. An example in which the model mistakenly predicted the negative class. For example, the model inferred that a particular email message was not spam (the negative class), but that email message actually was spam. An example in which the model mistakenly predicted the positive class. For example, the model inferred
the model. One row of a dataset. An example contains one or more features and possibly a label. See also labeled example and unlabeled example. An example in which the model mistakenly predicted the negative class. For example, the model inferred that a particular email message was not spam (the negative class), but that email message actually was spam. An example in which the model mistakenly predicted the positive class. For example, the model inferred
example and unlabeled example. An example in which the model mistakenly predicted the negative class. For example, the model inferred that a particular email message was not spam (the negative class), but that email message actually was spam. An example in which the model mistakenly predicted the positive class. For example, the model inferred
that a particular email message was not spam (the negative class), but that email message actually was spam. An example in which the model mistakenly predicted the positive class. For example, the model inferred
spam.
The x-axis in an ROC curve. The false positive rate is defined as follows:
False Positive Rate = False Positives / (False Positives + True Negatives)
An input variable used in making predictions.
The process of determining which features might be useful in training a model, and then converting raw data from log files and other sources into said features.
Feature engineering is sometimes called feature extraction.
 Overloaded term having either of the following definitions: Retrieving intermediate feature representations calculated by an unsupervised or pre-trained model for use in another model as input.
Synonym for feature engineering.
The group of features your machine learning model trains on. For example, postal code, property size, and property condition might comprise a simple feature set for a model that predicts housing prices.
Refers to your model's ability to make correct predictions on new, previously unseen data as opposed to the data used to train the model.
The correct answer. Reality. Since reality is often subjective, expert raters typically are the proxy for ground truth.
A quick solution to a problem, which may or may not be the best solution.
Synonym for class-imbalanced dataset.
Data drawn from a distribution that doesn't change, and where each value drawn doesn't depend on values that have been drawn previously. An i.i.d. is the ideal gas of machine learning—a useful mathematical construct but almost never exactly found in the real world. For example, the distribution of visitors to a web page may be i.i.d. over a brief window of time; that is, the distribution doesn't change during that brief window and one person's visit is generally independent of another's visit. However, if you expand that window of time, seasonal differences in the web page's visitors may appear.
In machine learning, often refers to the process of making predictions by applying the trained model to unlabeled examples. In statistics, inference refers to the process of fitting the parameters of a distribution conditioned on some observed data.
Synonym for example.
The degree to which a model's predictions can be readily explained. Deep models are often non- nterpretable; that is, a deep model's different layers can be hard to decipher. By contrast, linear regression models and wide models are typically far more interpretable.
An open source Java implementation of the C4.5 algorithm
In supervised learning, the "answer" or "result" portion of an example. Each example in a labeled datase consists of one or more features and a label. For instance, in a housing dataset, the features might nclude the number of bedrooms, the number of bathrooms, and the age of the house, while the label might be the house's price. In a spam detection dataset, the features might include the subject line, the sender, and the email message itself, while the label would probably be either "spam" or "not spam."
SFTFFATTER A SODEWILL STREET

57

Linear regression	A type of regression model that outputs a continuous value from a linear combination of input features.
Machine learning	A program or system that builds (trains) a predictive model from input data. The system uses the learned model to make useful predictions from new (never-before-seen) data drawn from the same distribution as the one used to train the model. Machine learning also refers to the field of study concerned with these programs or systems.
Majority class	The more common label in a class-imbalanced dataset. For example, given a dataset containing 99% non-spam labels and 1% spam labels, the non-spam labels are the majority class.
Matplotlib	An open-source Python 2D plotting library. matplotlib helps you visualize different aspects of machine learning.
Minority class	The less common label in a class-imbalanced dataset. For example, given a dataset containing 99% non-spam labels and 1% spam labels, the spam labels are the minority class.
ML	Abbreviation for machine learning.
Model training	The process of determining the best model.
Multi-class classification	Classification problems that distinguish among more than two classes. For example, there are approximately 128 species of maple trees, so a model that categorized maple tree species would be multi-class. Conversely, a model that divided emails into only two categories (spam and not spam) would be a binary classification model.
Multinomial classification	Synonym for multi-class classification.
Negative class	In binary classification, one class is termed positive and the other is termed negative. The positive class is the thing we're looking for and the negative class is the other possibility. For example, the negative class in a medical test might be "not tumor." The negative class in an email classifier might be "not spam." See also positive class.
Neural network	A model that, taking inspiration from the brain, is composed of layers (at least one of which is hidden) consisting of simple connected units or neurons followed by nonlinearities.
Node (decision tree)	A "test" on an attribute.
Noise	 Broadly speaking, anything that obscures the signal in a dataset. Noise can be introduced into data in a variety of ways. For example: Human raters make mistakes in labeling. Humans and instruments mis-record or omit feature values.
Normalization	The process of converting an actual range of values into a standard range of values, typically -1 to +1 or 0 to 1. For example, suppose the natural range of a certain feature is 800 to 6,000. Through subtraction and division, you can normalize those values into the range -1 to +1. See also scaling.
Numerical data	Features represented as integers or real-valued numbers.
Outliers	 Values distant from most other values. In machine learning, any of the following are outliers: Weights with high absolute values. Predicted values relatively far away from the actual values. Input data whose values are more than roughly 3 standard deviations from the mean.
	Outliers often cause problems in model training. Clipping is one way of managing outliers.
Overfitting	Creating a model that matches the training data so closely that the model fails to make correct predictions on new data.
Parameter	A variable of a model that the ML system trains on its own.
Performance	 Overloaded term with the following meanings: The traditional meaning within software engineering. Namely: How fast (or efficiently) does this piece of software run? The meaning within ML. Here, performance answers the following question: How correct is this model? That is, how good are the model's predictions?
Positive class	In binary classification, the two possible classes are labeled as positive and negative. The positive outcome is the thing we're testing for. (Admittedly, we're simultaneously testing for both outcomes, but play along.) For example, the positive class in a medical test might be "tumor." The positive class in an email classifier might be "spam."

	Contrast with negative class.
Precision	A metric for classification models. Precision identifies the frequency with which a model was correct when predicting the positive class. That is:
	Precision = True Positives / (True Positives + False Positives)
Prediction	A model's output when provided with an input example.
Pre-trained model	Models or model components that have been already been trained.
	Data used to approximate labels not directly available in a dataset.
Proxy labels	For example, suppose you want "is it raining?" to be a Boolean label for your dataset, but the dataset doesn't contain rain data. If photographs are available, you might establish pictures of people carrying umbrellas as a proxy label for "is it raining"? However, proxy labels may distort results. For example, in some places, it may be more common to carry umbrellas to protect against sun than the rain.
Rater	A human who provides labels in examples. Sometimes called an "annotator."
	A metric for classification models that answers the following question:
Recall	"Out of all the possible positive labels, how many did the model correctly identify?"
Recall	That is:
	Recall = True Positives / (True Positives + False Negatives)
Regression model	A type of model that outputs continuous (typically, floating-point) values. Compare with classification models, which output discrete values, such as "day lily" or "tiger lily."
Reinforcement learning	A machine learning approach to maximize an ultimate reward through feedback (rewards and punishments) after a sequence of actions. For example, the ultimate reward of most games is victory. Reinforcement learning systems can become expert at playing complex games by evaluating sequences of previous game moves that ultimately led to wins and sequences that ultimately led to losses.
Representation	The process of mapping data to useful features.
200.0	ROC = Receiver Operating Characteristic
ROC Curve	A curve of true positive rate vs. false positive rate at different classification thresholds.
Scaling	A commonly used practice in feature engineering to tame a feature's range of values to match the range of other features in the dataset. For example, suppose that you want all floating-point features in the dataset to have a range of 0 to 1. Given a particular feature's range of 0 to 500, you could scale that feature by dividing each value by 500.
	See also normalization.
Scikit-learn	A popular open-source ML platform. See www.scikit-learn.org.
Scoring	The part of a recommendation system that provides a value or ranking for each item produced by the candidate generation phase.
Semi-supervised learning	Training a model on data where some of the training examples have labels but others don't. One technique for semi-supervised learning is to infer labels for the unlabeled examples, and then to train on the inferred labels to create a new model. Semi-supervised learning can be useful if labels are expensive to obtain but unlabeled examples are plentiful.
Sequence model	A model whose inputs have a sequential dependence. For example, predicting the next video watched from a sequence of previously watched videos.
Serving	A synonym for inferring.
Static model	A model that is trained offline.
Stationarity	A property of data in a dataset, in which the data distribution stays constant across one or more dimensions. Most commonly, that dimension is time, meaning that data exhibiting stationarity doesn't change over time. For example, data that exhibits stationarity doesn't change from September to December.
Supervised machine learning	Training a model from input data and its corresponding labels. Supervised machine learning is analogous to a student learning a subject by studying a set of questions and their corresponding answers. After mastering the mapping between questions and answers, the student can then provide answers to new (never-before-seen) questions on the same topic.
	Compare with unsupervised machine learning.
Target	Synonym for label.

Training	The process of determining the ideal parameters comprising a model.
Training set	The subset of the dataset used to train a model.
Training set	Contrast with validation set and test set.
True negative (TN)	An example in which the model correctly predicted the negative class. For example, the model inferred that a particular email message was not spam, and that email message really was not spam.
True positive (TP)	An example in which the model correctly predicted the positive class. For example, the model inferred that a particular email message was spam, and that email message really was spam.
	Synonym for recall. That is:
True positive rate (TPR)	True Positive Rate = True Positives / (True Positives + False Negatives)
(True positive rate is the y-axis in an ROC curve.
	Producing a model with poor predictive ability because the model hasn't captured the complexity of the training data. Many problems can cause underfitting, including:
Underfitting	Training on the wrong set of features.
Onderniting	Training for too few epochs or at too low a learning rate.
	Training with too high a regularization rate.
	Providing too few hidden layers in a deep neural network.
Unlabeled example	An example that contains features but no label. Unlabeled examples are the input to inference. In semi-supervised and unsupervised learning, unlabeled examples are used during training.
	Training a model to find patterns in a dataset, typically an unlabeled dataset.
Unsupervised machine learning	The most common use of unsupervised machine learning is to cluster data into groups of similar examples. For example, an unsupervised machine learning algorithm can cluster songs together based on various properties of the music. The resulting clusters can become an input to other machine learning algorithms (for example, to a music recommendation service). Clustering can be helpful in domains where true labels are hard to obtain. For example, in domains such as anti-abuse and fraud, clusters can help humans better understand the data.
	Another example of unsupervised machine learning is principal component analysis (PCA). For example, applying PCA on a dataset containing the contents of millions of shopping carts might reveal that shopping carts containing lemons frequently also contain antacids.
	Compare with supervised machine learning.
Validation	A process used, as part of training, to evaluate the quality of a machine learning model using the validation set. Because the validation set is disjoint from the training set, validation helps ensure that the model's performance generalizes beyond the training set.
	Contrast with test set.
	A subset of the dataset—disjoint from the training set—used in validation.
Validation set	Contrast with training set and test set.
Weka	A collection of machine learning algorithms for data mining tasks. It contains tools for data preparation, classification, regression, clustering, association rules mining, and visualization.

Revision history

Date	Version	Changes
03-Oct-2019	1	Initial release
		Updated Figure 1. Machine Learning Core supervised approach
		Updated Section 1.1 Inputs
	2	Updated Section 1.2 Filters
		Updated Section 1.3 Features
		Updated Section 1.3.13 Selection of features
		Added Section 1.3.13.1 Histogram of a single feature (1D plot)
05-Aug-2020		Added Section 1.3.13.2 Visualization of two features (2D plot)
		Added Section 1.3.13.3 Ranking of features
		Updated Section 1.5 Meta-classifier
		Updated Section 2.1 Unico GUI
		Updated Section 2.2 Decision tree generation
		Updated Section 2.3 Configuration procedure
		Updated Section Appendix A Weka
	3	Updated Section 1.1 Inputs
13-Apr-2021		Updated maximum number of features in Section 1.3 Features
		Updated Table 6. Decision tree interrupts

Table 11. Document revision history

Contents

1	Мас	hine Lea	arning Core in the ISM330DHCX	.2
	1.1	Inputs		. 4
	1.2	Filters		. 6
		1.2.1	Filter coefficients	. 7
	1.3	Feature	es	. 9
		1.3.1	Mean	10
		1.3.2	Variance	10
		1.3.3	Energy	10
		1.3.4	Peak-to-peak	10
		1.3.5	Zero-crossing	11
		1.3.6	Positive zero-crossing	11
		1.3.7	Negative zero-crossing	12
		1.3.8	Peak detector	12
		1.3.9	Positive peak detector	13
		1.3.10	Negative peak detector	13
		1.3.11	Minimum	14
		1.3.12	Maximum	14
		1.3.13	Selection of features	15
	1.4	Decisio	on tree	19
		1.4.1	Decision tree limitations in the ISM330DHCX	20
	1.5	Meta-c	lassifier	21
		1.5.1	Meta-classifier limitations in the ISM330DHCX	21
	1.6	Finite S	State Machine interface	21
2	Мас	hine Lea	arning Core tools	22
	2.1	Unico (GUI	22
	2.2	Decisio	on tree generation	25
	2.3		uration procedure	
3		Ŭ	e examples	
-	3.1		on monitoring	
	3.2		intensity	
	V.2	motion	intenenty	50



3.3	6	D position recognition	35
3.4	Sı	ummary of examples	37
Appendix	Α	Weka	38
Appendix	B	RapidMiner	45
Appendix	C	Matlab	49
Appendix	D	Python	50
Appendix	Ε	Glossary	51
Revision	hist	tory	56
Contents			57
List of tab	oles	;	59
List of fig	ure	S	60

List of tables

Table 1.	Machine Learning Core output data rates	. 2
Table 2.	Filter coefficients	. 6
	Examples of filter coefficients	
Table 4.	Features	. 9
Table 5.	Decision tree results.	20
Table 6.	Decision tree interrupts.	20
Table 7.	Decision tree limitations in the ISM330DHCX.	20
	Meta-classifier example	
Table 9.	Meta-classifier limitations in the ISM330DHCX.	21
	Summary of examples	
Table 11.	Document revision history	56

List of figures

Figure 1.	Machine Learning Core supervised approach	1
Figure 2.	Machine Learning Core in the ISM330DHCX	2
Figure 3.	Machine Learning Core blocks	3
Figure 4.	MLC inputs (accelerometer)	4
Figure 5.	MLC inputs (gyroscope).	4
Figure 6.	Filter basic element.	6
Figure 7.	Peak-to-peak	0
Figure 8.	Zero-crossing	1
Figure 9.	Positive zero-crossing	
Figure 10.	Negative zero-crossing	2
Figure 11.	Peak detector	2
Figure 12.	Positive peak detector	3
Figure 13.	Negative peak detector	3
Figure 14.	Minimum	
Figure 15.	Maximum	4
Figure 16.	Distribution of single feature for three different classes	6
Figure 17.	Visualization of two features and two classes	
Figure 18.	Ranking from automated output tool	
Figure 19.	Decision tree node	
Figure 20.	Unico GUI	
Figure 21.	Machine Learning Core tool - Data Patterns	
Figure 22.	Machine Learning Core tool - Configuration	
Figure 23.	Decision tree generation in Unico	
Figure 24.	Weka preprocess	
Figure 25.	Weka classify	
Figure 26.	Decision tree format	
Figure 27.	Configuration procedure	
Figure 28.	Assigning a result to a data pattern	
Figure 29.	Configuration of Machine Learning Core	
Figure 30.	Configuration of filters	
Figure 31.	Configuration of features	
Figure 32.	ARFF generation	
Figure 33.	ARFF file	
Figure 34.	Configuration of results and decision tree.	
Figure 35.	Meta classifier and device configuration	
Figure 36.	Creation of configuration file	
Figure 37.	Unico load configuration	
Figure 38.	Unico Data window	
Figure 39.	Unico - Machine Learning Core source registers	
Figure 40.	Vibration monitoring decision tree	
Figure 41.	Motion intensity decision tree	
Figure 42.	6D positions	
Figure 43.	6D decision tree	
Figure 44.	ARFF example	
Figure 45.	Weka GUI Chooser	
Figure 46.	Weka Explorer	
Figure 47.	Weka Explorer - Attributes	
Figure 48.	Weka Classify	
Figure 49.	Weka Classify J48	
Figure 50.	Weka J48 classifier parameters	
Figure 51.	Correctly classified instances	
Figure 52.	Confusion matrix	

57

Figure 53.	RapidMiner Studio - Open File	45
Figure 54.	RapidMiner Studio - Read ARFF.	46
Figure 55.	RapidMiner Studio - Set Role	46
Figure 56.	RapidMiner Studio - Decision Tree operator	47
Figure 57.	RapidMiner Studio - Decision Tree graph	47
Figure 58.	RapidMiner Studio - Decision Tree description	48

IMPORTANT NOTICE - PLEASE READ CAREFULLY

STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST's terms and conditions of sale in place at the time of order acknowledgement.

Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of Purchasers' products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

ST and the ST logo are trademarks of ST. For additional information about ST trademarks, please refer to www.st.com/trademarks. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2021 STMicroelectronics - All rights reserved