Performance analysis of the *LAMDA* fuzzy algorithm improvements in different case studies

Luis A. Morales[a], *Frank A. Ruiz*[b], *Christian D. Moreno*[c], *Jose Aguilar*[d,e,f,*]

[a] Departamento de Automatización y Control Industrial. Escuela Politécnica Nacional, Quito, Ecuador
[b] Department of Mechanical Engineering. Institución Universitaria Pascual Bravo, Medellín, Colombia
[c] Department of Electronic Engineering. Universidad de Antioquia, Medellín, Colombia
[d] Departamento de Computación. CEMISID, Universidad de Los Andes, Mérida, Venezuela
[e] CIDITIC, Universidad EAFIT, Medellín, Colombia
[f] IMDEA Networks Institute, Legane´s, Madrid, Spain
[*] Corresponding author

Abstract: Learning Algorithm for Multivariable Data Analysis *(LAMDA)* is a fuzzy approach, which has been used in clustering and classification processes. Recently, extensions have been proposed of *LAMDA*, to improve its performance in classification tasks. The first one is called *LAMDA-FAR*, which proposes a new criterion to validate functional states after recognition, based on the minimum and maximum calculated distances between the two membership degrees with the highest values. The second extension is called *LAMDA-HAD*, which proposes two strategies to improve *LAMDA* performance. The first strategy calculates an adaptive Global Adequacy Degree (*GAD*) of the Non-Informative Class (*NIC*) to each class to prevent that correctly classified individuals will be assigned to the *NIC* class. The second strategy calculates the similarity among the *GAD* of an individual and all ones of each class, to make a more reliable assignment. This article analyzes the performance of these techniques for different classification problems. The goal is to define the application context for each one. Each case study was defined by a set of data in an operational context, which must be used by the classification techniques to obtain accurate results. *LAMDA-HAD* was better with unbalanced classes, while *LAMDA-FAR* was excellent for discovering new classes. Both algorithms worked well for different levels of noise (which can represent faults in the sensors), a factor important in diagnostic tasks. The aim of this paper is to determine the correct utilization profile of each *LAMDA* technique adjusted to the properties of the problems under study.

Keywords: classification problems, performance analysis, *LAMDA*.

1. Introduction

Classification problems are present in a lot of engineering processes. The main goal of a classification task is to assign objects to predefined categories. The classification task model can be used in different ways, most commonly as a descriptive model to explain the distinctions between objects in different classes, but also as a predictive model to forecast classes of unknown data [1]–[5]. Sometimes, the classification process may be challenging due to external disturbances, inaccuracy in measurement equipment, incipient faults not detected in the system, or simply, inherent classification techniques variances.

LAMDA is a fuzzy clustering algorithm proposed by (Aguilar-Martín and López De Mantaras, 1982 [5]), which uses probability density functions to compute the membership of an individual *i* to a class *k* considering the maximum value of a numerical array of membership degrees or Global Adequacy degrees (*GAD*), which varies between 0 and 1, where 1 represents the absolute membership of a data to a class and 0 represents non-membership to this.

Among some notable differences of *LAMDA* algorithm, compared to other algorithms [6], the following are related:

- This algorithm does not need to have data of all the possible classes of the system (unknown states) to generate new functional states even after its training stage.
- This algorithm can work in a supervised (scenario evaluated in this work) and unsupervised learning processes including both qualitative and quantitative data.
- The data processing time invested in the training/learning stage of the algorithm is relatively short because this is not an iterative process.
- The equations and internal structure of the algorithm are known, facilitating the modification of the classifier's characteristic parameters.
- Complex mathematical routines are not used to determine the membership of an individual to a class, which facilitates its implementation in different types of processes.
- Allowing it to be used in descriptive and classification tasks.

1.1. Related works

In the scientific literature, can be found abundant works related to data classification and clustering methods based on the functional states detection of different systems.
To deal with a lot of classification, clustering, or prediction problems, a general combination of neural networks and fuzzy systems have been proposed to solve them, Santos-Junior et al. developed a new method based on a Fuzzy ARTMAP neural network with continuous training which can be trained via classification or prediction methods [7]. Ramirez-Bautista et al. compared the obtained classification results of human plantar foot alterations employing Fuzzy Cognitive Maps (FCM) trained by Genetic Algorithm (GA) against a Multi-Layer Perceptron Neural Network (MLPNN) to detect gait disorders in a person. The tests were validated by a specialized physician of the Piédica diagnostic center, obtaining better performance the fuzzy method [8]. In the field of medicine, and especially in the diagnosis of pathologies through the analysis and treatment of biomedical images, computational intelligence methods have an important role, Das A et al. designed a classifier with a fuzzy decision method for biomedical images. Four heterogeneous base classifiers based on Neural Networks and a fuzzy min-max model were considered. Accuracy, precision, recall, specificity, sensitivity, and F1-score parameters were evaluated for each data set [9]. In the field of biology, considering sound databases of marine mammals, recognition and classification processes were carried out using the Fuzzy-ChOA algorithm (fuzzy-Chimp Optimization algorithm). This algorithm is a combination of ChOA as an artificial neural networks trainer (ANN) and fuzzy logic [10].

Some years ago, the *LAMDA* fuzzy algorithm has been employed as a helpful tool in medical and biological applications to detect anuran (amphibians) species through the identification of its calls. The Implemented methodology showed an excellent potential of recognition and high classification percentages and noise immunity [11].

In engineering processes, specifically those monitored by Artificial Intelligence (*AI*) systems, is important to identify accurately the functional states (classes), in this context *LAMDA* is very useful. Among some clustering and data classification works that have used the *LAMDA* algorithm in engineering processes the following stand out [12]–[14]. For example, *LAMDA* has been used in Fault Detection and Isolation (*FDI*) case studies, to detect operating states and avoid dangerous operating conditions [2]. Also, the algorithm was used to detect the functional states of a process in real time, identifying its normal and abnormal states [15], [16]. Another application has been to determine the fault location considering the information obtained from the signals of the system [12].

Other additional works related to *LAMDA* fuzzy algorithm are mentioned following. Morales et al. proposed the *LAMDA* algorithm to compute the sliding mode control continuous and discontinuous actions to obtain a chattering-free controller to apply it to a class of *SISO* systems. The experiments were compared with other control techniques, exhibiting good results and enhancing the performance of tanks control [17]. Additionally, some extensions have been proposed to improve the performance of *LAMDA*, a modification of the original algorithm proposed by the same authors named *LAMDA-RD* where an automatic merge technique to update the cluster partition was performed to improve the quality of the clusters, that proposal was applied to several benchmarks and was compared with different clustering algorithms and measured metrics [18]. In the field of artificial vision and image processing, a variation of the *LAMDA* algorithm (*T-LAMDA*) was used to perform color image segmentation procedures (*RGB* values), incorporating spatial information organized in a class tree which improved the accuracy method and increased the noise immunity [19]. *LAMDA* too was used to perform the trajectory tracking control of a robot. Different dynamic controllers based on this fuzzy algorithm were designed such as *LAMDA-PID, LAMDA-Sliding-Mode Control (LSMC),* and Adaptive *LAMDA* controllers. To perform a comparative analysis between them and the conventional *PID, SMC,* and Fuzzy-*PID* controllers, different trajectories both qualitatively and quantitatively results were evaluated [20]. Recently a soft computing algorithm for modeling and control of nonlinear complex systems applying online learning based on *LAMDA* was used to enhance the accuracy and performance of a controller [21]. In that work, the structure and learning methods of the original algorithm were modified, developing an adaptive approach that evaluates the closed-loop system [20]. These controllers have been tested in systems with different characteristics, such as non-linearities, systems with dead time, SISO and MIMO systems, etc, in which their operation has been validated and their performance analyzed [22]. Botia et al. too proposed a structural modification of the *LAMDA* algorithm adding to the model two functions: intuitionistic global adequacy degree (*IGAD*) and global typicality degree (*GTD*), later mixing both functions, they formed a new function called typicality and intuitionistic global adequacy degree (*TIGAD*). That proposal was applied in three study cases improving the data clustering process [23].

In the field of prediction industrial complex processes, Isaza et al. proposed an approach based on *LAMDA* and *Markov's* theory to classify and estimate functional states respectively, that work was tested on a boiler subsystem of a steam generator and a power transmission system [24]. In the automotive sector, *LAMDA* fuzzy algorithm was used in supervisory

learning mode to diagnose the current faults in a vehicle. The algorithm identified different functional states such as normal driving behavior, aggressive driving, or mechanical failure. That approach achieves 92.52% of correct identification with a low computational cost [24]. It has been shown that the limitations of the algorithm are related to datasets that have descriptors that do not adequately characterize the classes [25], Therefore, it is appropriate to carry out a previous stage of data science to know the most representative descriptors that provide relevant information to the model that the algorithm will generate. The main foundation of LAMDA is fuzzy and this feature is used to create new classes not considered in the training, however, sometimes this functionality creates classes excessively, which has been a problem that has been tried to solve. by researchers in order to improve the performance of the algorithm as in [16], [25].
From the review of related works, it is evident that there is a large amount of information in this regard, in which new modifications to the algorithm are presented for use in the field of classification, clustering and even control. In the context of supervised learning, there is no formal research that allows knowing a priori which of the algorithms is the most appropriate when working with data sets of different characteristics and that allows an adequate selection of the different LAMDA methodologies (extensions).

The motivation of this work arises from this lack of information, so it is proposed to carry out a performance analysis of the improvements of the LAMDA fuzzy algorithm in different case studies in which several modifications are made to evaluate the cases in which each one allows for better results. Specially, we are interested in two recent improvements in classification tasks. One is *LAMDA-FAR* [20], which takes as basic information the measure of two distances computed among the two highest *GAD* in each class. Using these distances, it is evaluated if the *GAD* of an individual is within those ranges to assign it to a class; otherwise, it is sent to the *NIC* class. The other one is *LAMDA-HAD* [25], which proposes two strategies to improve the efficiency of the original algorithm. The first strategy defines an adaptable *GAD* of the *NIC* to each class to avoid that correctly classified individuals will be assigned to the *NIC* class; and the second strategy calculates a similarity measure between the GAD of an individual and all the others of each class, to make a more reliable assignment. In this paper, we are going to test the performance of these recent extensions of the *LAMDA* algorithm, in different classification problems, to determine the utilization profile of each one. The utilization profile of a technique is defined based on the characteristics of the descriptors, classes, and data, among other things, of the classification problems where it gives the best performances.
This paper is organized as follows: In the next section, we introduce *LAMDA* and in section 3 its recent extensions. Section 4 presents the three case studies. Section 5 shows the results and defines the utilization profile of each technique according to the results obtained. Finally, Section 6 presents our conclusions.

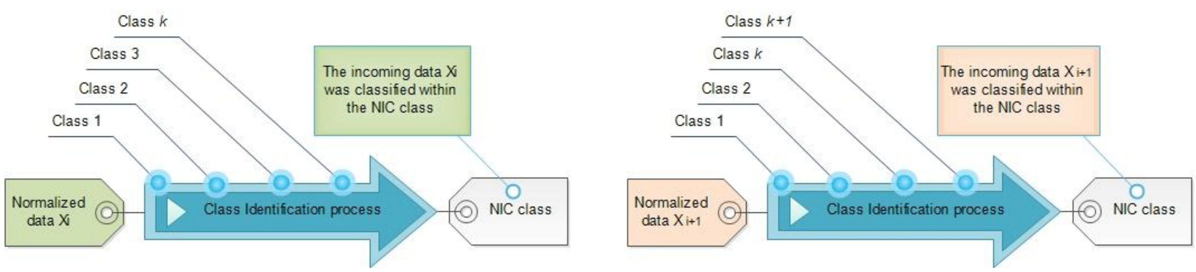
2. Learning Algorithm for Multivariable Data Analysis (LAMDA)

LAMDA is a fuzzy algorithm that combines the concepts of neural networks and fuzzy clustering [5]. The algorithm is based on the calculation of the GADs (see equation 6) or membership degrees matrix, which in turn depend on the Marginal Adequacy Degrees matrix

(MAD) (see equation 2), calculated using probability density functions (binomial function, Gaussian function, Poisson function, etc.), to find the functional state or class to which an individual X belongs.

In this paper, the *k-th* class is denoted by the lowercase and italic letter k, with $1 < k < m$, where m is the total number of classes in the system, and the *d-th* descriptor or attribute is denoted by the lowercase letter d, with $1 < d < D$, where D is the total number of descriptors.

One of the main advantages of *LAMDA* over other fuzzy classification algorithms is that this algorithm can create new classes even after its training stage. When data does not conform to the characteristics of the pre-established classes, *LAMDA* has a class called the Non-Informative Class (*NIC)*, to which this data will be assigned. If the system where the algorithm is applied is being trained in a supervised manner, then all new incoming data X, with $X = [x_1, x_2, ..., x_d, ..., x_D]$ that do not meet the selection criteria of the original classes will be assigned to the *NIC* class (see Figure 1.a). In the same way, when the training is performed in an unsupervised mode, the characteristics of the algorithm would allow the construction of new classes to which these individuals would be assigned, distinguishing between themselves according to their characteristics (see Figure 1.b).



a) b)

Figure 1. Creation of new classes when incoming data is classified within the NIC class

In LAMDA, it is necessary to work with normalized data in the algorithm, with the purpose that all the descriptors are in the same subspace $[0,1]$. For this operation, the maximum $x_{max,d}$ and minimum $x_{min,d}$ values of each descriptor must be considered, this normalization is shown in equation 1.

$$\bar{x}_d = \frac{x_d - x_{min,d}}{x_{max,d} - x_{min,d}} \tag{1}$$

The *MAD* is a parameter used to measure the similarity of a descriptor with the same descriptor in each class k. To compute *MADs* are used probability density functions like the binomial function:

$$MAD_{[\bar{X},K,D]}(\bar{x}_d, \rho_{k,d}) = \rho_{k,d}{}^{\bar{x}_d}(1-\rho_{k,d})^{(1-\bar{x}_d)} \tag{2}$$

where $\rho_{k,d}$ is the average value for the class k, calculated according to equation 3, in the case of supervised training:

$$\rho_{[K,D]}(\bar{x}_d, T_k) = \frac{1}{T_k}\sum_{t=1}^{t=T_k} \bar{x}_d(t) \tag{3}$$

where T_k is the number of data belonging to class k.

LAMDA algorithm uses one of two types of connectors to obtain the GAD from the MAD, *Product-Probabilistic sum (equation 4)* or *Minimum-Maximum (equation 5).*

$$\gamma(a,b) = ab; \beta(a,b) = a + b - ab \tag{4}$$

$$\gamma(a,b) = \min(a,b); \beta(a,b) = \max(a,b) \tag{5}$$

where a and b are fuzzy sets (in LAMDA are the MADs of class k), γ is the t-norm and β is the s-norm of the fuzzy connectors.

GAD function can be obtained according to equation 6. The degree of exigency to classify the data depends upon the parameter α, with $0 \leq \alpha \leq 1$. When α increases, then the classification turns out to be stricter, and when α decreases, then the classification is more permissive.

$$GAD(\bar{X}, K_k) = \alpha\gamma[MAD_1(\bar{x}_1, K_k), \dots, MAD_d(\bar{x}_d, K_k), \dots, MAD_D(\bar{x}_D, K_k)] \\ +(1-\alpha)\beta[MAD_1(\bar{x}_1, K_k), \dots, MAD_d(\bar{x}_d, K_k), \dots, MAD_D(\bar{x}_D, K_k)] \tag{6}$$

Finally, the normalized individual $\bar{X}$ is assigned to a class where the maximum *GAD* value is reached. Figure 2 shows the original *LAMDA* classification structure.

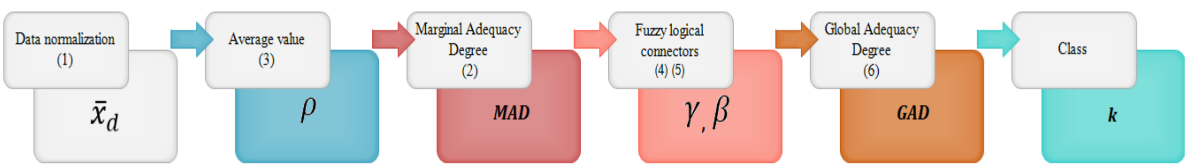


Figure 2. Original *LAMDA* classification structure

3. Improvements to the *LAMDA* algorithm

3.1 LAMDA-FAR algorithm

The *LAMDA-FAR (LAMDA-Functional States After Recognition)* algorithm in its training stage, calculates the $d_{max}(k)$ and $d_{min}(k)$ distances (see Figure 3) between the two membership degrees with the highest GAD values for each incoming data X and for each class k.

"The $d_{max}(k)$ distance (equation 7) is described as the difference between the maximum value of the uppermost *GAD* (GAD_{top}) which are the highest membership degrees values for each class k, and the minimum value of the *GAD* immediately below (GAD_{low})" [16].

$$d_{max}(k) = \max\left(GAD_{top}(k)\right) - \min\left(GAD_{low}(k)\right) \tag{7}$$

"The $d_{min}(k)$ distance (Equation 8) represents the difference among the minimum value of uppermost *GAD* (GAD_{top}), and the maximum value of *GAD* immediately below (GAD_{low})" [20].

$$d_{min}(k) = \min\left(GAD_{top}(k)\right) - \max(GAD_{low}(k)) \tag{8}$$

Once $d_{max}(k)$ *and* $d_{min}(k)$ distances for each class *k* are computed, the differences between the two higher membership degrees are evaluated for each incoming data $\bar{X}$. In other words, when the membership degrees of an individual X to each class *k* (*GAD* (*k*)) are found, then they are sorted from highest to lowest, and then the difference of the first two values is computed (the two membership degrees of higher value). If the distances obtained are lower than $d_{min}(k)$ or higher than $d_{max}(k)$, then data X is classified into the NIC class. In order to carry out the previous procedure, it is clarified that the membership degrees associated with the NIC class will not be considered. If the distances computed from the data X are within the thresholds, then this will be assigned into the preexisting class *k* defined by the original *LAMDA* algorithm.

To understand the algorithmic way of how *LAMDA-FAR* works, consider the following steps:

Step 1: sort from highest to lowest the *GAD's* for each individual *X*.

$$sort\ ([GAD(1), GAD(2), \dots GAD(k), \dots\ GAD(m)])$$

Step 2: the two highest values are selected, the difference between them is computed and the distance is obtained

$$distance = GAD_{1-max} - GAD_{2-max}$$

GAD_{1-max} represents the highest membership degrees value and GAD_{2-max} represents the second highest value.

Step 3: the calculated distance is compared with the distances $d_{max}(k)$ and $d_{min}(k)$ obtained in the training stage.

$$if\left(distance < d_{min}(k)\right)\ or\ if\left(distance > d_{max}(k)\right)$$

$$then\ (k = NIC)$$

$$else\ \left(k = k_{original}\right)$$

$k_{original}$ is the class found by the original *LAMDA* algorithm.

Figure 3, shows an example of the maximum ($d_{max}(k)$) and minimum ($d_{min}(k)$) distances obtained for class $k\ = 2$ between the two membership degrees with the higher GAD values for each incoming data applied by the *LAMDA-FAR* algorithm in the training stage.

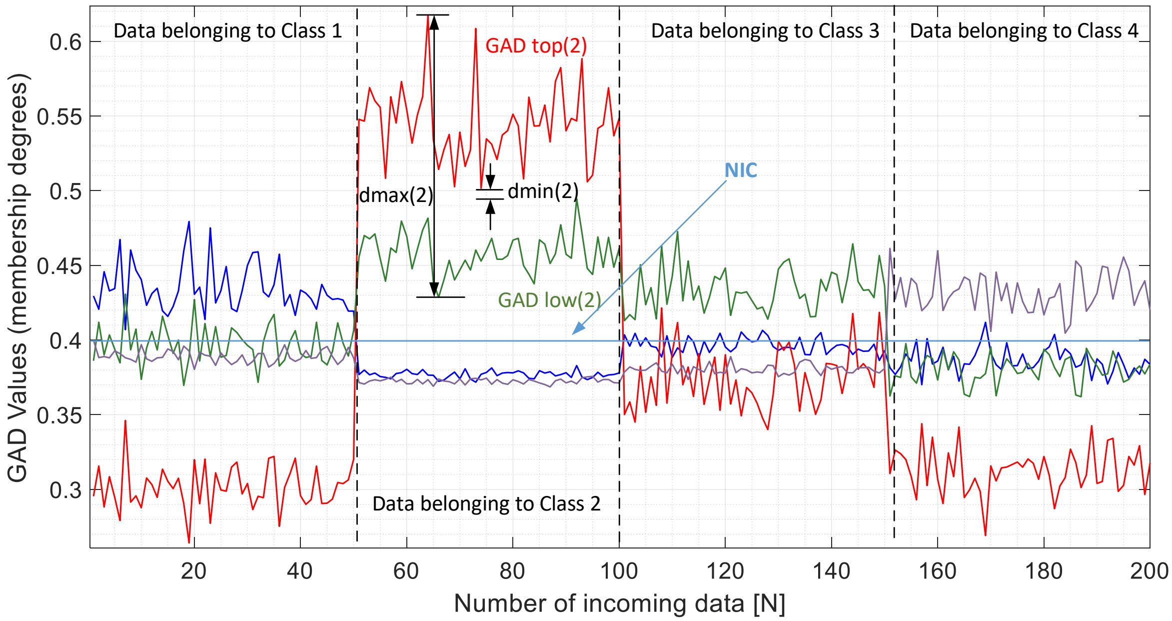


Figure 3. Example of the maximum ($d_{max}(k)$) and minimum ($d_{min}(k)$) distances for class $k\ = 2$ in a training data base with 4 classes with 50 samples each one.

If the original *LAMDA* algorithm recognizes that a new individual belongs to the *NIC*, then the *LAMDA-FAR* criterion does not apply. However, if the class is any other, then the classification will be validated. *LAMDA-FAR* criterion is used to validate the classification process of the original *LAMDA* algorithm, establishing each class or functional state by using a membership degrees analysis.

3.2 LAMDA-HAD algorithm

LAMDA-HAD solves some problems presented in the original algorithm. In certain applications, the original algorithm tends to incorrectly send well classified objects to the *NIC*. On the other hand, depending on the similarity of the descriptors of an object between two classes, it could perform an incorrect classification process (misclassification) [25]. To solve these drawbacks, *LAMDA-HAD* proposes two strategies:

- To compute as many *NICs* as the number of classes. The *NICs* are obtained using the intrinsic features of each class, to prevent sending well-classified individuals to the *NIC*.
- To calculate the Higher Adequacy Degree (*HAD*), a measure of the similarity degree of the *GAD* of an individual related with the average of the *GADs* of the classes using probabilistic functions. The *HAD* allows a more accurate object assignment to the class that really corresponds [26].

The *LAMDA-HAD* algorithm is similar to *LAMDA* in the procedure shown from equations (1)-(6). Starting from this, *LAMDA-HAD* requires the computation of the average values of the *GADs* of the class p for each individual in each class k ($MGAD_{k,p}$). These parameters are obtained as:

$$MGAD_{[k,p]}\left(GAD_{t,p}, T_k\right) = \frac{1}{T_k}\sum_{t=1}^{t=T_k} GAD_{t,p} \tag{9}$$

where $p = \{1, .., m\}$ are the pre-existing classes, therefore $GAD_{t,p}$ is the GAD of the individual t for the class p, in the class k.

Figure 4, shows the same example of Section 3.1 where are presented the location of some *GADs* (colored lines) and $MGAD_{k,p}$ (dashed lines), in a training database with 4 classes with 50 samples each one.

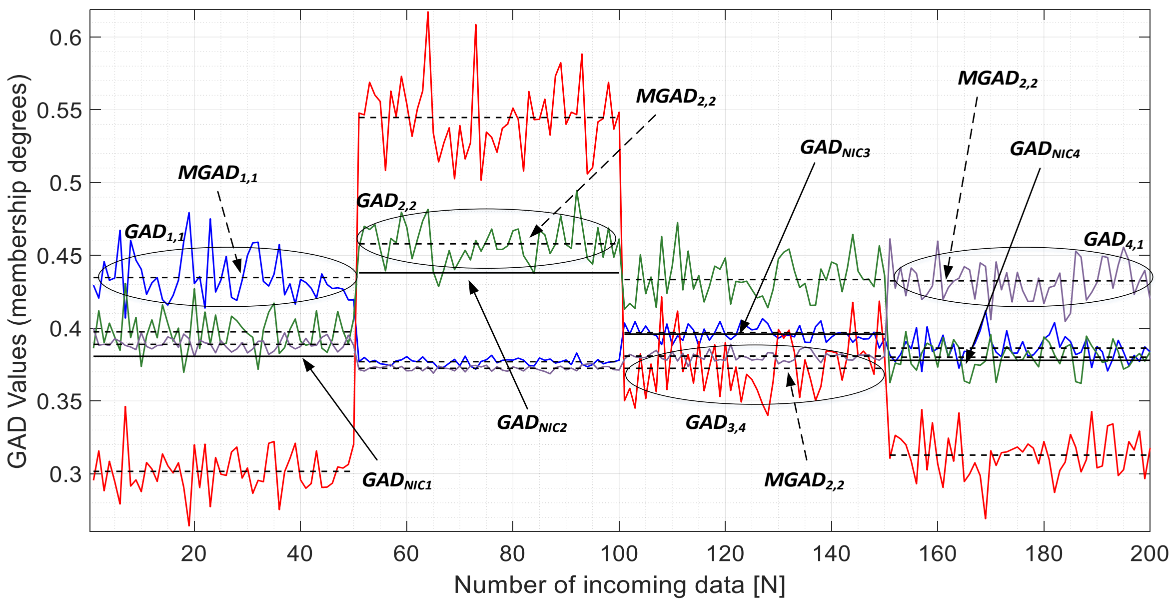


Figure 4. Example of *MGAD* obtained for each *GAD* in the training database with 4 classes and 50 samples each one.

With the *MGAD*, the next parameters are computed:

Adaptable GAD_{NIC}*:* The *GAD* of the *NIC* of each class k is calculated by equation 10, and it corresponds to the mean value of all *MGADs* in each class k.

$$GAD_{NIC_k}(MGAD_{k,p}, m) = \frac{1}{m}\sum_{p=1}^{p=m} MGAD_{k,p} \tag{10}$$

This is the new threshold established to define whether or not an individual should be assigned to the class k. As mentioned before, in the original proposal a single general NIC is calculated, while in *LAMDA-HAD*, the *NIC* is adapted to each class. In the example of Figure 3, the GAD_{NIC} are the solid black lines in each class.

Adequacy Degree of the GAD (AD_{GAD})*:* this parameter computes the adequacy degrees of the *GAD* of the object with respect to the $MGAD_{k,p}$, it is obtained evaluating $\bar{X}$ in each class as:

$$AD_{GAD_{[\bar{X},k,p]}}(MGAD_{k,p}, GAD_{\bar{X},p}) = MGAD_{k,p}{}^{GAD_{\bar{X},p}}(1 - MGAD_{k,p})^{(1-GAD_{\bar{X},p})} \tag{11}$$

Higher Adequacy Degree (HAD): this parameter is computed adding the AD_{GAD} for each class:

$$HAD_{[\bar{X},k]}(AD_{GAD_{\bar{X},k,p}}) = \sum_{p=1}^{p=m} AD_{GAD_{\bar{X},k,p}} \tag{12}$$

Using the probability function presented in equation (11), the *HAD* computes with greater certainty the membership degree of the individual $\bar{X}$ based on its $GADs$, which strengthens the assignment process, since the similarity analysis, in this case, is performed concerning the *GADs* of all the individuals in each class. As a result, *LAMDA-HAD* improves the performance of the classification in unbalanced class scenario.

The maximum *HAD*, (Equation 13) allows establishing the index (label) E_I of the class to which the object has a greater probability of belonging.

$$E_I(HAD_{\bar{X},k}) = \arg\max(HAD_{\bar{X},1}, \dots, HAD_{\bar{X},k}, \dots, HAD_{\bar{X},m}) \tag{13}$$

Finally, it is necessary to verify if the maximum *GAD* of the object in the estimated class E_I is greater than the corresponding GAD_{NIC} (equation 14) in the estimated class. If this condition is met, then the object is assigned to the class E_I, otherwise is assigned to the *NIC* class.

$$index\ (GAD_{E_I,\bar{X}}, GAD_{NIC_{E_I}}) = \arg\max(GAD_{E_I,\bar{X}}, GAD_{NIC_{E_I}}) \tag{14}$$

4. Case studies

In engineering processes, it is important to identify accurately their functional states (classes), to diagnose typical and atypical states, monitor the normal operation of processes, detect fault to take corrective actions, among others. The case studies considered in this section are real applications. The goal is to identify the correct functional states of the systems under different conditions in the datasets. These are: balanced and unbalanced datasets, clean and noisy datasets, and datasets with incomplete data to detect states not considered in the training, which will allow a rigorous analysis of the tested algorithms. The used datasets are of Wells based on the Artificial Gas Lift, Diesel Engines and of Driver States [1], [3], [4], [16], [27], [28].

4.1 Wells based on the Artificial Gas Lift (AGL) method

4.1.1 Theoretical framework

The flow to the well depends on the pressure exerted downhole in the well (*Pwf*), and the static pressure exerted on the tank (*Pws*). In the well, the fluids rise through the production pipe-line overcoming the friction of the internal walls and gravity. At the wellhead, the resulting pressure corresponds to *Pwh*. The production capacity of the well corresponds to the balance between the energy input capacity of the reservoir and the energy requirement of the installation to bring the fluids outside. [27].

Gas lift is a method used to extract oil in wells that have low pressure in the reservoir. For this it is necessary to reduce the hydrostatic pressure in the pipe [3], [27]. The gas is drawn into the piping and combines with the fluid in the reservoir (see Figure 5). The gas decreases the density of the fluid in the pipe, which decreases *Pwf*, which increases the production of the reservoir. The flow dynamics in a gas well can be explained as: *i*) the gas from the casing flows into the pipe. When gas enters the pipeline, the pressure in the pipeline decreases which speeds up gas entry; *ii*) the gas pushes the liquid out of the pipeline; *iii*) the liquid in the pipe creates a blockage in the injection hole. Then the pipe is filled with liquid and the annular space with gas, *iv*) a new cycle will start when the pressure at the injection port exceeds the pressure at the pipe side.

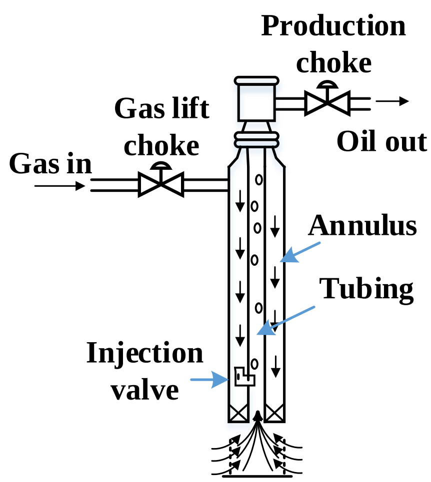


Figure 5. The Artificial Gas Lift (Image taken from [6])

The operation of the AGL well is presented in Figure 6. The graph shows that by increasing the gas injection rate, production also increases until it reaches its maximum; however, further increases in gas injection would cause a decrease in production [3], [27]–[29].

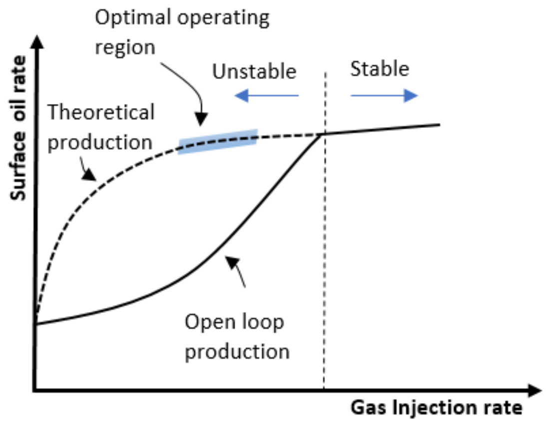


Figure 6. Artificial Gas Lift well behavior's model (Image taken from [28])

Application of the *AGL* method in the field requires instrumentation and control monitoring [28], [29], for measuring and controlling the variables presented in Figure 7. These variables are Differential Pressure of the Gas Injected (*GLDP*), Pressure of the Tubing of Production (*THP*), Pressure of the Gas Injected (*GLP*), Pressure of the Casing (*CHP*), and the Pressure of the Line of Production (*PLP*), Flow of Lift (*FGL)*, and the Rate of Production (Qprod).

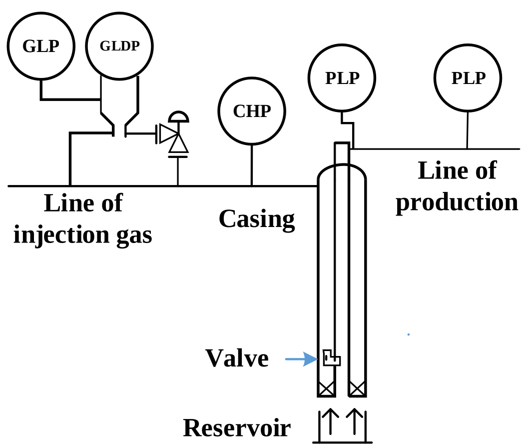


Figure 7. Representation of a Gas Lift Method in a Well(Image taken from [28])

4.1.2 Experimental Setup

The database corresponding to Gas Lift Wells consists of 1186 instances, which have 4 descriptors: Casing Pressure (*CHP*), Production Tubing Pressure (*THP*), Gas Lift Flow (*FGL*), and Bottom Pressure (*Pwf*), with 4 classes corresponding to the rate of production (*Qprod*). Values corresponding to the classes are the following [28], [29]:

- Class 1: $Q_{prod} \leq 100$
- Class 2: 100< $Q_{prod} \leq 215$
- Class 3: 215< $Q_{prod} \leq 300$
- Class 4: 300 < Q_{prod}

The classes are balanced, with the following number of instances: Class 1, Class 2, and Class 3, with 297 instances in each one, and Class 4 with 295 instances. Previously, data science tasks have been performed to avoid the existence of atypical data, and data that may have null values. Also, the descriptors have been normalized to values between 0 to 1.

In order to carry out the classification tests, 10 different settings were proposed, in which the classifiers have been trained only once with 80% of the data. The different settings vary according to the percentage of noise added to one or more descriptors in the validation data, detailed as follows:

Setting 1: Original database. The algorithm was trained with 80% of the total data of the 4 states (classes) of *Qprod*, which were randomly chosen. The remaining 20% were used for the validation of the algorithms, this means, they are the original data obtained from the process.

Setting 2, 3 and 4: Original database plus white noise in Pwf descriptor. To confuse the algorithm and hinder its classification process, white noise of 10%, 20% and 30%, respectively, was added to the *Pwf* descriptor of the validation data, which corresponds to 20% of the dataset. It is an important test because if this measurement fails (sensor fails), then the modeling and controlling of the system can have considerable negative effects.

Setting 5, 6, and 7: Original database plus white noise in CHP and THP descriptors. In this case, white noise of 10%, 20% and 30%, respectively, was added to the *CHP* and *THP* descriptors of the validation data. It is an error that could occur due to the failure of the sensors measuring these variables, or possible effects of their disarrangement. As in the previous case, the validation samples correspond to 20% of the dataset.

Setting 8, 9 and 10: Original database plus white noise in Pwf, CHP and THP descriptors. We consider these the worst scenarios, in which all the samples in the testing data have errors, which could considerably confuse and reduce the performance of the classifiers. White noise of 10%, 20% and 30%, respectively were added to the Pwf, CHP and THP descriptors. As in the previous cases, the validation data correspond to 20% of the dataset.

The procedure for the validation of the algorithms in the oil process is presented in Figure 8.

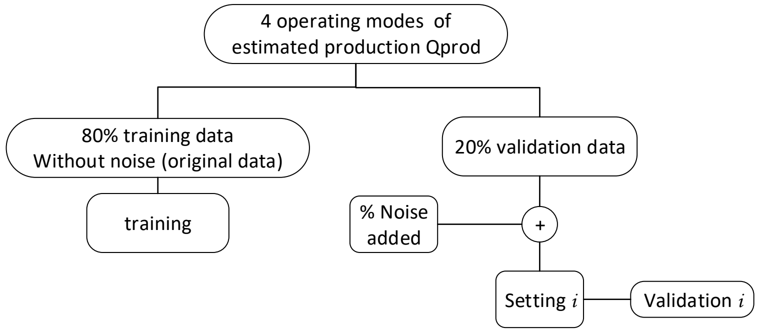


Figure 8. Experimental process in the Oil Context

4.2 Diesel Engines

4.2.1 Theoretical framework

In this case study, we use a turbocharged, 4-cylinder, 2.5 L, pre-euro automotive diesel engine. It has 17 steady-state operating modes, defined by engine torque (Nm) and engine speed (rpm), as is shown in Figure 9. The operating modes were determined using a mathematical model of longitudinal dynamics and automotive simulation for the vehicle that carries this engine (Chevrolet D-max), following the FTP-75 driving cycle. To validate the performance of the algorithms, input variables such as the position of the accelerator, exhaust gas temperature, engine speed were measured. These variables were selected because they are easy to measure in any conventional vehicle, and they give a good indication of the functional state of the engine [16], [30].

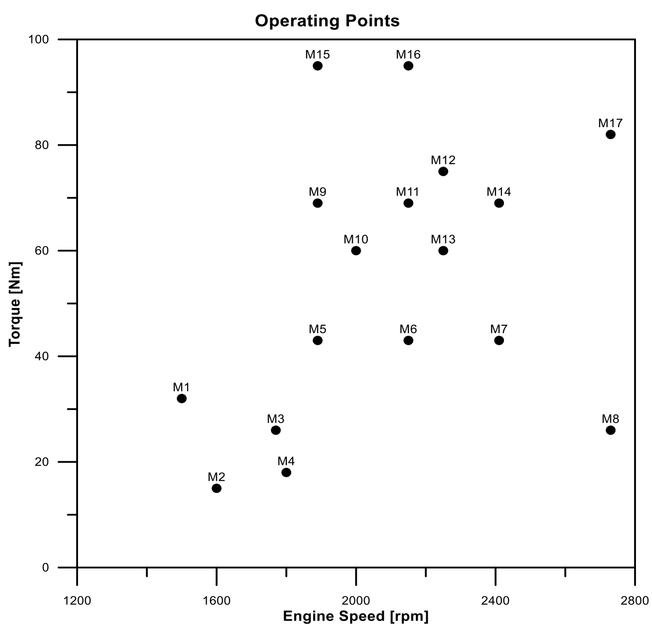


Figure 9. Stationary operating modes

To measure the torque of the diesel engine, a Shenck E90 eddy current dynamometer equipped with a U2A load cell was used. Engine speed was measured with a Heidenhain

ROD426 TTL angular encoder with a resolution of 1024 pulses/rev. Fuel consumption was measured by gravimetric techniques using a Shimadzu electronic balance (0.01g). The exhaust gas temperature was measured with a type K thermocouple and the throttle opening percentage was obtained through the voltage reading provided by a linear potentiometer located on the pedal. The experimental context is shown in Figure 10.

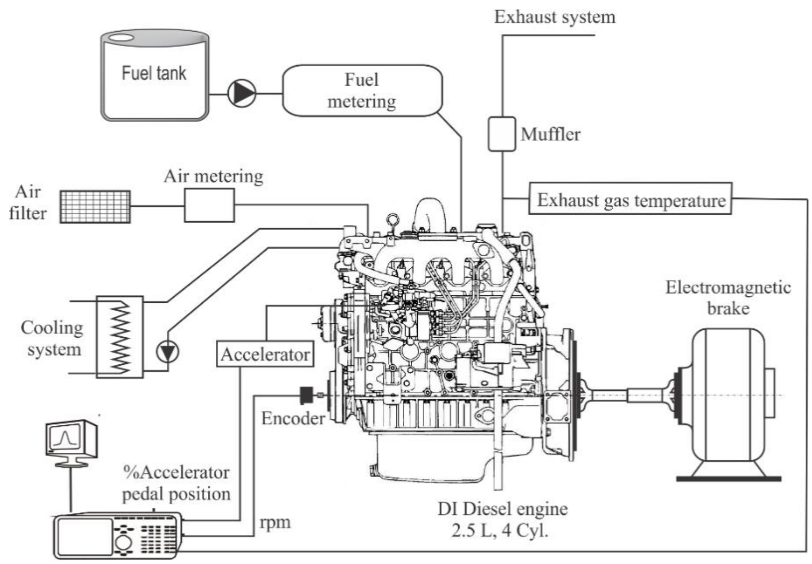


Figure 10. Experimental setup for the diesel engine (Image taken from [16])

4.2.2 Experimental setup

Three hundred (300) instantaneous pieces of data were obtained at each engine operating mode by engine speed, temperature of the exhaust and pedal position of the accelerator, conforming a database of 5100 data points. This amount of data was enough to provide reliable information about the functional state of the engine, given that, according to [16], 100 data per operating mode is enough to have satisfactory classification results. This database was normalized to values between 0 to 1. To perform the data classification tests, 4 different settings were established as follows:

Setting 1: Original and complete database. The algorithm was trained with 80% of the total data belonging to the 17 operating modes chosen randomly. The remaining 20% were utilized for the validation stage of the algorithm.

*Setting 2: Original database plus white noise. T*o confuse the algorithm and hinder its classification process, white noise was added to the descriptors of the validation data in the ranges specified in Table 1. The percentages of training and validation were 80% and 20%, respectively.

Table 1. White noise levels added to the descriptors of the system

Descriptor	White noise levels
Engine speed [rpm]	±20 rpm
Exhaust gas temperature [°C]	±5 °C
Accelerator pedal position [%]	±1 %

Setting 3: *Separate original database*. Fourteen operating modes were chosen for the training phase, while the other three modes were utilized for the validation (see Figure 11). The training stage was carried out with 80% of the historical database of the fourteen operating modes. The remaining 20% of the data and the three operating modes not considered during the training phase were chosen for the validation (testing data).

Setting 4: *Separate original database plus white noise*. Same as setting 3; however, this option includes the addition of white noise for each descriptor of the remaining 20% of the data of the fourteen operating modes in the validation data, according to Table 1.

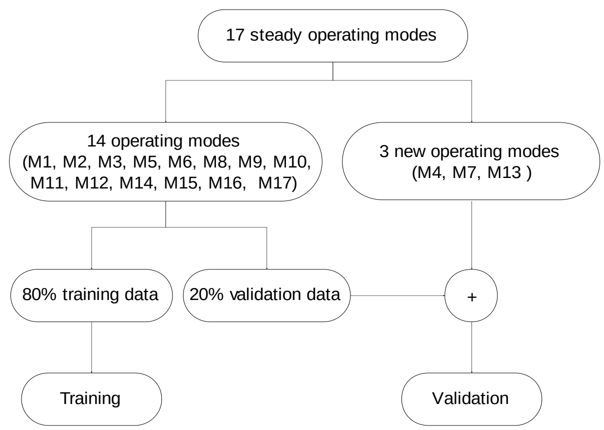


Figure 11. Usage of experimental data for setting 3 in diesel engine case study

4.3 Driver State

4.3.1 Theoretical framework

An Advanced Driver-Assistance Systems (ADAS) aim to help the driver in the driving process. In the context of ADAS, the behavior of the driver is very important to analyze. The driving styles, driver emotions and driver states for ADAS have been studied in the literature [1], [4], [31]. One of the main factors for the identification of driving styles, driver emotions, and driver states is the characterization of the patterns with their respective descriptors. Based on the patterns, it is possible to select to define algorithms focused on recognition. So, the first step is to carry out an analysis of the definition of the patterns. In the works [1], [4] different kinds of descriptors have been defined to have a good characterization of the context, but especially, a hierarchical pattern that combines this set of characteristics. The Hierarchical pattern proposed in [1], [4] is made up of three levels, with descriptors that can be inferred in a real ADAS.

In this paper, we studied the recognition problem of the driver states (second level). This level describes the states of the car driver, which can be: awake, concentrated, fatigued, stressed, lethargic, impatient, pleasant, calm, bored, asleep, etc. [32], [33]. To identify the current status of the driver, the descriptors shown in Table 2 have been selected.

Table 2. Descriptors of the pattern of the driver state [1], [4]

Descriptor	Description
Class of vehicle	Describes the type of vehicle. For example a car, a SUV, a minivan, etc.
Control Action on the vehicle	Describes the current action of the driver of the car. For example, if the driver is braking, accelerating, etc.
Emotion of the driver	Defines the emotional state of the driver, and it is defined by the third level of our pattern
Vehicle condition	Defines the current conditions of the vehicle, for example, if it has a mechanical failure, an electrical failure, if it has a lack of fuel, among other things.
Characteristics of the driver	Defines the profile of age, or physical condition, of the driver. For example, if the driver is a teen, is an older adult, if the driver has physical limitations, etc.
Driving experience	Defines the experience, for example little, medium, or large experience.
Driving hour	Defines the current hour of the day

The main objective is to recognize the driver state in order to be used by the ADAS. Because each descriptor can be obtained in a different way (vision, sound, etc.), The ADAS requires different types of sensors. [34]. This implies the use of a system of sound sensors, cameras, and devices that can process the information acquired quickly and efficiently.

4.3.2 Experimental Setup

The database consists of 145 instances, which have 7 descriptors corresponding to: Class of the vehicle, Control Action of the Vehicle, Driver's Emotions, Vehicle Condition, Characteristics of the Driver, Driving Experience, and Driving Hour, with 3 classes corresponding to the Driver's Mood. These states are the following [35]:

- Class 1: Stressed
- Class 2: Fatigue
- Class 3: Relaxed

This case study is an unbalanced dataset, which will allow observing the algorithm behavior in applications with these characteristics. The corresponding classes have the following number of instances: Class 1: 44 instances, Class 2: 2 instances and Class 3: 99 instances. As we have explained previously, data analytics tasks have been performed to avoid the existence of atypical data, and data that may have null values, to reduce the probability of errors in the classification tasks of the algorithms.

As in the previous case, to carry out the classification tests, different settings were proposed, in which the classifiers had been trained with 80% of the data, and the different settings vary according to the percentage of noise added to one or more descriptors in the validation data, detailed as follows:

Setting 1: Original database. The algorithm was trained with 80% of the total data belonging to the original data. The remaining 20% were used for the validation of the algorithms.

Setting 2: Original database plus noise in Driver's Emotions descriptor. To confuse the algorithm and hinder its classification process, the Driver's Emotions descriptor was modified to incorporate noise into the validation or testing data set. It is an important test because if there are problems in this descriptor, then we need to determine the negative effects in the recognition process.

Setting 3: Original database plus noise in Driver's Emotions and Vehicle Condition descriptors. In this case, noise to the Driver's Emotions and Vehicle Condition descriptors of the validation data was added. As in the previous case, the samples correspond to 20% of the dataset.

5 Results and discussion

As described above, the tests are carried out in case studies in which different modifications have been made. In order not to extend the paper significantly and cover the greatest number of possible cases that can be found in datasets from different applications, the following aspects have been considered:

1. Tests were performed on the three datasets, which have data with homogeneously distributed system descriptors, as well as non-homogeneous data.
2. Tests were carried out with the original data of each system and with modified data simulating the presence of noise in them.
3. Classification tests were carried out with known data for the algorithms (during training stages) and also with new validation data (data that were not part of the training) in order to observe the behavior of the data inclusion at the pre-existing classes and the generation or creation of new classes using the NIC.
4. Tests were performed omitting important descriptor data from the system (simulating sensor damage) and also with the original dataset.

5.1 *Selection of LAMDA, LAMDA-FAR and LAMDA-HAD parameters*

Figures 12 and 13 are examples that exhibit a general and illustrative behavior of how the geometric grouping in the data space would be used with binomial and Gaussian probability density functions.

To estimate the MAD array, the fuzzy binomial function was selected in all algorithms (equation 2), because this type of function uses hyper-planes to carry out the clustering process, which allowed an adequate classification of the data in each evaluated system (see example showed in Figure 12). On the other hand, if the Gaussian function was used to determine the data clustering, and knowing previously that this function uses hyper-spheres as geometric space for the grouping criterion, some data would be left out of the proposed groups (see example showed in Figure 13a), a situation which would imply increasing the exigency parameter α (equation 6) of the algorithm and, consequently, the number of classes in each system (see example showed in Figure 13b).

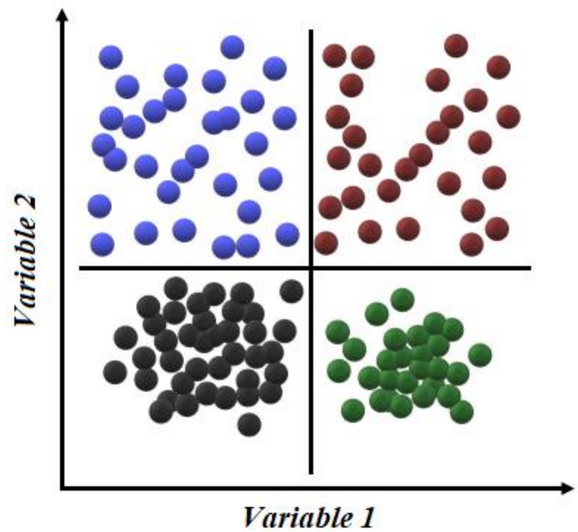
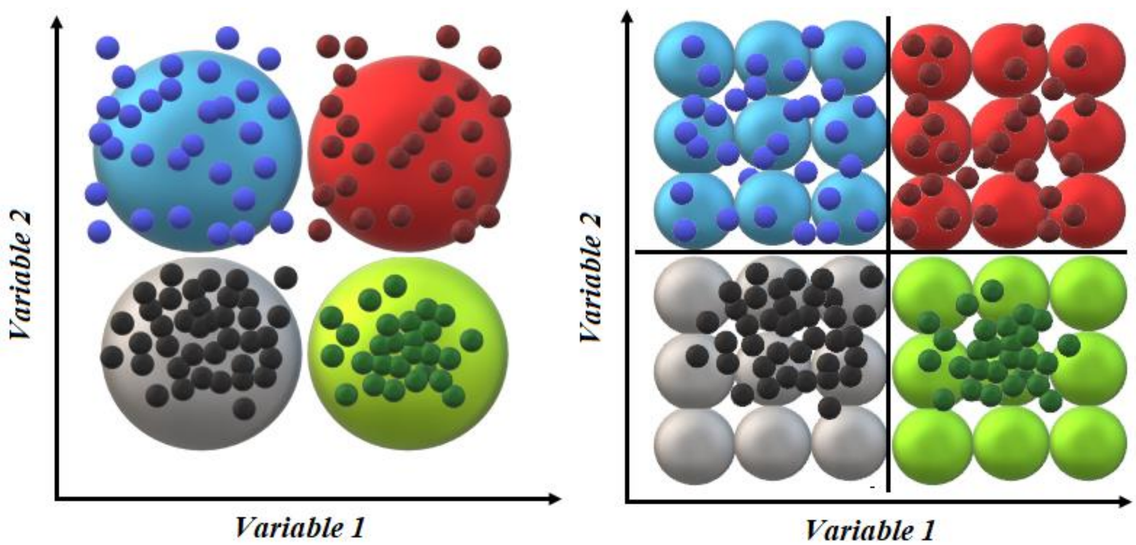


Figure 12. Division of the data space by hyper-planes using a fuzzy binomial function.



a) b)

Figure 13. Division of the data space by hyper-spheres using a fuzzy Gaussian function

The original data sets in the different systems were tested, a classification of 100% of well-classified individuals was obtained using the Product-Probabilistic sum fuzzy connector (equation 4), for this reason it was not necessary to explore other fuzzy connectors alternatives. The parameter of exigency level was constant all time and fixed in a value $\alpha = 1$, to compare the results of the original LAMDA algorithm in its maximum value, with the results achieved using the FAR and HAD algorithms versions. Table 3, shows the parameters used.

Table 3. Parameters used for the classifiers

Algorithms	Fuzzy clustering method parameters			
	Method	**Exigency**	**MAD Type**	**Connector**
LAMDA *LAMDA-FAR* *LAMDA-HAD*	Supervised	α=1	Binomial function	Probabilistic sum

5.2 AGL Well results

In this case study, the results of the classification are shown for two extreme experiments, the first one, for setting 1, in which the original data (without noise) was tested, and the second one, represents the worst-case scenario, that is, setting 10, which has the highest level of noise in most of its descriptors. Figure 14, shows the classification performed by the algorithms *LAMDA-FAR* and *LAMDA-HAD*.

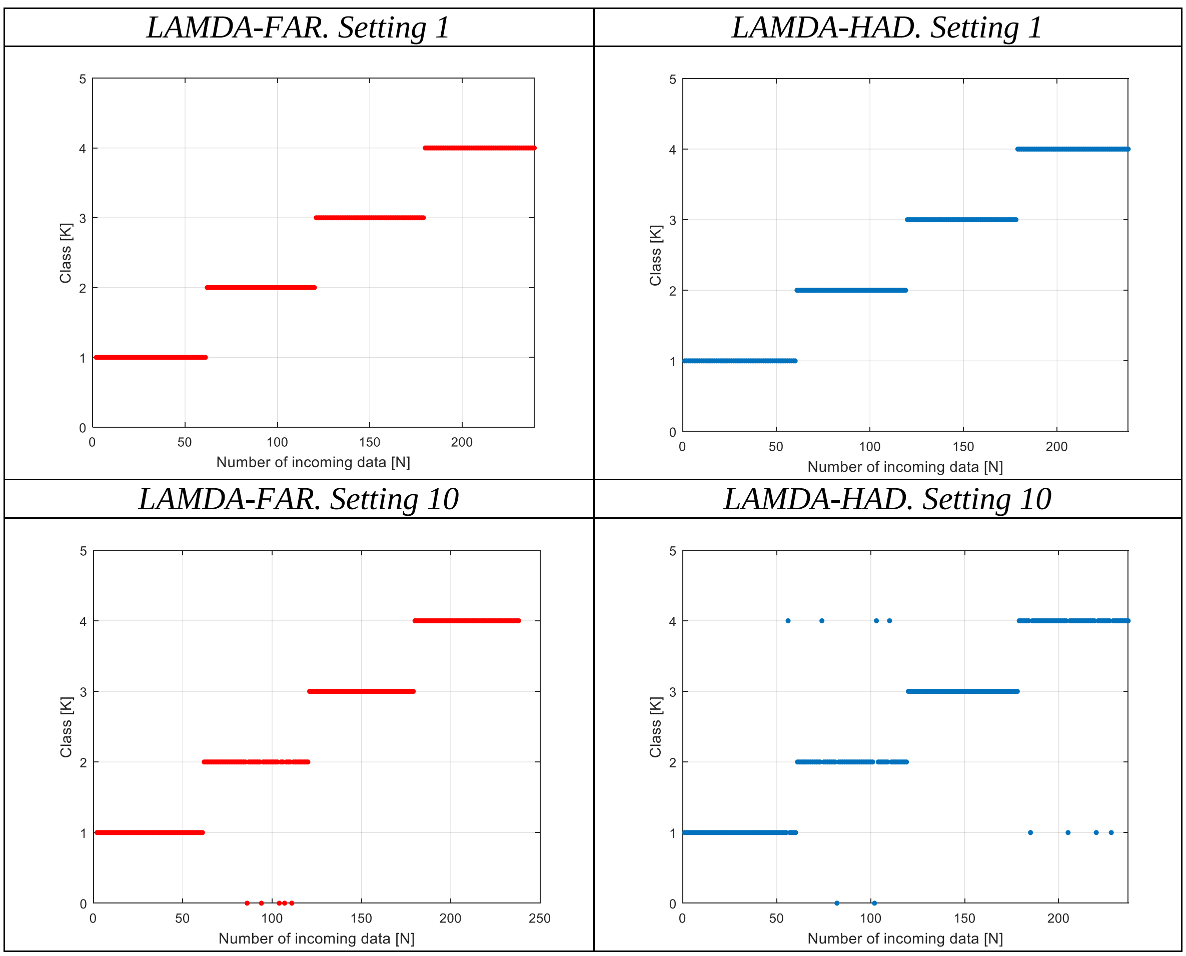


Figure 14. Classification results for validation data using *LAMDA-FAR* and *LAMDA* HAD algorithms in the AGL wells case study

Table 4, shows the results of the metrics used to compare the algorithms for each test or setting in the AGL wells case study. In this case study, in all scenarios *LAMDA* has the worst results, and among *LAMDA-HAD* and *LAMDA-FAR* in some cases, one is better than the other or vice versa. We could not define one overriding rule for determining when one algorithm is better than the other because in some scenarios, one is more precise than the other, even when considering different levels of noise. Overall, the differences are small, but when an algorithm is better, normally it is better in all the metrics.

Table 4. Results of the metrics used to compare the algorithms in the AGL wells case study

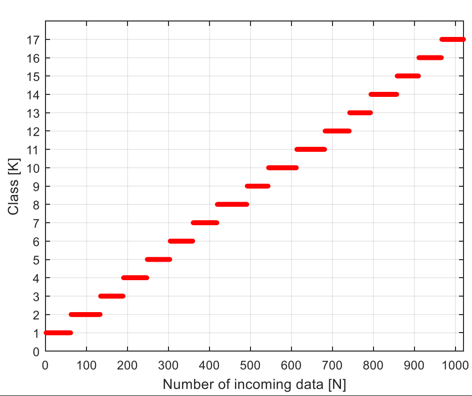
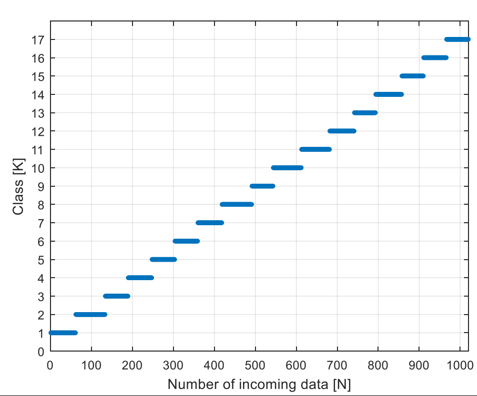
Setting	Algorithm	Accuracy	Precision	Recall	F-Measure
	LAMDA	0,9958	0,9916	0,9958	0,9937
1	*LAMDA-FAR*	1,0000	1,0000	1,0000	1,0000
	LAMDA-HAD	1,0000	1,0000	1,0000	1,0000
	LAMDA	0,9916	0,9875	0,9916	0,9895
2	*LAMDA-FAR*	0,9873	0,9749	0,9873	0,9810
	LAMDA-HAD	**0,9958**	**0,9959**	**0,9958**	**0,9958**
	LAMDA	0,9873	0,9791	0,9874	0,9832
3	*LAMDA-FAR*	0,9873	0,9749	0,9873	0,9809
	LAMDA-HAD	**0,9958**	**0,9959**	**0,9958**	**0,9958**
	LAMDA	0,9747	0,9712	0,9747	0,9727
4	*LAMDA-FAR*	**0,9958**	**0,9916**	**0,9958**	**0,9936**
	LAMDA-HAD	0,9789	0,9797	0,9789	0,9790
	LAMDA	1,0000	1,0000	1,0000	1,0000
5	*LAMDA-FAR*	0,9915	0,9832	0,9915	0,9873
	LAMDA-HAD	**1,0000**	**1,0000**	**1,0000**	**1,0000**
	LAMDA	0,9789	0,9626	0,9790	0,9707
6	*LAMDA-FAR*	0,9873	0,9749	0,9873	0,9810
	LAMDA-HAD	**0,9916**	**0,9916**	**0,9916**	**0,9916**
	LAMDA	0,9789	0,9666	0,9790	0,9727
7	*LAMDA-FAR*	**0,9915**	0,9832	**0,9915**	0,9873
	LAMDA-HAD	0,9873	**0,9875**	0,9874	**0,9874**
	LAMDA	0,9831	0,9751	0,9832	0,9791
8	*LAMDA-FAR*	0,9746	0,9504	0,9746	0,9620
	LAMDA-HAD	**0,9916**	**0,9919**	**0,9915**	**0,9916**
	LAMDA	0,9747	0,9584	0,9746	0,9662
9	*LAMDA-FAR*	0,9831	0,9667	0,9831	0,9747
	LAMDA-HAD	**0,9873**	**0,9875**	**0,9874**	**0,9874**
	LAMDA	0,9409	0,9139	0,9410	0,9272
10	*LAMDA-FAR*	**0,9788**	**0,9585**	**0,9788**	**0,9682**
	LAMDA-HAD	0,9578	0,9509	0,9577	0,9537

For Setting 1, corresponding to the LAMDA-FAR and LAMDA-HAD algorithms, the metrics presented in Table 4 show perfect performance, i.e., the algorithms properly classified all individuals. Setting 4 corresponding to the addition of 30% white noise in the Pwf descriptor and shows that LAMDA-FAR is the most robust algorithm, decreasing its performance in terms of accuracy: 0.0042 and F-Measure: 0.0064, values that demonstrate a good tolerance when affecting that descriptor. Setting 7, which corresponds to the addition of 30% white noise in the CHP and THP descriptors, shows that LAMDA-FAR and LAMDA-HAD are tolerant of added noise, with decreases in terms of accuracy (LAMDA-FAR: 0.0085 and LAMDA-HAD: 0.0127) and in terms of F-Measure (LAMDA-FAR: 0.0127 and LAMDA-HAD: 0.0126), low values compared to the affectation suffered by two of the four descriptors. In setting 10, which corresponds to the addition of 30% white noise in the descriptors CHP, THP and Pwf, it shows that LAMDA-FAR is the method that has the best tolerance to added noise, with decreases in terms of accuracy: 0.0212 and F -Measure 0.0318. That is, adding a large amount of noise to confuse the algorithms has obtained, in the worst case, an average decrease that does not exceed 2.89% considering the metrics in Table 4, which demonstrates the great effectiveness of the LAMDA-FAR algorithm under these conditions. On the other hand, LAMDA-HAD in the worst case (setting 10) presents a decrease of 4.5% in terms of performance average, and LAMDA of 6.93% in this case study.

5.3 Diesel Engine results

Figure 15 shows classification results for validation data using *LAMDA-FAR* and *LAMDA-HAD* algorithms. In this case study, the results of the classification are shown for two extreme experiments; the first one (setting 1) represents the original data (without noise) composed by 17 different operating modes, and the second one (setting 4) contains the three new operating modes (not considered during the training stage) and white noise applied to each descriptor. As it is shown, using both algorithms, all the functional states were successfully classified in its respective class (in setting 1) resulting in zero misclassified individuals. For setting 4, both algorithms have classification problems with some individuals. While *LAMDA-FAR* classifies those individuals, who do not fit their training parameters into the *NIC* class, *LAMDA-HAD* tries to assign them to the pre-existing classes. The misclassification detected are related to the noise levels incorporated into the data.

Table 5, shows all the results of the metrics used to compare the algorithms for each test or setting in the diesel engine case study. In this case, while analyzing the benefits of the LAMDA family, especially in cases where the identification of new functional states is intended, the metrics obtained by two of the best classification algorithms that currently present better results in terms of performance, are shown. These are: Linear Discriminant Analysis (LDA) and Random Forest (RF).

LAMDA-FAR. Setting 1	*LAMDA-HAD. Setting 1*
	
LAMDA-FAR. Setting 4	*LAMDA-HAD. Setting 4*

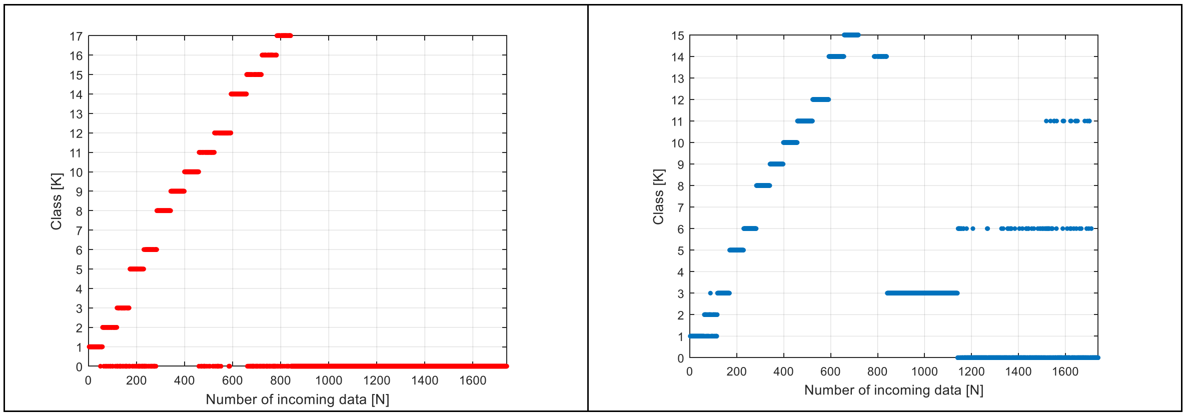


Figure 15. Classification results for validation data using *LAMDA-FAR* and *LAMDA HAD* in the diesel engines case study

Table 5. Results of the metrics used to compare the algorithms in the diesel engine case study

Setting	Algorithm	Accuracy	Precision	Recall	F_Measure
	LAMDA	1,0000	1,0000	1,0000	1,0000
	LAMDA-FAR	1,0000	1,0000	1,0000	1,0000
1	*LAMDA-HAD*	1,0000	1,0000	1,0000	1,0000
	LDA	1,0000	1,0000	1,0000	1,0000
	RF	1,0000	1,0000	1,0000	1,0000
	LAMDA	0,7284	0,3623	0,7307	0,4816
	LAMDA-FAR	0,8243	0,3488	0,8205	0,4885
2	*LAMDA-HAD*	0,9431	0,9621	0,9455	0,9419
	LDA	**1,0000**	**1,0000**	**1,0000**	**1,0000**
	RF	**1,0000**	**1,0000**	**1,0000**	**1,0000**
	LAMDA	0,4828	0,7249	0,9333	0,7634
	LAMDA-FAR	**1,0000**	**1,0000**	**1,0000**	**1,0000**
3	*LAMDA-HAD*	0,8264	0,9412	0,9776	0,9591
	LDA	0,4828	0,7651	0,9333	0,7881
	RF	0,4828	0,7664	0,9333	0,7901
	LAMDA	0,4477	0,6722	0,8650	0,6843
	LAMDA-FAR	**0,8953**	**0,9888**	0,7969	**0,8671**
4	*LAMDA-HAD*	0,7506	0,8606	**0,9040**	0,8351
	LDA	0,4828	0,7607	0,8933	0,7860
	RF	0,4736	0,7248	0,8936	0,7531

Under Setting 1, all algorithms achieve a perfect classification rate. In Setting 2, noise decreases the performance of LAMDA-based algorithms. LDA and RF show perfect results, while LAMDA-HAD (in this case, the best of the LAMDA family) has a decrease of 5% in performance terms. In the last two settings, the contribution of LAMDA is fully appreciated, since it is evident that the improvements make a good classification and identify new functional states. Under setting 3, LAMDA-FAR performs a perfect classification and identification, followed by LAMDA-HAD. In setting 4 (in which noise has been added), a better performance of the LAMDA-based proposals can also be observed due to its new class identification feature, LAMDA-FAR has an average performance decrease of 11.3%, LAMDA-HAD: 16.2%, LDA: 26.9%, and RF: 28.9%. Again, the results of our algorithms are very varied. It is not possible to define when an algorithm is better that the other. For example, *LAMDA-HAD* showed good result in scenarios with noise, but *LAMDA-FAR* showed very good performance when discovering new classes.

5.4 Driver State results

Figure 16 shows classification results for validation data using *LAMDA-FAR* and *LAMDA-HAD* algorithms in the driver state case study for settings 1 and 3. Table 6, shows the results of the metrics used to compare the algorithms for each test or setting in the driver state case study. As can be seen, due to the imbalance of classes, and to the noise levels incorporated into the descriptors, the metrics decrease immensely when all algorithms are compared. In general, *LAMDA-HAD* obtains the best results, and when the noise is not very important (setting 2) its results are very good.

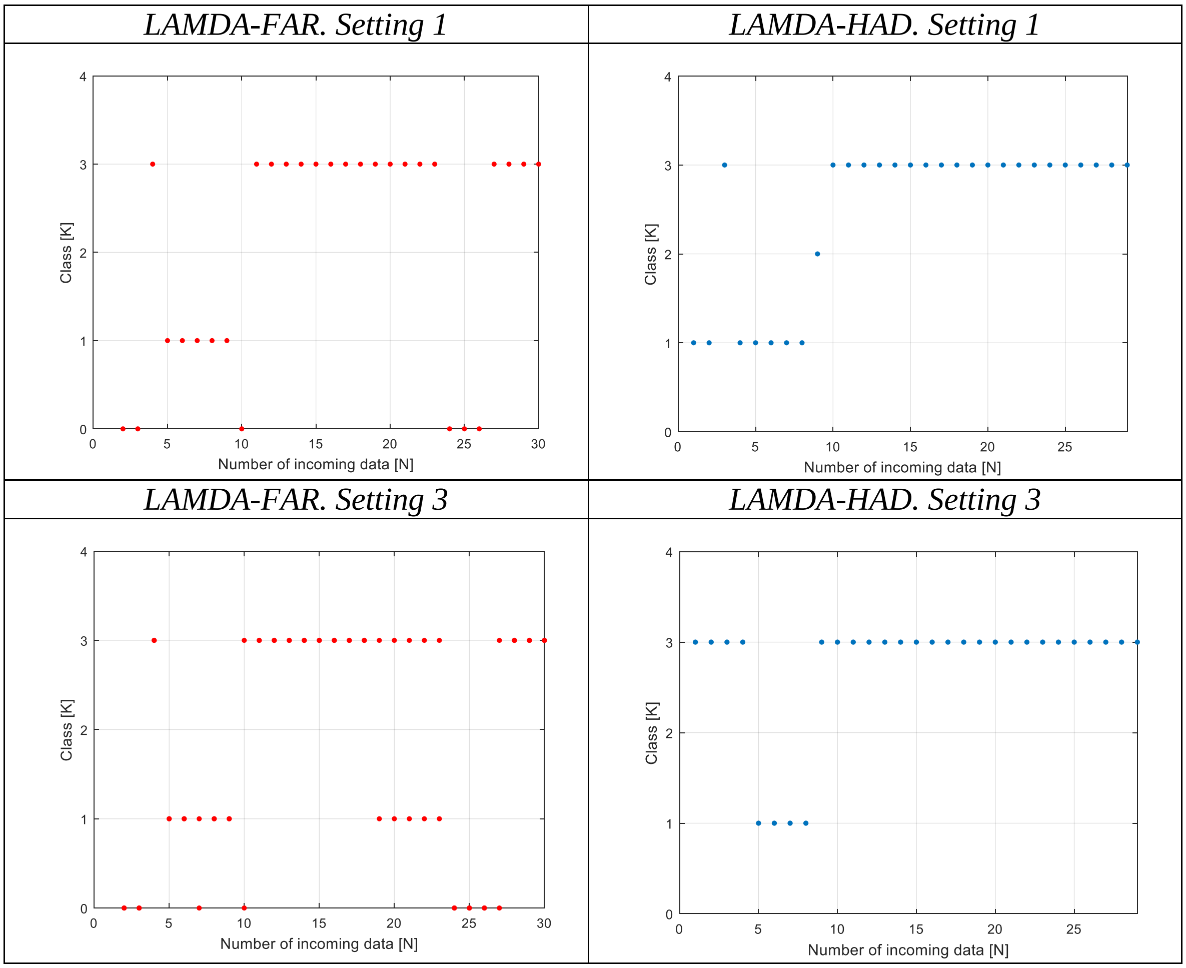


Figure 16. Classification results for validation data using *LAMDA-FAR* and *LAMDA-HAD* in the driver state case study

Table 6. Results of the metrics used to compare the algorithms in the driver state case study

Setting	Algorithm	Accuracy	Precision	Recall	F_Measure
1	*LAMDA*	0,7931	0,5939	0,8250	0,6430
	LAMDA-FAR	0,7857	0,4986	0,5214	0,5097
	LAMDA-HAD	**0,9655**	**0,9841**	**0,9583**	**0,9696**
2	*LAMDA*	0,7586	0,5639	0,7833	0,6051
	LAMDA-FAR	0,6071	0,4692	0,7238	0,5176
	LAMDA-HAD	**0,8621**	**0,9444**	**0,8333**	**0,8586**

	LAMDA	0,6207	0,3845	0,4000	0,3921
3	*LAMDA-FAR*	0,5357	0,3403	0,3738	0,3441
	LAMDA-HAD	**0,8276**	**0,6000**	**0,5000**	**0,5185**

The results for Setting 1 in Table 6, show a fairly good classification in terms of performance metrics. For example, for LAMDA-HAD: 96.9%, LAMDA-FAR: 57.2% and LAMDA: 71.4%. The performance decreases when adding noise in the Driver's Emotions descriptor, obtaining average performance values of LAMDA-HAD: 87.5%, LAMDA-FAR: 57.2% and LAMDA: 67.8%. Also, in setting 3, when adding noise in Driver's Emotions and Vehicle Condition descriptors, the obtained performance averages are LAMDA-HAD: 61.2%, LAMDA-FAR: 39.8% and LAMDA: 44.9%. The results show that the algorithms are quite sensitive to the addition of noise. Therefore, noise should be corrected in the descriptor engineering stage so that it does not affect the performance of the algorithms.

5.5 Determination of the diagnostic profile of the improved *LAMDA* algorithms

The ROC (Receiver Operating Characteristic) curves for the tested models are presented below for the different case studies, to analyze the sensitivity and specificity in the diagnostic tasks (see Figures 17, 18, 19). In general, methods with good sensitivity are required for diagnostic, since each state of the system requires a positive result for the diagnostic test, based on the class that corresponds to each functional state. Also, diagnostic methods with great specificity are necessary because it is interesting to see negative results when an operating state has not been considered in the classes considered for learning. With ROC, it is possible to calculate the area under the curve, called AUC (Area Under Curve), which takes values between 0 and 1. The required value of the ROC is close to the coordinate (0, 1), the which represents high sensitivity and specificity indicating that it is a diagnostic method of good quality.

ROC curves shown in Figures 17, 18 and 19 have been drawn for each class in the two extreme settings of the different case studies, since these are multiclass problems. In the same way, in the Tables 7, 8 and 9 are shown the average value of the AUC metrics of the classes in all the settings of the case studies. Additionally, in Table 8 the results of the LAMDA family are compared with LDA and RF.

Again, we have situations where *LAMDA-HAD* and *LAMDA-FAR* have a very similar behavior like in the AGL well case study, where *LAMDA-HAD* has better results. In this case study with a lot of noise exposure *LAMDA-FAR* shows the best classification results. In the diesel engine case study *LAMDA-FAR* has better results, and it can discover new classes, Finally, with unbalanced classes (driver state case study), *LAMDA-HAD* given very good results. In this case with noise, *LAMDA-FAR* has the worst results. As diagnostic methods, we obtain a similar behavior as in the previous subsections (5.2 to 5.4), where we have analyzed classification metrics. In contexts with noises, due for example to sensor problems, *LAMDA-HAD* given good results. Similarly, in the case where there are important imbalances in the data of the classes of the problem (see subsection 5.4). When it is necessary to discover new classes, even with the noise, *LAMDA-FAR* gives excellent results.

ROC CURVES Setting 1

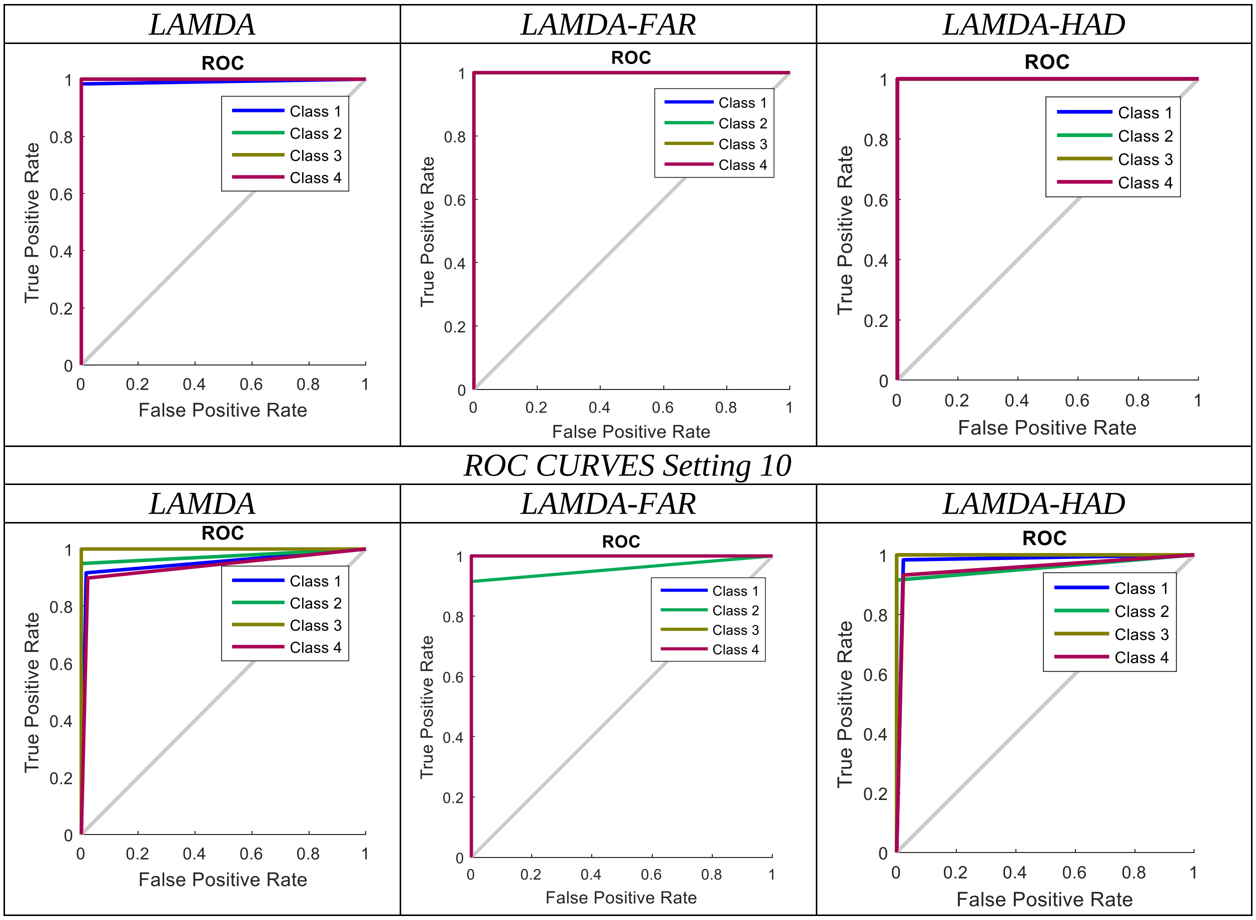


Figure 17. Comparison of sensitivity and specificity for *AGL* Wells

Table 7. Results of the diagnostic metrics of the algorithms in the AGL wells case study

Setting	Algorithm	Sensitivity	Specificity	AUC
	LAMDA	0,9958	0,9944	0,9951
1	*LAMDA-FAR*	1,0000	1,0000	1,0000
	LAMDA-HAD	1,0000	1,0000	1,0000
	LAMDA	0,9916	0,9930	0,9923
2	*LAMDA-FAR*	0,9873	0,9833	0,9853
	LAMDA-HAD	**0,9958**	**0,9986**	**0,9972**
	LAMDA	0,9874	0,9875	0,9875
3	*LAMDA-FAR*	0,9873	0,9833	0,9853
	LAMDA-HAD	**0,9958**	**0,9986**	**0,9972**
	LAMDA	0,9747	0,9874	0,9811
4	*LAMDA-FAR*	**0,9958**	**0,9944**	**0,9951**
	LAMDA-HAD	0,9789	0,9929	0,9859
	LAMDA	1,0000	1,0000	1,0000
5	*LAMDA-FAR*	0,9915	0,9888	0,9902
	LAMDA-HAD	**1,0000**	**1,0000**	**1,0000**
	LAMDA	0,9790	0,9766	0,9778
6	*LAMDA-FAR*	0,9873	0,9833	0,9853
	LAMDA-HAD	**0,9916**	**0,9972**	**0,9944**
	LAMDA	0,9790	0,9806	0,9798
7	*LAMDA-FAR*	**0,9915**	0,9888	0,9902
	LAMDA-HAD	0,9874	**0,9958**	**0,9916**
	LAMDA	0,9832	0,9861	0,9846
8	*LAMDA-FAR*	0,9746	0,9672	0,9709

	LAMDA-HAD	**0,9915**	**0,9972**	**0,9944**
	LAMDA	0,9746	0,9752	0,9749
9	*LAMDA-FAR*	0,9831	0,9779	0,9805
	LAMDA-HAD	**0,9874**	**0,9958**	**0,9916**
	LAMDA	0,9410	0,9526	0,9468
10	*LAMDA-FAR*	**0,9788**	**0,9725**	**0,9757**
	LAMDA-HAD	0,9577	0,9777	0,9677

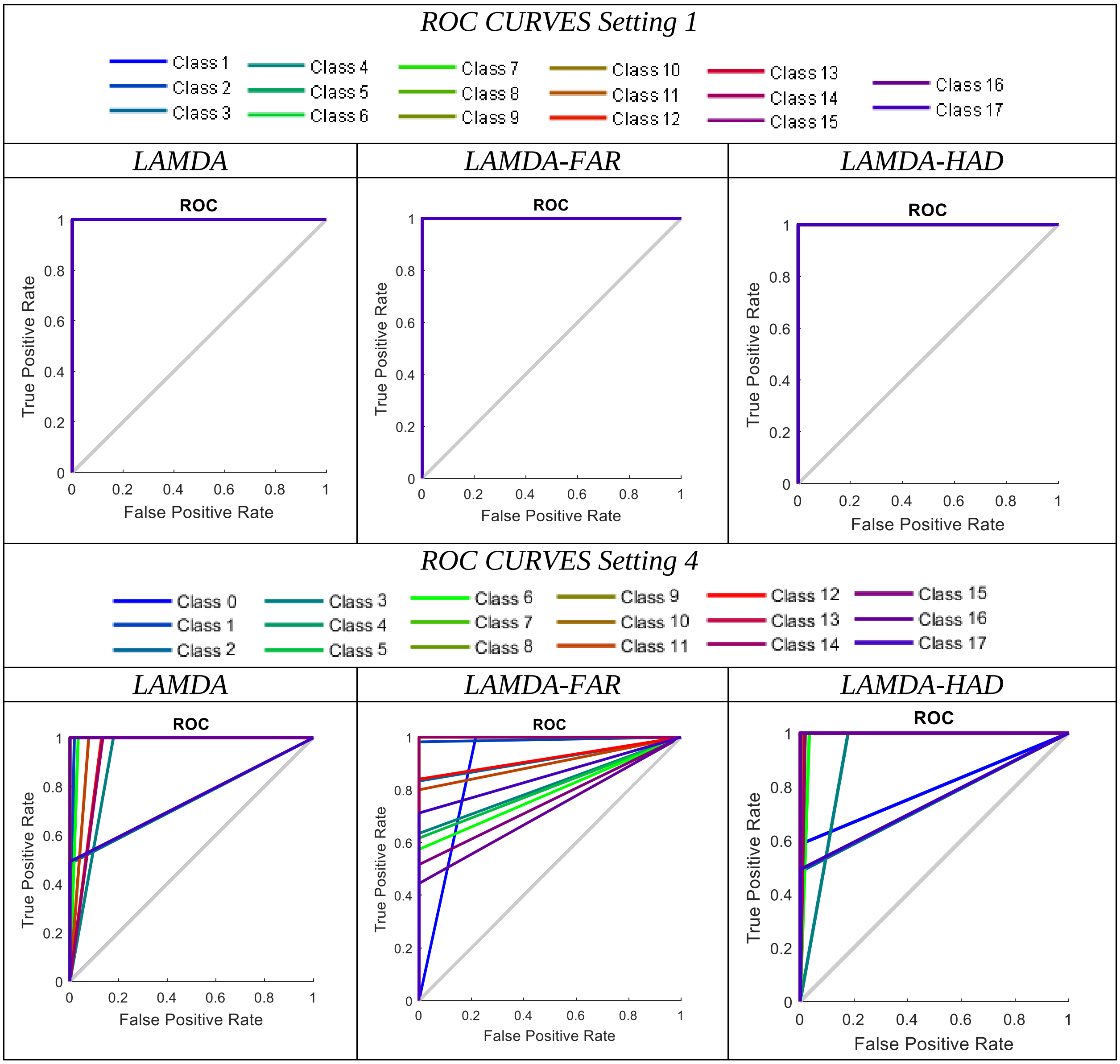


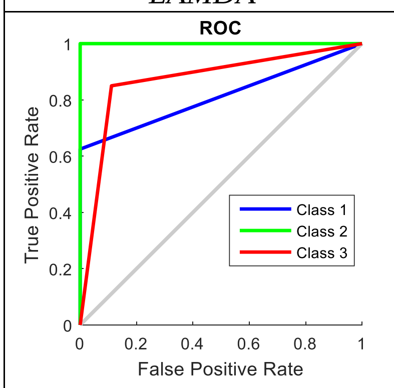
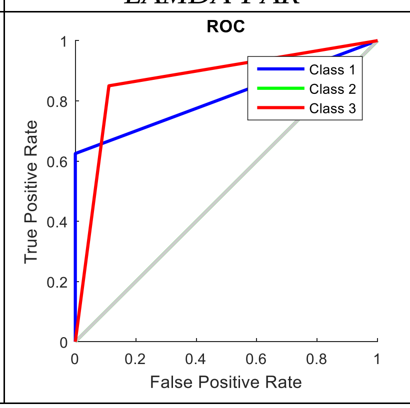
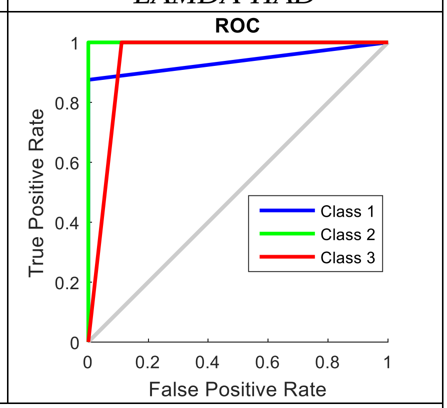
Figure 18. Comparison of sensitivity and specificity for the Diesel engine case

Table 8. Results of the diagnostic metrics of the algorithms in the diesel engine case study

Setting	Algorithm	Sensitivity	Specificity	AUC
	LAMDA	1,0000	1,0000	1,0000
	LAMDA-FAR	1,0000	1,0000	1,0000
1	*LAMDA-HAD*	1,0000	1,0000	1,0000
	LDA	1,0000	1,0000	1,0000
	RF	1,0000	1,0000	1,0000
	LAMDA	0,7307	0,8713	0,8010
	LAMDA-FAR	0,8205	0,8427	0,8316
2	*LAMDA-HAD*	0,9431	0,9621	0,9455
	LDA	**1,0000**	**1,0000**	**1,0000**
	RF	**1,0000**	**1,0000**	**1,0000**
	LAMDA	0,4828	0,7249	0,9333
	LAMDA-FAR	**1,0000**	**1,0000**	**1,0000**
3	*LAMDA-HAD*	0,9776	0,9881	0,9829
	LDA	0,9333	0,9644	0,9488
	RF	0,9333	0,9643	0,9488
	LAMDA	0,8650	0,9618	0,9134
	LAMDA-FAR	0,7969	**0,9855**	0,8912
4	*LAMDA-HAD*	**0,9040**	0,9828	**0,9434**
	LDA	0,8933	0,9064	0,8998
	RF	0,8936	0,8927	0,8931

The diagnostic measures show that the analyzed algorithms achieve very good results with the settings used for experimentation. It should be noted that when performing the analysis by class and averaging the values, the algorithms that have not been able to detect the new functional states show high results. These algorithms make a good classification with the trained classes (14 classes), although they are not good with the new classes (3 classes), that is, they are not identified. A real and more consistent analysis of the behavior and performance of the algorithms in this case study, with this metric, are those shown in Table 8. At Setting 1, all algorithms perform well, and Settings 3 and 4 show the obvious benefits of using LAMDA-based algorithms.

ROC CURVES Setting 1

LAMDA	*LAMDA-FAR*	*LAMDA-HAD*
		

ROC CURVES Setting 3

LAMDA	*LAMDA-FAR*	*LAMDA-HAD*

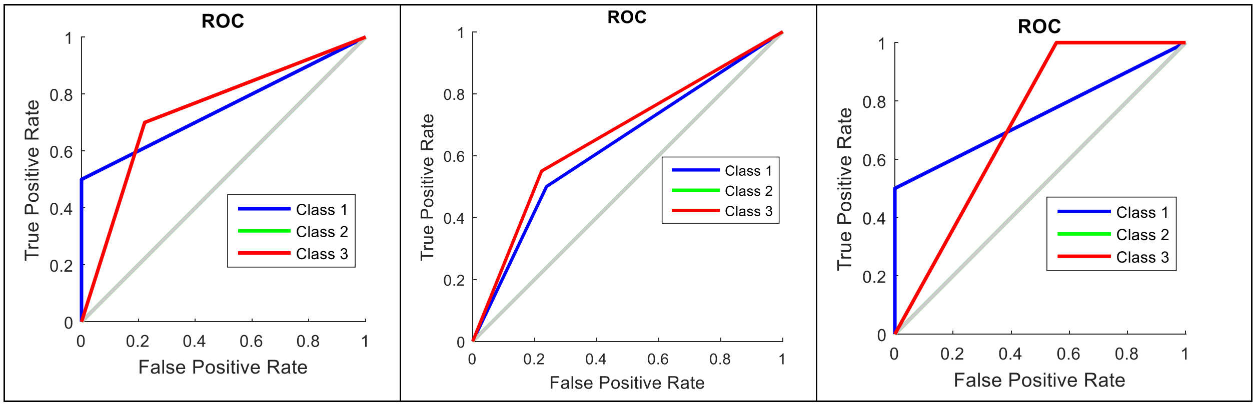


Figure 19. Comparison of sensitivity and specificity of the Driver State case

Table 9. Results of the diagnostic metrics of the algorithms in the driver state case study

Setting	Algorithm	Sensitivity	Specificity	AUC
1	*LAMDA*	0,8250	0,7425	0,7838
	LAMDA-FAR	0,5214	0,7300	0,6257
	LAMDA-HAD	**0,9583**	**0,9630**	**0,9606**
2	*LAMDA*	0,7833	0,7187	0,7510
	LAMDA-FAR	0,7238	0,6659	0,6948
	LAMDA-HAD	**0,8333**	**0,8519**	**0,8426**
3	*LAMDA*	0,4000	0,6152	0,5076
	LAMDA-FAR	0,3738	0,6131	0,4935
	LAMDA-HAD	**0,5000**	**0,8148**	**0,6574**

6 Conclusions

In this work, we have presented two of the latest improvements of the *LAMDA* algorithm regarding classification tasks, and we have compared them in different case studies. Each case study has a specific characteristic. In one case there are few well-balanced classes, but several levels of noise are introduced in almost all its descriptors; in the second one there are many classes and some of them must be discovered (they are not used to train the classifier), and in the other there is an important imbalance in the classes.

Based on our classification and diagnostic metrics, we have determined behavior profiles for algorithms. *LAMDA-HAD* is better with unbalanced classes, while *LAMDA-FAR* is excellent for discovering new classes. Both algorithms work well under different levels of noise (which can represent faults in the sensors), an important factor in diagnostic tasks.

Further research should be conducted that will allow us to determine the maximum acceptable noise level to diagnose, as well as the proportions of imbalance supported by each problem. For example, in the case study about the driver state, it seems that it is around 20% the noise level, but in other problems (e.g. the AGL wells), it seems that it is larger according to the results obtained (see table 4, Setting 10).

Declarations

Funding*:* Authors wish to acknowledge the Universidad de Antioquia, especially to the thermal machine laboratory- GIMEL research group and the Institución Universitaria Pascual Bravo. We gratefully acknowledge the financial support provided by the Colombia Scientific Program (SENECA) within the framework of the call Ecosistema Científico (Contract No. FP44842-218-2018). Special thanks to Ingenio Providencia S.A for the donation of ethanol fuel and to Ecopetrol for the donation of ULSD to carry out the diesel engine case study. Frank Ruiz acknowledges the Colombian science foundation (COLCIENCIAS) for his doctoral scholarship.

Conflicts of interest/Competing interests: The authors declare no conflict of interest. The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper

Availability of data and material: The data will be available if they are requested from the authors

Code availability*:* The code will be available if they are requested from the authors

References

[1] J. Aguilar, K. Aguilar, D. Chávez, J. Cordero, and E. Puerto, "Different Intelligent Approaches for Modeling the Style of Car Driving," in *Proceedings of the 14th International Conference on Informatics in Control, Automation and Robotics*, 2017, vol. 2, no. Icinco, pp. 284–291.

[2] J. F. Botía, C. Isaza, T. Kempowsky, M. V. Le Lann, and J. Aguilar-Martín, "Automaton based on fuzzy clustering methods for monitoring industrial processes," *Eng. Appl. Artif. Intell.*, vol. 26, no. 4, pp. 1211–1220, Apr. 2013.

[3] M. Araujo, J. Aguilar, and H. Aponte, "Fault detection system in gas lift well based on artificial immune system," in *Proceedings of the International Joint Conference on Neural Networks, 2003.*, 2003, vol. 3, no. June, pp. 1673–1677.

[4] J. Cordero, J. Aguilar, K. Aguilar, D. Chávez, and E. Puerto, "Recognition of the Driving Style in Vehicle Drivers," *Sensors*, vol. 20, no. 9, p. 2597, May 2020.

[5] J. Aguilar-Martín and R. López De Mantaras, "The process of classification and learning the meaning of linguistic descriptors of concepts," in *Approximate reasoning in decision analysis*, North-Holland Publishing Company, 1982, pp. 165–175.

[6] L. Morales, H. Lozada, J. Aguilar, and E. Camargo, "Applicability of LAMDA as classification model in the oil production," *Artif. Intell. Rev.*, vol. 53, no. 3, pp. 2207–2236, Mar. 2020.

[7] C. R. Santos-Junior, T. Abreu, M. L. M. Lopes, and A. D. P. Lotufo, "A new approach to online training for the Fuzzy ARTMAP artificial neural network," *Appl. Soft Comput.*, vol. 113, p. 107936, Dec. 2021.

[8] J. A. Ramirez-Bautista, J. A. Huerta-Ruelas, L. T. Kóczy, M. F. Hatwágner, S. L. Chaparro-Cárdenas, and A. Hernández-Zavala, "Classification of plantar foot

alterations by fuzzy cognitive maps against multi-layer perceptron neural network," *Biocybern. Biomed. Eng.*, vol. 40, no. 1, pp. 404–414, Jan. 2020.

[9] A. Das, S. K. Mohapatra, and M. N. Mohanty, "Design of deep ensemble classifier with fuzzy decision method for biomedical image classification," *Appl. Soft Comput.*, vol. 115, p. 108178, Jan. 2022.

[10] A. Saffari, M. Khishe, and S.-H. Zahiri, "Fuzzy-ChOA: an improved chimp optimization algorithm for marine mammal classification using artificial neural network," *Analog Integr. Circuits Signal Process.*, vol. 1, Mar. 2022.

[11] C. Bedoya, J. Waissman Villanova, and C. V. Isaza Narvaez, "Yager–Rybalov Triple Π Operator as a Means of Reducing the Number of Generated Clusters in Unsupervised Anuran Vocalization Recognition," 2014, pp. 382–391.

[12] C. Isaza, J. Aguilar-Martin, M. V. Le Lann, J. Aguilar, and A. Rios-Bolivar, "An Optimization Method for the Data Space Partition Obtained by Classification Techniques for the Monitoring of Dynamic Processes," *Artif. Intell. Res. Dev.*, vol. 146, pp. 80–87, 2006.

[13] H. R. Hernandez, J. L. Camas, A. Medina, M. Perez, and M. Veronique Le Lann, "Fault Diagnosis by LAMDA methodology Applied to Drinking Water Plant," *IEEE Lat. Am. Trans.*, vol. 12, no. 6, pp. 985–990, Sep. 2014.

[14] J. Waissman, R. Sarrate, T. Escobet, J. Aguilar, and B. Dahhou, "Wastewater treatment process supervision by means of a fuzzy automation model," *IEEE Int. Symp. Intell. Control - Proc.*, no. Isic, pp. 163–168, 2000.

[15] J. Mora-Florez, V. Barrera-Nunez, and G. Carrillo-Caicedo, "Fault Location in Power Distribution Systems Using a Learning Algorithm for Multivariable Data Analysis," *IEEE Trans. Power Deliv.*, vol. 22, no. 3, pp. 1715–1721, 2007.

[16] F. Ruiz, C. Isaza, A. Agudelo, and J. Agudelo, "A new criterion to validate and improve the classification process of LAMDA algorithm applied to diesel engines," *Eng. Appl. Artif. Intell.*, vol. 60, pp. 117–127, 2017.

[17] L. Morales, J. Aguilar, O. Camacho, and A. Rosales, "An intelligent sliding mode controller based on LAMDA for a class of SISO uncertain systems," *Inf. Sci. (Ny).*, vol. 567, pp. 75–99, Aug. 2021.

[18] L. Morales and J. Aguilar, "An Automatic Merge Technique to Improve the Clustering Quality Performed by LAMDA," *IEEE Access*, vol. 8, pp. 162917–162944, 2020.

[19] A. Doncescu, J. Aguilar-Martin, and J.-C. Atine, "Image color segmentation using the fuzzy tree algorithm T-LAMDA," *Fuzzy Sets Syst.*, vol. 158, no. 3, pp. 230–238, Feb. 2007.

[20] L. Morales, M. Herrera, O. Camacho, P. Leica, and J. Aguilar, "LAMDA Control Approaches Applied to Trajectory Tracking for Mobile Robots," *IEEE Access*, vol. 9, pp. 37179–37195, 2021.

[21] L. Morales, J. Aguilar, A. Rosales, D. Chávez, and P. Leica, "Modeling and control of nonlinear systems using an Adaptive LAMDA approach," *Appl. Soft Comput.*, vol. 95, Oct. 2020.

[22] L. Morales, J. Aguilar, A. Rosales, and D. Pozo-Espin, "A Fuzzy Sliding-Mode Control based on Z-Numbers and LAMDA," *IEEE Access*, vol. PP, pp. 1–1, 2021.

[23] J. F. Botía Valderrama and D. J. L. Botía Valderrama, "On LAMDA clustering method based on typicality degree and intuitionistic fuzzy sets," *Expert Syst. Appl.*, vol. 107, pp. 196–221, Oct. 2018.

[24] C. V. Isaza, H. O. Sarmiento, T. Kempowsky-Hamon, and M.-V. LeLann, "Situation

prediction based on fuzzy clustering for industrial complex processes,” *Inf. Sci. (Ny).*, vol. 279, no. 7, pp. 785–804, Sep. 2014.

[25] L. Morales, J. Aguilar, D. Chávez, and C. Isaza, “LAMDA-HAD, an extension to the LAMDA classifier in the context of supervised learning,” *Int. J. Inf. Technol. Decis. Mak.*, vol. 19, no. 1, 2020.

[26] L. Morales, C. A. Ouedraogo, J. Aguilar, C. Chassot, S. Medjiah, and K. Drira, “Experimental comparison of the diagnostic capabilities of classification and clustering algorithms for the QoS management in an autonomic IoT platform,” *Serv. Oriented Comput. Appl.*, vol. 13, no. 3, pp. 199–219, Sep. 2019.

[27] E. Camargo, J. Aguilar, A. Ríos, F. Rivas, and J. Aguilar-Martin, “Nodal analysis-based design for improving gas lift wells production,” *WSEAS Trans. Inf. Sci. Appl.*, vol. 5, no. 5, pp. 706–715, 2008.

[28] E. Camargo and J. Aguilar, “Hybrid intelligent supervision model of oil wells,” in *2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2014, no. November 2014, pp. 934–939.

[29] E. Camargo and J. Aguilar, “Advanced Supervision Of Oil Wells Based On Soft Computing Techniques,” *J. Artif. Intell. Soft Comput. Res.*, vol. 4, no. 3, pp. 215–225, 2014.

[30] F. A. Ruiz, M. Cadrazco, A. F. López, J. Sanchez-Valdepeñas, and J. R. Agudelo, “Impact of dual-fuel combustion with n-butanol or hydrous ethanol on the oxidation reactivity and nanostructure of diesel particulate matter,” *Fuel*, vol. 161, no. August, pp. 18–25, Dec. 2015.

[31] A. Verbeke, “Advanced Driver Assistance System,” *Int. J. Recent Technol. Eng.*, vol. 8, no. 6, pp. 3481–3487, Mar. 2020.

[32] C. Guoying, “Study on Identification of Driver Steering Behavior Characteristics Based on Pattern Recognition,” *Int. Robot. Autom. J.*, vol. 1, no. 1, pp. 22–28, Oct. 2016.

[33] C. Lisetti and F. Nasoz, “Affective intelligent car interfaces with emotion recognition,” *Proc. 11th Int. Conf. Hum. Comput. Interact.*, no. July, pp. 1–10, 2005.

[34] L. Kessous, G. Castellano, and G. Caridakis, “Multimodal emotion recognition in speech-based interaction using facial expression, body gesture and acoustic analysis,” *J. Multimodal User Interfaces*, vol. 3, no. 1–2, pp. 33–48, Mar. 2010.

[35] M. Kuderer, S. Gulati, and W. Burgard, “Learning driving styles for autonomous vehicles from demonstration,” in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, vol. 2015-June, no. June, pp. 2641–2646.