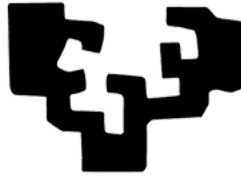


eman ta zabal zazu



Universidad  
del País Vasco

Euskal Herriko  
Unibertsitatea

PhD Thesis

---

**Design and Development of Soft-sensors in the  
Context of Oil and Gas 4.0: an AutoML Approach  
based on Machine Learning Feature Engineering**

---

Author:

Iratxe Niño Adan

Supervisors:

Eva Portillo Pérez  
Itziar Landa Torres

Department of Automatic Control and Systems Engineering

**2021**

**Iratxe Niño Adan**

*Design and Development of Soft-Sensors in the Context of Oil and Gas 4.0: an AutoML Approach based on Machine Learning Feature Engineering*

PhD Thesis, 2021

Supervisors:

**Eva Portillo Pérez**

Automatic Control and Systems

Engineering

eman ta zabal zazu



Universidad  
del País Vasco

Euskal Herriko  
Unibertsitatea

**Itziar Landa Torres**

Petronor Innovación S.L.



**Petronor**  
Innovación

Thesis developed at Tecnalia Research and Innovation.

Supervisor from the company: **Diana Manjarrés Martínez**

**tecnalia**

MEMBER OF BASQUE RESEARCH  
& TECHNOLOGY ALLIANCE

*Por mí y por todos mis compañeros.*



Es de buen nacido ser agradecido. En primer lugar me gustaría agradecer a Tecnalia por acogerme durante estos 4 años. Como me dijo un día mi profesor de geometría José Luis Mencía: La UPV/EHU, y especialmente el departamento de matemáticas iba a ser mi alma matter por ser el sitio donde me había formado. Bueno, pues Tecnalia por ser el centro donde he hecho el doctorado se ha ganado también el título de alma matter en mi corazón. Y como Tecnalia son las personas que lo conforman, agradezco a Joseba Laka e Iñigo Arizaga por su apoyo, a Isidoro Cirión por ser mi gran jefe y a la gente de OPTIMA, por su compañía y aliento a lo largo de este camino. He disfrutado enormemente a vuestro lado.

También agradezco a Petronor Innovación y su equipo, al cual he podido visitar en alguna ocasión y me han hecho sentir como en casa. Por introducirme y despertar en mí el interés por el mundo de la refinería. Ha sido un placer bajar del mundo abstracto al industrial con vosotros. Especialmente, agradezco a Lucía Orbe, por compartir su conocimiento experto y sus detalladas explicaciones sobre los procesos.

Por supuesto, he de agradecer a mis amigas. Las amigas de Trigueros del Valle, Leti, Bea y Rocío, que a pesar de la distancia y los años seguimos siendo igual de cercanas. Al grupo de la Uni: Josué, Oihane, Maider, Julene, Alaitz, Andrea, porque somos unos cracks! A las amigas del colegio, especialmente Nieves, Lucía y Alejandra, las de siempre, a las que aguanto y las que me aguantan, las que me sacan fotos a traición y las que me llenan de alegría. También incluyo en este apartado a Adriana y Alejandro, doctorandos en Tecnalia como yo que a lo largo de estos años se han ganado el título de amigos.

Agradezco especialmente a mi familia: mi madre, mi padre, mi hermano, mis abuelos, mis tíos, primos e incluso a antepasados a los que siento que conozco. A todos vosotros gracias por vuestro cariño y ser como sois. Adaptando la frase de Ortega y Gasset, “ Yo soy yo por mis circunstancias”. Por tanto, he de agradecer a la aleatoriedad o al destino el haber tenido la suerte de nacer en la familia que me ha tocado, a la cual atribuyo en gran medida las cualidades que se puedan destacar de mi persona. Os quiero a todos.

Y sobre todo, agradezco principalmente a mis tutoras, Itziar Landa y Eva Portillo, y a la supervisora por parte de Tecnalia Diana Manjarres. A ellas las menciono junto a mi familia por el lugar tan especial que han pasado a ocupar en mi vida. A todas ellas agradecer y reconocer su esfuerzo, interés y paciencia. A Diana por su apoyo y el día a día. A Eva por su confianza, su cercanía, su dedicación, su energía y por compartir conmigo su conocimiento . Y a Itziar por seguir conmigo, por su fe en mí, por los cientos de oportunidades que me ha

---

brindado y por darme alas, no solo en lo científico también en lo personal.

El teneros a las tres ha sido un privilegio. Cada una de vosotras sois la personificación de la excelencia y en conjunto formáis un equipo digno de mención. Sois el espejo en el que me gustaría verme reflejada. He sido una afortunada por haberos tenido de ejemplo tanto en lo profesional como en lo personal.

Mil gracias.

# List of Papers

## Paper I

Niño-Adan , I. Manjarres, D. Landa-Torres I. and Portillo E. “Feature Weighting methods: A review”. In: *Expert systems with Applications*. Vol. 184, (2021), 115424. DOI: 10.1016/j.eswa.2021.115424.

**JCR: 6.954**, Category: *Computer Science, Artificial Intelligence*, 23/139, Q1

## Paper II

Niño-Adan, I., Landa-Torres I., Portillo E. and Manjarres, D. To appear in “Influence of Statistical Feature Normalisation Methods on K-Nearest Neighbours and K-Means in the Context of Industry 4.0”. In: *Engineering Applications of Artificial Intelligence*.

(Second round of revision. Accepted by three out of four reviewers. Pending approval of the fourth reviewer.)

**JCR: 6.212**, Categories: *Computer Science, Artificial Intelligence*, 26/139, Q1, *Automation & Control Systems*, 6/63, Q1

## Paper III

Niño-Adan, I., Portillo E. , Landa-Torres I. and Manjarres, D. “Normalization Influence on ANN-Based Models Performance: A New Proposal for Features’ Contribution Analysis”. In: *IEEE Access*. Vol. 9, (2021), pp. 125462-125477. DOI: 10.1109/ACCESS.2021.3110647.

**JCR: 3.367**, Category: *Computer Science, Information Systems*, 65/161, Q2

## Paper IV

Niño-Adan, I., Landa-Torres I., Portillo E. and Manjarres, D. “Soft-sensor design for vacuum distillation bottom product penetration classification”. In: *Applied Soft Computing*. Vol. 102, (2021), 107072. DOI: 10.1016/j.asoc.2020.107072.

**JCR: 6.725**, Categories: *Computer Science, Interdisciplinary Applications, 11/112, Q1, Computer Science, Artificial Intelligence, 24/139, Q1*

## **Paper V**

Niño-Adan, I., Landa-Torres I., Portillo E. and Manjarres, D. and Orbe L. “Soft-sensor for Class Prediction of the Percentage of Pentanes in Butane at a Debutanizer Column”. In: *Sensors*. Vol. 21, no. 12, 3991, (2021), DOI: 10.3390/s21123991.

**JCR: 3.576**, Category: *Instruments & Instrumentation, 14/64, Q1*



# Contents

<b>List of Papers</b>	<b>v</b>
<b>Contents</b>	<b>vii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Introduction . . . . .	1
1.2 State-of-the-art . . . . .	5
1.3 Hypotheses and objectives . . . . .	15
1.4 Results and discussion . . . . .	18
1.5 References . . . . .	38
<b>2 Conclusions</b>	<b>49</b>
2.1 Future work . . . . .	51
<b>Papers</b>	<b>54</b>
I Feature Weighting methods: A review . . . . .	55
II Influence of Statistical Feature Normalisation Methods on K-Nearest Neighbours and K-means in the Context of Industry 4.0 . . . . .	116
III Normalisation Influence on ANN-Based Models Performance: A New Proposal for Features' Contribution Analysis	147
IV Soft-sensor design for vacuum distillation bottom product penetration classification . . . . .	165
V Soft-Sensor for Class Prediction of the Percentage of Pentanes in Butane at a Debutanizer Column . . . . .	189



# Chapter 1

## Introduction

### 1.1 Introduction

Industry is defined as the economic activity concerned with processing raw materials and manufacturing goods in factories. The first industrial revolution took place in the 18<sup>th</sup> century and as Figure 1.1 shows, since then, the industry has continuously evolved, mostly powered by economic, environmental and technological factors.

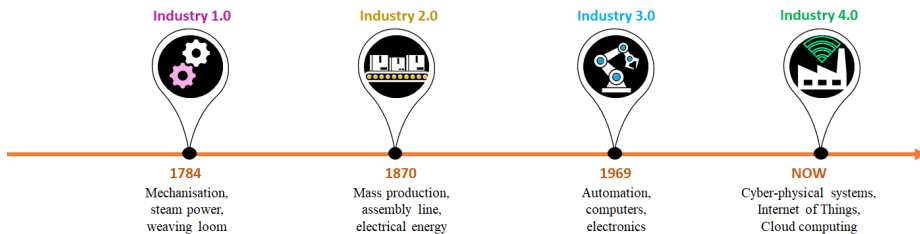


Figure 1.1: Industrial revolutions.

Nowadays, we are witness of a new industrial revolution, referred to as “Industry 4.0”. Industry 4.0 lies in industrial transformation, revitalisation and development [1] by the upgrading of manufacturing technologies by cyber-physical systems, the Internet of Things (IoT) and cloud computing [2]. As a result of integrating new emerging and disruptive information technologies in the industrial plants, the traditional factories are converted into Smart factories or Factories of the Future.

Among the sectors that have adopted the Industry 4.0 philosophy, refineries are one of them, giving place to “Oil and Gas 4.0” which core goal, similar to Industry 4.0, is to achieve higher value by employing digital technologies. The Oil and Gas industry is complex and diverse, and it can be divided into three different sectors depicted in Figure 1.2: upstream, midstream and downstream.



Figure 1.2: Oil and Gas Streams. (Source: [3])

“Upstream” refers to the activities that concern the exploration and production of oil and natural gas. The “midstream” segment is related to

## 1. Introduction

the transportation and storage of crude oil and natural gas. The final sector of the oil and natural gas industry is known as “downstream”. It includes everything involved in converting crude oil and natural gas into thousands of marketable subproducts. Such conversion is done in the refinery, which is a group of manufacturing plants composed of several unit operations that convert raw crude into valuable subproducts. Crude oil is a mixture of thousands of different hydrocarbon compounds that, in the refining process, are separated into fractions of varying boiling ranges. These fractions are further processed to be converted into subproducts, such as butane, gasoline, or diesel fuels, among others [4]. The separation of the crude oil into valuable subproducts is done based on the boiling points (Figure 1.3).

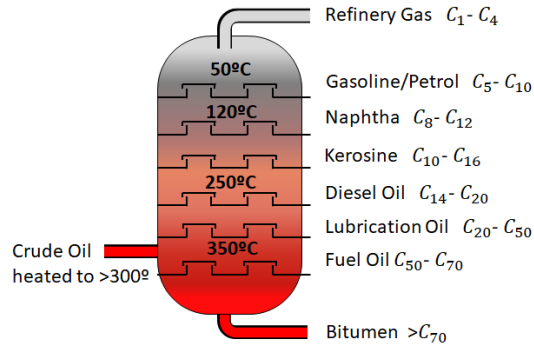


Figure 1.3: Crude oil fractional distillation.

Refineries already produce vast volumes of subproduct. However, the direction of the refining industry is determined by 1) increased operating costs or investments due to stringent environmental regulations and 2) accelerating globalisation resulting in stronger international petroleum price scenarios [5]. In consequence, the profit obtained by the refineries per unit of subproduct is considerably small, as exemplified in Figure 1.4 in which the European refining margins in the last two years are shown.

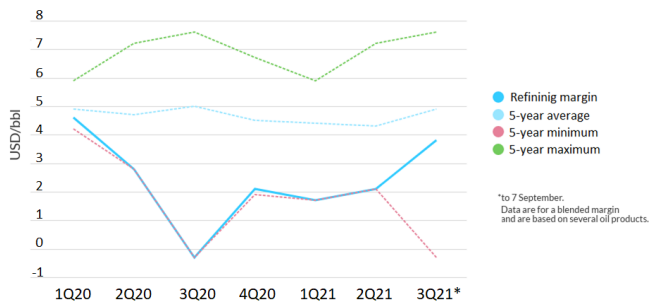


Figure 1.4: European refining margins in 2020 and 2021 in terms of dollars per barrel. (Source: [6]).

Optimising the refinery operations to reduce costs and improve efficiency is mandatory to achieve the difference between profit and loss and be globally competitive. In this regard, the digitalisation in the refinery sector is essential to cope with future challenges and achieve a profitable outcome [5].

The irruption of digitalisation is driven by the development of 1) faster and more informative sensing technology able to collect and store information from multiple sources, as well as 2) the increment of computational resources [7] that allow to process large amounts of data by advanced analytical platforms. This way, the key enablers of the smart factories are technology, data and analytics. Refineries are already collecting information from process units and streams (e.g., temperature, flow, pressure, pH, conductivity), resulting in massive amounts of data that may give place to data-rich scenarios. The phrase “data-rich and information poor” describes organisations rich in data but with a lack of processes that produce meaningful information and create a competitive advantage from such data [8]. In fact, nowadays, it is said that “the data is the new oil”, so similar to the crude oil, data must be refined to extract valuable information. For doing so, data analysis is mandatory and a principal actor in industrial digitalisation.

Among the data analysis techniques, Artificial Intelligence (AI) in general and Machine Learning (ML) methods, in particular, can ingest hundreds of features and, if proper data preprocessing is conducted, learn and extract hidden patterns between the input and output variables for modelling the problem at hand. Thus, these methods represent an advantage over traditional first-principle approaches that must be explicitly modelled based on the physical knowledge of the analysed problem, which is often unfeasible in complex industrial plants with several interconnected equipment.

As a consequence, AI is gaining territory in refining plants through soft-sensors development. *Soft-sensors* (also called observed-based sensors, virtual or inferential sensors) refer to mathematical models that enable to infer any system variable or product quality based on some other available and related variables [9, 10]. The term soft-sensor derives from incorporating the word “software” to the term “sensor” to distinguish from their counterparts so as computer programs perform the estimations. Soft-sensors, which complement hardware sensors, are easily implemented on existing hardware, suppose a low-cost alternative to devices required to work in hostile environments and estimate data in real-time, enabling real-time optimisation of the entire value chain [1].

It is important to note that soft-sensors are of special utility along all the streams of the Oil and Gas industry. For instance, they have been employed for the optimisation of the total refinery profit [11], the optimal scheduling for crude oil loading and unloading [12] or steam injection [13], multi-level production planning [14], fault [15] or leakage detection system [16] and maintenance decision of petrochemical plant [17]. They are also of special interest for subproduct quality estimation. Due to the efficiency demands and the environmental regulations, the subproducts extracted from the crude raw must meet the defined standards. A violation of such standards can result in production waste, reprocessing, energetic inefficiencies or even expensive taxes. To comply with

## 1. Introduction

the regulations, subproduct quality is currently estimated by online analysers or laboratory tests. Online analysers are physical sensors equipped with an automatic sample acquisition tool for quality analysis which results are included in the computer system of the plant. However, since the analysis is performed once the refining process ends, the use of online analysers has limitations for preserving the quality of subproducts, which delays the possible corrective actions. Besides, since online analysers can be exposed to extreme conditions, they can be subject to failure making the measurements unreliable. Laboratory tests produce the most reliable quality estimations, but the delay between the sample collection and the laboratory results is even longer than online analysers, generally of hours. In order to circumvent the limitations of the online analysers and the laboratory tests, in the digitalisation era, soft-sensors are designed and developed to complement the current instruments, enhancing the industrial systems' flexibility, adaptability and resilience, which exponentially improves productivity and economic growth, ensuring the sustainability of the industry.

In this context, this thesis focuses on automatising the design and development of new data-driven soft-sensors for real refineries use cases, specifically for measuring different subproducts quality. In order to contextualise this thesis in the current state-of-the-art, Section 1.2 surveys different data-driven soft-sensors proposed for refineries' product quality estimation. Furthermore, given the importance of data preprocessing for developing reliable soft-sensors, the existing approaches for features' processing are carefully analysed, giving place to a literature review of data preprocessing methods, specifically Feature Selection (FS), Feature Weighting (FW) and Feature Normalisation (FN). Once the related literature has been revised, the hypothesis and the goals of this thesis are presented in Section 1.3. Finally, Section 1.4 discusses and shows the significant contributions of this thesis to both Oil and Gas 4.0 and Machine Learning scopes. Figure 1.5 illustrates a high-level diagram of the key concepts considered in this thesis and the research contributions derived from this work.

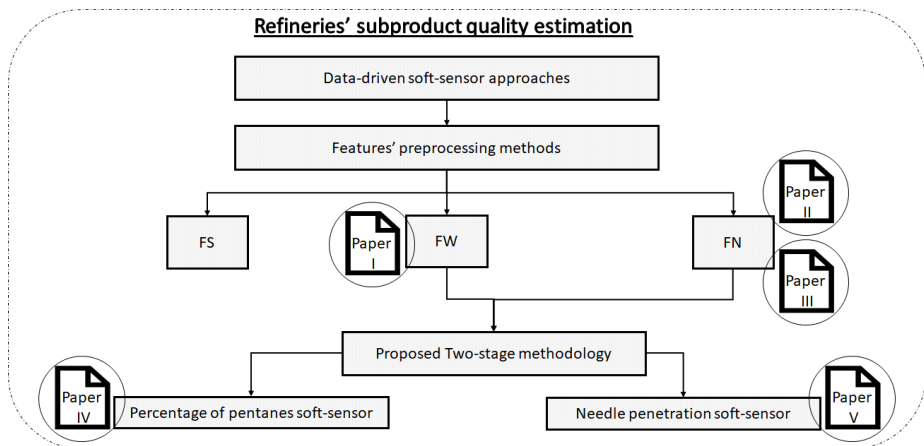


Figure 1.5: Key concepts and contributions of this thesis.

## 1.2 State-of-the-art

Refineries are complex industrial systems that transform crude oil into more valuable subproducts. One of their primary concerns is to ensure high-quality final subproducts that meet rigorous government regulations to maximise profit for commercialising them. The traditional quality control systems are online analysers and laboratory tests. However, as explained in Section 1.1, these methods present limitations that the complementary use of soft-sensors can overcome. Until the date, several soft-sensors have been developed to supplement diverse laboratory tests, such as 95% ASTM-D86 of naphtha, gas oil and kerosene [18–21], 10%, 50% and 90% boiling points of diesel [22] and the carbon decomposition rate [23], among others. Online analysers are also complemented by recently developed soft-sensors designed to estimate the octane number in gasoline in a power-former unit [24], the side-cut and overhead stream in a desihexanizer column [25], toluene estimation [26] or the butane content on the bottom of a debutanizer column [27].

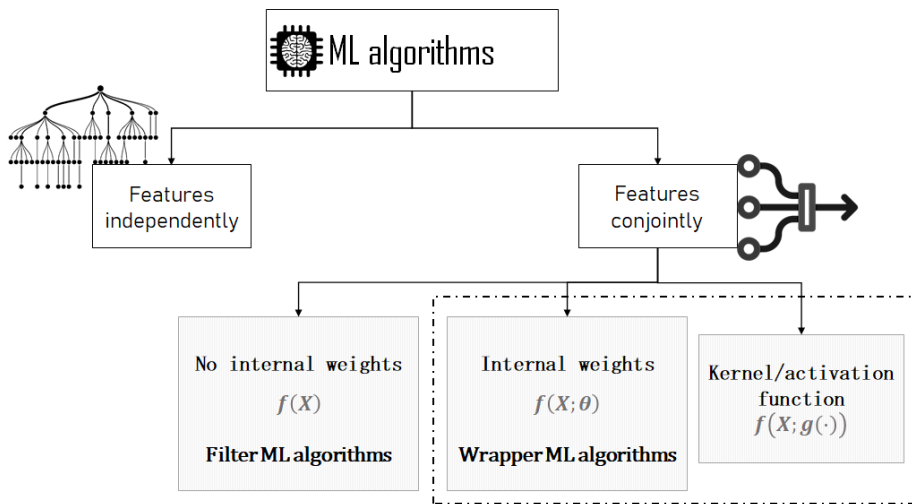


Figure 1.6: Properties of the ML algorithms in terms of the features' processing.

The revised soft-sensors are generally modelled by Partial Least Squares (PLS) or Artificial Neural Networks (ANN) based algorithms, like in [28] and [18, 19, 21, 29, 30], respectively. Approaches based on Support Vector Machines (SVM) or Decision Trees (DT) can also be found. A DT-based model can be viewed as a flowchart diagram where one feature is processed at each node. Thus, a decision tree computes the output by utilising the features of the dataset independently. In contrast, the majority of the ML-based methods consider the conjoint information of the features to estimate the output of the model. In the latter cases, contrary to some ML algorithms, such as K-Nearest Neighbours or K-means which compute the output based on the given input features (defined in Section 1.4.2.1 as *filter ML algorithms*), algorithms like PLS [31], ANN [32] or

## 1. Introduction

---

SVM [33] (defined in Section 1.4.2.2 as *wrapper ML algorithms*) present internal weights to, in a wrapper manner, adjust the contribution of the features to the model. In addition to the internal weights, algorithms like SVM or ANNs are formulated with a kernel or an activation function, respectively. Depending on the kind of function, these algorithms are able to model not only linear but also non-linear relationships between the features. Thus, these ML algorithms are widely employed for modelling use cases related to refineries due to the complexity of the non-linear phenomena involved in the operation process and the great amount of historical operational data available. Figure 1.6 depicts the described properties of the ML algorithms in terms of the above-mentioned processing approaches of features.

*However, in the literature there is no formal definition about which ML algorithm to select for a particular problem, nor which considerations must be taken.*

The proper ML algorithm selection and configuration is essential for developing reliable soft-sensors that extract hidden patterns from the data. It is so the utilisation of relevant historical information. In this case, in order to infer the subproduct quality, process information related to the particular subproduct is needed. Refineries are organised as a chain of units. In fact, several distillation columns are needed to separate each fraction and, then, the subproducts. The separation of the hydrocarbons mixture depends on their boiling point. In order to reach the desired temperature inside the distillation tower, reflux and condensers are adjusted to reach and maintain the optimal temperature. Each column is equipped with several sensors for monitoring flow, temperature and pressure. Different online analysers to control the material properties, like pH or Reid Vapour Pressure, are also common. Hence, as Figure 1.7 illustrates, several process variables are involved in the disunion of the subproducts and, consequently, several process variables must be considered in order to infer the subproducts quality. For instance, the soft-sensors proposed in [20, 34, 35] employ between 51 and 84 features for modelling the problem at hand, while authors in [22] utilise up to 143 features to estimate different diesel boiling points. Nevertheless, some authors exploit a reduced number of features for modelling the problem since a “small number of variables are easily maintainable in the industrial setting” [28]. This way, some works employ less than 10 features, directly related with the resultant subproduct quality. It must be noticed that these latter works are generally focused on estimating the subproduct quality considering the process variables of the last unit of the distilling chain, like [36–39] which infer butane concentration on the debutanizer bottom based on sensors’ data from such column. Of course, the decision about which features to include for modelling the problem must be guided by expert knowledge. As previously mentioned, the features must be informative enough for describing the process, avoiding redundancy that might disturb the modelling process and increase the computational cost.



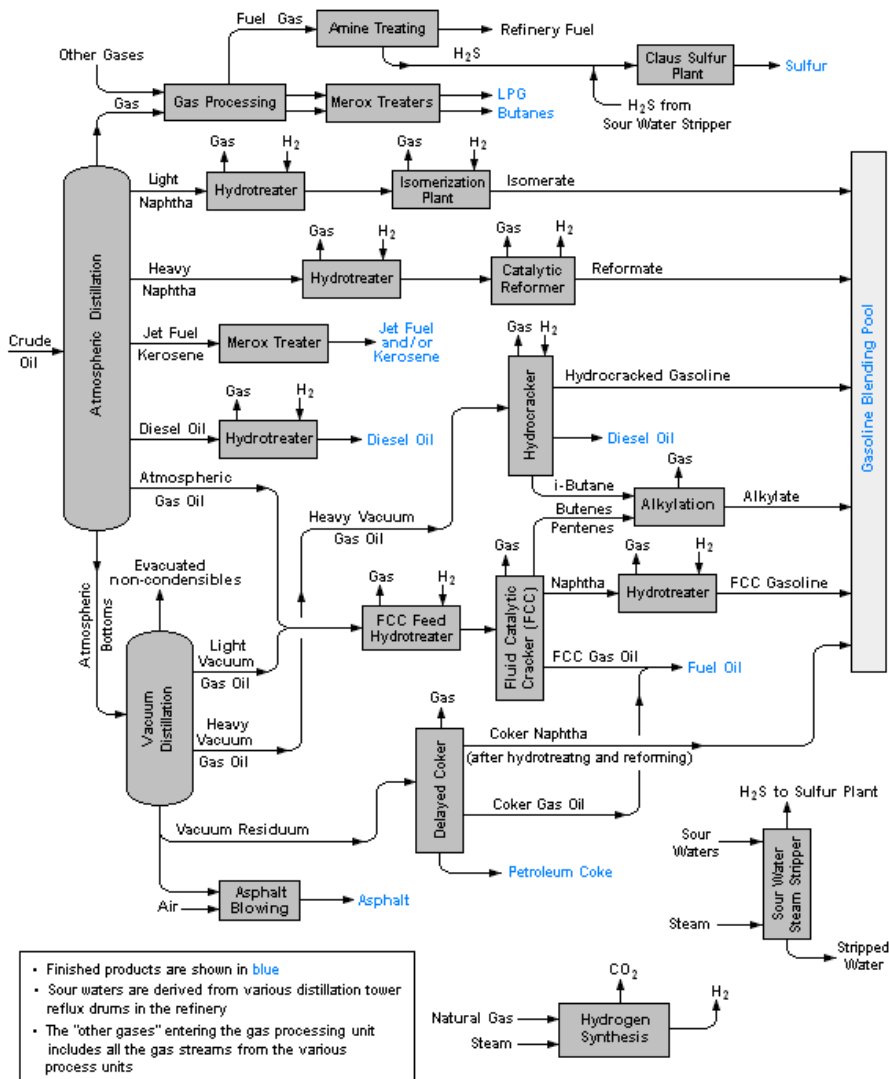


Figure 1.7: Refinery flow chart. (Source: [40])

The utilisation of excessive features may also produce over-fitting [41], corrupting the generalisation ability of the model, especially when the number of training samples is insufficient. In order to circumvent it, authors in [18, 24, 25, 28, 42, 43] employ Feature Selection (FS) to choose the most representative features of the dataset for modelling the problem at hand. It must be noticed that these works employ less than 2400 samples for training the models. However, the physical sensors used at the refineries commonly monitor the operational variables continuously. Assuming a sampling rate of 10 minutes for the physical

## 1. Introduction

---

sensors, there would be 1008 samples per week and 4320 per month. Thus, depending on the quantity and the variability of the training data, a reduced number of features may prevent the algorithm from capturing hidden patterns from the data, causing under-fitting [44]. In this sense, beyond FS, authors in [27] propose a soft-sensor modelled by a variable-wise weighted Auto Encoder, which includes a Feature Weighting method. It assigns to each feature a weight representative of its relevance for estimating the output, determined by the correlation between the feature and the output data.

*There is no explicit methodology proposed in the literature to guide the criteria for inferring the proper number of features depending on the quantity and characteristics of the dataset. While including irrelevant or redundant features disturbs the model's performance and its generalisation ability, selecting too little features can affect the algorithms ability for capturing hidden patterns in the data.*

Lastly, despite the mentioned algorithms ability to internally transform the features' influence on the model, it must be noticed that prior to the ML algorithm utilisation, dataset features are usually scaled by FN methods such as Min-Max normalisation [19, 23, 34], or z-score approach [21, 22, 45] to avoid features' magnitude difference to affect the modelling process.

As observed along the data-driven soft-sensors literature revision, in addition to the proper ML algorithm selection, the features' preprocessing strategy to enhance the model's performance is of great importance. Therefore, in the following, a literature review about the highlighted features' preprocessing methods (Figure 1.8) is conducted.

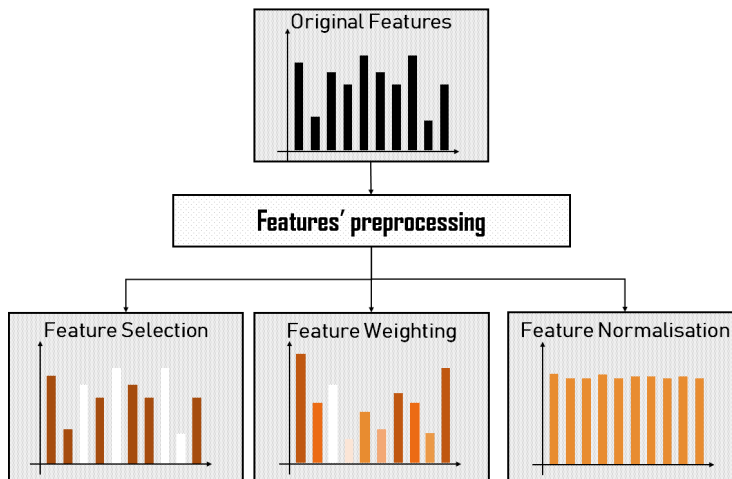


Figure 1.8: Features' preprocessing approaches. The colours represent the weights' value associated to each feature, from white which symbolises zero, to dark orange associated to the weight value equal to one.

The literature illustrates that, in addition to the ML algorithm, some authors focus on the preprocessing stage for data-driven soft-sensors development to enhance the reliability of the models. As above-mentioned, features' preprocessing is essential to reflect features' real problem description ability on the model.

The first feature preprocessing approach previously remarked is Feature Selection (FS). FS is utilised to choose a subset of features of the input dataset by assigning a weight equal to 1 to the relevant features. On the contrary, the discarded non-relevant features for modelling the problem are multiplied by 0. Although unsupervised FS methods exist, the most widely utilised approaches in soft-sensors literature are the supervised ones. As observed in the works [18, 45], FS is conducted by, or takes into consideration, expert knowledge. However, there are different mathematical FS approaches proposed to rank and select the most relevant features [46–50]. Figure 1.9 presents a taxonomy of the supervised FS methods.

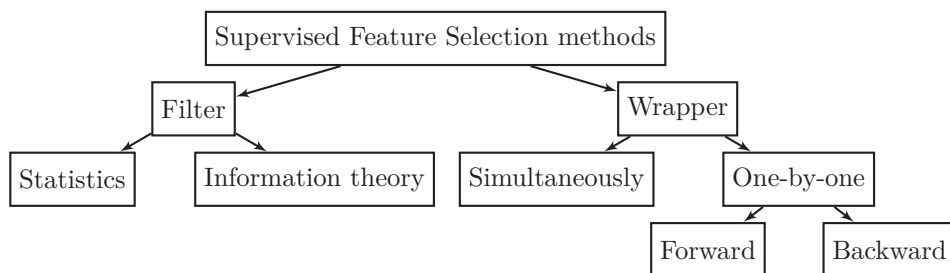


Figure 1.9: Taxonomy of the supervised FS methods.

Regarding the reviewed research papers about quality monitoring soft-sensors, an extended FS method is based on Pearson correlation coefficient [20, 25, 43]. In this approach the correlation between each feature of the input space and the output is estimated, and the features with the highest Pearson correlation coefficient are selected. Another method presented in [28] uses an approach derived from a Genetic Algorithm (GA) to study the features' relevance for modelling the problem based on PLS regression in terms of Root Mean Squared Error Cross-Validation. The difference of [28] with respect to the other mentioned FS works is the technique applied to choose the features. FS based on Pearson correlation suggests which features to select without considering the ML algorithm used for modelling the soft-sensor. This approach is known as filter. In contrast, [28] employs a wrapper method for FS. Both methods are considered in [51] where authors revise the utilised FS approach to develop industrial soft-sensors. Regarding the filter methods, those derived from correlation analysis or Mutual Information are the most extended ones. The computation results quantify each feature's importance for representing the input space. Also, they can be viewed as a ranking of the features' relevance. The main challenge of filter methods is to define the threshold from which features are discarded. Feature Selection wrapper methods, in contrast, select, in an iterative manner, the subset of features that obtains the highest model's performance. Some wrapper

## 1. Introduction

---

methods, like those based on GA, search for the best subset of features by means of evaluating at each iteration a different configuration of the selection parameters, over all the features simultaneously, until the optimal one is reached. In contrast, other approaches conduct a forward or backward search considering the features one-by-one for its selection. This means that, one feature is added or deleted from the feature space at each iteration, respectively. Then, the model's performance is computed. Finally, the subset with the best trade-off between the number of features and performance reached is selected. Although discarding or adding one-by-one the features allows to quantify their influence on the model's performance, note that the order in which the features are removed/included should be also considered.

In contrast to the filter methods, the wrapper ones obtain the highest performance at the cost of computational efficiency.

*However, wrapper or filter FS methods assign a weight equal to 1 to the selected features. Hence, the assignment of the same weight implicitly means that the selected features are equally important for the model, which may not be realistic.*

A generalisation of FS is Feature Weighting (FW) [52–54]. FW assigns different weights to each feature according to its relative relevance for representing the output. Traditionally, the weights range from 0 to 1 so that the weights' sum equals 1. The higher the features' relevance, the higher the feature weight value resulting from the computing. Moreover, since feature weights can take value 0, FW implicitly conducts feature selection discarding the non-relevant features.

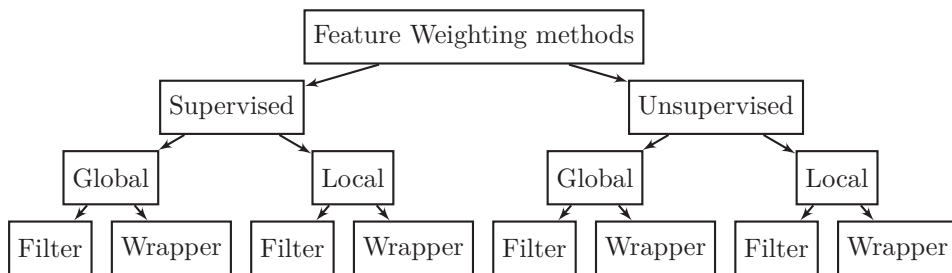


Figure 1.10: Taxonomy of the FW methods. (Source: Paper I)

In the literature, there are several Feature Weighting methods. Figure 1.10 illustrates a general classification of the FW approaches. Some FW methods utilise label information to estimate the features' relevance for modelling the problem, like FS methods. However, FW also computes the weights of the features in an unsupervised fashion based on features' distribution properties. Then, supervised and unsupervised FW methods can be found.

Among the supervised FW methods, similar to FS, filter or wrapper approaches can be differentiated. Filter methods consider statistical or

information theory-based approaches to estimate the relevance of the features. Among the statistical methods, those based on correlation analysis are commonly used [55]. Still, other statistical approaches like  $\chi^2$  or Fisher score (F-score) [56] are utilised to quantify the features' importance. Finally, regarding the information theory-based FW methods, they commonly employ Mutual Information [57–60].

*The filter methods are computationally cheaper and independent from the ML algorithm. However, the resultant weights are dependent on the selection of the quantification method, which is not straightforward.*

Concerning the wrapper FW methods, the most extended practice is the use of a Genetic Algorithm (GA) for the weights' calculation in order to improve the model's performance in terms of a given metric [61–63]. In addition to GA, Evolutionary algorithms (EA) [64], Differential Evolution (DE) algorithms [65] or Particle Swarm Optimisation (PSO) [66] can be found in the literature.

*The wrapper methods are performance oriented, but they focus on reaching the highest performance for the particular ML algorithm and involve high computational cost.*

The mentioned FW approaches calculate a weight per feature. These methods are known as global. Global approaches consider that a given feature presents the same importance level along all its samples when modelling a problem. Local methods consider that a subset of samples of a given feature may present a different importance level when modelling a problem. Local FW methods can also be differentiated between filter and wrapper. However, few supervised filter local approaches are found in the literature. Authors in [67] reformulate Gini and Entropy diversity index formulation to estimate local weights based on information theory techniques. Other supervised filter local approaches [68, 69] are based on the RELIEF algorithm, which is used in classification problems and computes class-dependant weight to maximise the class separability. Regarding the wrapper local FW methods with supervised learning of the weights, similar to the global methods, some approaches are based on Differential Evolution [70], or Evolutionary approaches [71]. However, most of the works reformulate the objective function of the problem to include the local weights, which are adjusted by Gradient Descent [72–74].

*Local methods may be more realistic but defining the optimal number of weights per feature and the characterisation of the samples for discriminating among the different weights is not straightforward.*

The supervised FW methods can be characterised by filter or wrapper and global or local. The same classification can be made for the unsupervised FW methods. Due to the absence of information about the real labels, unsupervised

## 1. Introduction

---

FW methods estimate the weights by means of specific characteristics and relations between the features or by the obtained cluster structure. The filter unsupervised FW methods generally apply first a clustering algorithm to group the data samples into different partitions, and then, based on the obtained cluster structure, the feature weights are calculated. Some examples can be found in [75, 76], where the feature weights are calculated as the ratios of the means of the features to the centroids. In contrast, in wrapper FW approaches the weighting strategy is commonly embedded into the clustering objective function, and iteratively, both the cluster structure and the feature weights are obtained. One of the first works that embrace this approach is [77] in which the authors adapt the K-means algorithm by means of including the feature weights into the formula giving place to the Weighted K-Means (W-k-means). The weights optimisation is done by partial optimisation. This idea has been widely employed by other authors, which modified the weights adjustment strategy, like in [78].

*Unsupervised global FW methods are generally based on clustering K-means algorithm to characterise the dataset and then measure the features' contribution to such characterisation. However, K-means is based on samples dispersion, which may not be the main property that describes the use case at hand.*

Regarding the local weights, similar to FS, their use in filter strategies is not common. However, some exciting research works that use unsupervised filter local FW methods are found. Among them, [79] proposes the Weighted Dynamic Time Warping (WDTW), which includes weights into the formulation where the phase difference between a reference point and the tested one is considered, and large phase differences are penalised in order to prevent minimum distance distortion caused by outliers. Later, the same authors in [80] proposed WDTW as a kernel function into a multi-class Support Vector Machine for time series classification. Additionally, filter approaches for unsupervised local feature weights have been employed for static data with missing values [81, 82].

Given the clustering ground of unsupervised FW methods, the use of local wrapper weights is most extended compared to the supervised FW methods. In this case, there are as many local weights as clusters. Similar to the global methods, these approaches include local weights into the problem formulation, and the weights are adjusted iteratively according to the prior clustering classification [83]. Other works include Evolutionary algorithms [84, 85] to estimate such weights.

*Unsupervised local FW methods derived from clustering are dependant on the proper selection of the number of clusters. Those based on the phase difference between samples can be more representative of continuous industrial processes. However, they are still not common and the time delay between operational parameters adjustment and its impact on the industrial final product is not always known.*

As it can be observed in the literature review, there is a wide range of FW methods. However, each approach presents advantages and limitations. Given that unsupervised approaches do not consider the labels to estimate the features' relevance, the estimated weights may not represent the importance of the features for modelling the use case. Regarding the filter methods, the selection of the relationship between the input and the output is crucial to estimate a representative weight that properly guides the training of the ML algorithm. Besides, filter methods are not expressly oriented to the model's performance maximisation, since they do not consider the ML algorithm.

With respect to the wrapper methods, due to their iterative nature, they require more computational resources. Besides, since the weights are estimated based on the particularly used samples and labels, the generalisation ability of the model may be compromised. Finally, regarding local FW methods, they are expected to capture more realistically the importance of each feature and each subset of samples for modelling the use case, especially in refining cases where different production modes are used, or when the phase difference between the samples may influence the output estimation. However, there is a memory and computational cost associated with this approach. Besides, local weights may disturb the output estimation when the subset of samples differentiation is not straightforward, or the temporal relationships are dynamic or evolve. Therefore, for selecting the FW method, a thorough reflection must be done.

Furthermore, as Figure 1.11 illustrates, in the case of FS and FW, it must be noticed that the related weights take values at most equal to 1. Then, the differences between the original magnitudes of the features can result in both FS and FW approaches in an over-influence of a set of features on the ML algorithm's metric. Thus, in FS, features with higher magnitude can dominate the calculations. On the other hand, the magnitude differences can disturb the influence of the FW-based weights on transforming the space. For all these reasons, the third preprocessing technique highlighted in the state-of-the-art is Feature Normalisation (FN).

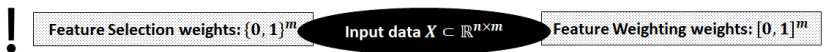


Figure 1.11: Scale difference between feature selection or weighting weights and input data.

As observed in the reviewed literature about soft-sensors, FN is widely applied. It is claimed to be particularly useful in statistical learning methods [86], since it avoids magnitude differences among the features. Besides, FN is thought to equalise the features' contribution to the ML algorithm calculations. Thus, it is considered a necessary step especially for methods that use distance measures [87] and for Neural Networks-based approaches. Such is the importance of FN that, for example, authors in [88] utilise FN in conjunction with other data

## 1. Introduction

---

preprocessing techniques explicitly to enhance the performance of the proposed machine-fault diagnosis system.

Among the several existing FN techniques, Min-Max normalisation [89–92] and Standardisation [51, 91, 93–96] are the most popular ones. However, despite the general use of FN for the development of data-driven methods, no general agreement about which particular method to choose for a particular scenario is settled. For instance, it is common to scale features’ values within the range  $[0, 1]$  by Min-Max normalisation in Neural Network-based approaches. However, authors in [21, 45, 97] utilise Standardisation in the preprocessing stage.

*Despite the common utilisation of FN, in the literature, the reasoning about the selection of a particular FN method is not generally included.*

In the literature, some works can be found that empirically aim at investigating the most suitable FN technique for a particular problem. For instance, authors in [98] study the effect of eight FN methods on the recovery cluster structure over four agglomerate clustering methods applied on synthetic data. Their results reveal that the range-based normalisation methods present consistently superior recovery of the underlying cluster structure and that the traditional z-score is the less effective on the analysed problem. Regarding clustering approaches, a similar work can be found in [99] where authors conduct an empirical comparison of the effect of six normalisation methods on clustering results. In this case, real data from the social sciences field is employed. In the chemical field also experimental analysis of the normalisation effect on two clustering algorithms can be found in [100]. In addition to the analysis over clustering algorithms, [101] compares different linear and non-linear normalisation methods and their impact on a Deep Recurrent Neural Network for predicting different Stock Exchange closing indexes. Besides, particular interest has been put on the effect of FN for speech recognition problems [102, 103].

Recently, authors in [104] present a study aiming to investigate the impact of 14 data normalisation methods on K-Nearest Neighbours (K-NN) algorithm classification performance, considering FS and FW preprocessing techniques. Specifically, authors employ the Ant Colony optimisation algorithm to, in a wrapper manner, find the optimal feature selection configuration and the optimal feature weights that maximise the K-NN accuracy. Based on their results, authors conclude that data normalisation affects the datasets’ features properties, which changes the feature relevance. Besides, regarding the applied normalisation methods and the performance results, it is stated that median-based measures and minimum and maximum statistical measures should not be used for normalising data, being the mean and standard deviation measures the most recommendable ones to transform the original dataset.

*In contrast to the extended thought that FN equalises the features’ contribution, the literature review shows that the selection of the FN method alters the model’s output.*



Despite the general assumption of the FN importance prior to the algorithm's training and the interest in analysing the FN impact on the models' performance, the experimental researches diverge on their conclusion. Therefore, until the date, the selection of the FN approach for scaling the features still depends on the analyst experience and criteria.

*In the literature, there is no agreement about the proper FN method for a given problem. Besides, since the FN method selection influence on the ML algorithms has been experimentally analysed, the results directly depend on the properties of the input datasets.*

To summarise, from the literature review, the following conclusions that motivate the presented thesis are drawn:

The data-driven soft-sensors demonstrate to be a promising complement to the online analysers and laboratory tests to monitor the subproducts quality continuously in the refinery industry. However, no roadmap for selecting the proper ML algorithm for modelling a given use case was still proposed. Until the date, such choice is conducted by the data analyst, conditioned by their expertise and intuition about the proper ML algorithm selection and configuration for discovering the hidden patterns. In addition to the ML algorithm selection, the features' preprocessing stage is difficult and time-consuming. Despite its importance for creating reliable soft-sensors, the features' preprocessing influence on the model's performance has not been formally analysed. In this sense, the relevance of selecting informative features for properly modelling the problem and saving computational resources is commonly acknowledged.

Since the primary goal of ML algorithms is to discover hidden patterns in the historical data, among the reviewed feature preprocessing methods, FW, which considers all the features suggested by the expert knowledge and implicitly discards or reduces the less informative features' contribution to the model will be considered in this thesis. Furthermore, since the feature weights generally take values within  $[0, 1]$  and the operational variables of the refinery often present significant magnitude differences, FN is also considered in this thesis. However, due to the general utilisation of FN and the lack of formal analysis about its influence on the model's performance, it is necessary to understand and formalise its impact when transforming the input features.

## 1.3 Hypotheses and objectives

This Section thoroughly describes the hypotheses and objectives derived from the literature review.

### 1.3.1 Hypotheses

The general hypothesis of this thesis is that it is possible to:

**H0 design and develop a methodology that selects the proper feature preprocessing and ML algorithm for creating reliable soft-sensors in the context of Oil and Gas 4.0.**

The proposed methodology is focused on analysing and selecting the proper features' preprocessing methods to enhance the ability of ML algorithms for creating reliable soft-sensors. In this line, particular hypothesis regarding ML, FW and FN are established in this thesis:

**H1** ML algorithms: It is assumed that not all the features are equally important for modelling ML based soft-sensors for the refinery industry.

Hence, features' preprocessing is an essential and a relevant part of the Machine Learning-based model. More concretely, regarding the features' contribution to the model, particular hypotheses are established for both FW and FN:

**H2** Feature Weighting: The proper representation of features' relevance is essential to guide the ML algorithm towards the maximum model's performance.

**H3** Feature Normalisation: FN does not equalise the features' contribution to the model. Thus, this work assumes that it is possible to formalise a methodology for quantifying the FN impact on transforming the features, and hence, on model's performance.

Once stated the general and particular hypothesis, next the objectives of this thesis are described.

### 1.3.2 Objectives

The main objective of this thesis is to:

**O0 Automate data-driven soft-sensors development for refinery's subproduct quality estimation by proposing a methodology that considers different ML algorithms and features' preprocessing techniques, specifically FW and FN.**

In order to reach the main objective of this thesis, first, specific objectives in the scope of Data Analysis are established for the design and development of reliable soft-sensors, in particular with regards to

**O1** Feature Weighting:

**O1.1** Analyse the FW methods from both the mathematical formulation and the practical use on industrial use cases perspectives considering:

- The characteristics of the dataset.
- The practical orientation of the problem at hand.

**O2.2** Validate that FW can improve the soft-sensor performance.

**O2** Feature Normalisation:

**O1.1** Prove that the selection of a particular FN method influences the features' contribution to the model and consequently the model's performance, considering:

- Filter ML algorithms: ML algorithms without internal weights in their formulation.
- Wrapper ML algorithms: ML algorithms with internal weights in their formulation.

**O2.2** Propose a methodology to quantify the influence of the FN method selection on the features' contribution to the model.

**O3** ML algorithms: Select and apply a set of candidate ML algorithms and training methodologies to avoid over and under fitting.

Once reached the specific goals regarding the Machine Learning stages, next the objectives of this thesis in the context of Oil and Gas 4.0 are to:

**O4** create a flexible methodology for soft-sensors design and development that selects, for the problem at hand, the proper features' preprocessing method and the proper ML algorithm.

**O5** validate the designed methodology with a real use case on laboratory tests in the refinery industry.

**O6** validate the designed methodology with a real use case on online analysers in the refinery industry.

Table 1.1 relates the presented objectives and the papers in which they have been evolved.

Objectives	Associated Paper
<b>O0</b>	Paper IV & Paper V
<b>O1</b>	<b>O1.1</b> <b>O2.2</b> Paper I
<b>O2</b>	<b>O1.1</b> <b>O2.2</b> Paper II & Paper III
<b>O3</b>	Paper IV & Paper V
<b>O4</b>	Paper IV & Paper V
<b>O5</b>	Paper IV
<b>O6</b>	Paper V

Table 1.1: Association between the objectives of the thesis and the papers in which they are addressed.

## 1.4 Results and discussion

This Section outlines the results of this thesis in terms of the objectives mentioned above. These results are of theoretical and practical nature.

### 1.4.1 Feature Weighting

First, given the importance of features' preprocessing to create reliable soft-sensors, a thorough revision of the literature concerning FW methods has been conducted. As mentioned in Section 1.2, FW methods aim at transforming the features of the dataset in order to represent their proportional relevance at estimating the output and thus, enhance the performance of the model. There are plenty of FW methods in the literature. In fact, until 2020, up to 16200 works could be found on Google Scholar with the exact words "Feature Weighting". However, among those works, only five surveys can be found and as Table 1.2 shows, these focus on a subset of FW methods. Therefore, it was deemed necessary a new review work to establish a general taxonomy for the FW methods based on their formulation and present a concise review about FW methods concerning different aspects.

Review work	Reviewed FW methods
[52, 57]	FW methods for K-NN-based algorithms
[105]	FW methods used in case-based reasoning problems
[53]	FW methods for K-means-based algorithms
[54]	Wrapper FW methods for clustering methods

Table 1.2: Summary of surveys about FW methods and their main focus.

In this line, first a general taxonomy which classifies the FW methods according to the learning approach (supervised or unsupervised), the way the weights are applied (global or local) and the estimation strategy (filter or wrapper) is described in **Section 2 of Paper I** and graphically represented in Figure 1.10 (**Figure 3 of Paper I**). In Figure 1.10, it is observed that the three taxonomy levels define each FW method. In order to clarify how each level is integrated into the FW methods, **Figures 1 and 2 from Paper I** illustrate a flow chart of filter (Figure 1.12) and wrapper (Figure 1.13) FW approaches, respectively, considering whether supervised/unsupervised and global/local weights are estimated.

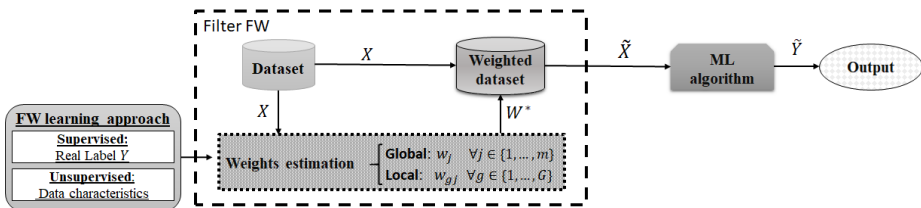


Figure 1.12: Flow chart of filter FW approaches. (Source: Paper I)

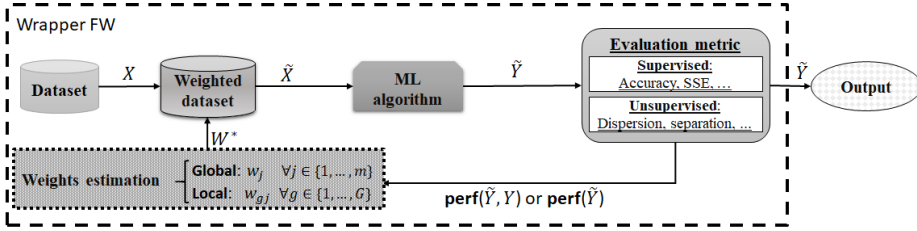


Figure 1.13: Flow chart of wrapper FW approaches. (Source: Paper I)

Once defined the general taxonomy for classifying the FW methods, a literature revision is conducted with special attention on 1) the FW technique description, 2) the input features of the problem, 3) the field of application, 4) the metric utilised for the evaluation of the ML-based model and 5) the comparison with related works. The revised works are collected in **Table 1 from Paper I** specifying the information of these five categories. In addition, the reviewed FW techniques are grouped and described in **Section 3 of Paper I** according to the proposed taxonomy. At the same time, each subsection includes the survey about the FW approaches and a pseudo-algorithm representative of the described FW methods.

As a result of the revision and comparison of the works, **Sections 3.1.3, 3.2.3, 4.1, 4.2 and 4.3 of Paper I** collect concluding remarks, advantages and limitations of the FW methods regarding their formulation with a special focus on their impact on ML algorithms-based solutions. Finally, based on these conclusions and with practical purposes, **Section 4.4 in Paper I** provides a recommendation guide for the optimal selection of the FW approach according to the characteristics and objectives of the problem at hand, with regards to 1) labels' availability, 2) high-dimensional dataset, 3) dimensionality reduction, 4) dataset understanding, 5) features' contribution estimation, 6) missing values, 7) imbalanced dataset, 8) outliers, 9) noise, 10) interpretability, 11) condition-based problems, 12) temporal dependency, 13) model's performance maximisation, 14) semi-supervised learning and 15) online learning. Figure 1.14 (**Figure 4 in Paper I**) depicts the recommendations into the different levels of the proposed taxonomy.

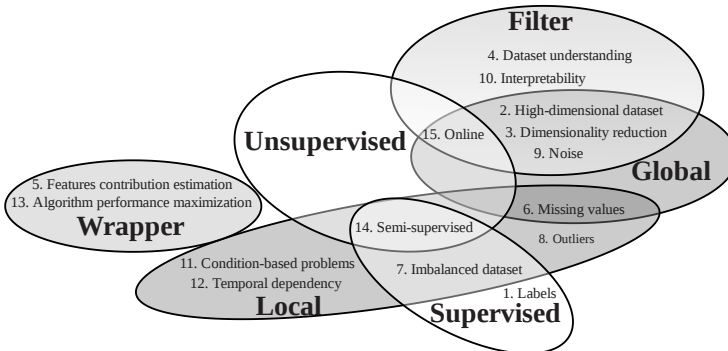


Figure 1.14: Recommendation for the proper FW method selection. (Source: Paper I)

## 1. Introduction

---

As a result of the former analysis and the recommendations proposed in Paper I, since in the real use cases of this thesis the measures of subproduct quality are continuously registered, supervised FW methods will be applied. Likewise, given the high-dimensionality of the refinery use cases, the possible noisy data, the intention of enhancing the interpretability of the models and, in future work, of including into the proposed soft-sensors online learning, global filter FW methods are considered for designing and developing soft-sensors for subproduct quality estimation in the refinery industry.

### 1.4.2 Feature Normalisation

Next, as mentioned in Sections 1.2 and 1.3, this thesis claims that FN does not equalise the features' contribution to the model. Actually, since the works revised from the literature study the impact of FN in an experimental manner, the conclusions are not extensible to new datasets. Then, this work formally analyses the impact of FN on transforming the dataset. This thesis will focus on FN methods that linearly transform the features based on statistical factors. In this line, this work considers two hypotheses, described in **Section 3.1 of Paper IV**:

- Hypothesis 1: Different FN methods transform a given dataset differently.
- Hypothesis 2: A given FN method transforms each feature of a dataset differently.

In order to formally study the FN impact on transforming the features of the dataset, first of all, a formulation of statistical-based FN methods is presented in **Equations 1 and 3 of Paper II and Paper III**, respectively.

$$\tilde{X}^{Norm} = \frac{X - \mathbf{pos}(\mathbf{X})}{\mathbf{dis}(\mathbf{X})} \quad (1.1)$$

In Equation 1.1,  $\mathbf{pos}(\mathbf{X})$  refers to the position or central tendency vector that centres the values of the feature, and  $\mathbf{dis}(\mathbf{X})$  corresponds to the dispersion statistical vector, which scales the features.

With the aim of highlighting the magnitude of each feature, and thus compare the magnitude differences among the features, *decimal notation* is defined in Equation 1.2 (**Equations 2 and 4 of Paper II and Paper III**, respectively):

$$x_{ij} = \mathit{sign}(x_{ij}) \cdot 0.d_1d_2d_3 \dots \cdot 10^{n_j} = \widehat{x}_{ij} \cdot 10^{n_j} \quad (1.2)$$

where  $d_1, d_2, \dots \in \{0, \dots, 9\}$  and  $n_j \in \mathbb{Z}$  in such a way that,  $\forall j$ ,  $|n_j|$  is the minimum number which fulfils  $X_j = \widehat{X}_j \cdot 10^{n_j}$  and  $\max\{|\widehat{X}_j|\} < 1$ . Then,  $10^{n_j}$  represents the  $j$ -th feature's magnitude factor. Thus, when comparing  $n_j \forall j$  the original magnitude differences among the features can be analysed.

Moreover, when applying decimal notation in the formulation of the statistical-based FN methods it can be concluded that, with FN, the magnitude differences

among the features disappears, as shown in Equation 1.3 (**Equations 3 and 5 of Paper II and Paper III**, respectively).

$$\widetilde{X}_j^{Norm} = \frac{\widehat{X}_j \cdot 10^{n_j} - pos(\widehat{X}_j) \cdot 10^{n_j}}{dis(\widehat{X}_j) \cdot 10^{n_j}} = \frac{\widehat{X}_j - pos(\widehat{X}_j)}{dis(\widehat{X}_j)} \quad (1.3)$$

However, the de-magnified position and dispersion vectors still transform the features of the dataset. As mentioned before,  $pos(\widehat{X}_j)$  vector only translates the values of the features. But  $dis(\widehat{X}_j)$  expands or compresses the features of the dataset in order to fulfil that  $dis(\widehat{X}_j) = 1 \forall j$ , as demonstrated in Equation 1.4 (**Equation 4 of Paper II**).

$$dis(\widetilde{X}_j^{Norm}) = dis\left(\frac{\widehat{X}_j}{dis(\widehat{X}_j)} - \frac{pos(\widehat{X}_j)}{dis(\widehat{X}_j)}\right) = dis\left(\frac{\widehat{X}_j}{dis(\widehat{X}_j)}\right) = \frac{dis(\widehat{X}_j)}{dis(\widehat{X}_j)} = 1 \quad (1.4)$$

Thus, it is demonstrated that FN does not equalise the features' contribution but the features' dispersion in terms of the statistic of dispersion applied by the FN approach. Table 1.3 (**Table 1 in Paper II**) collects different statistical-based FN methods.

Normalisation method	pos(X)	dis(X)
Standardisation (ST) [98, 104]	$\bar{X}$	$\frac{\sigma_X}{\bar{X}}$
Variable Stability Scaling [106, 107]	$\bar{X}$	$\frac{\sigma_X}{\bar{X}}$
Pareto scaling [108, 109]	$\bar{X}$	$\sqrt{\sigma_X}$
Min-Max normalisation (MM) [98, 104]	min(X)	range(X)
Range scaling [110, 111]	$\bar{X}$	range(X)
Unitisation [98]	0	range(X)
Robust scaler [112, 113]	Me(X)	IQR(X)
MAD normalisation [114, 115]	Me(X)	mad(X)

Table 1.3: Linear-based normalisation methods. (Source: Paper II)

Min-Max normalisation, Range scaling and Unitisation utilise the same dispersion factor to normalise the features. Then, in these cases, FN transforms the features equally. However, for the rest of cases in Table 1.3, FN transforms the features to fulfil Equation 1.4 in terms of different dispersion statistics. Consequently, since each FN method is based on different dispersion statistics, each FN will transform a given dataset differently. Thus, **hypothesis 1 of Paper IV** is validated. Besides, since each feature presents a different dispersion value, each feature will be differently transformed in order to fulfil Equation 1.4. So, **hypothesis 2 of Paper IV** is also validated. And, consequently, hypothesis **H3** of Section 1.3.1.

Therefore, it is demonstrated that FN does not equalise the features' contribution to the model but equalise their dispersion based on the dispersion factor utilised by the selected approach. Consequently, FN can be viewed as a particular case of unsupervised FW where the inverse of the dispersion

## 1. Introduction

factors acts as feature weights  $w^{Norm}$ . In this sense, it is possible to extract information about the impact of FN when transforming the dataset by analysing the dispersion factors. Thus, in **Section VI.B.1 Paper III** the dispersion weights obtained from a set of FN methods for four datasets from the UCI repository [116] are analysed in order to infer similarities/dissimilarities between different FN methods when transforming the dataset. Intending to quantify the FN impact when transforming the features of a given dataset, in **Section 2.4 of Paper II** a new metric referred to as *Normalisation weight* is presented for measuring the over or under-influence of each feature resultant from the application of the dispersion factor. Thus, this metric symbolises how each feature is transformed by a FN method.

In fact, as it can be observed in **Table 7 or Figure 6 of Paper II** (Figure 1.15) with the Normalisation weight, it can be estimated a priori the degree of expansion/compression applied to each feature by the dispersion factor of a particular FN method.

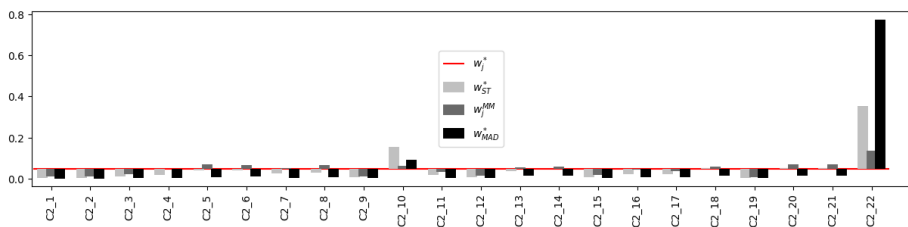


Figure 1.15: Difference between the ideal weights  $w^*$  and those assigned by each normalisation method  $w^{norm}$ . (Source: Paper II)

Once demonstrated that FN, and precisely the dispersion factor, transforms the features of a given dataset differently, it is expected that FN affects the features' contribution to the model and, hence, its performance. Consequently, in the following, the influence of ML algorithm-based models is studied. As specified in Figure 1.6 of Section 1.2, according to the properties of the ML algorithms, two kinds of methods can be distinguished in terms of features' contribution: 1) filter ML algorithms with no internal weights which directly operate over the features introduced in the algorithm, or 2) wrapper ML algorithms which include into their formulation internal weights that adjust the contribution of the features based on the algorithm's performance measure.

For filter ML algorithms, it is expected that, since no internal transformation of the features is performed, the influence of FN when transforming the features will impact on the performance of the models. In fact, most of the experimental analysis found in the literature about the impact of FN on the models' performance is conducted over this kind of algorithms, like in [98–100, 104]. However, it should be noticed that, out of this thesis, no theoretical investigation has been carried out in this field. On the other hand, regarding the wrapper ML algorithms in terms of features' contribution, due to the internal



weights, it is thought that any previous contribution of the features can be altered and avoided by the tuning phase of the ML algorithm. Nevertheless, the hypothesis **H3** of this thesis also considers that FN influences the features' contribution to the wrapper ML algorithm-based models, and hence the models' performance.

The research conducted on the filter and wrapper ML algorithms is described below in Subsections 1.4.2.1 and 1.4.2.2, respectively.

### 1.4.2.1 Filter ML algorithms

Concerning the filter ML algorithms in terms of features' contribution, the most representative ones are those that estimate the output based on the distance between the samples.

A distance function measures how far two elements are from each other. The most commonly distance function used in ML algorithms is the Euclidean distance, which calculates the closeness degree between  $\mathbf{x}, \mathbf{y} \in \mathbb{R}^m$  as defined in **Equation 5 of Paper II** as

$$d^2(\mathbf{x}, \mathbf{y}) = \sum_{j=1}^m (x_j - y_j)^2 \quad (1.5)$$

By utilising the decimal notation previously defined into the formulation of the Euclidean distance, as Equation 1.6 (**Equation 6 of Paper II**) shows, the magnitude differences between the features directly influences the distance computation.

$$d_E^2(\mathbf{x}, \mathbf{y}) = \sum_{j=1}^m ((\hat{x}_j - \hat{y}_j) \cdot 10^{n_j})^2 \quad (1.6)$$

That is why the application of FN before the distance calculation is traditionally recommended. When including decimal notation and the formulation of FN for defining the Euclidean distance between two normalised samples, it is observed in Equation 1.7 (**Equation 7 of Paper II**) that the magnitude factor disappears, and by the definition of decimal notation, the numerator of each component  $\hat{x}_j - \hat{y}_j$  takes values in  $(-2, 2)$ . Nevertheless, due to the dispersion factor, each term of the sum presents a different influence on the Euclidean distance computation.

$$d_E^2(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}) = \sum_{j=1}^m \left( \frac{(\hat{x}_j - \hat{y}_j) \cdot 10^{n_j}}{\text{dis}(\hat{X}_j) \cdot 10^{n_j}} \right)^2 = \sum_{j=1}^m \left( \frac{(\hat{x}_j - \hat{y}_j)}{\text{dis}(\hat{X}_j)} \right)^2 \quad (1.7)$$

Taking advantage of the formulation of FN methods contributed in this thesis, by comparing Equation 1.6 with the weighted Euclidean distance

$$wd_E^2(\mathbf{x}, \mathbf{y}) = \sum_{j=1}^m w^2 \cdot ((\hat{x}_j - \hat{y}_j) \cdot 10^{n_j})^2 \quad (1.8)$$

## 1. Introduction

---

it is clear that the Euclidean distance between two normalised samples is equivalent to Equation 1.8 (**Equation 9 in Paper II**) when the weights are computed as the inverse of the dispersion factors. Thus, it can be concluded that FN influences the features' contribution to the Euclidean distance. Moreover, if a subset of features present significantly lower dispersion than the resting ones, as Equation 1.9 (**Equation 8 in Paper II**) illustrates, such subset can dominate the distance results.

$$d_E(\tilde{\mathbf{x}}, \tilde{\mathbf{y}})^2 = \sum_{s \in S} \left( \frac{(\hat{x}_s - \hat{y}_s)}{\text{dis}(\hat{X}_s)} \right)^2 + \sum_{h \in H} \left( \frac{(\hat{x}_h - \hat{y}_h)}{\text{dis}(\hat{X}_h)} \right)^2 \approx \sum_{s \in S} \left( \frac{(\hat{x}_s - \hat{y}_s)}{\text{dis}(\hat{X}_s)} \right)^2 \quad (1.9)$$

Therefore, this thesis also aims at quantifying the degree of influence of each feature on the Euclidean distance-based algorithms. This way, it would be possible to compare the impact of different FN approaches on the distance calculation or even decide which FN technique may enhance the model's performance.

In the literature review, the work presented in [104] analyses the features' influence based on their range, estimated as the difference between the maximum and the minimum values of the feature. The features with the highest range were considered the most influencing ones on the ML algorithms' calculations. However, some FN methods utilise the range as dispersion factor. Then, according to Equation 1.9 (**Equation 8 of Paper II**), by estimating the features' contribution according to their range, it could be interpreted that some FN approaches equalise the contribution of the features to the model. To circumvent it, this thesis proposes a new metric for measuring the degree of influence of each feature on Euclidean distance-based ML algorithms, referred to as *Proportional influence*. In contrast to the range, this metric uses the extreme and the mean value of the feature for measuring its influence. This way,  $\text{Infl}(X_j)$  defined in Equation 1.10 (**Equation 11 of Paper II**) considers the mean around which the samples of the features are concentrated and the maximum separation from the samples to the mean.

$$\text{Infl}(X_j) = \max \{ |(\max(X_j) - \overline{X_j})|, |(\min(X_j) - \overline{X_j})| \} \quad (1.10)$$

In order to compare the feature's relative influence on the Euclidean distance-based ML algorithm's calculations, the *Proportional influence IN* is proposed:

$$\text{IN}(X_j) = \text{Infl}(X_j) / \max \{ \text{Infl}(X_j) | j = \{1, \dots, m\} \} \quad (1.11)$$

With the proportional influence, as illustrated in Figure 1.16 (**Figure 3 of Paper II**) the original features' contribution and the resultant one from the application of a given FN method can be analysed and compared before the dataset transformation.

For instance, Figure 1.16 (**Figure 3f in Paper II**) shows in the first row the original features' proportional influence and the resultant from applying standardisation (ST), Min-Max (MM) and Median Absolute Deviation (MAD)

normalisation methods from Table 1.3. When comparing, it is clear that the features' contribution to the Euclidean distance-based models will differ depending on the selected transformation. For instance, feature 4 presents a proportional influence value close to 0.6 with MAD, close to 0.75 with ST, and around 0.9 with MM.

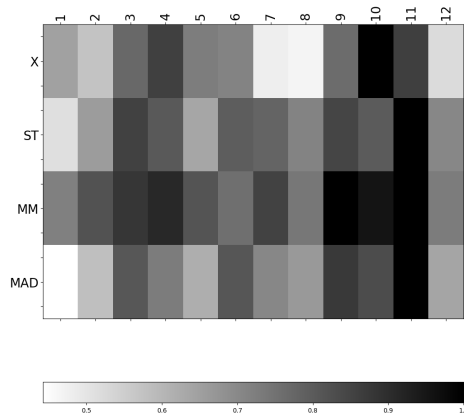


Figure 1.16: Proportional influence of the features of Accent dataset from UCI repository in the original space and the resultant from applying different FN methods (ST, MM and MAD). (Source: Paper II)

Once proven the impact of FN on the features' contribution to the Euclidean distance-based models, this thesis experimentally validates the obtained conclusions over two widely extended Euclidean distance-based ML algorithms:  $K$ -Nearest Neighbours and  $K$ -means.

**$K$ -Nearest Neighbours** ( $K$ -NN) [117] is a classification algorithm that establishes the label of a given sample based on the class membership of its  $K$  closest samples in terms of Euclidean distance.

Since each sample is classified based on the  $K$  closest neighbours and these are selected according to the spatial distribution of the samples, if FN influences the Euclidean distance, the neighbours' selection and hence, the model's performance are expected to be also affected by the FN method. In Paper II, it is also experimentally validated. As thoroughly described in **Section 3.3.1 from Paper II**, the neighbours' distribution is analysed in terms of Kendall's  $\tau$ , which measures pairwise the rank similarity or in this case, the neighbours' distribution similarity between a given dataset normalised by two different FN methods. In **Table 3 of Paper II**, it is depicted that for different UCI datasets, the mean  $\tau$  value ranges from 0.566 to 0.972. Of course, this neighbours' distribution similarity is measured for the entire neighbourhood, not only the selection of the closest  $K$  neighbours. However, an intuition about the FN influences on the  $K$ -NN selection can be derived from it. In fact, when analysing the FN influence on the performance of  $K$ -NN for different values of  $K$ , it is observed

## 1. Introduction

that there are differences up to 6.445% in terms of accuracy, and up to 10.417% in terms of recall. Similarly, if comparing the precision reached for a given  $K$  and for a dataset normalised by two different FN methods, the differences reach 21.041%. A similar analysis has been conducted over a dataset of a real use case from a refinery, which is described below in Section 1.4.4.2. In this case, the mean  $\tau$  values obtained when analysing the neighbours' distribution are higher than 0.779 in Table 11. But, when calculating the percentage of samples –normalised by different FN methods– equally classified, in **Table 9d of Paper II**, in some cases only 86.143% of the samples are labelled within the same class. Furthermore, it is translated into a difference of up to 1.9% in terms of precision, or even up to 3% of recall (**Tables 12a and 12b of Paper II**, respectively). For all the mentioned above, it is proven that FN influences the  $K$  closest neighbours selection and then the model's performance.

**$K$ -means** [118] is a clustering algorithm that groups the samples of the dataset in  $K$  different disjoint groups. The groups are formed so that the distribution of the samples maximises the intra-group cohesion, i.e. the distance of the samples to their centroid.

According to the demonstration of the FN influence on the Euclidean distance calculations, and as observed for K-NN, it is expected that FN affects the samples' distribution between the clusters, and therefore, the  $K$ -means performance. But, in addition, this thesis aims to demonstrate that the features' dominance derived from the dispersion factors differences conditions the K-means convergence. Since the clusters are formed in order to minimise the intra-cluster dispersion, if a subset of features presents a significantly lower dispersion factor, as demonstrated from Equation 1.9 (**Equation 8 in Paper II**), K-means primarily allocate the centroids in such a way that the dispersion in the mentioned components is minimised. Thus, the higher the difference between the features' proportional influence the more conditioned the search space. When comparing **Figures 5 and 8 with Figure 10 in Paper II**, the raw and the dataset normalised with MAD, which both present the highest difference with respect to the proportional influence, the algorithm converges to a solution in less than five iterations.

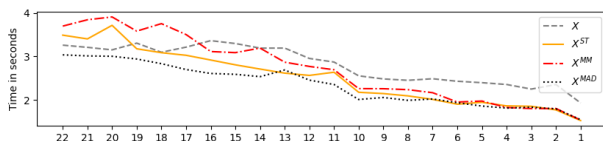


Figure 1.17: Time in seconds employed by K-means to do 100 initialisation for raw, ST, MM and MAD datasets, in a 16 GB RAM Dell Latitude 5580 workstation equipped with Intel Core i7-7600U CPU running at Microsoft Windows 10 Enterprise. (Source: Paper II)

Accordingly, an intuition about the algorithm's convergence is derived from knowing the features' dominance in advance. Furthermore, by understanding

the features' contribution to the model, discarding noncontributing features may reduce the computational cost Figure 1.17 (**Figure 13 in Paper II**) without affecting the K-means performance, as illustrated in 1.18.

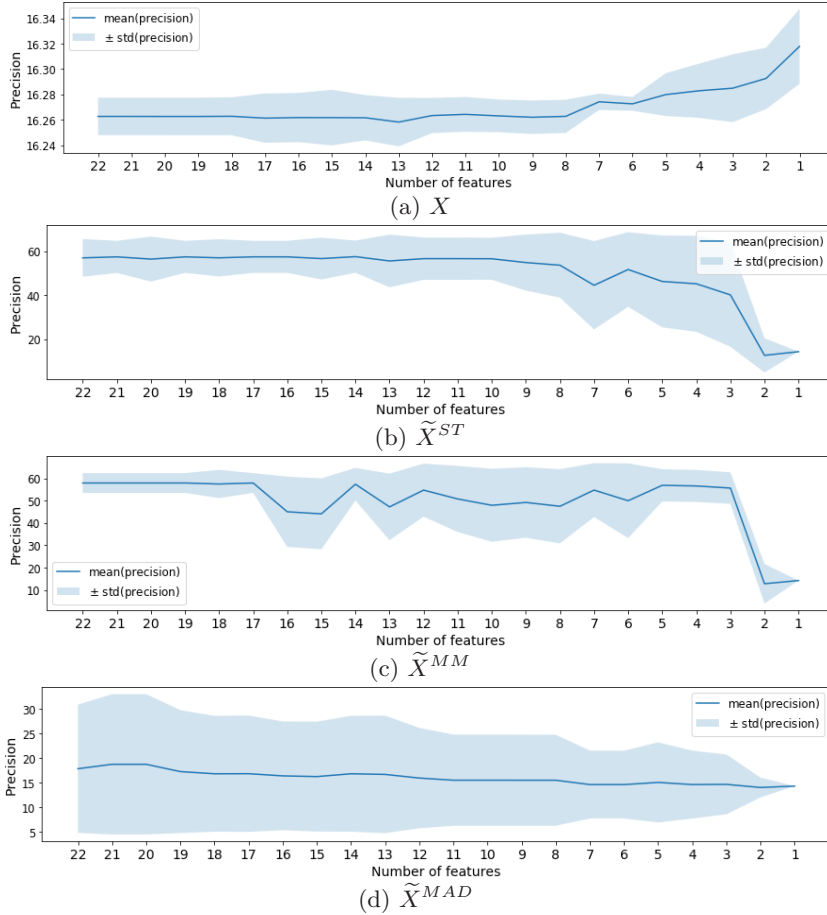
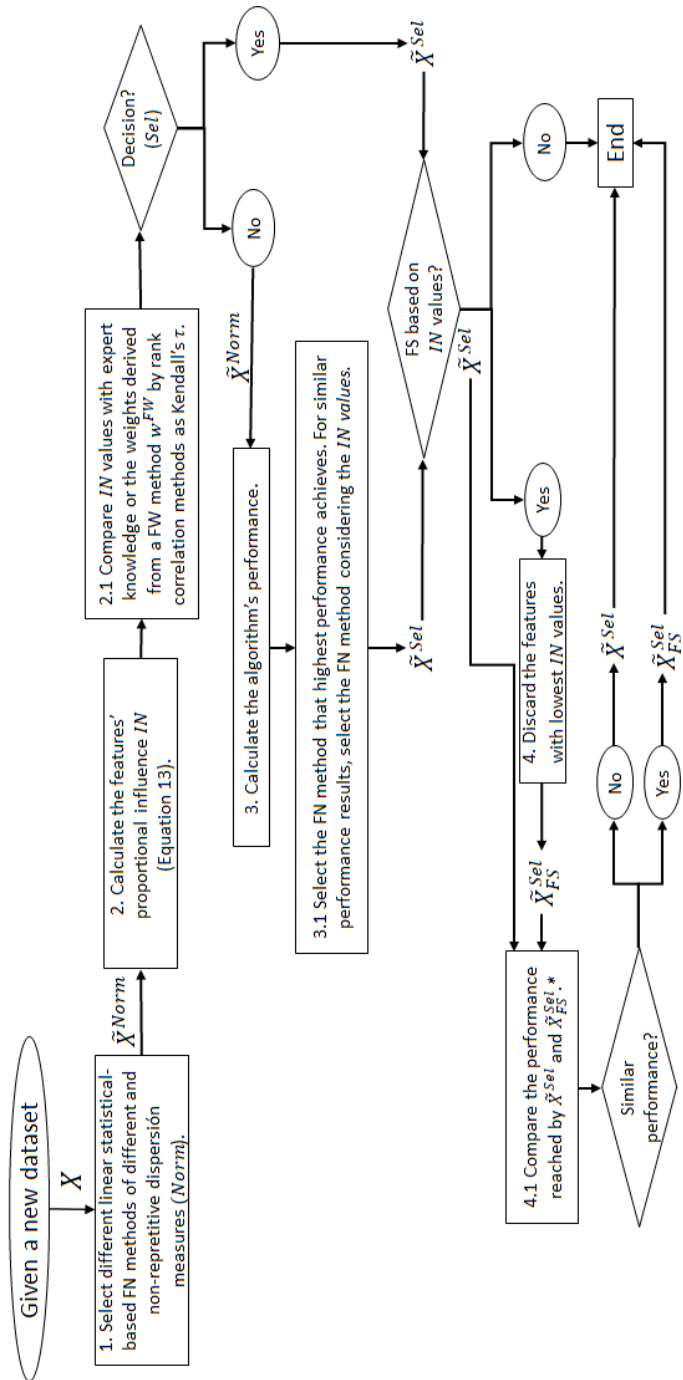


Figure 1.18: K-means results when applying Feature Selection based on the values of proportional influence (Source: Paper II).

In this case, as Figure 1.18 illustrates (**Figure 11 of Paper II**), by removing one by one the features with the lowest proportional influence value, and especially for the original and MAD datasets, no significant differences in the model's precision is observed even when maintaining only the most influencing feature of the dataset.

As a result of the described work about FN impact on Euclidean distance-based ML algorithms, a roadmap for the proper selection of the FN method when starting a new data analysis problem illustrated in Figure 1.19 is presented in **Section 6 of Paper II**.



\* Note that performance must be calculated in case of not going through step 3.

Figure 1.19: Roadmap for the proper FN method selection. (Source: Paper II)

### 1.4.2.2 Wrapper ML algorithms

As previously mentioned, wrapper ML algorithms, in terms of features' contribution, include internal weights into their formulation. Examples of this kind of ML algorithm are linear regression, logistic regression, SVM, or ANNs. The internal weights of these algorithms are iteratively updated during the training phase in order to maximise the model's performance. Therefore, the features' contributions continuously change until the optimal configuration is reached. In fact, it is widely assumed that FN is applied to equalise the features' contribution in order to "speed up the learning process in ANNs, helping the weights to converge faster" [86].

This thesis focuses on ANN-based models to demonstrate for the wrapper ML algorithms the influence of FN on the features' contribution to the model and hence the model's performance. For doing so, first a mathematical formulation of the ANN is presented in Equation 1.12 (**Equation 1 in Paper III**).

$$Y = \varphi \left( \dots \varphi \left( X \cdot W^{(1)} + b_1 \right) \dots W^{(H+1)} + b_{H+1} \right) \quad (1.12)$$

For  $\varphi(x) = x$ , (1.12) can be rewritten as in **Equation 2 from Paper III**

$$\hat{Y} = X \cdot \left( \prod_{h=1}^{H+1} W^h \right) + cte = X \cdot \mathbb{W} + cte. \quad (1.13)$$

As it can be observed, the matrix  $W^h$  represents the internal weights of the  $h$ -th layer of the network. In ANN-based models, especially when the activation function is linear, the ANN's weights are the fundamental parameters that relate the input data with the estimated output. By taking advantage of the formulation of the statistical-based FN methods and the decimal notation proposed in this thesis, an ANN with linear activation function for a normalised dataset can be formulated as

$$\hat{Y} = \tilde{X} \cdot \left( \prod_{h=1}^H W^h \right) + cte = X \cdot \mathbb{D} \cdot \left( \prod_{h=1}^H W^h \right) + cte \quad (1.14)$$

In Equation 1.14 (**Equation 8 of Paper III**) it is shown that the dispersion factors, collected in the diagonal matrix  $\mathbb{D}$ , are fixed weights contributing along with the internal weights of the network in the output estimation. Therefore, it could be assumed that during the training, the influence of the dispersion factor can be discarded if interpreting  $W = \mathbb{D}^{-1} \cdot \hat{W}$ . In such a case, contrary to the filter ML algorithms, the influence of FN would disappear.

However, this thesis demonstrates that  $\mathbb{D}$  influences the search space, the final features' contribution to the model, and consequently, the model's performance.

**Table 3 of Paper III** depicts significant differences in the outputs estimated from differently normalised datasets. For instance, in Table 3b, when comparing the outputs of the training data for a dataset normalised by ST and by MAD FN methods a mean RMSE of 86.412 is obtained. Such differences in the estimated outputs are translated into differences in the model's performance. For the

mentioned case, in Table 5b, the difference between the mean RMSE of the real labels with respect to the output estimated for the dataset normalised by ST and MAD is 856.307. Then, it is demonstrated that FN influences the ANN-based models' performance. This conclusion is also validated through the Wilcoxon signed-rank test. Furthermore, in **Section VI.B.1 of Paper III**, an analysis of the dispersion factors as explanatory factors of the similitude and differences between the performance reached by models trained by differently normalised datasets is presented. It demonstrates that the higher the dissimilarity between the dispersion factors of different FN methods, the higher the difference expected between the model's output estimations and consequently, the models' performance.

In contrast to filter ML algorithms, due to the internal weights of the wrapper approaches, the established techniques for analysing the features' contribution to the model compute the calculations in terms of the network's internal weights once the network has been trained. These methods are referred to as weight matrix analysis techniques, and Garson or Yoon's methods, defined in Equations 6 and 7 respectively, are well-known examples. However, until the date, the impact of FN on the ANNs has not been considered. Since this work proves their impact on the model's performance, an adaptation of Garson and Yoon's methods, which includes the dispersion factors to calculate the final features' contribution is proposed in Equations 1.15 and 1.16, respectively. (Analogously, **Equations 9 and 10 in Paper III**).

$$\widehat{Garson}_j = \frac{(\mathbb{D} \cdot |\mathbb{W}|)_j}{\sum_{j=1}^m |(\mathbb{D} \cdot \mathbb{W})_j|} \in [0, 1] \quad (1.15)$$

$$\widehat{Yoon}_j = \frac{(\mathbb{D} \cdot \mathbb{W})_j}{\sum_{j=1}^m |(\mathbb{D} \cdot \mathbb{W})_j|} \in [-1, 1] \quad (1.16)$$

The proposal is defined in **Equation 10 of Paper III**. After a thorough analysis and comparison of the traditional and the proposed adaptation of these weight matrix analysis methods along Section VI.C, it is concluded that the inclusion of the dispersion factors in the features' contribution calculation significantly improves the estimation of the real features' influence on the model.

All in all, it is demonstrated the relevance of the proper FN method selection for enhancing reliable results in ML-based approaches.

### 1.4.3 Methodology for soft-sensors design and development

As demonstrated above, the selection of the FN technique influences the features' contribution to the model. Simultaneously, FW transforms the features to represent their relative importance by means of features' weights ranging in  $[0, 1]$ . The scale of the features' weight derived from FW is not enough for compensating the magnitude differences among the features. Specially supervised filter FW methods that estimate the weights for each feature independently from the



ML are generally invariant to scale. Therefore, in order to conduct a proper features' preprocessing, the conjoint application of both FN and FW is mandatory. Consequently, this thesis proposes the Two-stage methodology (**Section 3.2. and Section V from Paper IV and Paper V**, respectively), given place to a new methodology for the proper features' preprocessing.

The Two-stage methodology combines both Feature Normalisation and Feature Weighting to transform the raw dataset intelligently. By this means, normalisation acts over the magnitude differences among the features to extol the resulting importance representative of the features.

Then, the Two-stage based transformation results from multiplying, for  $j \in \{1, \dots, m\}$ , each normalised feature by the corresponding feature weight,

$$(\tilde{X}_{FW}^{Norm})_j = w_j^{FW} \cdot (w_j^{Norm} \cdot X_j)$$

This way, the Two-stage methodology is flexible and allows the combination of any selected FN and FW methods.

After establishing the methodology for the proper features' preprocessing, it must be noticed that a priori, in the literature, it has not been defined an explicit methodology for the proper selection of the ML algorithm based on some properties or characteristics of the dataset or the application field. Besides, given that the features' preprocessing impacts the features' contribution to the ML algorithm-based model, and hence its performance, this thesis proposes a flexible methodology that considers several ML algorithms with different hyper-parameter configurations. This way, the proposed methodology for the soft-sensors design and development, as depicted in Figure 1.20, considers different features' preprocessing methods and ML algorithms configurations. Finally, this methodology, referred to as autoML for soft-sensor design and development, automatically selects the proper features' preprocessing methods and ML algorithms configuration.

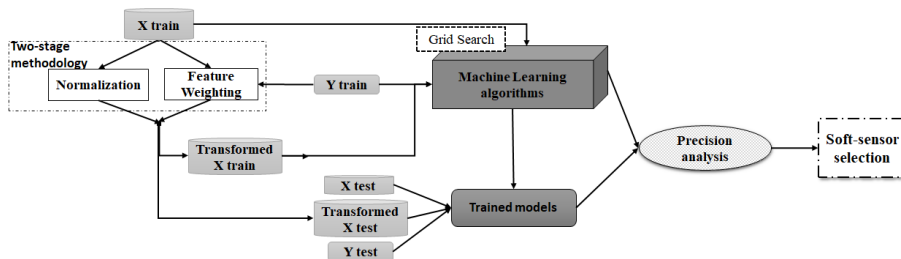


Figure 1.20: High-level diagram of the autoML for soft-sensor design and development. (Source: Paper V)

The main goal of the soft-sensors developed in this thesis is to estimate if the subproducts quality meets the standards. Then, since most of the FW literature is focused on classification problems and given that the final goal is

to obtain a *yes* or *no* about the fulfilment of the requirements, the refinery use cases are faced as classification problems. In the literature review of global filter methods for classification problems, it was observed that Pearson correlation was extended for estimating the relevance of a continuous feature with respect to a categorical one. However, the Pearson correlation coefficient measures the degree of linear correlation between two continuous variables. Therefore, this thesis also proposes the *adapted Pearson correlation*, defined in **Section 3.2.2 of Paper IV**. The adapted Pearson correlation is a new FW method specifically designed to estimate the relative importance of real-valued features with respect to categorical labels in terms of class separability. It is done by encoding the labels as centroids of the different classes, calculated according to the corresponding samples from the analysed feature. In fact, by the rationale and formulation of the proposed approach, the point-biserial correlation coefficient can be taken as a particular case of the proposed adapted Pearson correlation when the number of classes equals two.

All in all, the suitability of the proposed methodology for soft-sensor design and development are validated in the context of Oil and Gas 4.0.

### 1.4.4 Application of the proposed autoML for soft-sensor design and development in Oil and Gas 4.0

As mentioned in Sections 1.1 and 1.2, soft-sensors are of special interest for subproduct quality estimation in order to complement the current laboratory tests and online analysers. Thus, this thesis applies the proposed autoML approach for two different real use cases from a refinery in order to validate it.

#### 1.4.4.1 Laboratory test real use case- Needle penetration soft-sensor

The first application of the proposed autoML is presented in Paper IV. As it is remarked in Section 1.1, industrial petroleum refineries are complex distillation systems, composed by chain units where physical reactions aim at converting crude in high quality subproducts.

Figure 1.21 depicts a high-level flow diagram of the crude refining process. Firstly, the input crude properties, described in the dataset by 33 features (C\_1:C\_33), determined by experts in the field and updated every six months – or with a higher frequency if major changes regarding the use of new crude occurs – is injected into Process 1 and Process 2 where the chemical and physical reactions start the refining process. From Process 1 and 2 the processed crude is pumped into the Process 3, where the refining process continues. The behaviour of the Process 1, 2 and 3 is shown through 15 (P1\_1:P1\_15), 10 (P2\_1:P2\_10) and 27 (P3\_1:P3\_27) features, respectively, that are continuously monitored, providing information of flow, pressure and temperature. Along all the units of the chain system, 9 online chemical quality analysers (A1\_1:A\_9) monitor the process development.

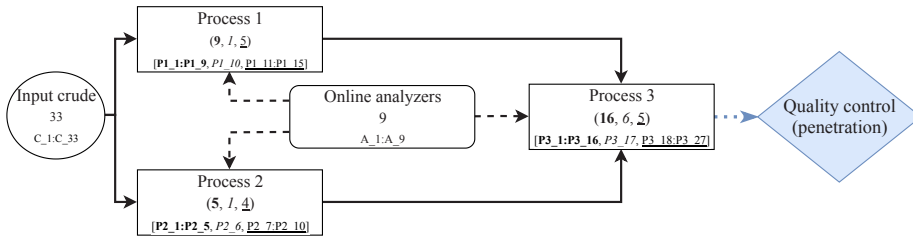


Figure 1.21: Flow diagram of the crude refining process. At each process unit, the features related to flow, pressure and temperature are remarked with bold, italic and underlined text, respectively. (Source: Paper IV)

The information of the crude refining process is collected in a dataset, consisting of 33485 samples described by the mentioned 94 features recorded every 15 minutes during 349 days.

The interest of the refinery is to maintain a high penetration quality of the bottom product of the vacuum distillation unit located at the end of Process 3. In order to control the quality specifications, some samples are tested in the laboratory according to [119]. Depending on the result, the samples are classified into two groups: group 0 represents the samples that fulfil penetration quality standards, and group 1 refers to the samples that do not meet the constraints. In practice, if a sample does not meet the designed quality standard, the plant operators adjust the operational variables aiming at correcting the outcome quality at the end of the process. However, in practice, such laboratory test is not performed following a predefined schedule and the results are available after four hours, which entails a significant delay in knowing the penetration quality of the product with the consequent economic loss for the refinery. In fact, from the whole historical data, only 268 samples are labelled. Therefore, in order to circumvent such drawback this thesis proposes complementary to the laboratory test a soft-sensor that continuously estimates, based on the operational variables, the subproduct quality.

For doing so, the presented autoML approach is applied. For the described problem, this work considers ST, MM and MAD FN methods for the normalisation stage of Figure 1.20. The proposed adapted Pearson correlation (P) and Random Forest (RF) for features' relevance estimation are candidates of FW methods. Regarding the ML algorithms, for the development of the soft-sensor K-means, K-NN, Random Forest classifier (RFc), Support Vector Machine and MultiLayer Perceptron (MLP) ML algorithms are selected. Except for K-means, which only hyper-parameter  $K$  is known in advance ( $K = 2$ ), for the rest of ML algorithms, a set of parameters configuration is stated. By design, as shown in Figure 1.20, the autoML includes a Grid Search for the optimal hyper-parameters configuration. In **Table 1 of Paper IV** the values of the optimal hyper-parameters for each features' preprocessing configuration is presented. Table 1.4 collects the models' accuracy results obtained from the different ML algorithms with the optimal hyper-parameters configuration.

## 1. Introduction

		raw	Normalization			Two-stage methodology					
			ST	MM	MAD	ST-P	MM-P	MAD-P	ST-RF	MM-RF	MAD-RF
K-means	mean	51.852	70.37	70.37	40.741	70.37	<b>81.407</b>	40.741	70.37	77.778	40.741
	std	0	0	0	0	0	0.519	0	0	0	0
	max	51.852	70.37	70.37	40.741	70.37	81.481	40.741	70.37	77.778	40.741
	min	51.852	70.37	70.37	40.741	70.37	77.778	40.741	70.37	77.778	40.741
K-NN	acc	70.37	77.778	77.778	77.778	<b>81.481</b>	<b>81.481</b>	77.778	<b>81.481</b>	<b>81.481</b>	77.778
RFc	mean	52.185	52.185	52.185	53.334	52.741	74.37	52.778	52.926	<b>74.445</b>	52.778
	std	3.535	3.535	3.535	4.709	3.652	6.132	4.479	4.188	5.803	4.479
	max	59.259	59.259	59.259	77.778	66.667	77.778	70.37	66.667	77.778	70.37
	min	44.444	44.444	44.444	44.444	44.444	55.556	44.444	44.444	55.556	44.444
SVC	acc	51.852	<b>81.481</b>	<b>81.481</b>	77.778	77.778	77.778	70.37	<b>81.481</b>	<b>81.481</b>	74.074
MLP	mean	51.667	67.259	65.444	64.519	74.815	<b>77.37</b>	66.852	74.852	67.555	63.889
	std	14.372	8.5	11.018	4.971	6.317	12.839	3.261	7.884	10.622	5.918
	max	81.481	81.481	85.185	74.074	85.185	85.185	74.074	88.889	85.185	70.37
	min	29.63	44.444	40.741	44.444	59.259	40.741	40.741	51.852	55.556	37.037

Table 1.4: Accuracy statistics (mean, std, max, min) obtained in 100 MonteCarlo simulations for each dataset and ML algorithm. Since K-NN and SVC are deterministic only the accuracy value (acc) is shown. (Source: Paper IV)

As it can be observed in Table 1.4, the suitability of the proposed Two-stage methodology is validated. In fact, in all the cases, the mean accuracy values obtained by the features preprocessed with the Two-stage methodology equal or outperform those yielded by the application of only FN. As depicted in Table 1.4, the maximum accuracy reached by the deterministic algorithms (K-NN and SVC) is 81.481%. For some transformations of the dataset, K-means and MLP also reach maximum accuracy values higher than 80%, but the unique model that in terms of mean accuracy takes values higher than 80% is the one based on K-means with MM and P preprocessing methods. In addition, it must be noticed that the main interest of the refinery is the prompt detection of improvable subproduct quality. Then, precision is also considered for the selection of the soft-sensor. For the models based on K-NN and SVC with accuracy higher than 80%, the associated precision is lower than 88%. In contrast, the K-means-based model obtains a precision value equal to 93.8%. Furthermore, between K-NN, SVC and K-means, the principle of parsimony suggests to select, in equal conditions, the simplest one, which corresponds to the model based on K-means. Besides, given the formulation of K-means, the resulting model can be easily explained and interpreted. Therefore, among the obtained models, the autoML approach selects the resulting from combining MM FN and the adapted Pearson correlation FW methods for the preprocessing stage and K-means ML algorithm for designing the soft-sensor.

As explained in **Section 4.3 of Paper IV**, based on the knowledge obtained from the analysis of the FN and FW influence on the features' contribution to the model, the proposed approach enables the interpretability of the soft-sensor, which demonstrates the reliability of the soft-sensor for its application. Specifically, it is concluded that the predominant features for the soft-sensor modelling are those related to the streams of atmospheric residue and vacuum residue. At the same time, these properties are relevant measures to infer the penetration quality standard of the bottom subproduct of the vacuum distillation

unit. Such correspondence between the soft-sensor and the physical process validates the suitability of the proposed soft-sensors, which, at the same time, it is expected to enhance the trust from the operator in the proposed soft-sensor's results. Moreover, the proposed soft-sensor is able to continuously estimate with a precision higher than 93% the subproduct quality based on the operation variables, anticipating at least four hours the results of the laboratory test and accelerating the subproduct preprocessing which impacts on the refinery's profit.

#### 1.4.4.2 Online analyser real use case- Percentage of pentanes in butane soft-sensor

Figure 1.22 depicts a high-level diagram of the analysed unit chain, in which crude oil is converted into high-quality gas subproducts.

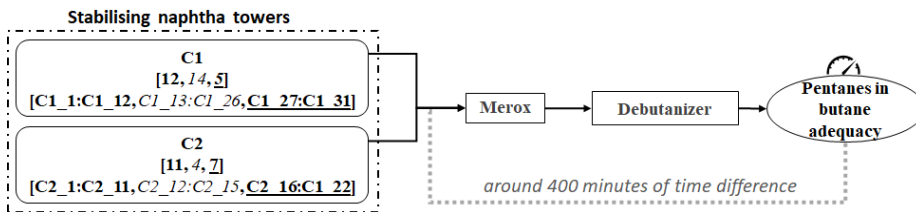


Figure 1.22: High level diagram of the pentanes use case. (Source: Paper V)

Columns C1 and C2 in Figure 1.22 represent two different stabilising naphtha towers. After a refining process of the raw crude, stabilised naphtha and Liquefied Petroleum Gas (LPG) are obtained at the bottom and the top of the columns C1 and C2, respectively. The resulting LPG is then pumped from the top of columns C1 and C2 to Merox, a gas sweetening unit in which the sulphur is removed. Finally, the sweetened gases pass to the debutanizer column, where propane and butane are separated. The estimated duration of the described unit chain, from stabilising naphtha columns to the end of the debutanizer column, is 400 minutes.

In order to fulfil the specification standards [120], the resultant butane must not exceed a certain threshold of percentage of pentanes (1.5%). According to the mentioned threshold, the refinery's interest is to classify the percentage of pentanes in butane as adequate (class 0) or improvable (class 1). Currently, such measurement is conducted by an online analyser over samples extracted from the debutanizer column. In contrast, this thesis aims to propose a soft-sensor that, based on the refining process of C1 and C2, estimates 400 minutes in advance the percentage of pentanes in butane. The final goal of the soft-sensor is to facilitate as soon as possible the re-processing of the subproduct saving costs to the refinery.

With that purpose, from the top of C1 and C2, 31 (C1\_1:C1\_31) and 22 (C2\_1:C2\_22) features are collected, respectively. These features at each column gather information about flow, temperature and pressure. The number of features of each column regarding each of these properties are presented in

## 1. Introduction

Figure 1.22 with bold, italic and underlined text, respectively. The process variables information and the pentanes percentage output are recorded every ten minutes for 465 days, from 24-10-2017 to 31-01-2019. Thus, the dataset consists of 66847 samples described by the 53 features described above.

Again, the proposed autoML approach is applied to, in this case, design and develop a soft-sensor for complementing the current online analyser in the subproduct quality estimation.

In this use case, given the fully supervised available data, first, as described in **Section 3.1 of Paper V**, a strategy for the selection of the optimal training window based on 1) a time domain feature evaluation considering seasonality, trend and stationarity, and 2) a class occurrence frequency analysis is conducted. After choosing the proper training data, the proposed methodology illustrated in Figure 1.20 is applied. Again, ST, MM and MAD methods are included in the normalisation phase. For the FW stage, in this case, in addition to the adapted Pearson correlation and Random Forest, Mutual Information (MI) is also considered for the estimation of the features' relevance. Regarding the ML algorithms, this work considers Quadratic Discriminant Analysis (QDA), K-NN, SVM, Ridge regression (RID), Logistic regression (LOG), MLP and Stochastic Gradient Descent (SGD). For the different combinations of features' preprocessing techniques, the proposed autoML searches with the Grid Search algorithm the optimal hyper-parameters configuration to train the selected ML algorithms. Table 1.5 collects the performance results, in terms of precision reached by each ML algorithm configured with the optimal hyper-parameter for each transformation of the dataset.

Algorithm	Raw		Normalisation			Proposed methodology								
	X	$\hat{X}^{ST}$	$\hat{X}^{MM}$	$\hat{X}^{MAD}$	$\hat{X}_P^{ST}$	$\hat{X}_P^{MM}$	$\hat{X}_P^{MAD}$	$\hat{X}_{RF}^{ST}$	$\hat{X}_{RF}^{MM}$	$\hat{X}_{RF}^{MAD}$	$\hat{X}_{MI}^{ST}$	$\hat{X}_{MI}^{MM}$	$\hat{X}_{MI}^{MAD}$	
QDA	24.414	62.304	64.286	61.340	0.000	38.506	40.909	53.548	72.973	47.689	90.000	0.000	<b>1.00</b>	
KNN	27.551	23.192	40.554	26.359	41.429	39.370	42.529	24.724	38.998	<b>43.416</b>	35.057	32.113	37.956	
SVC	56.897	0.000	7.368	0.000	22.562	16.068	20.564	52.250	16.333	<b>65.079</b>	21.914	16.071	24.145	
RID	81.507	38.517	86.957	51.598	22.938	98.734	54.028	20.511	<b>1.00</b>	51.598	21.807	96.000	51.835	
LOG	90.164	92.029	0.000	<b>1.00</b>	97.872	0.000	85.714	80.000	<b>1.00</b>	90.698	92.381	0.000	75.000	
MLP	Max	100	82.178	84.647	83.974	88.587	93.878	77.500	78.599	100	75.646	85.976	91.509	75.954
	Mean	34.595	51.622	68.631	58.136	80.558	82.055	73.460	71.364	<b>95.180</b>	71.675	76.591	73.593	72.384
	std	36.386	20.673	11.569	19.943	4.064	5.892	2.066	2.014	3.809	2.049	3.601	6.334	2.106
	Min	0.000	18.171	38.836	18.825	74.717	72.549	69.283	68.910	87.500	65.549	72.852	66.997	68.506
SGD	Max	26.606	46.868	18.929	75.862	71.795	0.000	81.022	41.640	0.000	44.660	23.343	0.000	30.334
	Mean	8.013	42.328	13.236	41.045	34.554	0.000	<b>42.824</b>	37.842	0.000	41.662	18.144	0.000	26.709
	std	6.171	2.028	1.850	13.979	13.019	0.000	10.755	1.365	0.000	1.465	2.359	0.000	1.245
	Min	0.000	38.636	10.304	17.804	15.139	0.000	24.967	35.431	0.000	39.130	12.405	0.000	24.194

Table 1.5: Precision reached by each ML algorithm over the raw, normalised and transformed datasets. (Source: Paper V)

As it can be observed in Table 1.5, the two-stage methodology reaches superior performance results than its counterparts. Moreover, the proposed methodology enables to design soft-sensors with a precision higher than 90%. In fact, the selected soft-sensor by the autoML for the subproduct quality estimation, designed as a combination of ST with the adapted Pearson correlation and logistic

regression, obtains a precision of 98.925% for predicting the resultant product of improvable quality.

Hence, this work demonstrates again the viability of the autoML approach in terms of performance. Moreover, since the subproduct quality estimation is computed based on the operational variables in columns C1 and C2, the soft-sensor detects deviations from the specifications 400 minutes in advance with respect to the online analyser, which directly translates to profit benefits for the refinery.

Figure 1.23 depicts the tons of butane per hour that do not fulfil the specification requirements.

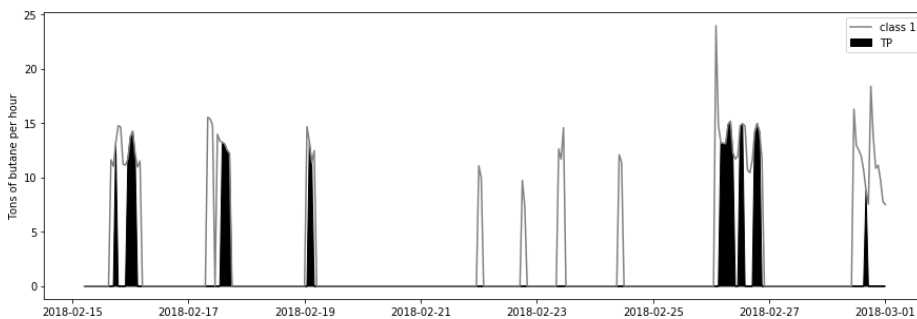


Figure 1.23: Tons of butane per hour that do not fulfil the specification requirements. (Source: Paper V)

The grey line in Figure 1.23 (**Figure 13 in Paper V**) depicts the total amount of butane per hour resulting from the distilling process that does not fulfil the specification requirements. The black area of Figure 1.23 represents the amount of butane that does not meet the specifications correctly detected by the soft-sensor. The quantity of butane resulting from the distilling process is calculated based on data from the refinery. For the units conversion, from the  $\text{m}^3/\text{l}$  of butane flow measured at the end of the unit chain to the tons of butane (Figure 1.23) utilised to calculate the final profit, a product density value equal to  $0.575 \text{ kg/l}$  is utilised according to the refinery's laboratory test conducted over real data from February of 2018. As observed in Figure 1.23, in some hours up to 14.56 tons of butane do not meet the specification requirements, which forces the refinery to re-inject such subproduct in the distillation process until fulfilling the specification, which ultimately results in a decrease in the amount of butane to sale. However, a prompt prediction of the butane quality in terms of percentage of pentanes allows to readjust the process in advance (400 minutes before) and reduce the profit losses.

Due to the time-frame needed to reach the new operation point, and considering a conservative approach, only the benefit over the 80% of the correctly detected improvable butane is calculated. Thus, in the analysed period and discarding the 20% of the detected improvable butane, 258.22 tons of butane

## 1. Introduction

---

that do not fulfil the specification requirements are correctly detected by the proposed soft-sensor. Besides, each refinery sets its own sale price for each subproduct. In the refinery from where the data come, the sale price of a ton of butane in February of 2018 was 459.74\$. Thus, in the studied two weeks, the profit derived from the online prediction of the subproduct quality with the proposed soft-sensor would be a total of 111,939.35\$. Then, the suitability of the autoML approach for the design and development of a soft-sensor for the percentage of pentanes in butane estimation and its profitability for the refinery are demonstrated.

## 1.5 References

- [1] Ocident Bongomin et al. “Industry 4.0 Disruption and Its Neologisms in Major Industrial Sectors: A State of the Art”. In: *Journal of Engineering* 2020 (2020).
- [2] Ray Y Zhong et al. “Intelligent manufacturing in the context of industry 4.0: a review”. In: *Engineering* 3.5 (2017), pp. 616–630.
- [3] “UPSTREAM? MIDSTREAM? DOWNSTREAM? WHAT’S THE DIFFERENCE?” In: *Energy HQ* (2017). URL: <https://energyhq.com/2017/04/upstream-midstream-downstream-whats-the-difference/>.
- [4] James H. Gary. “Refineries”. In: *Macmillan Encyclopedia of Energy* (2021). URL: <https://www.encyclopedia.com/environment/encyclopedias-almanacs-transcripts-and-maps/refineries>.
- [5] James G Speight. *The refinery of the future*. Gulf Professional Publishing, 2020.
- [6] FITCH WIRE. “Europe’s Refining Margins Rebound, but Remain Exposed to Covid-19 Setbacks”. In: *FITCH Ratings* (2021), p. 107617. URL: <https://www.fitchratings.com/research/corporate-finance/europes-refining-margins-rebound-remain-exposed-to-covid-19-setbacks-16-09-2021>.
- [7] Marco S Reis and Geert Gins. “Industrial process monitoring in the big data/industry 4.0 era: From detection, to diagnosis, to prognosis”. In: *Processes* 5.3 (2017), p. 35.
- [8] Angie Sticher. “Data rich but information poor”. In: *IIoT world* (2021). URL: <https://iiot-world.com/industrial-iiot/connected-industry/data-rich-and-information-poor/>.
- [9] Luigi Fortuna et al. *Soft sensors for monitoring and control of industrial processes*. Vol. 22. Springer, 2007.
- [10] Ali Al-Jlibawi et al. “The efficiency of soft sensors modelling in advanced control systems in oil refinery through the application of hybrid intelligent data mining techniques”. In: *Journal of Physics: Conference Series*. Vol. 1529. 5. IOP Publishing. 2020, p. 052049.



- 
- [11] Leonardo de Pádua Agripa Sales, Francisco Murilo Tavares de Luna, and Bruno de Athayde Prata. “An integrated optimization and simulation model for refinery planning including external loads and product evaluation”. In: *Brazilian Journal of Chemical Engineering* 35.1 (2018), pp. 199–215.
- [12] Georgios KD Saharidis, Michel Minoux, and Yves Dallery. “Scheduling of loading and unloading of crude oil in a refinery using event-based discrete time formulation”. In: *Computers & Chemical Engineering* 33.8 (2009), pp. 1413–1426.
- [13] Leonid Sheremetov, Jorge Martinez-Muñoz, and Manuel Chi-Chim. “Two-stage genetic algorithm for parallel machines scheduling problem: Cyclic steam stimulation of high viscosity oil reservoirs”. In: *Applied Soft Computing* 64 (2018), pp. 317–330.
- [14] Sandeep Singh Chauhan and Prakash Kotecha. “An efficient multi-unit production planning strategy based on continuous variables”. In: *Applied Soft Computing* 68 (2018), pp. 458–477.
- [15] P Gil et al. “Data Anomaly Detection in Wireless Sensor Networks with Application to an Oil Refinery”. In: *2018 13th APCA International Conference on Automatic Control and Soft Computing (CONTROLO)*. IEEE, 2018, pp. 425–429.
- [16] Peng Xu, Rui Du, and Zhongbao Zhang. “Predicting pipeline leakage in petrochemical system through GAN and LSTM”. In: *Knowledge-Based Systems* 175 (2019), pp. 50–61.
- [17] Bin Zhao et al. “Maintenance decision methodology of petrochemical plant based on fuzzy curvelet neural network”. In: *Applied Soft Computing* 69 (2018), pp. 203–212.
- [18] Luigi Fortuna, Salvatore Graziani, and Maria Gabriella Xibilia. “Comparison of soft-sensor design methods for industrial plants using small data sets”. In: *IEEE Transactions on instrumentation and measurement* 58.8 (2009), pp. 2444–2451.
- [19] Lukasz Pater. “Application of artificial neural networks and genetic algorithms for crude fractional distillation process modeling”. In: *arXiv preprint arXiv:1605.00097* (2016).
- [20] Bahareh Bidar et al. “A data-driven soft-sensor for monitoring ASTM-D86 of CDU side products using local instrumental variable (LIV) technique”. In: *Journal of the Taiwan Institute of Chemical Engineers* 84 (2018), pp. 49–59.
- [21] Yajun Fan et al. “A Data-Driven Soft Sensor Based on Multilayer Perceptron Neural Network with a Double LASSO Approach”. In: *IEEE Transactions on Instrumentation and Measurement* (2019).

- [22] Yalin Wang, Dongzhe Wu, and Xiaofeng Yuan. “A two-layer ensemble learning framework for data-driven soft sensor of the diesel attributes in an industrial hydrocracking process”. In: *Journal of Chemometrics* 33.12 (2019), e3185.
- [23] Wang Yuqiao et al. “A soft sensor for carbon content of spent catalyst in a continuous reforming plant using LSSVM-GA”. In: *Proceedings of the 31st Chinese Control Conference*. IEEE. 2012, pp. 7056–7060.
- [24] Luigi Fortuna et al. “Virtual instruments based on stacked neural networks to improve product quality monitoring in a refinery”. In: *IEEE Transactions on Instrumentation and Measurement* 56.1 (2007), pp. 95–101.
- [25] Srećko Herceg, Željka Ujević Andrijić, and Nenad Bolf. “Development of soft sensors for isomerization process based on support vector machine regression and dynamic polynomial models”. In: *Chemical Engineering Research and Design* 149 (2019), pp. 95–103.
- [26] Ivan Mohler, Željka Ujević Andrijić, and Nenad Bolf. “Soft sensors model optimization and application for the refinery real-time prediction of toluene content”. In: *Chemical Engineering Communications* 205.3 (2018), pp. 411–421.
- [27] Xiaofeng Yuan et al. “Deep learning-based feature representation and its application for soft sensor modeling with variable-wise weighted SAE”. In: *IEEE Transactions on Industrial Informatics* 14.7 (2018), pp. 3235–3243.
- [28] David Wang, Jun Liu, and Rajagopalan Srinivasan. “Data-driven soft sensor approach for quality prediction in a refining process”. In: *IEEE Transactions on Industrial Informatics* 6.1 (2009), pp. 11–17.
- [29] Luigi Fortuna, Salvatore Graziani, and Maria Gabriella Xibilia. “Cross correlation analysis of residuals for the selection of the structure of Virtual Sensors in a refinery”. In: *2005 IEEE Conference on Emerging Technologies and Factory Automation*. Vol. 1. IEEE. 2005, 6–pp.
- [30] Ridong Zhang and Qibing Jin. “Design and Implementation of hybrid modeling and PFC for oxygen content regulation in a coke furnace”. In: *IEEE Transactions on Industrial Informatics* 14.6 (2018), pp. 2335–2342.
- [31] V Esposito Vinzi et al. *Handbook of partial least squares*. Vol. 201. 0. Springer, 2010.
- [32] John J Hopfield. “Artificial neural networks”. In: *IEEE Circuits and Devices Magazine* 4.5 (1988), pp. 3–10.
- [33] Corinna Cortes and Vladimir Vapnik. “Support vector machine”. In: *Machine learning* 20.3 (1995), pp. 273–297.
- [34] Xu Zhang et al. “A weighted auto regressive LSTM based approach for chemical processes modeling”. In: *Neurocomputing* 367 (2019), pp. 64–74.

- 
- [35] Chao Jiang et al. “Real-time semisupervised predictive modeling strategy for industrial continuous catalytic reforming process with incomplete data using slow feature analysis”. In: *Industrial & Engineering Chemistry Research* 58.37 (2019), pp. 17406–17423.
- [36] Xiaofeng Yuan et al. “Soft sensor modeling of nonlinear industrial processes based on weighted probabilistic projection regression”. In: *IEEE Transactions on Instrumentation and Measurement* 66.4 (2017), pp. 837–845.
- [37] Kangcheng Wang et al. “Automatic hyper-parameter tuning for soft sensor modeling based on dynamic deep neural network”. In: *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2017, pp. 989–994.
- [38] Kumar Siddharth, Amey Pathak, and Ajaya Kumar Pani. “Real-time quality monitoring in debutanizer column with regression tree and ANFIS”. In: *Journal of Industrial Engineering International* 15.1 (2019), pp. 41–51.
- [39] Xiaofeng Yuan, Lin Li, and Yalin Wang. “Nonlinear dynamic soft sensor modeling with supervised long short-term memory network”. In: *IEEE transactions on industrial informatics* 16.5 (2019), pp. 3168–3176.
- [40] “Flow diagram of typical refinery”. In: *Expect Asia Co.* (2021). URL: <https://www.expect.com/process-plants/flow-diagram-of-typical-refinery/>.
- [41] Douglas M Hawkins. “The problem of overfitting”. In: *Journal of chemical information and computer sciences* 44.1 (2004), pp. 1–12.
- [42] Bram Steurtewagen and Dirk Van den Poel. “Machine learning refinery sensor data to predict catalyst saturation levels”. In: *Computers & Chemical Engineering* 134 (2020), p. 106722.
- [43] Srećko Herceg, Ž Ujević Andrijić, and Nenad Bolf. “Support Vector Machine-based Soft Sensors in the Isomerisation Process”. In: *Chemical and Biochemical Engineering Quarterly* 34.4 (2020), pp. 243–255.
- [44] WMP van der Aalst. “Process mining: On the balance between underfitting and overfitting”. In: *Proceedings of the ECML-PKDD Workshop on Induction of Process Models (IPM08)*. Citeseer, 2008, pp. 1–2.
- [45] Salvatore Graziani and Maria Gabriella Xibilia. “Deep structures for a reformer unit soft sensor”. In: *2018 IEEE 16th International Conference on Industrial Informatics (INDIN)*. IEEE, 2018, pp. 927–932.
- [46] Girish Chandrashekar and Ferat Sahin. “A survey on feature selection methods”. In: *Computers & Electrical Engineering* 40.1 (2014), pp. 16–28.
- [47] Jundong Li et al. “Feature selection: A data perspective”. In: *ACM Computing Surveys (CSUR)* 50.6 (2017), pp. 1–45.
- [48] Mehrdad Rostami et al. “Review of swarm intelligence-based feature selection methods”. In: *Engineering Applications of Artificial Intelligence* 100 (2021), p. 104210.

- [49] Mario Di Mauro et al. “Supervised feature selection techniques in network intrusion detection: A critical review”. In: *Engineering Applications of Artificial Intelligence* 101 (2021), p. 104216.
- [50] Hossein Hassani et al. “Unsupervised concrete feature selection based on mutual information for diagnosing faults and cyber-attacks in power systems”. In: *Engineering Applications of Artificial Intelligence* 100 (2021), p. 104150.
- [51] Francesco Curreri, Salvatore Graziani, and Maria Gabriella Xibilia. “Input selection methods for data-driven Soft Sensors design: application to an industrial process”. In: *Information Sciences* (2020).
- [52] Dietrich Wettschereck, David W Aha, and Takao Mohri. “A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms”. In: *Artificial Intelligence Review* 11.1-5 (1997), pp. 273–314.
- [53] Renato Cordeiro de Amorim. “A survey on feature weighting based K-Means algorithms”. In: *Journal of Classification* 33.2 (2016), pp. 210–242.
- [54] Zhaohong Deng et al. “A survey on soft subspace clustering”. In: *Information sciences* 348 (2016), pp. 84–106.
- [55] Lizhi Peng et al. “A fast feature weighting algorithm of data gravitation classification”. In: *Information Sciences* 375 (2017), pp. 54–78.
- [56] Emrehan Kutlug Sahin, Cengizhan Ipbuker, and Taskin Kavzoglu. “A comparison of feature and expert-based weighting algorithms in landslide susceptibility mapping”. In: *Procedia Earth and Planetary Science* 15 (2015), pp. 462–467.
- [57] Dietrich Wettschereck and Thomas G Dietterich. “An experimental comparison of the nearest-neighbor and nearest-hyperrectangle algorithms”. In: *Machine learning* 19.1 (1995), pp. 5–27.
- [58] Pedro J Garcia-Laencina et al. “K nearest neighbours with mutual information for simultaneous classification and missing data imputation”. In: *Neurocomputing* 72.7-9 (2009), pp. 1483–1493.
- [59] Hong-jie Xing et al. “Linear feature-weighted support vector machine”. In: *Fuzzy Information and Engineering* 1.3 (2009), pp. 289–305.
- [60] Syed Fawad Hussain. “A novel robust kernel for classifying high-dimensional data using Support Vector Machines”. In: *Expert Systems with Applications* 131 (2019), pp. 116–131.
- [61] Anh Viet Phan, Minh Le Nguyen, and Lam Thu Bui. “Feature weighting and SVM parameters optimization based on genetic algorithms for classification problems”. In: *Applied Intelligence* 46.2 (2017), pp. 455–469.
- [62] Maciej Komosiński and Krzysztof Krawiec. “Evolutionary weighting of image features for diagnosing of CNS tumors”. In: *Artificial Intelligence in Medicine* 19.1 (2000), pp. 25–38.

- 
- [63] Hyunchul Ahn and Kyoung-jae Kim. “Global optimization of case-based reasoning for breast cytology diagnosis”. In: *Expert Systems with Applications* 36.1 (2009), pp. 724–734.
- [64] Daniel Mateos-Garcia, Jorge Garcia-Gutiérrez, and José C Riquelme-Santos. “On the evolutionary weighting of neighbours and features in the k-nearest neighbour rule”. In: *Neurocomputing* (2017).
- [65] Isaac Triguero et al. “Integrating a differential evolution feature weighting scheme into prototype generation”. In: *Neurocomputing* 97 (2012), pp. 332–343.
- [66] Mahmood Sotoodeh, Mohammad Reza Moosavi, and Reza Boostani. “A novel adaptive LBP-based descriptor for color image retrieval”. In: *Expert Systems with Applications* 127 (2019), pp. 342–352.
- [67] Lifei Chen and Gongde Guo. “Nearest neighbor classification of categorical data by attributes weighting”. In: *Expert Systems with Applications* 42.6 (2015), pp. 3142–3149.
- [68] Elena Marchiori. “Class dependent feature weighting and k-nearest neighbor classification”. In: *IAPR International Conference on Pattern Recognition in Bioinformatics*. Springer. 2013, pp. 69–78.
- [69] Turgay Yilmaz, Adnan Yazici, and Masaru Kitsuregawa. “RELIEF-MM: effective modality weighting for multimedia information retrieval”. In: *Multimedia systems* 20.4 (2014), pp. 389–413.
- [70] Daniel Mateos-Garcia, Jorge Garcia-Gutiérrez, and José C Riquelme-Santos. “On the evolutionary optimization of k-NN by label-dependent feature weighting”. In: *Pattern Recognition Letters* 33.16 (2012), pp. 2232–2238.
- [71] Ammar W Mohemmed and Mengjie Zhang. “Evaluation of particle swarm optimization based centroid classifier with different distance metrics”. In: *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*. IEEE. 2008, pp. 2929–2932.
- [72] Roberto Paredes and Enrique Vidal. “Learning weighted metrics to minimize nearest-neighbor classification error”. In: *IEEE Transactions on Pattern Analysis & Machine Intelligence* 7 (2006), pp. 1100–1110.
- [73] Roberto Paredes and Enrique Vidal. “A class-dependent weighted dissimilarity measure for nearest neighbor classification problems”. In: *Pattern Recognition Letters* 21.12 (2000), pp. 1027–1036.
- [74] Zhenwen Ren et al. “Simultaneous learning of reduced prototypes and local metric for image set classification”. In: *Expert Systems with Applications* 134 (2019), pp. 102–111.
- [75] Salih Güneş, Kemal Polat, and Şebnem Yosunkaya. “Efficient sleep stage recognition system based on EEG signal using k-means clustering based feature weighting”. In: *Expert Systems with Applications* 37.12 (2010), pp. 7922–7928.

- [76] Hüseyin Gürüler. “A novel diagnosis system for Parkinson’s disease using complex-valued artificial neural network with k-means clustering feature weighting method”. In: *Neural Computing and Applications* 28.7 (2017), pp. 1657–1666.
- [77] Joshua Zhexue Huang et al. “Automated variable weighting in k-means type clustering”. In: *IEEE Transactions on Pattern Analysis & Machine Intelligence* 5 (2005), pp. 657–668.
- [78] Chieh-Yuan Tsai and Chuang-Cheng Chiu. “Developing a feature weight self-adjustment mechanism for a K-means clustering algorithm”. In: *Computational statistics & data analysis* 52.10 (2008), pp. 4658–4672.
- [79] Young-Seon Jeong, Myong K Jeong, and Olufemi A Omitaomu. “Weighted dynamic time warping for time series classification”. In: *Pattern Recognition* 44.9 (2011), pp. 2231–2240.
- [80] Young-Seon Jeong and Raja Jayaraman. “Support vector-based algorithms with weighted dynamic time warping kernel function for time series classification”. In: *Knowledge-based systems* 75 (2015), pp. 184–191.
- [81] Qingchen Zhang and Zhikui Chen. “A distributed weighted possibilistic c-means algorithm for clustering incomplete big sensor data”. In: *International Journal of Distributed Sensor Networks* 10.5 (2014), p. 430814.
- [82] Shounak Datta, Debaleena Misra, and Swagatam Das. “A feature weighted penalty based dissimilarity measure for k-nearest neighbor classification with missing features”. In: *Pattern Recognition Letters* 80 (2016), pp. 231–237.
- [83] Elaine Y Chan et al. “An optimization algorithm for clustering using weighted dissimilarity measures”. In: *Pattern recognition* 37.5 (2004), pp. 943–952.
- [84] Pierre Gañçarski and Alexandre Blansch . “Darwinian, Lamarckian, and Baldwinian (Co) Evolutionary Approaches for Feature Weighting in K-means-Based Algorithms”. In: *IEEE Transactions on Evolutionary Computation* 12.5 (2008), pp. 617–629.
- [85] Kalyanmoy Deb et al. “A fast and elitist multiobjective genetic algorithm: NSGA-II”. In: *IEEE transactions on evolutionary computation* 6.2 (2002), pp. 182–197.
- [86] Salvador Garcia, Juli n Luengo, and Francisco Herrera. *Data preprocessing in data mining*. Springer, 2015.
- [87] Xi Hang Cao and Zoran Obradovic. “A robust data scaling algorithm for gene expression classification”. In: *2015 IEEE 15th International Conference on Bioinformatics and Bioengineering (BIBE)*. IEEE, 2015, pp. 1–4.

- 
- [88] KT Sreekumar et al. “Performance enhancement of the machine-fault diagnosis system using feature mapping, normalisation and decision fusion”. In: *IET Science, Measurement & Technology* 13.9 (2019), pp. 1287–1298.
- [89] Gunhan Park, Yunju Baek, and Heung-Kyu Lee. “Re-ranking algorithm using post-retrieval clustering for content-based image retrieval”. In: *Information processing & management* 41.2 (2005), pp. 177–194.
- [90] Xueheng Qiu et al. “Oblique random forest ensemble via Least Square Estimation for time series forecasting”. In: *Information Sciences* 420 (2017), pp. 249–262.
- [91] Zheng Chu, Jiong Yu, and Askar Hamdulla. “LPG-model: A novel model for throughput prediction in stream processing, using a light gradient boosting machine, incremental principal component analysis, and deep gated recurrent unit network”. In: *Information Sciences* (2020).
- [92] Zhendong Peng et al. “ABFL: An autoencoder based practical approach for software fault localization”. In: *Information Sciences* 510 (2020), pp. 108–121.
- [93] Linhong Zhu, Aixin Sun, and Byron Choi. “Detecting spam blogs from blog search results”. In: *Information processing & management* 47.2 (2011), pp. 246–262.
- [94] ZN Sadough Vanini, Kash Khorasani, and Nader Meskin. “Fault detection and isolation of a dual spool gas turbine engine using dynamic neural networks and multiple model approach”. In: *Information Sciences* 259 (2014), pp. 234–251.
- [95] Fu-Yuan Hsu et al. “Automated estimation of item difficulty for multiple-choice tests: An application of word embedding techniques”. In: *Information Processing & Management* 54.6 (2018), pp. 969–984.
- [96] Paweł Pławiak et al. “DGHNL: A new deep genetic hierarchical network of learners for prediction of credit scoring”. In: *Information Sciences* 516 (2020), pp. 401–418.
- [97] Ajaya Kumar Pani, Krunal G Amin, and Hare Krishna Mohanta. “Soft sensing of product quality in the debutanizer column with principal component analysis and feed-forward artificial neural network”. In: *Alexandria Engineering Journal* 55.2 (2016), pp. 1667–1674.
- [98] Glenn W Milligan and Martha C Cooper. “A study of standardization of variables in cluster analysis”. In: *Journal of classification* 5.2 (1988), pp. 181–204.
- [99] Catherine M Schaffer and Paul E Green. “An empirical comparison of variable standardization methods in cluster analysis”. In: *Multivariate Behavioral Research* 31.2 (1996), pp. 149–167.
- [100] Chia-Wei Chu, John D Holliday, and Peter Willett. “Effect of data standardization on chemical clustering and similarity searching”. In: *Journal of chemical information and modeling* 49.2 (2009), pp. 155–161.

## 1. Introduction

---

- [101] Samit Bhanja and Abhishek Das. “Impact of data normalization on deep neural network for time series forecasting”. In: *arXiv preprint arXiv:1812.05519* (2018).
- [102] DS Kumar. “Feature normalisation for robust speech recognition”. In: *arXiv preprint arXiv:1507.04019* (2015).
- [103] Tshephisho Joseph Sefara. “The effects of normalisation methods on speech emotion recognition”. In: *2019 International Multidisciplinary Information Technology and Engineering Conference (IMITEC)*. IEEE, 2019, pp. 1–8.
- [104] Dalwinder Singh and Birmohan Singh. “Investigating the impact of data normalization on classification performance”. In: *Applied Soft Computing* (2019), p. 105524.
- [105] Boyce Sigweni and Martin Shepperd. “Feature weighting techniques for CBR in software effort estimation studies: a review and empirical evaluation”. In: *Proceedings of the 10th International Conference on Predictive Models in Software Engineering*. 2014, pp. 32–41.
- [106] Robert A van den Berg et al. “Centering, scaling, and transformations: improving the biological information content of metabolomics data”. In: *BMC genomics* 7.1 (2006), p. 142.
- [107] Jan Walach, Peter Filzmoser, and Karel Hron. “Data Normalization and Scaling: Consequences for the Analysis in Omics Sciences”. In: *Comprehensive Analytical Chemistry*. Vol. 82. Elsevier, 2018, pp. 165–196.
- [108] Isao Noda. “Scaling techniques to enhance two-dimensional correlation spectra”. In: *Journal of Molecular Structure* 883 (2008), pp. 216–227.
- [109] Paul J Trim et al. “Matrix-assisted laser desorption/ionisation mass spectrometry imaging of lipids in rat brain tissue with integrated unsupervised and supervised multivariate statistical analysis”. In: *Rapid Communications in Mass Spectrometry: An International Journal Devoted to the Rapid Dissemination of Up-to-the-Minute Research in Mass Spectrometry* 22.10 (2008), pp. 1503–1509.
- [110] Nathan C Hurley et al. “Visualization of Emergency Department Clinical Data for Interpretable Patient Phenotyping”. In: *arXiv preprint arXiv:1907.11039* (2019).
- [111] Kyle D Julian, Mykel J Kochenderfer, and Michael P Owen. “Deep neural network compression for aircraft collision avoidance systems”. In: *Journal of Guidance, Control, and Dynamics* 42.3 (2019), pp. 598–608.
- [112] DK Thara, BG PremaSudha, and Fan Xiong. “Auto-detection of epileptic seizure events using deep neural network with different feature scaling techniques”. In: *Pattern Recognition Letters* 128 (2019), pp. 544–550.
- [113] R Vaitheeshwari and V SathieshKumar. “Performance Analysis of Epileptic Seizure Detection System Using Neural Network Approach”. In: *2019 International Conference on Computational Intelligence in Data Science (ICCIDS)*. IEEE, 2019, pp. 1–5.



- 
- [114] Sourav Kundu and Samit Ari. “Score normalization of ensemble SVMs for brain-computer interface P300 speller”. In: *2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*. IEEE. 2017, pp. 1–5.
- [115] Karianne J Bergen and Gregory C Beroza. “Earthquake fingerprints: Extracting waveform features for similarity-based earthquake detection”. In: *Pure and Applied Geophysics* 176.3 (2019), pp. 1037–1059.
- [116] Dheeru Dua and Casey Graff. “UCI Machine Learning Repository”. In: (2017). URL: <http://archive.ics.uci.edu/ml>.
- [117] Padraig Cunningham and Sarah Jane Delany. “k-Nearest Neighbour Classifiers–”. In: *arXiv preprint arXiv:2004.04523* (2020).
- [118] Anil K Jain. “Data clustering: 50 years beyond K-means”. In: *Pattern recognition letters* 31.8 (2010), pp. 651–666.
- [119] Aenor. “UNE-EN 1426 : Bitumen and bituminous binders—determination of needle penetration”. In: (2015), pp. 1–16.
- [120] BOE-A-2006-2779. *Real Decreto 61/2006, de 31 de enero, por el que se determinan las especificaciones de gasolinas, gasóleos, fuelóleos y gases licuados del petróleo y se regula el uso de determinados biocarburantes. Spain, Ministerio de Industria, Turismo y Comercio.* <https://www.boe.es/buscar/act.php?id=BOE-A-2006-2779&tn=1&p=20060928>. 2015.



## Chapter 2

# Conclusions

The main objective of this thesis is to automatise the data-driven soft-sensors development for refinery's subproduct quality estimation. For doing so, in this thesis a methodology which bases on the proper features' preprocessing techniques and ML algorithms selection and configuration has been proposed, referred to as autoML approach. The proposed autoML approach has demonstrated to meet such goal. In fact, with the autoML approach two soft-sensors that reach precision values higher than 90% have been developed for different real use cases in the context of Oil and Gas 4.0. For the autoML design a thorough analysis of different features' preprocessing techniques has been conducted.

First, Feature Weighting methods have been studied. A general taxonomy based on the formulation of the FW methods which enables the comparison of different approaches has been proposed. In addition, a broad revision of the literature has been conducted and published, which serves as guide for researches about the state-of-the-art in the field. **From the literature revision the main conclusions are twofold. First, it is concluded that FW methods improve the model's performance. Second, for the proper FW method selection both the particular characteristics of the dataset and the practical orientation of the problem at hand must be considered.** Consequently, for such purpose, a recommendation guide has been also proposed. Based on such guide, **it is concluded that the supervised filter global FW methods are the most appropriate ones for the proposal of the autoML approach in the context of Oil and Gas 4.0.**

Feature Normalisation methods have been also investigated. **This thesis concludes that FN does not equalise the features' contribution to the model.** Actually, it has been demonstrated that FN transforms differently the features of a given dataset, altering their contribution to the model, **also concluding that FN is a particular case of unsupervised FW methods.** To be consistent with the filter and wrapper FW methods, this thesis defines the filter and wrapper ML algorithms in terms of the absence or inclusion of internal weights to adjust the features' contribution, respectively. In contrast to the general unconcern about the FN methods, **this thesis concludes that FN considerably impacts not only on the filter but also on the wrapper ML algorithm-based models' performance. Such conclusion enhances the understanding of the influence of FN on the data-driven models and, further, enables the proposal of more realistic features' relevance analysis methods by including also the influence of FN,** as demonstrated in this thesis by the results obtained for the proposal of the adapted Garson's and Yoon's methods.

From the understanding of the implications and the impact of the features'

## 2. Conclusions

---

preprocessing on the ML algorithms **it is concluded the necessity of considering both FW and FN in order to properly represent the features' contribution to the model.** Such conclusion has given place to the proposal of the Two-stage methodology that integrates FN and FW to intelligently transform the features of a given dataset. However, there is not any ideal feature's preprocessing method or ML algorithm for all the use cases. And, furthermore, it has been demonstrated the influence of features' preprocessing on the features' contribution to the ML algorithms and the model's performance. **Therefore, this thesis determines the suitability of a flexible approach to select the proper configuration for the problem at hand from a set of candidate features' preprocessing methods and ML algorithms.** Thus, in this thesis, a flexible methodology that considers different ML algorithms and searches for the best ML algorithms' hyper-parameters configuration based on applicant algorithms and preprocessing techniques is presented. Such methodology, the proposed autoML approach, automatically searches the best features' preprocessing and ML algorithm for the soft-sensor design and development. As previously mentioned, the autoML approach has been validated on two real use cases, developing soft-sensors with promising results.

The first described soft-sensor is designed to complement a costly laboratory test that analyses needle penetration over samples collected from the bottom of a vacuum distillation unit and whose results are available around four hours after the sample collection. Such delay interferes with possible corrective measures over the distillation process and may imply a waste of a significant amount of subproduct that must be reprocessed if the specifications are not fulfilled. Until the date, no data-driven soft-sensor was found for the vacuum distillation bottom product quality estimation. Thus, **it is clear the interest in developing a complementary soft-sensor.** In fact, this thesis contributes to the state-of-the-art by proposing the first soft-sensor for the described application. The proposed soft-sensor is able of continuously detecting with high precision if the subproduct does not meet the standard requirements. In addition, the knowledge extracted from the analysis of the features' preprocessing methods and the two-stage methodology enables the interpretability of the model. As a result, **the consistency** between the most influencing features on the soft-sensors and the key parameters for the physical analysis of the subproduct's quality validates the reliability of the soft-sensor for its applicability.

The second proposed soft-sensor is designed by the autoML approach for the estimation of the percentage of pentanes in butane. Currently, the refineries analyse the concentration of pentanes in butane with an online analyser located at the debutanizer column. Therefore, corrective measures are not applicable until the distilling process has reached the debutanizer column. In the literature some soft-sensors have been proposed to analyse the quality of subproducts obtained at the debutanizer column. However, non of these enable a prompt inference of the subproduct quality. **Then, this thesis states the importance of complementing the online analyser with a soft-sensor that estimates in advance the quality of the subproduct.** For doing so, the soft-sensor

computes its estimations based on the operational variables of the first distilling columns of the analysed unit chain, which represents an advance of 400 minutes with respect to the measurements of the current online analyser. In fact, the results show the high precision of the soft-sensor. This way, **it is concluded that the proposed soft-sensor enables the prompt reprocessing of the subproduct that does not meet the quality requirements, which directly impacts on the refinery's cost saving.**

In addition, through the designed and developed soft-sensors for both real use cases is concluded the validity of the proposed autoML approach for the automatic design and development of soft-sensors for the refineries' subproducts quality estimation.

For all the above-mentioned, this thesis significantly contributes in both the Machine Learning and Oil and Gas 4.0 fields.

## 2.1 Future work

From the analysis and approaches presented in this thesis, many extensions and improvements can be made.

Regarding the analysis of FN impact, filter and wrapper ML algorithms in terms of internal weights have been considered. However, it would be interesting to enlarge the study with further ML algorithms and to include ML algorithms with kernel or activation functions, such as polynomial kernel or ReLu. In addition, in the literature several researches have been conducted in order to find the proper weights initialisation for Artificial Neural Networks. It would be interesting to conduct a similar analysis considering that the dispersion factor of the FN also influences the search space.

Regarding the features' preprocessing approaches in general, it must be noticed that in this thesis the samples are utilised as static data. Therefore, an analysis of features' preprocessing methods for dynamic samples (time series) would also be of interest.

Concerning the use cases, more concretely, those associated to laboratory tests where the labels are scarce, an analysis of semi-supervised features' preprocessing techniques and semi-supervised ML algorithms can enhance the design of soft-sensors.

Finally, due to the evolving nature of the properties of the input raw crude to the distilling process, the inclusion of Concept drift detection techniques or Just-in-time methods into the autoML approach should be considered in the future.



# Papers





## I Feature Weighting methods: A review

**Iratxe Niño-Adan, Diana Manjarres, Itziar Landa-Torres, Eva Portillo**

Published in *Expert Systems with Applications*, June 2021, volume 184, 115424.

DOI: <https://doi.org/10.1016/j.eswa.2021.115424>.

**JCR: 6.954**

*Category: Computer Science, Artificial Intelligence, 23/139, Q1*



# Feature weighting methods: A review

Iratxe Niño-Adan<sup>a</sup>, Diana Manjarres<sup>a</sup>, Itziar Landa-Torres<sup>b</sup>, Eva Portillo<sup>c</sup>

<sup>a</sup>*TECNALIA, Basque Research and Technology Alliance (BRTA), 48160 Derio, Spain*

<sup>b</sup>*Petronor Innovación S.L., 48550 Muskiz, Spain*

<sup>c</sup>*Department of Automatic Control and System Engineering, Faculty of Engineering, University of the Basque Country, UPV/EHU, 48013 Bilbao, Spain*

---

## Abstract

In the last decades, a wide portfolio of Feature Weighting (FW) methods have been proposed in the literature. Their main potential is the capability to transform the features in order to contribute to the Machine Learning (ML) algorithm metric proportionally to their estimated relevance for inferring the output pattern. Nevertheless, the extensive number of FW related works makes difficult to do a scientific study in this field of knowledge. Therefore, in this paper a global taxonomy for FW methods is proposed by focusing on: 1) the learning approach (supervised or unsupervised), 2) the methodology used to calculate the weights (global or local), and 3) the feedback obtained from the ML algorithm when estimating the weights (filter or wrapper). Among the different taxonomy levels, an extensive review of the state-of-the-art is presented, followed by some considerations and guide points for the FW strategies selection regarding significant aspects of real-world data analysis problems. Finally, a summary of conclusions and challenges in the FW field is briefly outlined.

*Keywords:* Feature weighting, Feature importance, Feature relevance, Review.

---

\*Corresponding author. Tel.: +34 607 225 810

*Email addresses:* iratxe.nino@tecnalia.com (Iratxe Niño-Adan), diana.manjarres@tecnalia.com (Diana Manjarres), itziar.landa@repsol.com (Itziar Landa-Torres), eva.portillo@ehu.eus (Eva Portillo)

## 1. Introduction

Machine Learning (ML) algorithms are widely employed to successfully extract patterns and valuable information from data (Bishop et al., 1995). Nevertheless, their performance is highly dependant on the quality of the given dataset.

5 If it contains irrelevant or noisy information, reliable knowledge cannot be easily extracted (García et al., 2015). Consequently, data preprocessing, which transforms the raw data into a useful and understandable format, is a relevant stage in ML algorithms (García et al., 2016).

In the same context, the selection of the representative features that best define  
10 the output behaviour is an important task, which is frequently done with the aid of expert knowledge on the application field and/or by Feature Selection methods (Jović et al., 2015; Li et al., 2018; Saeys et al., 2007; Venkatesh & Anuradha, 2019). Traditionally, it has been assumed that all the selected features are equally important when estimating the output. However, if some  
15 features present higher scale than others, the results can be over-influenced by them, affecting the performance and the accuracy of the overall algorithm (Daszykowski et al., 2007). In order to minimise such dominance, normalisation methods (Jain et al., 2005; Milligan & Cooper, 1988; Panday et al., 2018) are usually employed in order to equalise the contribution of each feature on the  
20 algorithm metric (Aksoy & Haralick, 2001).

Nevertheless, it is widely known that all the features are usually not equally representative of the hidden pattern, especially in real-world problems. In this context, in the last decades a wide portfolio of Feature Weighting (FW) methods have been proposed with the aim of estimating the degree of relevance that each  
25 feature has for extracting the output pattern (Wei et al., 2015). Their main goal is to transform or weigh the features to contribute to the ML algorithm metric proportionally to their estimated relevance.

In this paper an extensive review of the state-of-the-art on FW methods is presented. Moreover, this paper depicts a classification of them focusing on:

1) the learning strategy followed (i.e. supervised or unsupervised), 2) the methodology used to calculate the weights (i.e. global or local), and 3) the feedback obtained from the ML algorithm when estimating the weights (i.e. filter or wrapper). In Section 2 the proposed taxonomy for their classification is shown. Following this taxonomy, Section 3 collects the FW methods proposed in the literature in the last years. Section 4 provides some considerations and guide points for the selection of optimal FW methods based on significant aspects to consider in real-world data analysis problems. To conclude this review, Section 5 gathers a summary of conclusions and challenges that remain unsolved in the FW field.

## 2. Taxonomy

FW methods are techniques that, given a dataset  $X \in \mathbb{R}^{n \times m}$  composed by  $n$  samples described by  $m$  features, obtain a set of weights  $W^*$  representing the relative relevance of the features of  $X$ . The obtained weights  $w_{ij}^*$  for  $i = \{1, \dots, n\}$  and  $j = \{1, \dots, m\}$ , generally ranging from 0 to 1 in such a way that  $\sum_{i,j} w_{ij} = 1$ , multiply each value  $x_{ij}$  of  $X$  in order to create a weighted dataset  $\tilde{X}$  representative of the relative importance each feature has for the given system. In this sense, the higher the weight value, the higher the relative importance of the corresponding feature. Besides, when  $w_{ij} = 0$  the feature weight acts as a Feature Selection factor, discarding non-relevant information from the dataset. The weighted dataset  $\tilde{X}$  is ultimately used by the ML algorithm to model the system.

From this basis, this section describes the taxonomy regarding the FW methods. At a first stage, the FW methods are classified based on the learning approach employed to estimate the weights, namely supervised or unsupervised. The supervised FW strategy refers to the methods that employ the information of the real labels  $Y$  to calculate the features weights. By contrast, unsupervised FW is considered when there is no information about the real labels. Instead, the feature weights are calculated considering other intrinsic characteristics of

the dataset, such as distance between samples of the features or respect to a  
 60 given point. In general, unsupervised FW methods employ clustering algorithms  
 to extract group structures from the dataset and utilise this information to  
 compute the weights.

At a second stage, the FW methods are classified based on the way the weights  
 are applied: global (i.e. over the entire instance space) or local (i.e. over  
 65 different parts of the instance space). Global FW approaches consider that the  
 feature has the same relevance for calculating the output for the whole target  
 population. For each feature  $X_j$ , a single global weight  $w_j$  is calculated, i.e.  $\forall i$   
 $w_{ij} = w_j$ . Local weights are employed when it is assumed that a given feature  
 70 presents different degrees of relevance, depending on its samples or subsets of  
 them, for estimating the output. Thus, more than one weight are assigned  
 to the same feature. In this case, the FW method obtains weights  $w_{gj}$ , with  
 $g \in \{1, \dots, G\}$  and  $1 < G \leq n$ , where  $G$  corresponds to the number of weights  
 assigned to each feature and which is generally equal to the number of classes  
 (supervised) or clusters (unsupervised).

75 Finally, the proposed taxonomy delves into the way the estimation strategy is  
 made. In particular, filter and wrapper methods are distinguished. As Figure  
 1 depicts, filter FW methods calculate the (global or local) feature weights as  
 the relationship between the features and a given reference which corresponds  
 to, based on the learning approach selected, the real labels  $Y$  in the supervised  
 80 case or intrinsic characteristics of the data in the unsupervised one.

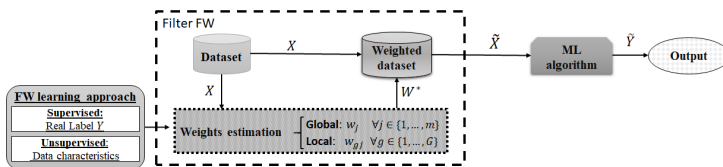


Figure 1: Flow chart of filter FW approaches.

In contrast, as shown in Figure 2, wrapper methods employ feedback from  
 a given ML algorithm to estimate the feature weights (global or local) in a

black-box iterative fashion. Thus, based on the performance obtained in the previous iteration calculated by supervised or unsupervised evaluation metrics, the method decides whether to adjust the weights or not in order to improve the model performance in the next iteration.

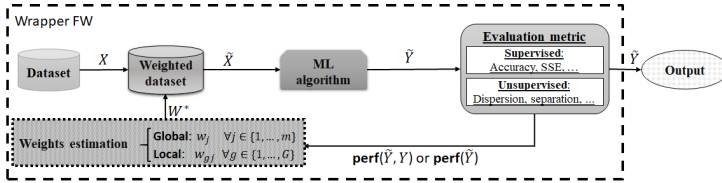


Figure 2: Flow chart of wrapper FW approaches.

In order to provide a comprehensive overview of the paper, Figure 3 depicts the proposed taxonomy of the FW methods, achieved from the combination of the different approaches per level, which is the basis of the next section schema.

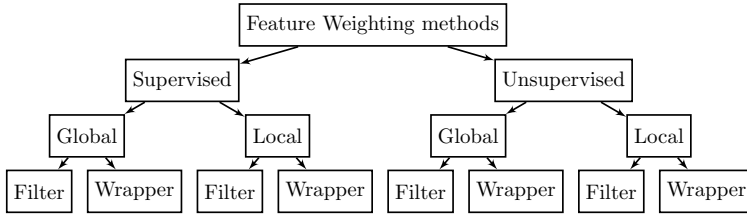


Figure 3: Proposed taxonomy of the FW methods.

### 3. Feature Weighting Methods

In this section an extensive review of FW methods is presented following the taxonomy proposed in Section 2. Besides, pseudo-algorithms are included to describe the main process of each group of FW methods. Note that, in the pseudo-algorithms, the learning approach (supervised/unsupervised) is remarked with underlined text, the way the weights are applied (global/local) with dotted box and the estimation strategy (filter/wrapper) with dashed box.

For the sake of our knowledge, there is no other review work in the literature that covers such a wide classification of FW methods, i.e. the encountered reviews are focused on a particular taxonomy level or on a subset of it, such as: FW for  
100 K-NN algorithms (Wettschereck & Aha, 1995; Wettschereck et al., 1997), FW for the K-means algorithm (de Amorim, 2016) or FW for different clustering methods but employing the wrapper methodology (Deng et al., 2016).

In this work special attention will be paid to particular characteristics and objectives of the problem at hand: (1) Labels' availability when is possible  
105 to obtain the labels. (2) High-dimensional dataset characterised by a high number of features compared to the number of observations. (3) Dimensionality reduction for those problems that require the reduction of the number of features due to high computational cost or other similar reasons. (4) Dataset understanding for those problems in which the main interest is to provide the  
110 domain expert with information about the influence of each feature on the output from the system/application's perspective. (5) Features contribution on the model for those problems in which the main objective is to infer the influence of every feature on the performance of the ML based model. (6) Missing values commonly caused by improper data collection or data acquisition  
115 fails. (7) Imbalanced dataset when the dataset has an unequal distribution of classes. (8) Outliers when the dataset contains observations that significantly deviate from most observations. (9) Noise when meaningless or corrupted features are introduced in the dataset. (10) Interpretability for those problems in which the main objective is to extract knowledge from the ML based model  
120 from the system/application's perspective. (11) Condition-based problems in which the relevance of the features varies depending on the operating condition of the system (for instance, type of material or material thickness). (12) Temporal dependency commonly found when time series are required to deal with the modelling task of interest. (13) Algorithm performance maximisation  
125 for those problems in which the main objective is to optimise the ML algorithm performance regardless the interpretability of the model. (14) Semi-supervised



learning if the dataset comprises both labelled and unlabelled samples. (15)  
Online learning for those systems whose properties change along the time.

### 3.1. Supervised Feature Weighting

130 This section presents the works in the literature focused on supervised feature weighting. As described previously, it takes advantage of the real labels  $Y$  to estimate the feature weights. Depending on the number of weights calculated per feature, global or local approaches can be distinguished.

#### 3.1.1. Global Supervised Feature Weighting

135 Global feature weighting methods look for the optimal weight  $w_j^*$  to be assigned to each feature  $X_j$  based on its relevance for estimating the output pattern. Since supervised learning strategy is considered, this relevance is computed as the relationship between the feature and the labels. The calculation of the global weights can be done in a filter or a wrapper approach.

140 *Filter Global Supervised Feature Weighting.* In the case of filter global supervised FW methods, the weights estimation is done by the employment of techniques on the field of Variable Importance Analysis (V.I.A.), also called Sensitivity analysis (Christopher Frey & Patil, 2002; Wei et al., 2015). As presented in Algorithm 1, the global weights  $w_j^*$  are calculated in line 3 utilising  
145 V.I.A methods to represent the importance of each feature  $X_j$  in terms of its relationship with the label  $Y$ . Once calculated the weights, these multiply each feature of the dataset (line 5) creating the weighted dataset  $\tilde{X}$  which is ultimately passed to the ML algorithm (line 6). These works Christopher Frey & Patil (2002); Wei et al. (2015) present an excellent introduction and description  
150 of the basis and terminology associated to Variable Importance Analysis.

In this context, two main approaches can be found to perform the Variable Importance Analysis (V.I.A.) in line 3 of Algorithm 1: one based on Information theory, and another one related to Statistical-based methods.

---

**Algorithm 1** Filter Global Supervised Feature Weighting methods

---

1: Given a dataset  $X \subset \mathbb{R}^{n \times m}$  and **the labels**  $Y$

2: **for**  $j = 1 : m$  **do**

3:  $w_j^*$   $\leftarrow$   $V.I.A.(X_j, Y)$

4: **for**  $j=1:m$  **do**

5:  $\tilde{X}_j \leftarrow w_j \cdot X_j$

6:  $\tilde{Y} \leftarrow$  ML algorithm to  $\tilde{X}$

---

Regarding the former, a commonly used V.I.A. measure employed for weight  
155 estimation is Mutual Information (MI) defined as

$$I(X_j, Y) = \sum_{x_{ij} \in X_j} \sum_{y_i \in Y} P(x_{ij}, y_i) \cdot \log \left( \frac{p(x_{ij}, y_i)}{p(x_{ij})p(y_i)} \right) \quad (1)$$

where  $p(\cdot)$  is the probability distribution function and  $p(X_j, Y)$  the conditional  
probability distribution function of  $X_j$  and  $Y$ . Several works apply Equation 1  
to estimate the weights in line 3 of Algorithm 1 (García-Laencina et al., 2009;  
Wettschereck & Dietterich, 1995). For instance, the Linear Feature Weighted  
160 SVM (LFWSVM) algorithm (Xing et al., 2009), the MI-MCS-FWSVM method  
(Giveki et al., 2012), the Correlation-based Feature Weighting (CFW) filter  
method (Jiang et al., 2018) and the proposal (Hussain, 2019) estimate the  
features relevance respect to the label  $Y$  and calculate the weights employing  
the MI measure in line 3 of Algorithm 1. Similarly to the MI measure, the  
165 Information Gain (IG) and the Regularised Entropy (re) measures are employed  
in (Chen & Hao, 2017; Wu et al., 2017), respectively.

From the variable importance analysis field, the employment of Statistical-based  
methods to estimate the feature relevance is another extended approach. In  
(Sahin et al., 2015)  $\chi^2$  and Fisher (F-score) algorithms are employed to measure  
170 in line 3 of Algorithm 1 the quality of the features respect to the label. Likewise,  
Granger causality (Granger, 1988) and AHP (Saaty, 2014) are applied in  
(Bhattacharya et al., 2017) to improve the performance of the K-NN algorithm

(line 6 of Algorithm 1).

Heretofore, all the presented FW methods estimate the weights for each feature  
175 in an independent manner. However, as shown in (Elbasiony et al., 2013;  
Niño-Adan et al., 2019), methods that estimate the weights in a group manner,  
i.e. using the Random Forest (RF) algorithm, are also utilised as filter FW  
methods.

Although in most cases the main goal of the filter global supervised FW methods  
180 is to increase the accuracy of the model, some authors make use of the filter FW  
approaches for other purposes. This is the case of the proposed Fast Feature  
Weight algorithm for Data Gravitation Classification model (FFW-DGC) (Peng  
et al., 2017) which aims at *reducing the computational complexity* of the FW  
process in the DGC model. In this work the feature weights (line 3 of  
185 Algorithm 1) are calculated combining two fuzzy sets: 1) one relative to the  
feature discrimination capability, computed by means of the MI (Equation 1)  
between the discretized features and the labels, and 2) the second one related  
to the redundancy between features, calculated by Pearson correlation as the  
covariance of  $X_j$  and  $Y$  between their standard deviations:

$$\rho(X_j, Y) = \frac{COV(X_j, Y)}{\sigma_{X_j} \sigma_Y} \quad (2)$$

190 Finally, due to the presence of non-independent features in real problems,  
another common use of filter FW methods is to estimate the feature weights in  
order to alleviate the conditional independence assumption of the Naive Bayes  
(NB) algorithm (line 6 of Algorithm 1). In this context, the approach (Zhang  
et al., 2016) adapts two filter FW approaches for NB classifier in the field of text  
195 classification. The first approach employs in line 3 of Algorithm 1 Gain Ratio  
(GR) (Zhang & Sheng, 2004) assuming that the features take zero or nonzero  
values, while the second one obtains the feature weights (line 3 of Algorithm 1)  
from a Decision Tree (DT) classifier (Hall, 2007).

200 *Wrapper Global Supervised Feature Weighting.* Regarding the wrapper FW approach, as described in Algorithm 2 the weights  $w_j^*$  are first initialised with random values (line 2 of Algorithm 2) and then, adjusted (line 10 of Algorithm 2) in an iterative fashion (line 4 of Algorithm 2) considering the relationship between the estimated output  $\tilde{Y}$  and the real label  $Y$  in terms of a given supervised evaluation metric that measures the performance of the ML algorithm respect to the given label, i.e.  $\mathbf{perf}(\tilde{Y}, Y)$ . The iterative process ends after a predefined number of iterations  $n\_iter$ , when the performance outperforms a given threshold  $\theta$  or when the wrapper method converges, i.e. it does not present significant improvement respect to the previous iteration in terms of the performance measure (line 8 of Algorithm 2).

---

**Algorithm 2** Wrapper Global Supervised Feature Weighting methods

---

```

1: Given a dataset  $X \subset \mathbb{R}^{n \times m}$  and the labels  $Y$ 
2: Initialise  $w_j^{(0)} \in \mathbb{R}$ 
3:  $s \leftarrow 0$ 
4: while  $s < n\_iter$  do
5:   for  $j=1:m$  do
6:      $\tilde{X}_j^{(s)} \leftarrow w_j^{(s)} \cdot X_j$ 
7:    $\tilde{Y}^{(s)} \leftarrow$  ML algorithm to  $\tilde{X}^{(s)}$ 
8:   if  $\mathbf{perf}(\tilde{Y}^{(s)}, Y) > \theta$  or  $\mathbf{perf}(\tilde{Y}^{(s-1)}, Y) + \epsilon$  then
9:     Break
10:   $w_j^{(s+1)} \leftarrow$  Adjust  $w_j^{(s)}$ 
11:   $s \leftarrow s + 1$ 
12:  $w_j^* \leftarrow w_j^{(s)}$ 

```

---

210 The most extended practice to configure the wrapper strategy is the employment of a Genetic Algorithm (GA) for the weights calculation in order to improve the performance of the model in terms of accuracy (Komosiński & Krawiec, 2000; Phan et al., 2017).

The flexibility for encoding the individuals in the GA algorithm offers multiple

215 possibilities. For instance, the work (Ahn & Kim, 2009) proposes a Global  
Optimisation of Case-Based Reasoning (CBR), a hybrid model that employs  
a GA algorithm to simultaneously optimise the feature weights (line 10 of  
Algorithm 2), the instance selection and the  $K$  value for the K-NN algorithm  
based on the classification accuracy (line 8 of Algorithm 2). The authors employ  
220 the Wisconsin breast cancer image dataset from UCI repository in order to  
validate their method and conclude that the simultaneous optimisation of the  
feature weights, the instance selection and the  $K$  value obtains the highest  
accuracy (line 8 of Algorithm 2) for this problem.

Apart from the GA algorithm, other Evolutionary Algorithms (EA) are also  
225 employed for Feature Weighting (lines 2-11 of Algorithm 2). For instance,  
Mateos-García et al. (2017) present the Simultaneous Weighting of Attributes  
and Neighbours (SWAN) that employs an EA for adjusting the contribution  
of the neighbours and the significance of the features that minimise the  
cross-validation error (line 8 of Algorithm 2). Similarly, the work (Triguero  
230 et al., 2012) proposes a self-adaptive Differential Evolution (DE) algorithm in  
order to optimise the feature weights (line 12 of Algorithm 2) that maximise  
the performance of the K-NN for prototype generation (line 8 of Algorithm 2).  
In (Sotoodeh et al., 2019) a Particle Swarm Optimisation (PSO) algorithm is  
employed to generate the optimum feature weight vector (line 12 of Algorithm  
235 2) in terms of image retrieval system performance. In the same context, the  
research (Serrano-Silva et al., 2018) presents a performance comparison of DE,  
GA and Novel Bath Algorithm (NBA) for FW in terms of AUC and execution  
time over several financial datasets.

Nevertheless, other approaches, such as the Dynamic Representation and  
240 the Neighbour Sparse Reconstruction-based Relief (DRNSR-Relief) presented  
in (Huang et al., 2018b), decompose the nonlinear problem into locally  
linear problems. Specifically, the proposed algorithm represents the dynamic  
relationship between the margin and the weight vectors. In this proposal,  
the Gradient Ascent method is employed to calculate the expected margin

245 vector and to update the feature weights in line 10 of Algorithm 2 in a  
wrapper fashion. Similarly, Yang et al. (2012) employ the Gradient Ascent  
update method to estimate the feature weights that maximises the expected  
leave-one-out classification accuracy (line 8 of Algorithm 2). The same approach  
is applied by Raghu & Sriraam (2018) for the classification of EEG signals and  
250 by Romeo et al. (2020) for the prediction of heterogeneous machine parameters  
in Industry 4.0. Finally, Ouyed & Allili (2020) present a Multinomial Kernel  
Logistic Regression with Group of Features Relevance (GFR-MKLR) approach  
for human interaction recognition. The authors include into the kernel and the  
loss function the gestures' weights, estimated during the training phase utilising  
255 as wrapper FW approach the Newton-Raphson optimisation method.

### 3.1.2. Local Supervised Feature Weighting

Local FW methods aim at obtaining more than one optimal weight per feature,  
since it is assumed that the degree of relevance of each feature depends on the  
sample (or subsets of samples). Similarly to global feature weighting methods,  
260 the weights can be estimated in a filter or a wrapper approach.

*Filter Local Supervised Feature Weighting.* In contrast to global FW methods,  
very few works introduce a new filter approach for estimating local weights in a  
supervised fashion. In fact, most of them adapt the global methods to the local  
environment. In this case, as Algorithm 3 shows, the filter local supervised FW  
265 methods estimate per feature as many weights as classes  $C$  recorded in  $Y$ , and  
the weights in line 4 of Algorithm 3 are computed according to the distribution  
of the samples  $x_{ij}$  into the different classes. After the weights calculation, in line  
7 of Algorithm 3 each sample of the dataset is multiplied by the corresponding  
weight. The weighted dataset  $\tilde{X}$  is then employed by the ML algorithm (line 8)  
270 to ultimately estimate the output  $\tilde{Y}$ .

For instance, Marchiori (2013) proposes a decomposition of the well-known  
RELIEF online method, which processes the samples in a serial way one-by-one,  
into class dependant feature weights vectors (line 2 of Algorithm 3), in which

---

**Algorithm 3** Filter Local Supervised Feature Weighting methods

---

1: Given a dataset  $X \subset \mathbb{R}^{n \times m}$  and the labels  $Y \in \{1, \dots, C\}^n$

2: **for**  $c = 1 : C$  **do**

3:   **for**  $j = 1 : m$  **do**

4:      $w_{cj}^* \leftarrow \text{Distribution}(\{x_{ij} | y_i = c\})$

5: **for**  $j=1:m$  **do**

6:   **for**  $c = 1 : C$  **do**

7:      $\tilde{X}_{i'j} \leftarrow w_{cj}^* \cdot X_{i'j} \ \forall i' \in \{i | y_i = c\}$

8:  $\tilde{Y} \leftarrow$  ML algorithm to  $\tilde{X}$

---

each vector describes the relevance of features conditioned to each class (line 275 4 of Algorithm 3). Each class-dependant term generates a weighted distance by enlarging the sample margin of the corresponding class. Experiments conducted over two breast cancer datasets from UCI repository show that the accuracy obtained by the proposed decomposition is similar or superior than the reached one by the traditional RELIEF method. A similar work can be found in (Yilmaz et al., 2014) in which a RELIEF-based modality weighting 280 approach, named RELIEF-MM, is proposed for fusing multimodal information in multimedia data. Here, the authors convert the original RELIEF-f algorithm into a class-specific representation. The final values of the modality weights are obtained by grouping the training examples according to the classes and 285 processing samples of each class separately (line 4 of Algorithm 3).

In regards to the adaptations of commonly known filter methods, Chen & Guo (2015) reformulate the Simple Matching Coefficient (SMC) (line 8 of Algorithm 3) and propose the Weighted SMC distance (WSMD) (line 4 of Algorithm 3) aiming at including a supervised measurement of the contribution of the 290 categorical features (Entropy or Gini diversity index) for a global and a local approach. Experiments over real-world datasets support that the local weighting approach outperforms the accuracies obtained by the global proposal in *high*

dimensional datasets.

*Wrapper Local Supervised Feature Weighting.* Similar to the global approaches, as Algorithm 4 presents, the local wrapper supervised FW methods adjust the randomly initialised (line 1 of Algorithm 4) weights  $w_{cj}^{(s)}$  in an iterative fashion (line 11 of Algorithm 4) in order to maximise the performance of a given ML algorithm. In the local case, there are as many weights per feature as classes  $C$  and each sample of the dataset  $x_{ij}$  is multiplied by the local weight associated to the corresponding sample label  $y_i$  (line 7 of Algorithm 4). Once the iterative process ends (line 4 of Algorithm 4), the performance improves a given threshold  $\theta$  or the performance converges (line 9 of Algorithm 4), the lastly adjusted weights are selected as the optimal ones (line 13 of Algorithm 4).

---

**Algorithm 4** Wrapper Local Supervised Feature Weighting methods

---

```

1: Given a dataset  $X \subset \mathbb{R}^{n \times m}$  and the labels  $Y \in \{1, \dots, C\}^n$ 
2: Initialise  $w_{cj}^{(0)} \in \mathbb{R} \quad \forall c = 1 : C, \forall j = 1 : m$ 
3:  $s \leftarrow 0$ 
4: while  $s < n\_iter$  do
5:   for  $j=1:m$  do
6:     for  $c = 1 : C$  do
7:        $\tilde{X}_{i'j}^{(s)} \leftarrow w_{cj}^{(s)} \cdot X_{i'j} \quad \forall i' \in \{i | y_i = c\}$ 
8:        $\tilde{Y}^{(s)} \leftarrow$  ML algorithm to  $\tilde{X}^{(s)}$ 
9:       if  $\text{perf}(\tilde{Y}^{(s)}, Y) > \theta$  or  $\text{perf}(\tilde{Y}^{(s)}, Y) = \text{perf}(\tilde{Y}^{(s-1)}, Y) + \epsilon$  then
10:         Break
11:        $w_{cj}^{(s+1)} \leftarrow$  Adjust  $w_{cj}^{(s)}$ 
12:        $s \leftarrow s + 1$ 
13:  $w_{cj}^* \leftarrow w_{cj}^{(s)}$ 

```

---

Regarding wrapper local supervised approaches, Paredes & Vidal (2000) introduce the Class-Dependant Weighted (CDW) dissimilarity measure for NN classification (line 9 of Algorithm 4). The weights (line 11 of Algorithm 4)



are obtained through Fractional-Programming Gradient Descent (FPGD)-based  
minimisation of the ratio between intra-class and inter-class distances. Similarly,  
Paredes & Vidal (2006) reformulate the objective function as the sigmoid  
310 function of the ratios between intra-class and inter-class distances to emphasise  
the importance of the prototypes that are close to the boundaries. In line  
with this research, authors in (Ren et al., 2019) propose the Learning of  
Reduced Prototypes and Local Metric (LRPLM) that simultaneously learns  
a set of prototypes and optimal local feature-wise metric to minimise in line 9  
315 of Algorithm 4 the image set classification error probability.

However, Jiao et al. (2015) claim that the distance metric proposed by Paredes  
& Vidal (2000); Paredes & Vidal (2006) is not sufficient for characterising  
the particularities of the different classes in the feature space. Thus, the  
Evidential K-Nearest Neighbour classification method with Weighted attributes  
320 (WEK-NN) method (Jiao et al., 2013) is extended to propose a more general  
distance metric, named Class-Conditional Weighted (CCW), which is related  
to both the class labels of the prototypes and the query patterns. The idea  
of the CCW metric is employed few years later by the same authors in (Jiao  
et al., 2019). In this case, a new KNN-based classifier, called BPkNN, is  
325 developed based on pairwise distance metrics and Belief function theory. Instead  
of learning a global distance metric, it is decomposed into a group of pairwise  
distance metrics and K-NN (PkNN) sub-classifiers are adaptively designed to  
separate the classes.

Another related approach can be found in (Taheri et al., 2014) where the authors  
330 propose a novel Attribute Weighting method for a feature weighted NB classifier  
(AWNB), in which for each feature, a different weight per class is calculated.  
An objective function based on the structure of the NB for binary classification  
problems (line 8 of Algorithm 4) is modelled to optimise the feature weights  
by means of the quasisecant method. Similarly, Jiang et al. (2019), based  
335 on Weighting attributes to Alleviate Naive Bayes' Independence Assumption  
(WANBIA) (Zaidi et al., 2013), introduce the Class-specific Attribute Weighted

Naive Bayes (CAWNB) model, in which local weights are included into the conditional probability calculations of the Naive Bayes to consider the relative importance of the  $j^{\text{th}}$  feature for the given class. Then, based on the objective  
340 function, the authors present two CAWNB-based proposals to improve the prediction performance: CAWNB<sup>CLL</sup> that maximises in line 9 of Algorithm 4 the Conditional Log Likelihood (CLL) and CAWNB<sup>MSE</sup> which minimises (line 9 of Algorithm 4) the Mean Square Error (MSE).

Evolutionary algorithms are also employed to estimate the local weights in  
345 the wrapper supervised approach. For instance, Mohemmed & Zhang (2008) analyse the performance of the Particle Swarm Optimisation (PSO) employing different distance measures: the Euclidean, the class-dependant Mahalanobis and the CDW proposed in (Paredes & Vidal, 2006) for reducing the classification error of the Nearest Centroid Classifier (NCC) (lines 8 and 9 of Algorithm 4,  
350 respectively). The authors conclude that the PSO based NCC (PSO-NCC) approach performs well in this particular classification task. Furthermore, the k-Labels Dependant Evolutionary Distance Weighting (kLDEDW) presented in (Mateos-García et al., 2012) employs the Differential Evolution (DE) algorithm to optimise the local weights (line 11 of Algorithm 4) and the optimal value  
355 of  $K$  for the K-NN algorithm. The analysis conducted by the authors confirms that the proposed approach outperforms, in terms of accuracy, five classification algorithms including two above-mentioned local weighting methods: CDW based on Gradient Descent (Paredes & Vidal, 2006) and (AlSukker et al., 2010) which employs DE to estimate the weights. More recently, the work of Sinciya & Celin (2017) improves the performance of the Data Gravitation Classification (DGC) algorithm (line 8 of Algorithm 4) for high dimensional data by the  
360 employment of a PSO-based local feature weighting optimisation.

### *3.1.3. Concluding Remarks about Supervised Approaches*

Regarding supervised FW methods, it can be highlighted that filter global  
365 approaches are the most commonly employed ones in the literature. In this context, two approaches, namely Information theory and Statistical-based, can

be further distinguished. Most of the Information-based FW approaches need to know in advance the probability distribution of the features, which in real world continuous problems is hard to obtain. To circumvent this, a normal  
370 distribution of the continuous variables can be assumed, but this is not always realistic. Therefore, discretisation techniques are commonly applied with the consequent loss of information. On the other hand, the Statistical-based FW methods compute the relation between the features and the label by means of statistical measures. Consequently, the selection of the statistical tests is of vital  
375 importance for these approaches.

In contrast, wrapper methods do not have such above-mentioned limitations and the flexibility in the problem configuration enables its application for local FW strategies. In this regard, the selection of the ML algorithm, its configuration and encoding is crucial for the proper performance of the method and for not  
380 getting stuck in a local optimum. However, there is a lack of an extensive empirical evaluation of wrapper methods and, in particular, a comparison between filter and wrapper methods for the same problem. As wrapper methods employ feedback from a ML algorithm to evaluate alternatives based on an external validation measure, they are widely recognised to obtain *better results*  
385 *than filter approaches*. Nevertheless, due to the iterative search process, wrapper methods require a moderate complexity which results in a *high computational cost*, especially with more exhaustive search strategies or in *high-dimensional datasets*. Besides, the weights resultant from wrapper approaches are high dependant on the algorithm configuration and their contribution to the ML  
390 algorithm metric can be higher than their real relevance for estimating the output. In this sense, the filter approaches, which calculate the weights separately for each feature without the influence of the specific configuration of the algorithm, enable an *interpretation of the feature relevance* in terms of the employed Information theory or Statistical-based technique.

395 *3.2. Unsupervised Feature Weighting*

In this section, a review about FW methods framed within the unsupervised approach is presented. As there is no information about the real labels, the weight estimation is performed in most cases by means of certain characteristics and relations of the features or by the obtained cluster structure. These characteristics are generally calculated by means of unsupervised evaluation metrics. The works of Palacio-Niño & Berzal (2019) and Rendón et al. (2011) present an excellent introduction and description of the basis and terminology associated to clustering algorithms.

*3.2.1. Global Unsupervised Feature Weighting*

405 Global FW methods calculate a weight  $w_j$  per feature  $X_j$ ,  $j = \{1, \dots, m\}$ . However, since the unsupervised learning strategy does not consider  $Y$  to estimate the features relevance, in these cases the relative importance of each feature is estimated according to certain clustering structure obtained from the dataset.

410 *Filter Global Unsupervised Feature Weighting.* After analysing in detail the filter global unsupervised related literature, it must be highlighted that very few works follow a filter approach for estimating the global weights in an unsupervised fashion. Algorithm 5 describes the general procedure of this approach. Generally, first a clustering algorithm is applied to group the data samples into different partitions (line 2 of Algorithm 5), and then, as shown in line 4 of Algorithm 5, based on the obtained cluster structure the feature weights are calculated as the relation between the features and the extracted structures. Then, each weight multiplies the corresponding feature of the dataset (line 6) and the obtained  $\tilde{X}$  is passed in line 6 of Algorithm 5 to the ML algorithm.

420 An example can be found in (Gürüler, 2017) where the authors propose a novel hybrid algorithm, entitled KMCFW-CVANN, that combines the K-Means Clustering algorithm (line 2 of Algorithm 5) for Feature Weighting and a Complex-Valued Artificial Neural Network (line 7 of Algorithm 5) for Parkinson

---

**Algorithm 5** Filter Global Unsupervised Feature Weighting methods

---

- 1: Given a dataset  $X \subset \mathbb{R}^{n \times m}$  and **a clustering algorithm**
  - 2:  $centroids \leftarrow$  Clustering algorithm to  $X$
  - 3: **for**  $j = 1 : m$  **do**
  - 4:  $w_j^* \leftarrow \text{Relation}(X_j, centroids_j)$
  - 5: **for**  $j=1:m$  **do**
  - 6:  $\tilde{X}_j \leftarrow w_j^* \cdot X_j$
  - 7:  $\tilde{Y} \leftarrow$  ML algorithm to  $\tilde{X}$
- 

disease diagnosis. The feature weights in line 4 of Algorithm 5 are calculated  
425 as the ratios of the means of the features to the centroids. For the classification  
phase the CVANN (line 7 of Algorithm 5) is applied. The results of the  
experiments show that the KMCFW-CVANN reaches the highest diagnosis  
accuracy in comparison with the other diagnosis approaches from the literature  
(Polat, 2012; Sakar & Kursun, 2010) over the Parkinson dataset (Little et al.,  
430 2007).

Similarly, Güneş et al. (2010) present the K-Means Clustering based Feature  
Weighting (KMCFW) method. Their proposal first extracts frequency domain  
features for which the mean, minimum, maximum and standard deviation are  
computed as statistical features. In the second stage, (line 2 of Algorithm 5)  
435 the features are clustered by the K-means algorithm and the ratios of means  
of the features with respect to the obtained centroids are employed as feature  
weights (line 4 of Algorithm 5).

*Wrapper Global Unsupervised Feature Weighting.* The wrapper approach is the  
most applied procedure when considering unsupervised global FW problems. In  
440 this context, the weights are commonly embedded into the clustering objective  
function and both the cluster structure and the feature weights are iteratively  
obtained. Thus, as presented in Algorithm 6, for each feature  $X_j$  a weight  
 $w_j$  is associated. The weights are included into the ML algorithm (line 7 of

Algorithm 6) by multiplying the features of the dataset (line 6 of Algorithm  
 445 6) and depending on the performance obtained by the ML algorithm (line 9 of  
 Algorithm 6) the weights are adjusted.

---

**Algorithm 6** Wrapper Global Unsupervised Feature Weighting methods

---

```

1: Given a dataset  $X \subset \mathbb{R}^{n \times m}$ 
2: Initialise  $w_j^{(0)} \in \mathbb{R}$ 
3:  $s \leftarrow 0$ 
4: while  $s < n\_iter$  do
5:   for  $j=1:m$  do
6:      $\tilde{X}_j^{(s)} \leftarrow w_j^{(s)} \cdot X_j$ 
7:      $\tilde{Y}^{(s)} \leftarrow$  ML algorithm to  $\tilde{X}^{(s)}$ 
8:      $w_j^{(s+1)} \leftarrow$  Adjust  $w_j^{(s)}$ 
9:     if  $\text{perf}(\tilde{Y}^{(s)}) > \theta$  or  $\text{perf}(\tilde{Y}^{(s)}) = \text{perf}(\tilde{Y}^{(s-1)}) + \epsilon$  then
10:      Break
11:    $s \leftarrow s + 1$ 

```

---

One of the first works that follows a wrapper global unsupervised FW approach is (Huang et al., 2005), in which the authors adapt the K-means algorithm by means of including the feature weights into the formula giving place to the Weighted K-Means (W-k-means). In this work, partial optimisation is employed to iteratively obtain the samples membership for the different clusters (line 7 of Algorithm 6), to estimate the new centroids and finally, to recalculate the weights (line 8 of Algorithm 6) based on the current partition of the data. In order to obtain compact clusters, it considers the intra-cluster dispersion of the different clusters for the calculation of the weights in line 8 of Algorithm 6 as:

$$w_j = \frac{1}{\sum_{t \in H} [D_j / D_t]^{\frac{1}{\beta-1}}} \quad (3)$$

where  $D_j$  is the sum of the intra-cluster dispersion of the different  $K$  clusters in the feature  $j$  and  $H$  the set of features with  $D_j \neq 0$ . As it can be observed,

$w_j$  is dependant on a user-defined parameter  $\beta$ . Thus, the authors analyse the  
450 range of values that can be assigned to  $\beta$ , concluding that it is recommended  
to take values of  $\beta < 0$  or  $\beta > 1$ . Experiments over synthetic and real data  
sets validate the proposal and conclude that the W-k-means can effectively  
distinguish between noisy and normal features.

Since the W-k-means, several authors have proposed a new version of it for  
455 different applications. Thus, in (Chen et al., 2009) the W-k-means is adapted  
to text clustering, while Hung et al. (2011) employ a variant of it for colour  
image segmentation. Similarly, the proposal presented in (Saha & Das, 2015)  
adapts the W-k-means to the fuzzy K-modes algorithm (line 7 of Algorithm 6)  
in order to handle categorical data. In this case, the W-k-means distance and  
460 the degree of membership of the samples in the different clusters take values  
in  $\{0, 1\}$ . In the case of the Minkowski W-k-means (MW-k-means) approach  
proposed by De Amorim & Mirkin (2012), the Minkowski metric is employed  
as distance function in the K-means algorithm (line 7 of Algorithm 6) and the  
 $\beta$  parameter is fixed equal to the one utilised in the Minkowski distance and  
465 learned in a *semi-supervised* manner. Additionally, the authors propose the  
anomalous cluster step to intelligently set  $K$  and initialise the centroids. The  
proposal of De Amorim & Mirkin (2012) is employed later in (Panday et al.,  
2018) where the authors utilise feature weighting as an unsupervised *feature*  
*selection* tool.

470 Another adaptation is presented in (Chakraborty & Das, 2018) where the  
authors propose a version of (Huang et al., 2005) for the Gaussian Means  
algorithm which also estimates the optimal number of clusters in the iterative  
process. The proposed algorithm starts with a single cluster containing all  
the samples. At the beginning of each iteration the objective function of the  
475 W-k-means is calculated (line 7 of Algorithm 6) and the Anderson-Darling  
normality test is computed for the current centroids. If a cluster is not Gaussian,  
the centroid is replaced by two new centroids as described in (Hamerly & Elkan,  
2004). This iterative process ends when the objective function improvement is

lower than a predefined threshold (line 9 of Algorithm 6).

480 Following with the same line of research, the Feature Weight Self-Adjustment (FWSA) mechanism presented in (Tsai & Chiu, 2008), removes the  $\beta$  parameter in the objective function. The optimisation process is similar to (Huang et al., 2005), but the weights are updated at each iteration ( $s$ ) in line 8 of Algorithm 6 by adding an adjustment margin in the following manner:

$$w_j^{(s+1)} = w_j^{(s)} + \frac{b_j^{(s)}/a_j^{(s)}}{\sum_{j=1}^m b_j^{(s)}/a_j^{(s)}} \quad (4)$$

485 where, for the  $j^{\text{th}}$  feature,  $a_j, b_j$  are the sum of separations inter and intra clusters, respectively. Therefore, this proposal not only searches for compactness but also considers the separation between the clusters. The authors analyse the performance of their proposal over synthetic and real-world datasets. They also compare the FWSA against the W-k-means in terms of SSE, Entropy, ARI and number of iterations until convergence. Although FWSA is *computationally*  
 490 *more expensive*, it outperforms the W-k-means. In addition, the authors remark that the proposed algorithm do not need user-defined parameters as  $\beta$ .

Unsupervised wrapper FW methods have been also proposed to handle different kind of data. In (Benkabou et al., 2018) the Detection of Outlier Time Series  
 495 (DOTS) adapts the K-medians for time series clustering by minimising an Entropy and Dynamic Time Warping (DTW) based objective function (line 7 of Algorithm 6). In this proposal, the weights are assigned to each time series and the DTW between the time series and the medians of the clusters is employed to compute the distance in line 7 of Algorithm 6. In addition, the  
 500 penalised entropy of the weights is added to the objective function in order to control the distribution of the weights. The optimisation process is done in a partial manner, such as in (Huang et al., 2005), but the weights in line 8 of



Algorithm 6 are estimated as:

$$w_j = \frac{e^{-D_j/\lambda}}{\sum_{t=1}^m e^{-D_t/\lambda}} \quad (5)$$

where  $\lambda$  is the regularisation term for the weight entropy. The idea is to  
 505 assign small weights to *time series* that increase the intra-cluster distances and  
 therefore, are considered as outliers. Several experimental comparisons over  
 temporal data from datasets collected from the UCR repository are conducted  
 to evaluate the performance of the proposed approach in contrast to *outlier*  
 detection algorithms, such as: DTW+KM (Budalakoti et al., 2008), DTW+HC  
 510 (Portnoy, 2000), DTW+Spectral (Ng et al., 2002), DL-OCSVM (Bevilacqua  
 & Tsafaris, 2015), FD-OCSVM (Schölkopf et al., 2000) DTW+LOF (Breunig  
 et al., 2000) and Parzen-Window (Yeung & Chow, 2002).

Finally, unsupervised global FW methods for multi-view clustering can be found  
 in the literature. Authors Wang & Chen (2017) propose a multi-view fuzzy  
 515 clustering approach in which the weights are updated in line 8 of Algorithm  
 6 based on the weighted distance of the data points of each view and the  
 corresponding centroid. Similarly, Zhang et al. (2018) introduce the Two-level  
 Weighted Collaborative k-means (TW-CO-k-means) approach for analysing  
 data from multiple sources or views, while satisfying the consistency across  
 520 different views and the diversity within each view. The views and the features  
 in each view are assigned with weights that reflect their importance (line 6 of  
 Algorithm 6). The two-level methodology employs an objective function in line  
 7 of Algorithm 6 that considers the feature weighted intra-cluster dispersion  
 at each weighted view, a penalty term that measures the disagreement across  
 525 multiple views and two weight-entropy-based terms that adjust the distribution  
 of the weights.

### 3.2.2. Local Unsupervised Feature Weighting

Local unsupervised feature weighting methods estimate more than one weight  
 $w_{ij}$  per feature  $X_j$ . Filter approaches take into consideration the samples

530 relation with a given reference obtaining for each sample a different weight, while wrapper approaches utilise clusters structures to adjust the weights and obtain as many weights per feature as number of extracted clusters.

*Filter Local Unsupervised Feature Weighting.* Similar to global unsupervised FW, the filter approach is not a commonly employed technique for estimating the weights and few works are encountered in the literature. Most of works are related to *time series* data in order to include information about their temporal behaviour. In these cases, as Algorithm 7 shows, generally for each sample  $\mathbf{x}_i$  with  $i = \{1, \dots, n\}$  a local weight is estimated in line 3 of Algorithm 7, aiming at measuring common information shared with a given sample of reference  $\mathbf{z}$ .  
 540 Once the weights have been calculated, these in line 5 of Algorithm 7 multiply each sample of the dataset, creating the weighted dataset  $\tilde{X}$  which is finally utilised by the ML algorithm for the problem modelling (line 6).

---

**Algorithm 7** Filter Local Unsupervised Feature Weighting methods

---

```

1: Given  $X \subset \mathbb{R}^{n \times m}$  and a reference point  $\mathbf{z}$ 
2: for  $i = 1 : n$  do
3:    $\mathbf{w}_i^* \leftarrow \text{commonality}(\mathbf{x}_i, \mathbf{z})$ 
4: for  $i = 1 : n$  do
5:    $\tilde{X}_i \leftarrow w_i^* \cdot X_i$ 
6:  $\tilde{Y} \leftarrow \text{ML algorithm to } \tilde{X}$ 

```

---

In this context, the exponential fading function, a monotonic decreasing function that decays with the time, estimates the weights of the samples in line 3 of Algorithm 7 as a function of time respect to the moment of interest ( $\mathbf{z}$  in line 3 of Algorithm 7), assigning higher weight values to recent samples. This weighting function includes a decay rate  $\lambda$  which controls the importance of the historical information. The lower the value of  $\lambda$ , the higher the importance of the past data compared to more recent information. This filter local unsupervised weighting method is widely employed in temporal applications aiming at reducing the  
 550

influence of the past. For instance, authors in (Aggarwal et al., 2004) include the mentioned fading function in their proposed High-dimensional projected data stream clustering method (HPStream) (line 6 of Algorithm 7) and authors in (Ding & Li, 2005), aiming at assigning higher importance to recent data to  
555 the recommendation process of their collaborative filtering algorithm (line 6 of Algorithm 7), include the estimated weights in the preference prediction phase.

Conversely, Jeong et al. (2011) propose the Weighted Dynamic Time Warping (WDTW), a variant of the traditional DTW (line 6 of Algorithm 7). This variant includes weights into the formulation to consider, when creating the path matrix,  
560 the phase difference between two samples  $\mathbf{x}_i, \mathbf{x}'_i$  from two distinct sequences of different length. In order to measure such phase difference between the samples, the authors introduce the Modified Logistic Weight Function (MLWF) for the weight calculations in line 3 of Algorithm 7. This way, the authors aim at penalising large phase differences in order to prevent minimum distance  
565 distortion caused by *outliers*. In this work, the WDTW idea is also extended to other variants of the DTW such as DDTW (line 6 of Algorithm 7). Later, the authors in (Jeong & Jayaraman, 2015) employ the proposed WDTW as a kernel function into a multiclass Support Vector Machine (SVM) (line 6 of Algorithm 7) for *time series* classification and validate the proposed model over datasets  
570 from the UCR repository.

The Weighted Permutation Entropy (WPE) is proposed by Fadlallah et al. (2013), in which the motif counts from signal patterns are weighted aiming at retaining the amplitude information of nonlinear time series. Given the time-delay embedding representation  $X_j^{m,\tau}$  of a time series  $\{x_t\}_{t=1}^T$  being  
575  $j = 1, \dots, T - (m - 1)\tau$ ,  $m$  the embedding dimension and  $\tau$  the time delay, for each  $j$  (line 2 of Algorithm 7) the authors calculate the weights in line 3 of Algorithm 7 as the variance of each neighbours vector  $X_j^{m,\tau}$ . This way the proposed approach assigns different weights to neighbouring vectors with same ordinal patterns but with different amplitude variations. The weighted  
580 relative frequencies for each motif are employed to compute the WPE (line 6

of Algorithm 7). WPE can be utilised to detect abrupt changes in noisy or multi-component signals and it is successfully employed by Zhou et al. (2018) in combination with Ensemble Empirical Mode Decomposition (EEMD) and SVM ensemble classifier (line 6 of Algorithm 7) for fault diagnosis of rolling bearing.

585 Filter approaches for local feature weights are also employed for static data with *missing values*. For instance, author in (Datta et al., 2016) propose the Feature Weighted Penalty based Dissimilarity (FWPD), a measure that considers the number of *missing features*. More concretely, such dissimilarity measure includes a parameter  $\alpha$  to control the relative importance between two  
590 terms and a Feature Weighted Penalty that, for each pair of samples  $\mathbf{x}_i, \mathbf{x}_{i'}$  with  $i, i' \in \{1, \dots, n\}$ , calculates the penalty weight (line 3 of Algorithm 7) as the proportion of shared features without *missing values* respect to the total number of observed features. This FWPD is included into the K-NN classifier mechanism (line 6 of Algorithm 7) to estimate the similarity between neighbours  
595 in order to handle *missing features*.

*Wrapper Local Unsupervised Feature Weighting.* Wrapper FW is the most common approach for searching the optimal local weights. In this case, as can be observed in Algorithm 8, there are as many weights as clusters  $K$  per feature. These randomly initialised local weights (2 of Algorithm 8) are updated in each  
600 iteration of the algorithm (line 9 of Algorithm 8) based on the results obtained by the ML algorithm (line 8 of Algorithm 8). This process ends when (line 10 of Algorithm 8) the performance overcomes a certain threshold  $\theta$ , the algorithm converges, i.e., the performance do not present a significant improvement respect to the previous iteration, or until a number of iterations  $n\_iter$  is reached (line 4  
605 of Algorithm 8). The works related to wrapper local unsupervised FW methods found in the literature are mainly based on partitioning, hierarchical and fuzzy clustering. Other works based on classification and evolutionary algorithms can also be found.

Specifically, the Local Attribute Weighting K-Means (LKM) algorithm proposed

---

**Algorithm 8** Wrapper Local Unsupervised Feature Weighting methods

---

```
1: Given a dataset  $X \subset \mathbb{R}^{n \times m}$ , a clustering algorithm and  $K$ 
2: Initialise  $w_{kj}^{(0)} \in \mathbb{R} \quad \forall k = 1 : K, \forall j = 1 : m$ 
3:  $s \leftarrow 0$ 
4: while  $s < n\_iter$  do
5:   for  $j=1:m$  do
6:     for  $k = 1 : K$  do
7:        $\tilde{X}_{i'j}^{(s)} \leftarrow w_{kj}^{(s)} \cdot X_{i'j} \quad \forall i' \in \{i | \mathbf{x}_i = k\}$ 
8:        $\tilde{Y}^{(s)} \leftarrow$  ML algorithm to  $\tilde{X}^{(s)}$ 
9:        $w_{kj}^{(s+1)} \leftarrow$  Adjust  $w_{kj}^{(s)}$ 
10:      if  $\text{perf}(\tilde{Y}^{(s)}) > \theta$  or  $=\text{perf}(\tilde{Y}^{(s-1)}) + \epsilon$  then
11:        Break
12:       $s \leftarrow s + 1$ 
13:  $w_{kj}^* \leftarrow w_{kj}^{(s)}$ 
```

---

610 in (Chan et al., 2004) is akin to the one employed in (Huang et al., 2005), but the weights,  $w_{kj}$  with  $k = \{1, \dots, K\}$  and  $j = \{1, \dots, m\}$ , are related to a specific cluster  $k$  and feature  $j$ . The method for updating the weights in line 9 of Algorithm 8 relies on the cluster dispersion at each dimension. Similarly, Shen et al. (2006) propose the same idea adapted to the fuzzy C-means algorithm.

615 Another related proposal can be found in (Jing et al., 2007) where the Entropy Weighted K-Means algorithm (EWKM) for clustering high-dimensional objects in subspaces is presented. In this proposal the weight values are employed to identify the subsets of important dimensions that categorise different clusters. This is achieved by including the weight entropy in the objective function

620 (line 8 of Algorithm 8) that is minimised in the clustering process in line 10 of Algorithm 8. The experiments on both synthetic and real data have shown that the new algorithm can generate better clustering results than other subspace clustering algorithms, such as: FWKM (Jing et al., 2005), Bisecting K-means (Karypis et al., 2000), PROCLUS (Aggarwal et al., 1999), HARP

625 (Yip et al., 2004), LAC (Domeniconi et al., 2004) and COSA (Friedman &  
Meulman, 2004) and that is scalable to large high-dimensional data. Following  
with the same line of research, in (Chen et al., 2012) an extension of the  
EWKM algorithm (Jing et al., 2007) is introduced, called FG-k-means. It  
automatically calculates two types of local weights in line 9 of Algorithm 8,  
630 one for groups of features and the other one for the individual features in  
each cluster. The experimental results show the goodness of the FG-k-means  
over the other considered algorithms and its robustness to noise and missing  
values. Another work related to high-dimensional data is introduced by Gan  
& Ng (2015). The referred AFG-k-means algorithm, extends the idea of Chen  
635 et al. (2012) by incorporating an Automatic Feature Group selection algorithm  
which iteratively creates groups of features. Recently, Huang et al. (2018a)  
extend the weighted K-means algorithm presented in (Chan et al., 2004) with  
a  $l^2$  regularisation of the local features, obtaining the  $l^2$ -WKmeans. The  
authors introduce two versions of the algorithm to handle both numerical and  
640 categorical data clustering. Experimental results show that for both numerical  
and categorical data the proposal outperforms the state-of-the-art algorithms  
in terms of accuracy, Rand index, F-score and NMI.

In addition to the weighted K-means based algorithms, other clustering  
algorithms are adapted to include local weights into their formulation. In  
645 this sense, a new version of the Ward Hierarchical algorithm can be found in  
(de Amorim, 2015). It employs local cluster dependant weights calculated with  
the  $L_p$  norm in line 9 of Algorithm 8 over the normalised datasets. As the  
Ward algorithm, it does not need to know the number of clusters in advance.  
In addition, although the algorithm's performance depends on the term  $p$  of the  
650  $L_p$  norm, the value of  $p$  that maximises the Silhouette index is proposed. The  
proposal of Chen et al. (2016) presents a novel algorithm for Subspace Clustering  
of Categories (SCC) for clustering categorical data. The algorithm utilises a  
probabilistic distance function based on Kernel Density Estimation (KDE) to  
measure the dissimilarity between categorical objects. The local weights are

655 calculated at each iteration of Algorithm 8 in line 9 as the smoothed dispersion  
of the features in each cluster. The authors also propose a new categorical cluster  
validity index which evaluates the average intra-cluster scatter and inter-cluster  
separation. In the context of consensus clustering, authors in (Alguliyev et al.,  
2020) develop a weighted consensus clustering for Big data applications that  
660 assigns weights to single clustering methods based on purity utility function.

Finally, to conclude with the partitioning clustering FW methods, authors  
in (Hashemzadeh et al., 2019) present a Fuzzy C-means based method that  
automatically calculates local weights for the features and also includes in  
line 9 of Algorithm 8 a cluster weighting process to mitigate the initialisation  
665 sensitivity of the algorithm. During the optimisation process the feature  
weights are calculated in line 9 of Algorithm 8 based on the intra-cluster  
feature-weighted distance, while the cluster weights are recalculated by  
employing the intra-cluster distances. The authors compare their proposal with  
some state-of-the art algorithms (Frigui & Nasraoui, 2004; Tzortzis & Likas,  
670 2014; Zhi et al., 2014; Zhou et al., 2016) utilising large real world and synthetic  
datasets. The results confirmed that the proposed method is not sensitive  
to the cluster centroids initialisation and also that outperforms the clustering  
results obtained by its competitors. Furthermore, the authors conclude that the  
algorithm can be effectively employed on big data clustering and co-clustering  
675 applications used as *online* feature and clustering weighting method.

The local feature weighting strategy is also employed in conjunction with  
classification algorithms (line 8 of Algorithm 8). The Subspace Weighting Naive  
Bayes algorithm (SWNB) in (Chen & Wang, 2012) introduces a locally weighted  
probability model, in which the weights are optimised in line 9 of Algorithm 8  
680 to fit a Logitnormal priori distribution and the Maximum a Posteriori principle  
for Bayesian modelling in high dimensional spaces. The weights are then  
calculated by the Newton-Raphson method. The experiments conducted on  
document corpora and gene microarray datasets and the comparisons with other  
weighting algorithms (Frank et al., 2002; John & Langley, 1995; Lee et al.,

685 2011) demonstrate that the SWNB method is comparable or superior to its competitors.

Evolutionary algorithms are also applied to iteratively seek for the optimal local feature weights. In the research conducted by Gañçarski & Blansché (2008) new FW methods based on EAs are proposed: Darwinian, Lamackian  
690 and Baldwinian Evolutionary algorithms and their co-evolutionary approach (DE-LKM, LE-LKM, BE-LKM and DC-LKM, LC-LKM, BC-LKM). The algorithms utilise in line 10 of Algorithm 8 the cost function defined in LKM (Chan et al., 2004) as fitness function and estimate the internal quality of the unsupervised classification by means of the Wemmert-Gañçarski cluster quality  
700 index (Wemmert et al., 2000). Finally, the NSGA algorithm (Deb et al., 2002) is employed as optimisation tool for the multi-objective optimisation approach proposed by Zhou & Zhu (2018) with the aim of simultaneously estimating the cluster centres and, in line 9 of Algorithm 8, the local weights that minimise the intra-cluster dispersion and maximise the inter-cluster separation. Similarly,  
705 the work presented in (Liu et al., 2019) employs the Adaptive Shorting-based Evolutionary Algorithm (ASEA) to optimise a new clustering validity index for credit risk assessment based on four terms: 1) the weighted intra-cluster compactness, 2) the inter-cluster separation, 3) penalties for the feature, 4) penalties for the cluster weights. More concretely, the cluster weights and the  
705 term intra-cluster compactness are incorporated to address the issue of class imbalance.

### 3.2.3. Concluding Remarks about Unsupervised Approaches

Regarding unsupervised FW approaches, filter methods are not commonly employed in the literature and these obtain weights per sample, obviating  
710 the differences in relevance among dimensions. The reviewed works usually calculate the features relevance based on previous clustering results or, in the case of *time series*, estimate the relevance of the samples considering the temporal distance from the target. In these cases, the weights quality are dependant on the suitability of the clusters configuration or on the



715 availability of expert knowledge about the temporal dependency of the system,  
respectively. Wrapper approaches, instead, are the most widely applied and  
include the feature weights into the objective function of the ML algorithm.  
Nevertheless, the application of evolutionary techniques is not so popular in  
the unsupervised environment. The weights estimation is usually done as a  
720 function of the current partition, so the results are generally highly dependant  
on the initialisation of the algorithm.

Table 1 summarises the reviewed research works in the field of FW in the  
supervised and the unsupervised environment. It includes information about  
725 (a) the specific technique applied for FW, (b) the type of data of the features,  
(c) the application field and/or the datasets used in the research, (d) the research  
works and/or techniques used to compare the authors' proposal, and (e) the ML  
performance measure.

Ref.	FW technique	Features	Application	Comparison	ML performance measure
Xing et al. (2009)	MI	Real	UCI Dues & Graff (2017), Microarray, Ripley's synthetic dataset	Sun (2007); Wang et al. (2004)	Accuracy
Givokli et al. (2012)	MI	Integer, Real	Diabetes	SVM-based classifiers	Sensitivity, Specificity, Accuracy
Jiang et al. (2018)	MI	Categorical, Integer, Real	UCI Dues & Graff (2017)	Jiang et al. (2016); Lee et al. (2011); Wu & Cai (2011); Zaidi et al. (2013); Zhang & Sheng (2004)	CLL, AUC
Chen & Hao (2017)	MI	Real	Stock market indices prediction	SVM-KNN	MAPE, RSME
Wu et al. (2017)	Entropy	Text	Text categorisation	Jones et al. (2000); Lan et al. (2008); Martineau & Finin (2009); Paltoglou & Thebwall (2010); Sparck Jones (1972); Wu & Salton (1981)	Accuracy
Sahin et al. (2015)	$\chi^2$ , F-score	Image	Landslide susceptibility mapping	AHP model	Accuracy, Success rate curve
Bhattacharya et al. (2017)	Mean, Standard deviation	Integer, Real	UCI Dues & Graff (2017), Face, Hand-writing recognition	Metrics, FS	Accuracy
Niño-Adán et al. (2019)	Pearson / RF	Integer, Real	UCI Dues & Graff (2017)	normalisation methods	Accuracy
Elbasiony et al. (2013)	RF	Categorical, Real	NID Cup (1999)	Other NDI method	ROC, Detection rate, FP rate
Peng et al. (2017)	MI / Pearson	(Im)Balanced	DGC	Peng et al. (2014)	Accuracy, Computational time, AUC, Cohen's kappa coefficient
Zhang et al. (2016)	GR / DT	Binary	Text classification	Li et al. (2012); Wang et al. (2014)	Accuracy
Phan et al. (2017)	GA / Accuracy, F-measure, MCC	Real	UCI Dues & Graff (2017)	Jiang et al. (2016); Saez et al. (2014); Wu et al. (2015); Xiang et al. (2016)	Accuracy, Computational cost
Komonićki & Kraviec (2000)	GA	Images	Neuropathology	Feature selection	Accuracy
Ahn & Kim (2009)	GA/Accuracy	Real	Cancer	GA for GBR	Accuracy

Continued on next page

Table 1 – Continued from previous page

	Ref.	FW technique	Features	Application	Comparison	ML performance measure
	Matoos-García et al. (2017)	EA / CV error	Binary, Real	UCI Dues & Graff (2017)	Dudani (1976); García-Gutiérrez et al. (2014); Gou et al. (2012, 2011)	Accuracy
	Triguero et al. (2012)	DE/ Accuracy	Real	KEEL Alcala-Fdez et al. (2011)	Kononenko (1994); Tahir et al. (2007)	Accuracy, Reduction
	Serrano-Silva et al. (2018)	DE, GA, NBA / AUC	Categorical, Integer, Real	Financial risk	Between them	AUC, Execution time
	Huang et al. (2018b)	Gradient Ascent / Margin vector	Categorical, Integer, Real	UCI Dues & Graff (2017)	Cai et al. (2014); Kononenko (1994); Sun (2007)	Accuracy, Precision, Recall, F-measure
	Marchiori (2013)	Local Relief	Categorical, Integer, Real	Cancer	Relief	Accuracy
	Yilmaz et al. (2014)	Local Relief-f	Multimedia	Multimodal fusion	Relief-F, Exhaustive search	Accuracy, Time execution
	Chen & Guo (2015)	Entropy, Gini	Categorical	UCI Dues & Graff (2017)	Global version	Micro / Macro F1-measure
	Paredes & Vidal (2000)	PPGD / Intra, inter-class distances	Categorical, Integer, Real	UCI Dues & Graff (2017)	$L_2$ , MD, Class- dependant MD	Accuracy
	Paredes & Vidal (2006)	GD / Misclassification probability	Categorical, Integer, Real	Synthetic, UCI Dues & Graff (2017)	Disimilarity measures, Class and/or prototype dependant	Error rate
	Jiao et al. (2015)	MHNN / Centroids distance, feature distribution	Categorical, Integer, Real	Synthetic, UCI Dues & Graff (2017)	$L_2$ , Paredes & Vidal (2000), Paredes & Vidal (2006)	Run time, Error rate, Accuracy
	Jiao et al. (2019)	Maximum Likelihood / Pairwise distances	Categorical, Integer, Real	Synthetic, UCI Dues & Graff (2017)	Global, Local Pairwise $L_2$ , MD	Accuracy, Run time
	Taheri et al. (2014)	Quasiseccant method / NB binary classification accuracy	Categorical, Integer, Real	UCI Dues & Graff (2017)	NB variants	Accuracy
	Jiang et al. (2019)	GD / CLL, MSE	Categorical, Integer, Real	UCI Dues & Graff (2017)	Hall (2006); Jiang et al. (2016); Lee et al. (2011); Wu & Cai (2011); Zaidi et al. (2013); Zhang & Sheng (2004)	Accuracy
	Mohammed & Zhang (2008)	PSO / NCC classification error	Categorical, Integer, Real	UCI Dues & Graff (2017)	$L_2$ , Class-dependant MD, Paredes & Vidal (2006)	Accuracy
	Matoos-García et al. (2012)	DE / K-NN classification error	Categorical, Integer, Real	UCI Dues & Graff (2017)	ALSukker et al. (2010); Paredes & Vidal (2006)	Accuracy
	Shrivya & Culin (2017)	PSO / DGC accuracy	Real	UCI Dues & Graff (2017)	DGC, K-NN	Precision, Recall, F-measure

Continued on next page

Table 1 – Continued from previous page

	Ref.	FW technique	Features	Application	Comparison	ML performance measure
Filter	Gürler (2017)	Distance from centroids	Real	Parkinson Little et al. (2007)	Polat (2012); Saker & Kurman (2010)	Sensitivity, Specificity, Precision, Recall, F-measure, Kappa statistic
	Güneş et al. (2010)	Distance from centroids	Statistical features	Sleep stage recognition	Unweighted features	Success rate (Accuracy)
Wrapper	Huang et al. (2005)	Partial optimisation / Intra-cluster dispersion	Categorical, Integer, Real	Synthetic, UCI Dua & Graff (2017)	K-means, Different $\beta$ values, Feature selection	Accuracy
	Saha & Das (2015)	Alternative Optimisation / Cluster membership	Categorical	Synthetic; UCI Dua & Graff (2017)	Fuzzy K-modes	Rand index, Partition Coefficient, Partition Entropy
Global	De Amorim & Mirkin (2012)	Partial optimisation / Intra-cluster dispersion	Categorical, Integer, Real	Synthetic, UCI Dua & Graff (2017)	K-means Huang et al. (2005)	Accuracy
	Chakraborty & Das (2018)	Partial optimisation / Intra-cluster dispersion	Integer, Real	UCI Dua & Graff (2017), KEEL Alcalá-Fdez et al. (2011)	K-means, G-means, Huang et al. (2005)	NMI
Unsupervised	Tsai & Chin (2008)	Partial optimisation / Inter and intra cluster separation	Integer, Real	UCI Dua & Graff (2017)	K-means, Huang et al. (2005)	SSE, Entropy, ARI, Convergence
	Bankabou et al. (2018)	Partial optimisation / Intra-cluster dispersion, Weight entropy	Time series	Outlier detection	Bevilacqua & Tsafaris (2015); Breunig et al. (2000); Budalakoti et al. (2008); Ng et al. (2002); Portnoy (2000); Schölkopf et al. (2000); Young & Chow (2002)	AUC, Time complexity
Filter	Zhang et al. (2018)	Partial optimisation / Intra-cluster dispersion, Weight entropy	Multiple view	UCI Dua & Graff (2017)	Jing et al. (2007) and Multi-view clustering algorithms	NMI, Precision, F-score, Classification rate
	Jeong et al. (2011)	Phase difference	Time series	UCR Dau et al. (2018)	$L_2$ , DTW	Error rate, Entropy, F-measure, Average inter /intra cluster distance
Local	Jeong & Jayaraman (2015)	Phase difference	Time series	UCR Dau et al. (2018)	$L_2$ , DTW	Accuracy, Sensitivity, Specificity
	Fadhilah et al. (2013)	WPE	Time series	Electro - encephalogram	Entropy-based measurements	Entropy
Filter	Zhou et al. (2018)	WPE	Time series	Fault diagnosis	SVM, K-NN, Extreme Learning	Accuracy
	Chan et al. (2004)	Partial optimisation / Intra-cluster dispersion	Categorical, Integer, Real	Synthetic, UCI Dua & Graff (2017)	K-means, $\beta$	Accuracy

Continued on next page

Table 1 – Continued from previous page

Ref.	FW technique	Features	Application	Comparison	ML performance measure
Jing et al. (2007)	Partial optimisation / Intra-cluster dispersion, Weight entropy	Text	Synthetic, UCI Dna & Graff (2017)	Aggarwal et al. (1999); Domeniconi et al. (2004); Friedman & Meulman (2004); Jing et al. (2005); Karypis et al. (2000); Yip et al. (2004)	Entropy, F-score, NMI
Chen et al. (2012)	Partial optimisation / Intra-cluster distance, Weights entropy	Real	Synthetic, UCI Dna & Graff (2017)	K-means, Domeniconi et al. (2007); Huang et al. (2005); Jing et al. (2007)	Precision, Recall, F-measure, Accuracy
Gan & Ng (2015)	Partial optimisation / Intra-cluster distance, Feature and group weight entropy	Real	Synthetic, Gene expression	Chen et al. (2012); Domeniconi et al. (2007); Gan & Wu (2008); Jing et al. (2007)	Corrected Rand Index, Execution run time
Huang et al. (2018a)	Partial optimisation / $L_2$ norm regularisation, Weights entropy	Categorical	Synthetic, UCI Dna & Graff (2017)	Chan et al. (2004); Huang et al. (2005); Jing et al. (2007)	Accuracy, Rand Index, F-score, NMI
de Amorim (2015)	Partial optimisation / $L_p$ distance	Categorical, Integer, Real	Synthetic, UCI Dna & Graff (2017)	Ward	Adjusted Rand Index, Noise features contribution
Chen et al. (2016)	Bandwidth optimisation / Total error	Categorical	Synthetic, UCI Dna & Graff (2017)	Bai et al. (2011); Cao et al. (2013); Chan et al. (2004); Domeniconi et al. (2007); San et al. (2004)	Accuracy, NMI, Categorical cluster validity index, Run time
Hushemzadeh et al. (2019)	Partial optimisation / Intra-cluster dispersion	Categorical, Integer, Real	Synthetic, UCI Dna & Graff (2017)	Frigui & Nasraoui (2004); Tzortzis & Likas (2014); Zhi et al. (2014); Zhou et al. (2016)	Accuracy, NMI
Chen & Wang (2012)	Numerical methods/Logitnormal distribution	Real	Gene expression	Frank et al. (2002); Johns & Langley (1995); Lee et al. (2011)	Micro / Macro F-measure, Scalability
Gancwski & Blauche (2008)	Darwinian, Lameckian and Baldwinian (co)evolutionary algorithms/Intra-cluster distance	Categorical, Integer, Real	UCI Dna & Graff (2017)	K-means, Between them	Folks-Mallow index, Rand index, Jaccard coefficient
Zhou & Zhu (2018)	Multi-objective optimisation/Inter-intra cluster measures	Categorical, Integer, Real	UCI Dna & Graff (2017)	FW clustering	Accuracy, Rand index, NMI
Lin et al. (2019)	Multi-objective optimisation/Intra-cluster compactness inter-cluster separation	Categorical, Integer, Real	UCI Dna & Graff (2017)	Dunn (1973); Gan & Wu (2008); Parvin & Minaei-Bidgolhi (2013); Xia et al. (2013)	Rand Index, F-measure

Table 1: FW research works classified following the proposed taxonomy in Figure 3.

#### 4. Considerations for the Application of Feature Weighting Methods

730 In this section some guide points for the application of FW methods are proposed regarding the different taxonomy levels presented in Figure 3.

##### 4.1. Supervised versus Unsupervised Learning

At the first level of the proposed taxonomy, the FW methods are divided into supervised and unsupervised learning approaches. In our opinion, supervised  
735 FW approach is always advisable if *labels* are available (Ahn & Kim, 2009). By this way, the relevance can be computed as the informative ability of each feature for estimating the label (Chen & Hao, 2017; Jiang et al., 2018), increasing the output accuracy of the ML model (Sinciya & Celin, 2017). However, in many problems labels are not always available, thus, unsupervised FW methods can  
740 be applied as follows: 1) expert-knowledge of the problem field can be employed to approximate the influence of each feature on the physical process (Sahin et al., 2015); 2) temporal information, such as the distance of the samples to a reference temporal moment, or the phase difference (Benkabou et al., 2018; Jeong et al., 2011) can be considered to weigh the samples and 3) the estimation  
745 of the feature relevance can be done by means of the metric employed for the ML clustering algorithm, such as the cohesion or separation of the former clusters (De Amorim & Mirkin, 2012; Jing et al., 2007). In the latter case, the final result is highly dependant on the initialisation of the ML clustering algorithm (Gan & Ng, 2015). In order to circumvent this problem, some FW methods integrate  
750 an evolutionary algorithm for the optimisation process to be able to efficiently explore the solution space and not converge to a local optimum (Gançarski & Blansch e, 2008; Kuo & Nguyen, 2019).

##### 4.2. Global versus Local Approach

The second level of the proposed taxonomy discriminates between the way the  
755 weights are calculated, i.e., globally or locally. In the local supervised approach, the samples are merged according to their label and for each group of samples a local weight is commonly calculated (Chen & Guo, 2015). However, the

challenge is to assign, among all the estimated weights, the corresponding one to a new unlabelled sample. In this context, probabilistic methods are usually  
760 applied for estimating the class to which the new sample belongs, and therefore, assigning its weight. Another common strategy is to employ confidence-based methods for assigning a local weight to the new sample (Jiang et al., 2019). Nevertheless, depending on the number of local weights, the *computational cost* may increase and global FW methods might be a more advisable approach.

765 Regarding unsupervised environments, expert-knowledge based local weights can also be employed. Furthermore, in *condition-based systems*, in which some feature represents a specific condition of the physical system, makes advisable to assign different weights depending on the particular system condition. Similarly, if the *temporal evolution* of the analysed system is known, the local weight may  
770 consider the temporal distance of each sample to the referenced one.

Generally, in most cases a hybrid (global and local) approach can be applied. However, for *high dimensional datasets or when a high number of local weights* is presented, a global FW strategy may be recommended for reducing the computational cost.

#### 775 4.3. Filter versus Wrapper Method

In the last level of the proposed taxonomy, the estimation of the weights are divided into filter and wrapper methods.

In regards to unsupervised FW methods, the wrapper methods are the most employed ones in the literature. Moreover, in supervised approaches, they have  
780 also proven to *increase the algorithm performance* respect to filter approaches (Marchiori, 2013; Yilmaz et al., 2014). However, the calculated weights are highly dependant on the ML algorithm configuration (Phan et al., 2017), its hyper-parameters, or even the training dataset used (Peng et al., 2017). Therefore, in order to ensure *explainability*, i.e. for a decision support system  
785 (Chen & Guo, 2015) or *high dimensional datasets* (Yilmaz et al., 2014), filter methods are a preferable approach.

Nevertheless, some encoding considerations must be taken to handle categorical features with continuous ones in the same classification problem. Similarly, the estimation of the probability distribution in some real scenarios is not always possible. Then, instead of Information theory-based approaches, Statistical-based FW methods may be more suitable in these cases.

Finally, it must be taken into account that the differences among the features' magnitudes can influence the results more than the calculated weights by the FW methods. Consequently, prior to the weights calculation, additional preprocessing steps, like normalisation methods (Milligan & Cooper, 1988), and a rigorous analysis about the influence of such rescaling transformation (Niño-Adan et al., 2019) should be conducted.

#### 4.4. FW approach selection

Based on the conclusions drawn from the analysis of the literature presented in Section 3 as well as the considerations above, this Section provides recommendations for the optimal selection of the FW approach depending on the characteristics and objectives of the problem at hand.

1. **Labels:** If the dataset is labelled, **supervised** feature weighting approaches should be applied.
2. **High-dimensional dataset** (Peng et al., 2017): If the dataset comprises a large number of features, computationally cheaper strategies – like **global weights or filter methods** – should be considered.
3. **Dimensionality reduction** (Panday et al., 2018): If the reduction of the number of features is needed due to computational complexity problems, in order to select the most informative ones, **global filter** approaches should be applied as Feature Selection method.
4. **Dataset understanding:** If there is no prior information about the problem, the weights calculated by **filter methods** enhance the understanding of the relevance of each feature of the dataset.
5. **Features contribution estimation:** If the aim is to extract knowledge about the influence of each feature on the performance of the model



scoring, **wrapper methods** are recommended, since the weights obtained are representative values of the contribution each feature has on the outputs estimated by the model.

- 820 6. **Missing values** (Datta et al., 2016): If the dataset contains missing values and the application of some imputation method is required, the approaches are: 1) to assign a lower **global feature weight** to those features with missing values; 2) to employ **local weights** by means of assigning lower weight to the samples with missing values, and representative values – of the relevance between the sample and the label – to the remaining samples.
- 825 7. **Imbalanced dataset** (Liu et al., 2019): If the dataset is imbalanced, **local feature weighting** strategies aim at increasing the relevance of the minority class samples.
- 8 8. **Outliers** (Jeong et al., 2011): If the dataset contains outliers, **local feature weighting** methods allow assigning lower weight to such samples for minimising their influence.
- 830 9. **Noise**: If noisy features are expected in a dataset, using **global filter weighting approaches** the weight of an artificially-generated noisy feature can be estimated  $w^*$ . Then, the features of the dataset that present similar weights  $w_j$  (calculated by the same technique) to such estimated reference weight,  $w_j \leq w^* + \epsilon$  can be identified as noisy features.
- 835 10. **Interpretability** (Chen & Guo, 2015): If the interpretability of the model is imperative, as stated in Section 3.1.3, **filter approaches** based on Information theory or Statistical approaches should be considered.
- 840 11. **Condition-based problems**: If based on expert knowledge it is known that the analysed system presents different operational conditions (determined by the value of certain representative features can take), and thus, the relevance of the rest of features vary depending on the condition under the system is working, then **local feature weights** are recommended.
- 845 12. **Temporal dependency** (Fadlallah et al., 2013; Jeong et al., 2011): If the label of a sample depends not only on the values of such sample but

also on the values of the previous ones, it is recommended to assign **local weights** to the samples according to their temporal distance respect to the analysed one.

850

13. **Algorithm performance maximisation** (Ahn & Kim, 2009; Mateos-García et al., 2017; Triguero et al., 2012; Yang et al., 2012): If a maximisation of the algorithm performance – in terms of metric optimisation – is sought regardless the interpretability of the model, **wrapper weighting** methods that adjust the weights in order to maximise such metric are suggested.

855

14. **Semi-supervised learning** (De Amorim & Mirkin, 2012): If the dataset comprises both labelled and unlabelled samples, **local weights** to increase the impact of the labelled samples respect to the unlabelled ones are advisable.

860

15. **Online learning** (Marchiori, 2013; Yilmaz et al., 2014) : If an online fashion is faced and extreme label latency is expected, then **unsupervised weighting methods** are recommended. Besides, if concept drift is foreseen, for both quick discovering of such drift and rapid adjustment of the weights **filter approaches** are advised.

865

Figure 4 depicts the above mentioned recommendations into the different levels of the proposed taxonomy (Figure 3).

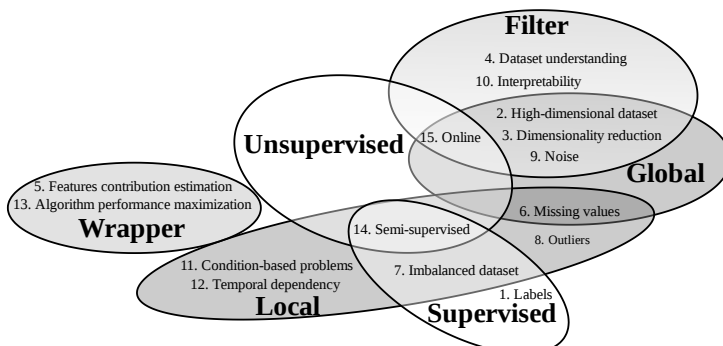


Figure 4: Recommendations for the optimal selection of the Feature Weighting approach.

## 5. Conclusions and Future Challenges

This work presents an extensive review of FW methods based on a proposed taxonomy or classification scheme, i.e: 1) At a first level, supervised and unsupervised methods are differentiated; 2) Then, regarding whether the application of the weights is over the entire or over a subset of the instance space, global and local approaches are presented, respectively; and 3) finally, the evaluation criteria and the interaction with the ML algorithm give rise to a filter/wrapper classification at a lower level. Moreover, some recommendations and guide points for the optimal selection of the FW approach are shown, regarding the characteristics and objectives of the problem at hand.

In light of the great interest that attracted the FW field in the last years, it is expected that the number of related works continues growing. As observed in the conducted review, the majority of works are focused on integer, real or categorical input data, being few the FW works applied to time series data. Since nowadays many real applications rely on time series data, future work may be focused on delving FW methods for time series data. Moreover, in order to be able to create a methodology for the application of a FW method based on the characteristics of the input data and the problem at hand, a complete comparison in theoretical and experimental terms of the different FW approaches has to be addressed. Similarly, an exhaustive study of the degree of susceptibility of the ML algorithms to the feature weighting transformation will be interesting and help the selection of the FW strategy.

## 6. Acknowledgments

This work has been supported in part by the ELKARTEK Research Programme of the Basque Government (Argia KK-2019/00068), the HAZITEK program (DATAlySE ZL-2018/00765), the Basque Government (IT1381-19), the DATAinc program (48-AF-W1-2019-00002), and a TECNALIA Research and Innovation PhD Scholarship.

## References

- Aggarwal, C. C., Han, J., Wang, J., & Yu, P. S. (2004). A framework for projected clustering of high dimensional data streams. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30* (pp. 852–863).  
900
- Aggarwal, C. C., Wolf, J. L., Yu, P. S., Procopiuc, C., & Park, J. S. (1999). Fast algorithms for projected clustering. *ACM SIGMOD Record*, 28, 61–72.
- Ahn, H., & Kim, K.-j. (2009). Global optimization of case-based reasoning for breast cytology diagnosis. *Expert Systems with Applications*, 36, 724–734.
- 905 Aksoy, S., & Haralick, R. M. (2001). Feature normalization and likelihood-based similarity measures for image retrieval. *Pattern recognition letters*, 22, 563–582.
- Alcalá-Fdez, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L., & Herrera, F. (2011). Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. *Journal of*  
910 *Multiple-Valued Logic & Soft Computing*, 17.
- Alguliyev, R. M., Aliguliyev, R. M., & Sukhostat, L. V. (2020). Weighted consensus clustering and its application to big data. *Expert Systems with Applications*, 150, 113294.
- 915 AlSukker, A., Khushaba, R., & Al-Ani, A. (2010). Optimizing the k-nn metric weights using differential evolution. In *2010 International Conference on Multimedia Computing and Information Technology (MCIT)* (pp. 89–92). IEEE.
- de Amorim, R. C. (2015). Feature relevance in ward’s hierarchical clustering using the  $l_p$  norm. *Journal of Classification*, 32, 46–62.  
920
- de Amorim, R. C. (2016). A survey on feature weighting based k-means algorithms. *Journal of Classification*, 33, 210–242.

- Bai, L., Liang, J., Dang, C., & Cao, F. (2011). A novel attribute weighting algorithm for clustering high-dimensional categorical data. *Pattern Recognition*, *44*, 2843–2861.
- 925
- Benkabou, S.-E., Benabdeslem, K., & Canitia, B. (2018). Unsupervised outlier detection for time series by entropy and dynamic time warping. *Knowledge and Information Systems*, *54*, 463–486.
- Bevilacqua, M., & Tsaftaris, S. (2015). Dictionary-decomposition-based one-class svm for unsupervised detection of anomalous time series. In *Proceedings of 23rd European signal processing conference (EUSIPCO)* (pp. 1776–1780).
- 930
- Bhattacharya, G., Ghosh, K., & Chowdhury, A. S. (2017). Granger causality driven ahp for feature weighted knn. *Pattern Recognition*, *66*, 425–436.
- 935
- Bishop, C. M. et al. (1995). *Neural networks for pattern recognition*. Oxford university press.
- Breunig, M. M., Kriegel, H.-P., Ng, R. T., & Sander, J. (2000). Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data* (pp. 93–104).
- 940
- Budalakoti, S., Srivastava, A. N., & Otey, M. E. (2008). Anomaly detection and diagnosis algorithms for discrete symbol sequences with applications to airline safety. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *39*, 101–113.
- Cai, H., Ruan, P., Ng, M., & Akutsu, T. (2014). Feature weight estimation for gene selection: a local hyperlinear learning approach. *BMC bioinformatics*, *15*, 70.
- 945
- Cao, F., Liang, J., Li, D., & Zhao, X. (2013). A weighting k-modes algorithm for subspace clustering of categorical data. *Neurocomputing*, *108*, 23–30.

- Chakraborty, S., & Das, S. (2018). Simultaneous variable weighting and  
950 determining the number of clusters- a weighted gaussian means algorithm.  
*Statistics & Probability Letters*, *137*, 148–156.
- Chan, E. Y., Ching, W. K., Ng, M. K., & Huang, J. Z. (2004). An optimization  
algorithm for clustering using weighted dissimilarity measures. *Pattern  
recognition*, *37*, 943–952.
- 955 Chen, L., & Guo, G. (2015). Nearest neighbor classification of categorical data  
by attributes weighting. *Expert Systems with Applications*, *42*, 3142–3149.
- Chen, L., & Wang, S. (2012). Automated feature weighting in naive bayes  
for high-dimensional data classification. In *Proceedings of the 21st ACM  
international conference on Information and knowledge management* (pp.  
960 1243–1252). ACM.
- Chen, L., Wang, S., Wang, K., & Zhu, J. (2016). Soft subspace clustering  
of categorical data with probabilistic distance. *Pattern Recognition*, *51*,  
322–332.
- Chen, X., Ye, Y., Xu, X., & Huang, J. Z. (2012). A feature group  
965 weighting method for subspace clustering of high-dimensional data. *Pattern  
Recognition*, *45*, 434–446.
- Chen, X., Yin, W., Tu, P., & Zhang, H. (2009). Weighted k-means algorithm  
based text clustering. In *2009 International Symposium on Information  
Engineering and Electronic Commerce* (pp. 51–55). IEEE.
- 970 Chen, Y., & Hao, Y. (2017). A feature weighted support vector machine and  
k-nearest neighbor algorithm for stock market indices prediction. *Expert  
Systems with Applications*, *80*, 340–355.
- Christopher Frey, H., & Patil, S. R. (2002). Identification and review of  
sensitivity analysis methods. *Risk analysis*, *22*, 553–578.

- 975 Cup, K. (1999). <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>.  
*The UCI KDD Archive*, .
- Daszykowski, M., Kaczmarek, K., Vander Heyden, Y., & Walczak, B. (2007). Robust statistics in data analysis - a review: basic concepts. *Chemometrics and intelligent laboratory systems*, *85*, 203–219.
- 980 Datta, S., Misra, D., & Das, S. (2016). A feature weighted penalty based dissimilarity measure for k-nearest neighbor classification with missing features. *Pattern Recognition Letters*, *80*, 231–237.
- Dau, H. A., Keogh, E., Kamgar, K., Yeh, C.-C. M., Zhu, Y., Gharghabi, S., Ratanamahatana, C. A., Yanping, Hu, B., Begum, N., Bagnall, A., Mueen, 985 A., & Batista, G. (2018). The ucr time series classification archive. URL: [https://www.cs.ucr.edu/~eamonn/time\\_series\\_data\\_2018/](https://www.cs.ucr.edu/~eamonn/time_series_data_2018/).
- De Amorim, R. C., & Mirkin, B. (2012). Minkowski metric, feature weighting and anomalous cluster initializing in k-means clustering. *Pattern Recognition*, *45*, 1061–1075.
- 990 Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, *6*, 182–197.
- Deng, Z., Choi, K.-S., Jiang, Y., Wang, J., & Wang, S. (2016). A survey on soft subspace clustering. *Information sciences*, *348*, 84–106.
- 995 Ding, Y., & Li, X. (2005). Time weight collaborative filtering. In *Proceedings of the 14th ACM international conference on Information and knowledge management* (pp. 485–492).
- Domeniconi, C., Gunopulos, D., Ma, S., Yan, B., Al-Razgan, M., & Papadopoulos, D. (2007). Locally adaptive metrics for clustering high 1000 dimensional data. *Data Mining and Knowledge Discovery*, *14*, 63–97.

- Domeniconi, C., Papadopoulos, D., Gunopoulos, D., & Ma, S. (2004). Subspace clustering of high dimensional data. In *Proceedings of the 2004 SIAM international conference on data mining* (pp. 517–521). SIAM.
- Dua, D., & Graff, C. (2017). UCI machine learning repository. URL: <http://archive.ics.uci.edu/ml>.  
1005
- Dudani, S. A. (1976). The distance-weighted k-nearest-neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics*, (pp. 325–327).
- Dunn, J. C. (1973). A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters, .
- 1010 Elbasiony, R. M., Sallam, E. A., Eltobely, T. E., & Fahmy, M. M. (2013). A hybrid network intrusion detection framework based on random forests and weighted k-means. *Ain Shams Engineering Journal*, 4, 753–762.
- Fadlallah, B., Chen, B., Keil, A., & Príncipe, J. (2013). Weighted-permutation entropy: A complexity measure for time series incorporating amplitude  
1015 information. *Physical Review E*, 87, 022911.
- Frank, E., Hall, M., & Pfahringer, B. (2002). Locally weighted naive bayes. In *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence* (pp. 249–256). Morgan Kaufmann Publishers Inc.
- Friedman, J. H., & Meulman, J. J. (2004). Clustering objects on subsets of  
1020 attributes (with discussion). *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 66, 815–849.
- Frigui, H., & Nasraoui, O. (2004). Unsupervised learning of prototypes and attribute weights. *Pattern recognition*, 37, 567–581.
- Gan, G., & Ng, M. K.-P. (2015). Subspace clustering with automatic feature  
1025 grouping. *Pattern Recognition*, 48, 3703–3713.
- Gan, G., & Wu, J. (2008). A convergence theorem for the fuzzy subspace clustering (fsc) algorithm. *Pattern Recognition*, 41, 1939–1947.



- Gançarski, P., & Blansch e, A. (2008). Darwinian, lamarckian, and baldwinian (co) evolutionary approaches for feature weighting in  $k$ -means-based algorithms. *IEEE Transactions on Evolutionary Computation*, *12*, 617–629.
- 1030 Garc a, S., Luengo, J., & Herrera, F. (2015). *Data preprocessing in data mining*. Springer.
- Garc a, S., Luengo, J., & Herrera, F. (2016). Tutorial on practical tips of the most influential data preprocessing algorithms in data mining. *Knowledge-Based Systems*, *98*, 1–29.
- 1035 Garc a-Guti errez, J., Mateos-Garc a, D., & Riquelme-Santos, J. C. (2014). Improving the  $k$ -nearest neighbour rule by an evolutionary voting approach. In *International Conference on Hybrid Artificial Intelligence Systems* (pp. 296–305). Springer.
- 1040 Garc a-Laencina, P. J., Sancho-G omez, J.-L., Figueiras-Vidal, A. R., & Verleysen, M. (2009).  $K$  nearest neighbours with mutual information for simultaneous classification and missing data imputation. *Neurocomputing*, *72*, 1483–1493.
- 1045 Giveki, D., Salimi, H., Bahmanyar, G., & Khademian, Y. (2012). Automatic detection of diabetes diagnosis using feature weighted support vector machines based on mutual information and modified cuckoo search. *arXiv preprint arXiv:1201.2173*, .
- Gou, J., Du, L., Zhang, Y., Xiong, T. et al. (2012). A new distance-weighted  $k$ -nearest neighbor classifier. *J. Inf. Comput. Sci*, *9*, 1429–1436.
- 1050 Gou, J., Xiong, T., & Kuang, Y. (2011). A novel weighted voting for  $k$ -nearest neighbor rule. *JCP*, *6*, 833–840.
- Granger, C. W. (1988). Causality, cointegration, and control. *Journal of Economic Dynamics and Control*, *12*, 551–559.

- Güneş, S., Polat, K., & Yosunkaya, Ş. (2010). Efficient sleep stage recognition  
1055 system based on eeg signal using k-means clustering based feature weighting.  
*Expert Systems with Applications*, *37*, 7922–7928.
- Gürüler, H. (2017). A novel diagnosis system for parkinson’s disease using  
complex-valued artificial neural network with k-means clustering feature  
weighting method. *Neural Computing and Applications*, *28*, 1657–1666.
- 1060 Hall, M. (2006). A decision tree-based attribute weighting filter for naive bayes.  
In *International Conference on Innovative Techniques and Applications of  
Artificial Intelligence* (pp. 59–70). Springer.
- Hall, M. (2007). A decision tree-based attribute weighting filter for naive bayes.  
*Knowledge-Based Systems*, *20*, 120 – 126. doi:[https://doi.org/10.1016/  
1065 j.knosys.2006.11.008](https://doi.org/10.1016/j.knosys.2006.11.008). AI 2006.
- Hamerly, G., & Elkan, C. (2004). Learning the k in k-means. In *Advances in  
neural information processing systems* (pp. 281–288).
- Hashemzadeh, M., Oskouei, A. G., & Farajzadeh, N. (2019). New fuzzy c-means  
clustering method based on feature-weight and cluster-weight learning.  
1070 *Applied Soft Computing*, *78*, 324–345.
- Huang, J. Z., Ng, M. K., Rong, H., & Li, Z. (2005). Automated variable  
weighting in k-means type clustering. *IEEE Transactions on Pattern Analysis  
& Machine Intelligence*, (pp. 657–668).
- Huang, X., Yang, X., Zhao, J., Xiong, L., & Ye, Y. (2018a). A new  
1075 weighting k-means type clustering framework with an l2-norm regularization.  
*Knowledge-Based Systems*, *151*, 165–179.
- Huang, X., Zhang, L., Wang, B., Zhang, Z., & Li, F. (2018b). Feature  
weight estimation based on dynamic representation and neighbor sparse  
reconstruction. *Pattern Recognition*, *81*, 388–403.

- 1080 Hung, W.-L., Chang, Y.-C., & Lee, E. S. (2011). Weight selection in wk-means  
algorithm with an application in color image segmentation. *Computers &  
Mathematics with Applications*, 62, 668–676.
- Hussain, S. F. (2019). A novel robust kernel for classifying high-dimensional  
data using support vector machines. *Expert Systems with Applications*, 131,  
1085 116–131.
- Jain, A., Nandakumar, K., & Ross, A. (2005). Score normalization in  
multimodal biometric systems. *Pattern recognition*, 38, 2270–2285.
- Jeong, Y.-S., & Jayaraman, R. (2015). Support vector-based algorithms with  
weighted dynamic time warping kernel function for time series classification.  
1090 *Knowledge-based systems*, 75, 184–191.
- Jeong, Y.-S., Jeong, M. K., & Omitaomu, O. A. (2011). Weighted dynamic time  
warping for time series classification. *Pattern Recognition*, 44, 2231–2240.
- Jiang, L., Li, C., Wang, S., & Zhang, L. (2016). Deep feature weighting for naive  
bayes and its application to text classification. *Engineering Applications of  
1095 Artificial Intelligence*, 52, 26–39.
- Jiang, L., Zhang, L., Li, C., & Wu, J. (2018). A correlation-based feature  
weighting filter for naive bayes. *IEEE Transactions on Knowledge and Data  
Engineering*, 31, 201–213.
- Jiang, L., Zhang, L., Yu, L., & Wang, D. (2019). Class-specific attribute  
1100 weighted naive bayes. *Pattern Recognition*, 88, 321–330.
- Jiao, L., Geng, X., & Pan, Q. (2019). Bp  $k$  nn:  $k$ -nearest neighbor classifier  
with pairwise distance metrics and belief function theory. *IEEE Access*, 7,  
48935–48947.
- Jiao, L., Pan, Q., & Feng, X. (2015). Multi-hypothesis nearest-neighbor classifier  
1105 based on class-conditional weighted distance metric. *Neurocomputing*, 151,  
1468–1476.

- Jiao, L., Pan, Q., Feng, X., & Yang, F. (2013). An evidential k-nearest neighbor classification method with weighted attributes. In *Proceedings of the 16th International Conference on Information Fusion* (pp. 145–150). IEEE.
- 1110 Jing, L., Ng, M. K., & Huang, J. Z. (2007). An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data. *IEEE Transactions on Knowledge & Data Engineering*, (pp. 1026–1041).
- Jing, L., Ng, M. K., Xu, J., & Huang, J. Z. (2005). Subspace clustering of text documents with feature weighting k-means algorithm. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining* (pp. 802–812).  
1115 Springer.
- John, G. H., & Langley, P. (1995). Estimating continuous distributions in bayesian classifiers. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence* (pp. 338–345). Morgan Kaufmann Publishers Inc.
- 1120 Jones, K. S., Walker, S., & Robertson, S. E. (2000). A probabilistic model of information retrieval: development and comparative experiments: Part 2. *Information processing & management*, 36, 809–840.
- Jović, A., Brkić, K., & Bogunović, N. (2015). A review of feature selection methods with applications. In *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)* (pp. 1200–1205). IEEE.
- 1125 Karypis, M. S. G., Kumar, V., & Steinbach, M. (2000). A comparison of document clustering techniques. In *TextMining Workshop at KDD2000 (May 2000)*.
- 1130 Komosiński, M., & Krawiec, K. (2000). Evolutionary weighting of image features for diagnosing of cns tumors. *Artificial Intelligence in Medicine*, 19, 25–38.
- Kononenko, I. (1994). Estimating attributes: analysis and extensions of relief. In *European conference on machine learning* (pp. 171–182). Springer.

- 1135 Kuo, R., & Nguyen, T. P. Q. (2019). Genetic intuitionistic weighted fuzzy  
k-modes algorithm for categorical data. *Neurocomputing*, *330*, 116–126.
- Lan, M., Tan, C. L., Su, J., & Lu, Y. (2008). Supervised and traditional term  
weighting methods for automatic text categorization. *IEEE transactions on  
pattern analysis and machine intelligence*, *31*, 721–735.
- 1140 Lee, C.-H., Gutierrez, F., & Dou, D. (2011). Calculating feature weights in  
naive bayes with kullback-leibler measure. In *2011 IEEE 11th International  
Conference on data mining* (pp. 1146–1151). IEEE.
- Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., & Liu,  
H. (2018). Feature selection: A data perspective. *ACM Computing Surveys  
(CSUR)*, *50*, 94.
- 1145 Li, Y., Luo, C., & Chung, S. M. (2012). Weighted naive bayes for text  
classification using positive term-class dependency. *International Journal on  
Artificial Intelligence Tools*, *21*, 1250008.
- 1150 Little, M. A., McSharry, P. E., Roberts, S. J., Costello, D. A., & Moroz, I. M.  
(2007). Exploiting nonlinear recurrence and fractal scaling properties for voice  
disorder detection. *Biomedical engineering online*, *6*, 23.
- Liu, C., Xie, J., Zhao, Q., Xie, Q., & Liu, C. (2019). Novel evolutionary  
multi-objective soft subspace clustering algorithm for credit risk assessment.  
*Expert Systems with Applications*, *138*, 112827.
- 1155 Marchiori, E. (2013). Class dependent feature weighting and k-nearest neighbor  
classification. In *IAPR International Conference on Pattern Recognition in  
Bioinformatics* (pp. 69–78). Springer.
- Martineau, J. C., & Finin, T. (2009). Delta tfidf: An improved feature space  
for sentiment analysis. In *Third international AAAI conference on weblogs  
and social media*.

- 1160 Mateos-García, D., García-Gutiérrez, J., & Riquelme-Santos, J. C. (2012). On  
the evolutionary optimization of k-nn by label-dependent feature weighting.  
*Pattern Recognition Letters*, *33*, 2232–2238.
- Mateos-García, D., García-Gutiérrez, J., & Riquelme-Santos, J. C. (2017).  
On the evolutionary weighting of neighbours and features in the k-nearest  
1165 neighbour rule. *Neurocomputing*, .
- Milligan, G. W., & Cooper, M. C. (1988). A study of standardization of variables  
in cluster analysis. *Journal of classification*, *5*, 181–204.
- Mohammed, A. W., & Zhang, M. (2008). Evaluation of particle swarm  
optimization based centroid classifier with different distance metrics. In  
1170 *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress  
on Computational Intelligence)* (pp. 2929–2932). IEEE.
- Ng, A. Y., Jordan, M. I., & Weiss, Y. (2002). On spectral clustering: Analysis  
and an algorithm. In *Advances in neural information processing systems* (pp.  
849–856).
- 1175 Niño-Adan, I., Landa-Torres, I., Portillo, E., & Manjarres, D. (2019). Analysis  
and application of normalization methods with supervised feature weighting  
to improve k-means accuracy. In *International Workshop on Soft Computing  
Models in Industrial and Environmental Applications* (pp. 14–24). Springer.
- Ouyed, O., & Allili, M. S. (2020). Group-of-features relevance in multinomial  
1180 kernel logistic regression and application to human interaction recognition.  
*Expert Systems with Applications*, *148*, 113247.
- Palacio-Niño, J.-O., & Berzal, F. (2019). Evaluation metrics for unsupervised  
learning algorithms. *arXiv preprint arXiv:1905.05667*, .
- Paltoglou, G., & Thelwall, M. (2010). A study of information retrieval weighting  
1185 schemes for sentiment analysis. In *Proceedings of the 48th annual meeting of  
the association for computational linguistics* (pp. 1386–1395). Association for  
Computational Linguistics.

- Panday, D., de Amorim, R. C., & Lane, P. (2018). Feature weighting as a tool for unsupervised feature selection. *Information Processing Letters*, *129*, 44–52.
- 1190 Paredes, R., & Vidal, E. (2000). A class-dependent weighted dissimilarity measure for nearest neighbor classification problems. *Pattern Recognition Letters*, *21*, 1027–1036.
- Paredes, R., & Vidal, E. (2006). Learning weighted metrics to minimize nearest-neighbor classification error. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (pp. 1100–1110).
- 1195 Parvin, H., & Minaei-Bidgoli, B. (2013). A clustering ensemble framework based on elite selection of weighted clusters. *Advances in Data Analysis and Classification*, *7*, 181–208.
- 1200 Peng, L., Zhang, H., Yang, B., & Chen, Y. (2014). A new approach for imbalanced data classification based on data gravitation. *Information Sciences*, *288*, 347 – 373. doi:<https://doi.org/10.1016/j.ins.2014.04.046>.
- Peng, L., Zhang, H., Zhang, H., & Yang, B. (2017). A fast feature weighting algorithm of data gravitation classification. *Information Sciences*, *375*, 54–78.
- 1205 Phan, A. V., Le Nguyen, M., & Bui, L. T. (2017). Feature weighting and svm parameters optimization based on genetic algorithms for classification problems. *Applied Intelligence*, *46*, 455–469.
- Polat, K. (2012). Classification of parkinson’s disease using feature weighting method on the basis of fuzzy c-means clustering. *International Journal of Systems Science*, *43*, 597–609.
- 1210 Portnoy, L. (2000). *Intrusion detection with unlabeled data using clustering*. Ph.D. thesis Columbia University.

- 1215 Raghu, S., & Sriraam, N. (2018). Classification of focal and non-focal eeg signals  
using neighborhood component analysis and machine learning algorithms.  
*Expert Systems with Applications*, *113*, 18–32.
- Ren, Z., Wu, B., Sun, Q., & Wu, M. (2019). Simultaneous learning of reduced  
prototypes and local metric for image set classification. *Expert Systems with  
Applications*, *134*, 102–111.
- 1220 Rendón, E., Abundez, I., Arizmendi, A., & Quiroz, E. M. (2011). Internal versus  
external cluster validation indexes. *International Journal of computers and  
communications*, *5*, 27–34.
- Romeo, L., Loncarski, J., Paolanti, M., Bocchini, G., Mancini, A., & Frontoni,  
E. (2020). Machine learning-based design support system for the prediction  
1225 of heterogeneous machine parameters in industry 4.0. *Expert Systems with  
Applications*, *140*, 112869.
- Saaty, T. L. (2014). Analytic heirarchy process. *Wiley statsRef: Statistics  
reference online*, .
- Saeyns, Y., Inza, I., & Larrañaga, P. (2007). A review of feature selection  
1230 techniques in bioinformatics. *bioinformatics*, *23*, 2507–2517.
- Sález, J. A., Derrac, J., Luengo, J., & Herrera, F. (2014). Statistical computation  
of feature weighting schemes through data estimation for nearest neighbor  
classifiers. *Pattern Recognition*, *47*, 3941–3948.
- Saha, A., & Das, S. (2015). Categorical fuzzy k-modes clustering with  
1235 automated feature weight learning. *Neurocomputing*, *166*, 422–435.
- Sahin, E. K., Ipbuker, C., & Kavzoglu, T. (2015). A comparison of feature  
and expert-based weighting algorithms in landslide susceptibility mapping.  
*Procedia Earth and Planetary Science*, *15*, 462–467.
- Sakar, C. O., & Kursun, O. (2010). Telediagnosis of parkinson’s disease using  
1240 measurements of dysphonia. *Journal of medical systems*, *34*, 591–599.



- San, O. M., Huynh, V.-N., & Nakamori, Y. (2004). An alternative extension of the k-means algorithm for clustering categorical data. *International Journal of Applied Mathematics and Computer Science*, *14*, 241–247.
- Schölkopf, B., Williamson, R. C., Smola, A. J., Shawe-Taylor, J., & Platt, J. C. (2000). Support vector method for novelty detection. In *Advances in neural information processing systems* (pp. 582–588).
- Serrano-Silva, Y. O., Villuendas-Rey, Y., & Yáñez-Márquez, C. (2018). Automatic feature weighting for improving financial decision support systems. *Decision Support Systems*, *107*, 78–87.
- Shen, H., Yang, J., Wang, S., & Liu, X. (2006). Attribute weighted mercer kernel based fuzzy clustering algorithm for general non-spherical datasets. *Soft Computing*, *10*, 1061–1073.
- Sinciya, P., & Celin, J. J. A. (2017). Weight optimized gravitational classifier for high dimensional numerical data classification. *International Journal of Pure and Applied Mathematics*, *116*, 251–263.
- Sotoodeh, M., Moosavi, M. R., & Boostani, R. (2019). A novel adaptive lbp-based descriptor for color image retrieval. *Expert Systems with Applications*, *127*, 342–352.
- Sparck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, *28*, 11–21.
- Sun, Y. (2007). Iterative relief for feature weighting: algorithms, theories, and applications. *IEEE transactions on pattern analysis and machine intelligence*, *29*, 1035–1051.
- Taheri, S., Yearwood, J., Mammadov, M., & Seifollahi, S. (2014). Attribute weighted naive bayes classifier using a local optimization. *Neural Computing and Applications*, *24*, 995–1002.

- Tahir, M. A., Bouridane, A., & Kurugollu, F. (2007). Simultaneous feature selection and feature weighting using hybrid tabu search/k-nearest neighbor classifier. *Pattern Recognition Letters*, *28*, 438–446.
- 1270 Triguero, I., Derrac, J., García, S., & Herrera, F. (2012). Integrating a differential evolution feature weighting scheme into prototype generation. *Neurocomputing*, *97*, 332–343.
- Tsai, C.-Y., & Chiu, C.-C. (2008). Developing a feature weight self-adjustment mechanism for a k-means clustering algorithm. *Computational statistics & data analysis*, *52*, 4658–4672.
- 1275 Tzortzis, G., & Likas, A. (2014). The minmax k-means clustering algorithm. *Pattern Recognition*, *47*, 2505–2516.
- Venkatesh, B., & Anuradha, J. (2019). A review of feature selection and its methods. *Cybernetics and Information Technologies*, *19*, 3–26.
- 1280 Wang, S., Jiang, L., & Li, C. (2014). A cfs-based feature weighting approach to naive bayes text classifiers. In *International Conference on Artificial Neural Networks* (pp. 555–562). Springer.
- Wang, X., Wang, Y., & Wang, L. (2004). Improving fuzzy c-means clustering based on feature-weight learning. *Pattern recognition letters*, *25*, 1123–1132.
- 1285 Wang, Y., & Chen, L. (2017). Multi-view fuzzy clustering with minimax optimization for effective clustering of data from multiple sources. *Expert Systems with Applications*, *72*, 457–466.
- Wei, P., Lu, Z., & Song, J. (2015). Variable importance analysis: a comprehensive review. *Reliability Engineering & System Safety*, *142*, 399–432.
- 1290 Wemmert, C., Gançarski, P., & Korczak, J. J. (2000). A collaborative approach to combine multiple learning methods. *International Journal on Artificial Intelligence Tools*, *9*, 59–78.

- 1295 Wettschereck, D., & Aha, D. W. (1995). Weighting features. In *International Conference on Case-Based Reasoning* (pp. 347–358). Springer.
- Wettschereck, D., Aha, D. W., & Mohri, T. (1997). A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. *Artificial Intelligence Review*, *11*, 273–314.
- 1300 Wettschereck, D., & Dietterich, T. G. (1995). An experimental comparison of the nearest-neighbor and nearest-hyperrectangle algorithms. *Machine learning*, *19*, 5–27.
- Wu, H., Gu, X., & Gu, Y. (2017). Balancing between over-weighting and under-weighting in supervised term weighting. *Information Processing & Management*, *53*, 547–557.
- 1305 Wu, H., & Salton, G. (1981). A comparison of search term weighting: term relevance vs. inverse document frequency. In *Proceedings of the 4th annual international ACM SIGIR conference on Information storage and retrieval: theoretical issues in information retrieval* (pp. 30–39).
- 1310 Wu, J., & Cai, Z. (2011). Attribute weighting via differential evolution algorithm for attribute weighted naive bayes (wnb). *Journal of Computational Information Systems*, *7*, 1672–1679.
- Wu, J., Pan, S., Zhu, X., Cai, Z., Zhang, P., & Zhang, C. (2015). Self-adaptive attribute weighting for naive bayes classification. *Expert Systems with Applications*, *42*, 1487–1502.
- 1315 Xia, H., Zhuang, J., & Yu, D. (2013). Novel soft subspace clustering with multi-objective evolutionary approach for high-dimensional data. *Pattern Recognition*, *46*, 2562–2575.
- 1320 Xiang, Z.-L., Yu, X.-R., & Kang, D.-K. (2016). Experimental analysis of naive bayes classifier based on an attribute weighting framework with smooth kernel density estimations. *Applied Intelligence*, *44*, 611–620.

- Xing, H.-j., Ha, M.-h., Hu, B.-g., & Tian, D.-z. (2009). Linear feature-weighted support vector machine. *Fuzzy Information and Engineering*, *1*, 289–305.
- Yang, W., Wang, K., & Zuo, W. (2012). Neighborhood component feature selection for high-dimensional data. *JCP*, *7*, 161–168.
- 1325 Yeung, D.-Y., & Chow, C. (2002). Parzen-window network intrusion detectors. In *Object recognition supported by user interaction for service robots* (pp. 385–388). IEEE volume 4.
- Yilmaz, T., Yazici, A., & Kitsuregawa, M. (2014). Relief-mm: effective modality weighting for multimedia information retrieval. *Multimedia systems*, *20*,  
1330 389–413.
- Yip, K. Y., Cheung, D. W., & Ng, M. K. (2004). Harp: A practical projected clustering algorithm. *IEEE Transactions on knowledge and data engineering*, *16*, 1387–1397.
- Zaidi, N. A., Cerquides, J., Carman, M. J., & Webb, G. I. (2013). Alleviating  
1335 naive bayes attribute independence assumption by attribute weighting. *The Journal of Machine Learning Research*, *14*, 1947–1988.
- Zhang, G.-Y., Wang, C.-D., Huang, D., Zheng, W.-S., & Zhou, Y.-R. (2018). Tw-co-k-means: two-level weighted collaborative k-means for multi-view clustering. *Knowledge-Based Systems*, *150*, 127–138.
- 1340 Zhang, H., & Sheng, S. (2004). Learning weighted naive bayes with accurate ranking. In *Fourth IEEE International Conference on Data Mining (ICDM'04)* (pp. 567–570). IEEE.
- Zhang, L., Jiang, L., Li, C., & Kong, G. (2016). Two feature weighting  
1345 approaches for naive bayes text classifiers. *Knowledge-Based Systems*, *100*, 137–144.
- Zhi, X.-b., Fan, J.-l., & Zhao, F. (2014). Robust local feature weighting hard c-means clustering algorithm. *Neurocomputing*, *134*, 20–29.

- Zhou, J., Chen, L., Chen, C. P., Zhang, Y., & Li, H.-X. (2016). Fuzzy clustering with the entropy of attribute weights. *Neurocomputing*, *198*, 125–134.
- <sup>1350</sup> Zhou, S., Qian, S., Chang, W., Xiao, Y., & Cheng, Y. (2018). A novel bearing multi-fault diagnosis approach based on weighted permutation entropy and an improved svm ensemble classifier. *Sensors*, *18*, 1934.
- Zhou, Z., & Zhu, S. (2018). Kernel-based multiobjective clustering algorithm with automatic attribute weighting. *Soft Computing*, *22*, 3685–3709.

## **II Influence of Statistical Feature Normalisation Methods on K-Nearest Neighbours and K-means in the Context of Industry 4.0**

**Iratxe Niño-Adan, Itziar Landa-Torres, Eva Portillo, Diana Manjarres**

To appear in *Engineering Applications of Artificial Intelligence*,  
(Second round of revision. Accepted by three out of four reviewers.  
Pending approval of fourth reviewer.)

**JCR: 6.212**

*Categories: Computer Science, Artificial Intelligence, 26/139, Q1,*  
*Automation & Control Systems, 6/63, Q1*

# Influence of Statistical Feature Normalisation Methods on K-Nearest Neighbours and K-Means in the Context of Industry 4.0.

Iratxe Niño-Adan<sup>a,b,1</sup>, Itziar Landa-Torres<sup>c</sup>, Eva Portillo<sup>b</sup>, Diana Manjarres<sup>a</sup>

<sup>a</sup>*TECNALIA, Basque Research and Technology Alliance (BRTA), 48160 Derio, Spain*

<sup>b</sup>*Department of Automatic Control and Systems Engineering, Faculty of Engineering, University of the Basque Country UPV/EHU, Bilbao, Spain*

<sup>c</sup>*Petronor Innovación S.L. 48550 Muskiz, Spain*

---

## Abstract

Normalisation is a preprocessing technique widely employed in Machine Learning (ML)-based solutions for industry to equalise the features' contribution. However, few researchers have analysed the normalisation effect and its implications on the ML algorithm performance, especially on Euclidean distance-based algorithms, such as the well-known K-Nearest Neighbours and K-means. In this sense, this paper formally analyses the impact of Normalisation yielding results significantly far from the state-of-the-art traditional claims. In particular, this paper shows that normalisation does not equalise the contribution of the features to distance-based ML algorithm and that each Normalisation method transforms the features of the dataset differently, with the consequent impact on the results and performance of the learning process for a particular problem. This paper concludes that normalisation can be viewed as an unsupervised Feature Weighting method. In this context, a new metric (*Normalisation weight*) for measuring the future impact of Normalisation on the features is presented. Likewise, an analysis of the normalisation effect on the Euclidean distance-based algorithms is conducted and a new metric referred to as *Proportional influence* that measures the features influence on the Euclidean distance-based ML algorithms is proposed. Both metrics enable the automatic selection of the most appropriate Normalisation method for a particular engineering problem. Furthermore, the optimal Normalisation method selection can significantly improve both the ML algorithm's computational cost and classification performance. The analytical conclusions are validated on well-known datasets from the UCI repository and a real-life application from the refinery industry.

*Keywords:* Feature Normalisation, Feature Weighting, Machine Learning, Euclidean distance, K-Nearest Neighbours, K-means

---

*Email address:* iratxe.nino@tecnalia.com (Iratxe Niño-Adan)

*Preprint submitted to Engineering Applications of Artificial Intelligence 24th November 2021*

## 1. Introduction

Every second a huge amount of data is collected in many engineering applications within the context of Industry 4.0. Machine Learning (ML) algorithms are commonly applied to extract valuable information from such data. The main goal of ML algorithms in this context is to build, based on input data  $X$  ( $n$  samples,  $m$  features), a mathematical model  $f: X \subset \mathbb{R}^{n \times m} \rightarrow \hat{Y}$  about an engineering system so as  $\hat{Y} \approx Y$  (real output), in order to make accurate predictions or decisions, among others.

In the literature several approaches rely on Euclidean distance-based ML algorithms, i.e. K-Nearest Neighbours (K-NN) and K-means. For instance, two versions of the K-NN are presented by Borghesan et al. (2019) for persistent disturbances prediction. Authors in (Semenov et al., 2020) employ K-NN for state analysis of hard-to-reach mechatronic units and devices. K-NN is also utilised by Chelmiah et al. (2020) for wear state classification and Remaining Useful Life (RUL) estimation of the mechanical rolling element bearings. Concerning the utilisation of K-means in industrial applications, Yao & Ge (2019) propose the K-means to discriminate the different process modes for Residual CO<sub>2</sub> content estimation in a Predecarburization unit. A methodology that combines Conditional Restricted Boltzmann Machines with K-means is presented in (Gutierrez-Torre et al., 2020). Its goal is to improve the quality of data sent to an Automatic Identification System. Also, Cavalcante et al. (2019) introduce a continuous learning-from-data algorithm for history-matching problems -classical petroleum reservoir engineering data assimilation process that reproduces the behaviour of a real reservoir-, that includes a Region Definition step based on K-means algorithm.

When modelling a problem with data-driven methods, as important as the selection of the ML algorithm, the collection of the data is. In fact, the modelling ability and reliability are dependent on the quality of the input data (Cortés-Ibáñez et al., 2020). If the acquired samples do not represent information about the engineering phenomenon or if the dataset contains disturbing features, the algorithm cannot create an accurate model. Consequently, data preprocessing (Famili et al., 1997; García et al., 2015) is a fundamental step in ML problems. In this context, one traditional approach is analysing and adapting the features that compose the dataset. Specifically, the expert knowledge or the employment of Feature Selection (FS) methods (Chandrashekar & Sahin, 2014; Li et al., 2017; Rostami et al., 2021; Di Mauro et al., 2021; Hassani et al., 2021), which select the relevant features for modelling the problem, are the mainly followed approaches in the literature. Note that FS is conducted by assigning a factor equal to 1 to the selected features and 0 to the others. Therefore, assigning the same factor value implicitly means that the selected features are equally important for the model, which may not be realistic. In order to circumvent it, a generalisation of FS is the Feature Weighting (FW) (Wettschereck et al., 1997; de Amorim, 2016; Deng et al., 2016). This approach assigns different weights to each feature according to its relative relevance for representing the output. Traditionally, the weights range from 0 to 1 so that the weights' sum equals 1.



However, the differences between the original magnitude of the features can result in both FS and FW approaches in an over-influence of a set of features in the ML algorithm’s metric. Thus, in FS, features with higher magnitude can dominate the calculations. On the other hand, the magnitude differences can disturb the influence of the FW-based weights on transforming the space. For all these reasons, the third preprocessing technique highlighted in the state-of-the-art is Feature Normalisation (FN).

FN is claimed to be particularly useful in statistical learning methods (García et al., 2015). It is an extended practice in ML problems in order to equate the magnitude of the features. Besides, it is thought to equalise their contribution in the ML algorithm calculations. There are several FN methods, of which Standardisation (Zhu et al., 2011; Vanini et al., 2014; Hsu et al., 2018; Curreri et al., 2020; Plawiak et al., 2020) and Min-max normalisation (Park et al., 2005; Qiu et al., 2017; Chu et al., 2020; Peng et al., 2020) are the most popular ones. Some works empirically investigate the most suitable FN technique for a particular problem. For instance, Milligan & Cooper (1988) study the effect of 8 FN methods over 4 agglomerate clustering algorithms applied on synthetic data. Results reveal that the range-based methods present consistently superior recovery of the underlying cluster structure. Similar experiments are conducted in (Schaffer & Green, 1996) and (Chu et al., 2009) in Social sciences and Chemical fields, respectively. In addition to the analysis over clustering algorithms, Bhanja & Das (2018) compare the impact of different linear and non-linear FN methods on a Deep Recurrent Neural Network for predicting the closing index of Bombay and New York Stock Exchange. Recently, Singh & Singh (2019) study the effect of 14 FN methods on K-NN classification performance considering FS and FW. Experimental analysis over 21 synthetic and real datasets from UCI repository are conducted. Based on the obtained results, the authors conclude that FN affects the datasets’ features properties, which causes a change in the features’ relevance. They also state that the mean and standard deviation measures are the most recommendable ones to transform  $X$ . As it can be noticed from the related works, there is no general agreement about the suitability of a particular FN method’s employment, since the conclusions are obtained from experimental analysis.

Then, this work advances over the state-of-the-art by presenting a theoretical analysis of the FN effect on the dataset transformation and its implications on distance-based ML algorithms. Specifically, in this work, the implications of the FN method selection on K-NN and K-means algorithms are analysed. Besides, two new metrics are proposed: a metric, called *Normalisation weight*, for measuring the normalisation effect on transforming each feature of the dataset, and a second metric for measuring the features’ influence on the Euclidean distance-based ML algorithm calculations, referred to as *Proportional influence*. The theoretical conclusions are experimentally validated with different well-known datasets from UCI repository as well as with real industrial use case. Furthermore, the implications on the Euclidean distance, more concretely on the neighbours’ selection of K-NN and its impact on the K-NN performance, and the convergence of K-means algorithms, are deeply analysed. The results show that

the optimal FN method selection can significantly improve both the classification performance and Euclidean distance-based ML algorithm’s computational cost.

95 The presented conclusions and novel metrics not only benefice the data analysis practitioners during the preprocessing, specifically in high-dimensional datasets composed of hundreds of features -such as in Big Data context or Industry 4.0- but also establish a systematic methodology for learning process automation or AutoML. In this line, a roadmap for the selection of the suitable  
100 FN method for a new given problem is also suggested.

This work is organised as follows: Section 2 presents the theoretical analysis of the normalisation effect on the dataset transformation and its implications on the Euclidean distance. As a result, a new metric for measuring the influence of the FN method is proposed. Section 3 describes the FN methods, the  
105 ML algorithms and the performance measures employed for the experimental analysis. Besides, the new metric for measuring the features influence on the Euclidean distance-based ML algorithm is presented. The experimental results regarding the UCI datasets and the industrial case are collected in Sections 4 and 5, respectively. Section 6 suggests a roadmap for the FN method selection.  
110 Finally, Section 7 gathers the conclusions and the paper’s contributions.

## 2. Hypothesis and Foundations

### 2.1. Notations and Definitions

The statistical FN methods analysed in this work are defined as:

$$\tilde{X}^{Norm} = \frac{X - \mathbf{pos}(\mathbf{X})}{\mathbf{dis}(\mathbf{X})} \quad (1)$$

115 where  $\mathbf{pos}(\mathbf{X})$  refers to the position or central tendency statistic vector<sup>1</sup> that centres the values of the features, and  $\mathbf{dis}(\mathbf{X})$  corresponds to the dispersion statistic vector which scales the features.

As above-mentioned, FN is a preprocessing technique usually employed to avoid magnitude differences among the features. In order to highlight the magnitude corresponding to the values  $x_{ij}$   $i = \{1, \dots, n\}$  of each feature  $X_j$ ,  
120  $j = \{1, \dots, m\}$ , they can be expressed by *decimal notation* as follows:

$$x_{ij} = \mathit{sign}(x_{ij}) 0.d_1d_2d_3 \dots \cdot 10^{n_j} = \widehat{x}_{ij} \cdot 10^{n_j} \quad (2)$$

where  $d_1, d_2, d_3, \dots \in \{0, 1, \dots, 9\}$  and  $n_j \in \mathbb{Z}$  in such a way that,  $\forall j$ ,  $|n_j|$  is the minimum number which fulfils  $X_j = \widehat{X}_j \cdot 10^{n_j}$  and  $\max\{|\widehat{X}_j|\} < 1$ . Then,  $10^{n_j}$  represents the  $j$ -th feature’s magnitude factor, and  $\widehat{X}_j$  is the demagnified feature where each value is  $|\widehat{x}_{ij}| < 1 \forall i$ .

---

<sup>1</sup>For the sake of brevity, the vector composed by position or central tendency statistic is referred as position statistic from now on.

125 *2.2. Normalisation Effect on Transforming the Dataset*

In order to analyse the effect of FN on the magnitude removal, decimal notation (Equation 2) is employed. Since the statistical factors are estimated by linear operations,  $pos(X_j) = pos(\widehat{X}_j) \cdot 10^{n_j}$  and  $dis(X_j) = dis(\widehat{X}_j) \cdot 10^{n_j}$ . Then, the linear-based statistical FN methods can be stated for each feature as:

130 
$$\widetilde{X}_j^{Norm} = \frac{\widehat{X}_j \cdot 10^{n_j} - pos(\widehat{X}_j) \cdot 10^{n_j}}{dis(\widehat{X}_j) \cdot 10^{n_j}} = \frac{\widehat{X}_j - pos(\widehat{X}_j)}{dis(\widehat{X}_j)} \quad (3)$$

As Equation 3 shows, the magnitude factor in the transformed dataset disappears. However, each feature  $j$  is scaled by a dispersion factor  $dis(\widehat{X}_j)$  calculated regarding the distribution of its values. It is thought that FN equalises the features' contribution. However, in Equation 3, since each feature presents a different dispersion value, each feature will be transformed differently. Besides, since each FN method employs different statistical dispersion measures, it is expected that each FN method will transform a given dataset diversely.

In fact, Equation 4 demonstrates that FN only equalises the dispersion of the features in the transformed space, not the feature's contribution.

$$dis(\widetilde{X}_j^{Norm}) = dis\left(\frac{\widehat{X}_j}{dis(\widehat{X}_j)} - \frac{pos(\widehat{X}_j)}{dis(\widehat{X}_j)}\right) = dis\left(\frac{\widehat{X}_j}{dis(\widehat{X}_j)}\right) = \frac{dis(\widehat{X}_j)}{dis(\widehat{X}_j)} = 1 \quad (4)$$

140 *2.3. Normalisation Effect on the Euclidean Distance*

So far, the FN effect on transforming the dataset has been analysed separately for each feature of the dataset. Nevertheless, the  $i$ -th sample of the dataset  $x_i = [x_{i1} \ x_{i2} \ \dots \ x_{im}]$  is defined by its values for every feature simultaneously. Therefore, not only the analysis of each transformed feature is important, but also the interrelation between the resultant features. In fact, except for particular cases, such as Random Forest, most of the metrics employed by ML algorithms are computed using the sample's joint information of all the features. Some ML algorithms, such as Support Vector Machines or Artificial Neural Networks, include internal weights that adjust the feature's contribution. However, for K-Nearest Neighbours (K-NN) and K-means Euclidean distance-based ML algorithms, the calculations are performed directly over the preprocessed dataset. Then, the relationship between the features directly determines the calculations of the metrics. Thus, for K-NN and K-means algorithms it is possible to analyse and analytically formalise the FN method's influence on the learning process.

A distance is a mathematical function that measures how far two points are located between them. One of the most commonly utilised distance metrics is the Euclidean, which is defined as the sum of the square difference for each component or feature between a pair of points:

$$d_E(\mathbf{x}, \mathbf{y})^2 = \sum_{j=1}^m (x_j - y_j)^2 \quad (5)$$

which, by the employment of Equation 2 can be rewritten as:

$$d_E^2(\mathbf{x}, \mathbf{y}) = \sum_{j=1}^m ((\hat{x}_j - \hat{y}_j) \cdot 10^{n_j})^2 \quad (6)$$

The features with higher  $n_j$  value will contribute more to Equation 6. Therefore, FN is traditionally recommended to avoid such dominance. The Euclidean distance between two normalised samples  $\tilde{\mathbf{x}}, \tilde{\mathbf{y}}$  is given by:

$$d_E^2(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}) = \sum_{j=1}^m \left( \frac{(\hat{x}_j - \hat{y}_j) \cdot 10^{n_j}}{\widehat{dis}(X_j) \cdot 10^{n_j}} \right)^2 = \sum_{j=1}^m \left( \frac{\hat{x}_j - \hat{y}_j}{\widehat{dis}(X_j)} \right)^2 \quad (7)$$

Again, the magnitude disappears, but the dispersion factor still remains in the formula. Moreover, since  $\hat{x}_j, \hat{y}_j \in (-1, 1)$ , then  $(\hat{x}_j - \hat{y}_j) \in (-2, 2) \forall j$ .

**Proof 1.** Based on the definition of the decimal notation it can be proven for Equation 7 by *reductio ad absurdum* that the value of  $|\hat{x}_j - \hat{y}_j| < 2$ .

Assuming  $|\hat{x}_j - \hat{y}_j| \geq 2$ :

- If  $y \geq 0$ :  $|x - y| = |x| + y \geq 2 \iff |x| \geq 2 - y$  and by definition of the decimal notation  $y < 1$ , thus  $|x| \geq 2 - y \implies |x| > 1$ , which is against Equation 2.
- If  $y < 0$ :  $|x - y| = |x| - y \geq 2 \iff |x| \geq 2 + y \xrightarrow{y \geq -1} |x| > 1$ .

Analogously, for  $x \geq 0$  and  $x < 0$ . Therefore, it is proven that  $|\hat{x}_j - \hat{y}_j|$  cannot be greater or equal to two. Thus,  $|\hat{x}_j - \hat{y}_j| < 2$ .

In this sense, despite each numerator in Equation 7 takes values in  $(-2, 2)$ , due to the dispersion factor each term will present a different influence on the Euclidean distance. This means that the dispersion factors derived from FN can alter the terms' or features' influence on the samples' distance calculations.

**Example 2.1.** Figure 1 depicts features 1 and 10 of Accent dataset from UCI and their normalisation by Standardisation (ST) and Min-Max (MM) methods.

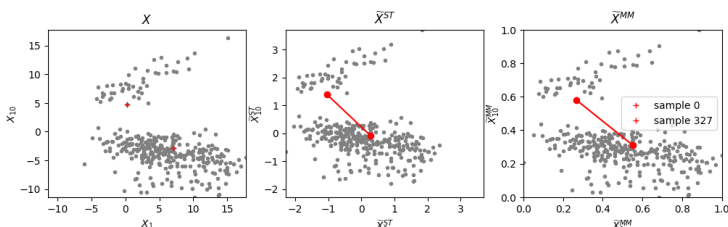


Figure 1: Example of FN influence on features transformation.

Feature 1 will over-influence the Euclidean distance calculation between samples  $\mathbf{x}$  and  $\mathbf{y}$  if  $DD = \widehat{dis}(X_1)/\widehat{dis}(X_{10}) < |x_1 - y_1|/|x_{10} - y_{10}| = dd$ . In this case, for samples 0 and 327,  $dd = 0.906$ . Features 1 and 10 have a std value of 5.105 and 5.042, respectively; and their range is equal to 23.818 and 27.756, respectively. Thus, when normalising the dataset with ST,  $DD = 1.013 > dd$ ,

and consequently, feature 10 over-influences the Euclidean distance calculation. In contrast, if MM FN method is employed, since  $DD = 0.858 < dd$ , feature 1 over-influences the Euclidean distance calculations.

In addition, depending on the difference of the dispersion factors, a subset of features can dominate the Euclidean distance calculations. Thus, if  $dis(\widehat{X}_S) \ll dis(\widehat{X}_H)$  for  $S \cup H = \{1, \dots, m\}$  and  $S \cap H = \emptyset$

$$d_E(\tilde{\mathbf{x}}, \tilde{\mathbf{y}})^2 = \sum_{s \in S} \left( \frac{(\hat{x}_s - \hat{y}_s)}{dis(\widehat{X}_s)} \right)^2 + \sum_{h \in H} \left( \frac{(\hat{x}_h - \hat{y}_h)}{dis(\widehat{X}_h)} \right)^2 \approx \sum_{s \in S} \left( \frac{(\hat{x}_s - \hat{y}_s)}{dis(\widehat{X}_s)} \right)^2 \quad (8)$$

In the light of the above, it is concluded that the statistical FN methods, more concretely, the dispersion factors, influence the Euclidean distance.

#### 2.4. Measuring the Normalisation Influence on the Features Transformation

As concluded above, the FN methods not only transform each feature of the dataset differently but also influence the calculation of the Euclidean distance. Conversely, as stated in Section 1, Feature Weighting assigns a weight to each feature proportional to a given property of the feature. This way, the Euclidean distance calculation between two weighted samples is defined as:

$$wd_E^2(\mathbf{x}, \mathbf{y}) = \sum_{j=1}^m (w_j^2 \cdot (\hat{x} - \hat{y}) \cdot 10^{n_j})^2 \quad (9)$$

By comparing Equations 9 with the Euclidean distance between two normalised samples (Equation 7) it is concluded that the inverse of the dispersion statistic acts as a feature weight. Thus, this work states that FN is a particular case of unsupervised FW where the feature weights are estimated according to the statistical dispersion of the features.

Based on this conclusion, and taking advantage of the FW perspective, where the feature weights  $w_j \in [0, 1]$  in such manner that  $\sum_{j=1}^m w_j = 1$ , if the dispersion statistical factor did not alter the features contribution, then  $w_j^* = 1/m$ . Thus, by comparing  $w_j^{norm} = dis(\widehat{X}_j)^{-1} / \sum_{j=1}^m dis(\widehat{X}_j)^{-1}$  respect to the ideal weight  $w_j^*$ , a metric referred as *Normalisation weight* ( $Nw$ ) for estimating the over or under-influence each feature will have on the calculations is stated:

$$Nw = \left[ sign(w_1^{norm} - w_1^*) \cdot \frac{w_1^{norm}}{w_1^*}, \dots, sign(w_m^{norm} - w_m^*) \cdot \frac{w_m^{norm}}{w_m^*} \right] \quad (10)$$

Each term of Equation 10 poses for each component  $j$  the difference between the ideal weight  $w^*$  and the one assigned by  $w^{norm}$ . The sign  $sign(w_j^{norm} - w_j^*)$  represents the over-influence (+) or under-influence (-) that the feature will have depending on the FN method due to the statistical dispersion factor. Thus, the metric proposed in Equation 10 depicts the transformation over the features caused by the FN method to equalise the features dispersion (Equation 4). Note that the values obtained by Equation 10 do not represent the relative importance of the ML calculations' features but symbolise how each feature is transformed by a FN method, which ultimately influences the ML calculations.

### 3. Methods

225 In order to complement the theoretical analysis conducted in Section 2, an empirical study of the FN methods influence is undertaken. For doing so, first the selected algorithms and FN methods are presented. Then, the methodology of the analysis is thoughtfully described.

#### 3.1. Normalisation Methods, Metrics and ML algorithms

230 Next, the normalisation methods, algorithms and metrics utilised in this work to experimentally validate the conclusions from Section 2 are described.

##### 3.1.1. Normalisation Methods

Normalisation method	pos(X)	dis(X)
Standardisation (ST) (Milligan & Cooper, 1988; Singh & Singh, 2019)	$\bar{X}$	$\frac{\sigma_X}{\bar{X}}$
Variable Stability Scaling (van den Berg et al., 2006; Walach et al., 2018)	$\frac{\bar{X}}{X}$	$\frac{\sigma_X}{\bar{X}}$
Pareto scaling (Noda, 2008; Trim et al., 2008)	$\bar{X}$	$\sqrt{\sigma_X}$
Min-Max normalisation (MM) (Milligan & Cooper, 1988; Singh & Singh, 2019)	min(X)	range(X)
Range scaling (Hurley et al., 2019; Julian et al., 2019)	$\bar{X}$	range(X)
Unitisation (Milligan & Cooper, 1988)	0	range(X)
Robust scaler (Thara et al., 2019; Vaitheswari & SathieshKumar, 2019)	Me(X)	IQR(X)
MAD normalisation (Kundu & Ari, 2017; Bergen & Beroza, 2019)	Me(X)	mad(X)

Table 1: Linear-based normalisation methods.

235 Table 1 collects diverse FN methods resulting from the combination of different position and dispersion statistics. As stated in Section 2, the dispersion statistic expands or compresses in different degree the values of the features. Accordingly, the FN methods based on the same dispersion statistic, such as Min-Max normalisation and Range scaling, will perform the same transformation over the data. Thus, the features transformed by any of these two methods will present the same features' influence on the Euclidean distance calculations.

240 In this work, three well-known FN methods from Table 1 will be utilised: ST, MM and MAD. The original and the normalised datasets  $X$ ,  $\tilde{X}^{ST}$ ,  $\tilde{X}^{MM}$ ,  $\tilde{X}^{MAD}$  are employed to validate the conclusions from Section 2.

##### 3.1.2. Features' Proportional Influence on distance-based ML Algorithms

245 As demonstrated in Section 2, each FN method transforms the features of a given dataset differently, which alters their contribution to the Euclidean distance-based ML calculations. Thus, the higher the separation between the samples or subsets of them, the higher the influence of such features for classifying or clustering the dataset samples in Euclidean distance-based algorithms.

250 The experimental research presented in (Singh & Singh, 2019) identifies the dominant features by comparing the features' range. The features with higher range are considered to have the most influence on the ML calculations. However, since MM normalisation employs the range to transform the features, according to Equation 4, by this approach it could be interpreted that MM method equalises the contribution of the features. The same reasoning is followed to discard any other statistical dispersion from Table 1. This is why in 255 this work, a new metric for measuring each feature's degree of influence on the

Euclidean distance-based ML algorithm calculations, referred to as *Proportional influence*, is presented.

In contrast to the range calculated with the extreme values of the feature, i.e. with the maximum and minimum values, the proposed metric employs not only the extreme values but also the mean value of the feature as follows:

$$Infl(X_j) = \max \{ |(\max(X_j) - \bar{X}_j)|, |(\min(X_j) - \bar{X}_j)| \} \quad (11)$$

where

$$Infl(X_j) < range(X_j) \leq 2 \cdot Infl(X_j) \quad (12)$$

Equation 11 measures the dispersion of each feature as the maximum difference, in absolute value, of the feature's mean respect to the most distant value (the maximum or the minimum value of the feature). In this sense,  $Infl(X_j)$  considers the mean around which the samples are allocated, and the maximum distance from the samples to the mean. If the samples of a feature are concentrated around a point, the value of  $Infl(X_j)$  is lower than if the samples are more scattered between them.

In order to compare the feature's relative influence on the Euclidean distance-based ML algorithm calculations, the proportional influence  $IN$  is considered:

$$IN(X_j) = Infl(X_j) / \max\{Infl(X_j) | j = \{1, \dots, m\}\} \quad (13)$$

### 3.1.3. Euclidean distance-based ML algorithms and performance measures

Two widely employed Euclidean distance-based ML algorithms are utilised to validate the conclusions stated in Section 2.3.

- K-Nearest Neighbours (K-NN) (Cunningham & Delany, 2020) is a classification algorithm that establishes the label of a given sample based on the class membership of its  $K$  closest samples in terms of Euclidean distance.
- K-means (Jain, 2010) is a clustering algorithm that groups the samples of the dataset in  $K$  different disjoint groups. The groups are formed in such a way that the distribution of the samples among the groups maximises the intra-group cohesion, i.e. the distance of the samples to its centroid.

In order to analyse and quantify the differences in the classification performance (Sokolova & Lapalme, 2009) resulting from the FN method selection, the accuracy, precision, and recall measures are employed.

- Accuracy is defined as the total number of correctly classified samples divided by the total number of samples. For imbalanced datasets, the accuracy value is dominated by the majority class.
- Precision measures the proportion of samples assigned label 1 correctly classified respect to all the samples classified with label 1.
- Recall represents the proportion of samples from class 1 correctly detected by the algorithm.

In multi-class classification problems, macro-precision and macro-recall measures are employed, estimated as the average of the precision and recall obtained for each class, respectively.

295 *3.2. Analysis of FN influence on transforming the dataset*

After normalising the raw dataset  $X$  with the three FN methods selected in Section 3.1.1, the *Normalisation weight* is calculated with the aim of quantifying the degree of compression/expansion imposed by each FN to each feature.

300 Then, the features' Proportional influence  $IN$  (Equation 13) presented in Section 3.1.2 is estimated to infer the features' contribution to the Euclidean distance-based models.

*3.3. Analysis of the FN influence on K-NN*

Next, the methodology employed to validate the FN method selection influence on the  $K$  closest neighbours selection and its implications on the K-NN classification performance is described.

*3.3.1. FN influence on the Neighbours Selection*

The FN impact on the neighbours' selection is analysed by studying the neighbours' distribution.

---

**Algorithm 1** K-NN neighbours similarity (Kendall's  $\tau$ )

---

```

1: Given a dataset  $\tilde{X}^* = \{X, X^{ST}, X^{MM}, X^{MAD}\} \subset \mathbb{R}^{n \times m}$ 
2: for  $i = 1 : n$  do
3:   for  $i' = 1 : n$  do
4:      $D^*(i, i') \leftarrow d_E(\mathbf{x}_i, \mathbf{x}_{i'})$ 
5: for  $D^*, D^+ = \{D, D^{ST}, D^{MM}, D^{MAD}\} \subset \mathbb{R}^{n \times m}, D^* \neq D^+$  do
6:   for  $i = 1 : n$  do
7:      $\tau_i(D^*(i), D^+(i))$ 
8:   Calculate the maximum, mean and minimum values of  $\tau$ 

```

---

310 As described in Algorithm 1, given a matrix  $D^*(i, i') = d_E(\mathbf{x}_i^*, \mathbf{x}_{i'}^*)$  that contains  $\forall i \in \{1, \dots, n\}, i' \in \{1, \dots, n\} \neq i$ , the pairwise Euclidean distances between each pair of samples of the dataset is calculated (line 4). Since K-NN selects the neighbours according to their distance to a given sample, each row  $i$  of  $D^*$  can be interpreted as the ordinal relationship between the dataset's samples respect to the sample  $i$ . Kendall's  $\tau$  correlation coefficient (Lapata, 2006) assesses the relation between two ordinal variables. Then, for each pair of transformations-based distances  $D^*, D^+$  (line 5) and for each sample (line 6), the similarity between neighbours' rank distribution is calculated in line 7. Kendall's  $\tau$  scores between -1 and 1 and  $\tau(x^*, x^+) = 1$  represents the total agreement between the rankings/neighbours obtained by both transformation methods over the same sample. For each pair of transformed datasets (line 7 of Algorithm 1) a total of  $n$  Kendall's  $\tau$  values are obtained. Then, in order to measure the neighbours' similarity, the maximum, mean and minimum  $\tau$  values are analysed.

*3.3.2. FN influence on K-NN performance*

325 In order to study the FN influence on the K-NN performance, the accuracy, precision and recall performance measures are estimated and compared.



### 3.4. Analysis of the FN influence on the K-means Convergence

Next, the methodology to analyse the influence of the dominant features influence (Equation 8) on the K-means convergence is assessed.

---

#### Algorithm 2 K-means convergence

---

```

1: Given  $X^* = \{X, X^{ST}, X^{MM}, X^{MAD}\}$ , the real labels  $Y$  and  $K = |Y|$ 
2: for  $t = 1 : n\_initialisation$  do
3:    $\tilde{Y} \leftarrow \text{K-means}(X^*)$ 
4:   Compute the confusion matrix between  $Y$  and  $\tilde{Y}$ 
5:   if  $TN + TP < FN + FP$  then
6:      $\tilde{Y} \leftarrow \tilde{Y} + 1$ 
7:      $\tilde{Y}[\tilde{Y} == 2] \leftarrow 0$ 
8:   Estimate Precision( $Y, \tilde{Y}$ ).

```

---

330 First, the convergence of K-means with all the features is considered (Al-  
 gorithm 2). For a given number of random initialisations, K-means is applied  
 to each transformed dataset (line 3). By understanding each cluster as a differ-  
 ent class, the results of K-means can be interpreted as the separability between  
 classes. So, for the binary problems, each cluster is encoded in lines 5-7 as 0  
 or 1 in such a way that the accuracy respect to  $Y$  is maximised. Finally, the  
 335 precision obtained at each iteration of the training process is calculated. Note  
 that the purpose of Algorithm 2 is to experimentally validate Equation 8.  
 Then, the effect of removing one-by-one the non-dominant features according  
 to their  $IN$  values is analysed as described in Algorithm 3.

---

#### Algorithm 3 K-means features dominance

---

```

1: Given a dataset  $X^* = \{X, X^{ST}, X^{MM}, X^{MAD}\} \subset \mathbb{R}^{n \times m}$ 
2: for  $n\_del = 1 : m - 1$  do
3:   Remove the  $n\_del$  features with lowest  $IN$  (Equation 13).
4:   Apply Algorithm 2.
5:   Estimate the mean and standard deviation values of the precision reached from the  

  different initialisations.

```

---

## 340 4. Experiments with UCI datasets

This Section experimentally validates over nine well-known datasets from  
 UCI repository (Dua & Graff, 2017) the conclusions presented in Section 2.

### 4.1. UCI datasets description

345 Table 2 describes the main characteristics of the nine well-known datasets  
 from UCI repository employed in this work.

	Breast	Blood	Cryotherapy	Immunotherapy	Parkinson	Accent	Glass	Vehicle	Wine
Features	30	4	6	7	22	12	9	18	13
Samples	569	784	90	90	195	329	214	864	178
Classes	2	2	2	2	2	6	6	4	3
Imbalanced	Yes	Yes	No	Yes	Yes	Yes	Yes	No	No

Table 2: Datasets description in terms of number of features, samples, classes and balance.

The box-plots from Figure 2 illustrate the features' properties of the datasets.  
 The bottom of the lower whisker and top of the upper whisker represent the  
 minimum and maximum values, respectively. Top, medium, and bottom of the

box depict 75, 50 and 25 percentiles, respectively, and the triangular-shaped marker the mean. Further, all the datasets have outliers at least in one feature and, except for Figure 2f, some features<sup>2</sup> present magnitude differences.

#### 4.2. Analysis of the normalised datasets

Figure 3 pictures, for each UCI dataset, the features' Proportional influences  $IN$  derived from each transformation (raw, ST, MM and MAD). Significant differences in the statistical distribution of the Proportional influence values are observed depending on FN. As observed in Figure 3e, feature 13 takes  $IN$  values higher than 0.93 with ST or MM, while  $IN_{MAD} = 0.728$ . Similarly, feature 10 for MM presents a  $IN = 0.908$ , but for ST and MAD the  $IN$  value of the mentioned feature is lower than 0.73. Then, from each FN the features' proportional influence varies, and thus from each normalised dataset different performance results are expected.

Table 6 shows the mean absolute differences between  $IN$  values if comparing two different FN methods. For the studied cases, the  $IN$  differences when comparing ST and MAD transformations are lower than those differences obtained from ST with MM or MAD with MM. These results prove that each FN method transforms a given dataset differently. Then, the selection of the FN method can affect the Euclidean distance-based ML algorithms' performance.

	Breast	Blood	Cryoth.	Immunoth.	Parkinson	Accent	Glass	Vehicle	Wine
$ IN_{ST} - IN_{MAD} $	0.071	0.069	0.137	0.096	0.126	0.046	0.157	0.131	0.035
$ IN_{ST} - IN_{MM} $	0.321	0.133	0.277	0.304	0.188	0.095	0.205	0.345	0.185
$ IN_{MM} - IN_{MAD} $	0.390	0.138	0.414	0.400	0.315	0.131	0.362	0.475	0.220

Table 6: Mean absolute difference between the features Proportional influence ( $IN$ ) derived from different FN methods.

#### 4.3. Normalisation Effect on the K-NN Neighbours selection

K-NN is employed to, first, analyse the FN influence on the  $K$  closest neighbours selection, and then, its impact on the K-NN performance.

##### 4.3.1. Analysis of FN influence on the neighbours' selection

Table 3 collects the results from applying Algorithm 1. The maximum, mean and minimum Kendall's  $\tau$  values result from comparing the neighbours of the raw respect to the normalised datasets (mean Kendall's  $\tau$  values are up to 0.899). Regarding the normalised datasets, the maximum differences between the neighbours' rank are obtained by comparing MAD with MM since mean  $\tau$  values range from 0.566 to 0.900. The second highest rank differences result from comparing ST respect to MM with mean  $\tau$  values between 0.678 and 0.928. Finally, the highest similitude between neighbours' rank results from comparing ST and MAD. These values are consistent with the results from Table 6, where the lowest and highest mean absolute differences between the  $IN$  values are obtained when comparing MAD with ST and MM, respectively.

<sup>2</sup>Further information about the features can be found in (Dua & Graff, 2017).





#### 4.3.2. Analysis of FN influence on the K-NN performance

Table 4 gathers, for each dataset and  $K = [3, 5, 9, 11, 19]$ , the classification performance obtained by K-NN applied to the different transformations.

In most cases, K-NN performance for a fixed value of  $K$  differs depending on the FN method. The maximum differences for the distinct values of  $K$  vary between 2.223% and 6.445% in terms of accuracy. In terms of recall, they range from 3.773% to 10.417%. These differences are even more significant in terms of precision ranging from 5.009% to 21.041%. These results show that the FN method selection can significantly affect the algorithm's performance.

The obtained differences are not only meaningful by themselves. Variations in accuracy of 1% - 4% and even lower than 1% are usually considered notable for UCI datasets. In fact, authors in (Basak & Huber, 2020) employ evolutionary methods to learn feature weights that improve the accuracy of the K-NN algorithm for different UCI datasets. The difference in the resulting accuracy is 0.15% and 3.38% for Breast and Immunotherapy datasets, respectively. In contrast, in this work the accuracy differences derived from the FN method selection for a fixed value of  $K$  are for most cases higher than the reported in (Basak & Huber, 2020). Also, in (Bian et al., 2020), the maximum accuracy differences obtained for Breast, Blood, Glass and Wine datasets by employing eight different FKNN-based algorithms are around 3%, 2%, 7% and 4%, respectively. However, it must be observed that only with the application of different FN methods to the traditional K-NN algorithm, maximum accuracy differences of 1.2%, 1.6%, 5.1% and 1.1% are achieved (Table 4).

Then, this work shows that the optimal FN method selection can be as influencing as the employment of more sophisticated K-NN-based algorithms.

#### 4.4. Normalisation Effect on the K-means Algorithm

In this section the results regarding the features dominance derived from the FN method selection (Equation 8) and their influence on K-means performance is analysed. Tables 5a to 5i collect the results from applying Algorithm 3. Diverse impact of the features' removal on the K-means convergence is observed. Two scenarios can be distinguished: when the features'  $IN$  values are distributed between 0 and 1, or where the minimum  $IN$  values are higher than 0.5. The first scenario, former by the transformations derived from ST and MAD, considers two cases: 1) when the number of features  $m > 10$ , like in Breast, Parkinson and Vehicle the removal of the features with lowest  $IN$  does not significantly affect the mean precision (Tables 5a, 5e and 5h for ST and MAD); 2) for datasets with  $m < 10$ , removing the two features with lowest  $IN$  may alter the mean precision around 1% for Cryotherapy or Immunotherapy with MAD, or up to 5% like in Table 5b for ST. Secondly, when the minimum features' proportional influence is up to 0.5, like in Breast with MM normalisation (Table 5a), smooth changes in mean precision values are observed. In contrast, significant changes –around 5%– result from removing 2 or 3 features from  $\tilde{X}^{ST}$ ,  $\tilde{X}^{MM}$  and  $\tilde{X}^{MAD}$ , such as in Table 5i. This behaviour can be due to the class conditioned distribution of the samples of the removed features, which may be

further analysed in future works. In any case, from the engineering application point of view, when a subset of features does not influence the learning process, these can be removed, saving memory and computational cost.

## 5. Real use case dataset

In this Section, real data obtained from a refinery located in The Basque Country is employed to validate the theoretical analysis conclusions.

### 5.1. Use case description

The data corresponds to a real-life refinery use case. Refineries are complex systems that extract higher quality subproducts from raw crude. Figure 4 presents a high-level diagram of the real-life use case employed in this work. Raw crude is injected into columns C1 and C2 where the distillation process separates Liquefied Petroleum Gas (LPG) from the top of the columns and heavier subproducts from the bottom. The LPG is then pumped into a sweetening unit where sulphur is removed (Chain unit 1). Finally, in Chain unit 2 LPG is further processed to obtain butane. According to the specification standards (BOE-A-2006-2779, 2015), the resultant butane must not exceed a certain threshold of percentage of pentanes (1.5%).

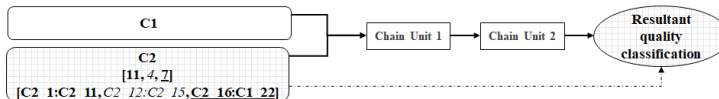


Figure 4: High-level flow diagram of the industrial refining process.

In order to validate the conclusions of Section 2, this work utilises information of the operational process gathered in 22 features regarding the distillation column C2. These features correspond to flow, pressure, and temperature measures (remarked in bold, italic, and underlined text respectively in Figure 4). Based on these 22 operational process features, the goal is to classify the resultant subproduct quality as acceptable or improvable with labels 0 or 1, respectively.

The dataset contains 12.960 samples described by the 22 above-mentioned features. These samples have been collected every 10 minutes during three months of the refining process. Each sample of the dataset has an associated label regarding the resulting quality, from which 18.32% correspond to class 1.

### 5.2. Normalisation Effect on Transforming the Dataset

Figure 5 illustrates the proportional influence of the features that compose the raw dataset  $X$ . Feature C2.4 presents the maximum  $IN$  value in Figure 5. The second feature in terms of proportional influence is C2.11, with  $IN_{C2.11} < 0.4 \cdot IN_{C2.4}$ . Then, if  $X$  is employed for the learning algorithm, C2.4 is expected to be the most influencing feature.

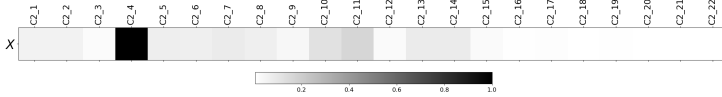
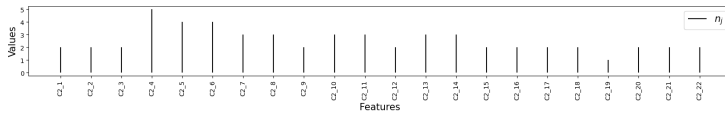


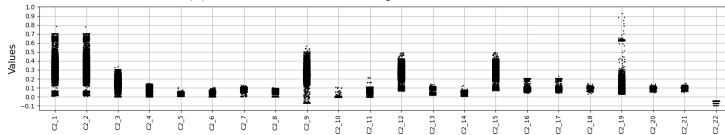
Figure 5: Proportional influence of the features of the raw dataset  $X$ .

Figure 6 depicts the magnitude of the features and the distribution of the values per feature resulting from applying the decimal notation (Equation 2).

As it is expected from Figure 5, C2.4 gathers in Figure 6a the maximum magnitude exponent value  $n_4 = 5$ , followed by C2.5 and C2.6 with  $n_5 = n_6 = 4$ . Then, the maximum magnitude factor is at least ten times higher than the others. However, although in Figure 5 the feature with the second maximum  $IN$  is C2.11,  $n_{11} = 3$ , which is lower than  $n_5$  or  $n_6$ . Hence, despite being the magnitude an important factor, it is not determinant for the features' dominance.



(a) Magnitude exponents  $n_j$  of the raw dataset  $X$



(b) Values gathered by each feature of  $\hat{X}$

Figure 6: Magnitude factors and  $\hat{X}$  obtained after decimal notation application (Equation 2).

Figure 6b presents for each feature the value of the samples of  $X$  after removing the magnitude factor, i.e.  $\hat{X}$ . From Equation 2  $|\hat{x}_{ij}| < 1$ , but depending on the feature the samples are more concentrated in smaller intervals. For instance, the samples of C2.5 in  $\hat{X}$  take values between 0 and 0.1, while C2.1 takes values from 0 to 0.8. Besides, contrary to the rest of the features which are more homogeneously distributed, in C2.22 the samples take only three differentiated values. Hence, despite  $|\hat{x}_{ij}| < 1$ , the features' distribution of the samples also determines the features influence.

In the following, the results obtained from normalising  $X$  are presented. By the application of the *Normalisation weight* (Equation 10), and prior to the dataset normalisation, Table 7 shows a measure about how much each feature will be expanded/compressed respect to an ideal weight  $w^* = 1/m$ .

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
ST	-0.10	-0.10	-0.21	-0.37	0.89	-0.91	-0.53	-0.61	-0.123	-0.40	-0.41	-0.18	-0.78	1.06	-0.18	-0.45	-0.45	1.13	-0.09	1.15	1.15	7.74
MM	-0.19	-0.19	-0.45	1.01	1.50	1.44	1.16	1.47	0.24	1.32	-0.69	0.33	1.22	1.25	-0.35	-0.92	-0.78	1.29	-0.17	1.53	1.52	2.96
MAD	-0.02	-0.02	-0.03	-0.08	-0.18	-0.23	-0.10	-0.13	-0.032	-0.02	-0.07	0.04	-0.28	-0.32	-0.04	-0.16	-0.16	-0.31	-0.03	-0.31	-0.32	17.07

Table 7: Normalisation weight: Percentage of expansion (+) or compression (-) each FN method imposes to each feature of column C2 of the real case study in comparison to  $w^*$ .

The value obtained for each feature in Table 7 differs depending on the FN method. For the features C2.1:C2.3, C2.9, C2.11, C2.15:C2.17 and C2.19, the

weight assigned by the normalisation methods is lower than  $w^*$ . Only features C2.10 and C2.22 present in the three cases normalisation weights higher than  $w^*$ . In the remaining features, either with ST and MM methods, the value obtained by Equation 10 is positive while the obtained for MAD is negative.

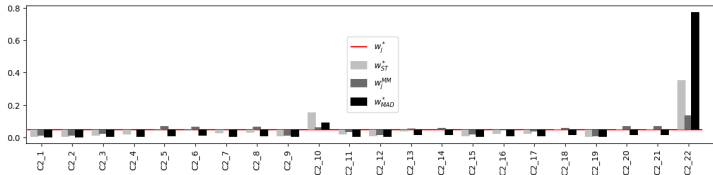


Figure 7: Difference between the ideal weights  $w^*$  and those assigned by each normalisation method  $w^{norm}$ .

In addition to Table 7, Figure 7 depicts a graphical representation of the difference between the ideal weight  $w^*$  and those  $w^{Norm}$  assigned by each FN method. It can be observed that C2.22 is the most expanded feature by the three FN methods. It is remarkable that, for the mentioned feature, MM presents a lower normalisation weight than ST and MAD, and MAD is the most discriminating one. For ST and MAD, the feature with the second-highest normalisation weight is C2.10. In this case, contrary to the observed in C2.22, the normalisation weight assigned for the feature by ST is higher than that given by MAD. Therefore, based on the results of the *Normalisation weight*, it can be known in advance how each FN method would transform each feature of Figure 6b.

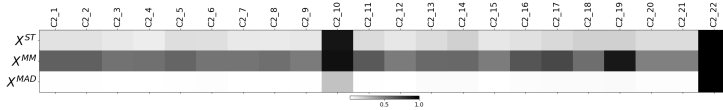


Figure 8: Proportional influence (IN) of the normalised datasets.

Figure 8 depicts the features' Proportional influence of the normalised datasets. Features C2.10 and C2.22 are in Figure 8 the most influencing ones, especially in  $\tilde{X}^{ST}$  and  $\tilde{X}^{MAD}$ . This was expected since these features present in Figure 7, by far, the highest normalisation weights. With MAD normalisation, the rest of features are insignificant in comparison to the most influencing ones. For ST normalisation, the features present a *IN* value between 10% and 50% the proportional influence value of C2.22 or C2.10. In the case of MM, there are no such significant differences between the highest and the lowest contributing features. Besides, C2.19 is also one of the most influencing features, jointly with C2.10 and C2.22. The rationale behind this can be explained due to a less uniform distribution of the samples of C2.10, C2.19 and C2.22 along the features (Figure 6b).

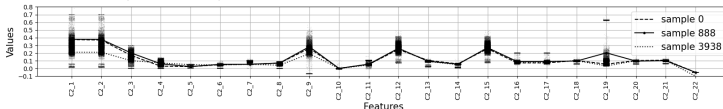


Figure 9: Example of samples values along the features of  $\tilde{X}$ .



510 As commented in Section 2, the FN effect on transforming the data impacts  
on each feature's influence on the algorithm. Nevertheless, the particularities  
of the samples that compose the dataset and their multivariate behaviour  
also affect the metric calculations. Despite the features' associated weight and  
 $|\widehat{x_{ij}}| < 1$ , if the compared pair of samples present the same value for such feature,  
515 the  $w^{Norm}$  losses its effect on the calculations. Figure 9 exemplifies the  
relationship of some samples between the features. For instance, samples 0 and  
888 take the same value in the feature C2.22. They also have similar values for  
the rest of features except for C2.19, where the value of sample 888 is close to  
0.25, while the value for sample 0 is around 0. Consequently, despite the high  
520 weight value assigned to feature C2.22, the distance between samples 0 and 888  
is more determined by the feature C2.19. In contrast, the highest differences  
between samples 888 and 3938 are found in the features C2.1, C2.2, C2.3, C2.9  
and C2.22. However, since the weight assigned to C2.22 is much higher than the  
weights of C2.1, C2.2, C2.3, C2.9, the distance in those components is less in-  
525 fluencing than in C2.22. Consequently, the calculation of the Euclidean distance  
for a pair of samples is affected by both  $IN$  and the particular interrelations  
between them, dependent at the same time on the selected FN method.

### 5.3. Normalisation Effect on the K-NN

530 Next, an analysis of, first, the FN influence on the selection of the  $K$  closest  
neighbours, and then, its implications on the K-NN performance is conducted.

#### 5.3.1. Analysis of FN influence on the neighbours' selection

Table 11 collects the maximum, mean and minimum Kendall's  $\tau$  values ob-  
tained from Algorithm 1. The highest rank disagreement is found when compar-  
ing the raw respect to the normalised datasets. Regarding the normalised ones,  
535 the most significant dissimilarities are observed when contrasting the minimum  
 $\tau$  values (between 0.129 and 0.587) so as for some samples, the FN changes  
drastically the distance rank from a given sample to the remaining ones. Be-  
sides, by comparing by pairs the ranks' similarity, the most significant differences  
are observed when comparing  $\tilde{X}^{MM}$  with  $\tilde{X}^{MAD}$ , followed by those resulting  
540 from the comparison of  $\tilde{X}^{ST}$  and  $\tilde{X}^{MAD}$ .

$\tau$	$X$	$\tilde{X}^{ST}$	$\tilde{X}^{MM}$	$\tilde{X}^{MAD}$	
X	max	1.000	0.593	0.708	0.550
	mean	1.000	0.294	0.368	0.270
	min	1.000	-0.105	-0.040	-0.122
$\tilde{X}^{ST}$	max	0.593	1.000	0.953	0.943
	mean	0.294	1.000	0.874	0.857
	min	-0.105	1.000	0.587	0.358
$\tilde{X}^{MM}$	max	0.708	0.953	1.000	0.921
	mean	0.368	0.874	1.000	0.779
	min	-0.040	0.587	1.000	0.129
$\tilde{X}^{MAD}$	max	0.550	0.943	0.921	1.000
	mean	0.270	0.857	0.779	1.000
	min	-0.122	0.358	0.129	1.000

Table 11: Maximum, mean and minimum  $\tau$  values obtained by comparing the ranks for each pair of dataset transformation.

It should be noted that, a difference in selecting a single neighbour can be  
enough to assign a different class label to a sample. Suppose the case where

K = 35						
X	X <sup>ST</sup>	X <sup>MM</sup>	X <sup>MAD</sup>	X	X <sup>ST</sup>	X <sup>MM</sup>
35	9	10	9	35	9	10
X <sup>ST</sup>	9	35	31	31	31	31
X <sup>MM</sup>	10	31	35	28	28	28
X <sup>MAD</sup>	9	31	28	35	35	35

(d) K = 35

K = 27						
X	X <sup>ST</sup>	X <sup>MM</sup>	X <sup>MAD</sup>	X	X <sup>ST</sup>	X <sup>MM</sup>
27	7	7	6	27	7	7
X <sup>ST</sup>	7	27	24	24	24	24
X <sup>MM</sup>	7	24	27	21	21	21
X <sup>MAD</sup>	6	24	21	27	27	27

(c) K = 27

K = 19						
X	X <sup>ST</sup>	X <sup>MM</sup>	X <sup>MAD</sup>	X	X <sup>ST</sup>	X <sup>MM</sup>
19	5	5	4	19	5	5
X <sup>ST</sup>	5	19	16	17	17	17
X <sup>MM</sup>	5	16	19	15	15	15
X <sup>MAD</sup>	4	17	15	19	19	19

(b) K = 19

K = 11						
X	X <sup>ST</sup>	X <sup>MM</sup>	X <sup>MAD</sup>	X	X <sup>ST</sup>	X <sup>MM</sup>
11	3	3	3	11	3	3
X <sup>ST</sup>	3	11	9	10	10	10
X <sup>MM</sup>	3	9	11	8	8	8
X <sup>MAD</sup>	3	10	8	11	11	11

(a) K = 11

Table 8: Mean number of equal neighbours for each pair of transformed datasets and different values of K.

K=35						
X	X <sup>ST</sup>	X <sup>MM</sup>	X <sup>MAD</sup>	X	X <sup>ST</sup>	X <sup>MM</sup>
100	34.418	33.876	33.906	100	34.418	33.876
X <sup>ST</sup>	78.459	100	<b>94.57</b>	91.355	91.355	91.355
X <sup>MM</sup>	79.173	<b>91.552</b>	100	<b>86.143</b>	86.143	86.143
X <sup>MAD</sup>	78.887	<b>93.242</b>	<b>90.821</b>	100	100	100

(d) K = 35

K=27						
X	X <sup>ST</sup>	X <sup>MM</sup>	X <sup>MAD</sup>	X	X <sup>ST</sup>	X <sup>MM</sup>
100	38.421	38.842	37.4	100	38.421	38.842
X <sup>ST</sup>	80.197	100	95.356	<b>91.695</b>	91.695	91.695
X <sup>MM</sup>	79.213	<b>93.164</b>	100	<b>87.457</b>	87.457	87.457
X <sup>MAD</sup>	80.32	<b>94.343</b>	<b>92.099</b>	100	100	100

(c) K = 27

K=19						
X	X <sup>ST</sup>	X <sup>MM</sup>	X <sup>MAD</sup>	X	X <sup>ST</sup>	X <sup>MM</sup>
100	41.839	41.987	41.474	100	41.839	41.987
X <sup>ST</sup>	79.607	100	<b>94.402</b>	<b>93.384</b>	93.384	93.384
X <sup>MM</sup>	79.193	<b>93.558</b>	100	<b>89.188</b>	89.188	89.188
X <sup>MAD</sup>	79.814	<b>94.459</b>	<b>90.999</b>	100	100	100

(b) K = 19

K=11						
X	X <sup>ST</sup>	X <sup>MM</sup>	X <sup>MAD</sup>	X	X <sup>ST</sup>	X <sup>MM</sup>
100	50.539	51.365	49.377	100	50.539	51.365
X <sup>ST</sup>	81.373	100	95.479	<b>94.008</b>	94.008	94.008
X <sup>MM</sup>	81.924	<b>94.216</b>	100	<b>90.316</b>	90.316	90.316
X <sup>MAD</sup>	81.294	96.127	<b>93.592</b>	100	100	100

(a) K = 11

Table 9: Percentage of samples equally classified as class 1 by each pair of normalisation methods for different values of K. Emphasised with bold the cases where the percentage is lower than 95%.

K=35						
Accuracy	X	X <sup>ST</sup>	X <sup>MM</sup>	X <sup>MAD</sup>	Accuracy	X
85.270	90.247	90.208	<b>90.486</b>	90.486	85.270	90.247
Precision	83.107	84.731	<b>85.714</b>	84.979	83.107	84.731
Recall	24.558	37.035	55.855	<b>58.382</b>	24.558	37.035

(d) K = 35

K=27						
Accuracy	X	X <sup>ST</sup>	X <sup>MM</sup>	X <sup>MAD</sup>	Accuracy	X
86.024	91.289	91.111	<b>91.636</b>	91.636	86.024	91.289
Precision	84.623	86.682	<b>86.852</b>	86.942	84.623	86.682
Recall	28.981	49.963	60.657	<b>63.943</b>	28.981	49.963

(c) K = 27

K=19						
Accuracy	X	X <sup>ST</sup>	X <sup>MM</sup>	X <sup>MAD</sup>	Accuracy	X
86.698	92.747	92.238	<b>92.986</b>	92.986	86.698	92.747
Precision	83.644	89.018	7.541	<b>89.403</b>	83.644	89.018
Recall	34.033	68.913	67.186	<b>70.008</b>	34.033	68.913

(b) K = 19

K=11						
Accuracy	X	X <sup>ST</sup>	X <sup>MM</sup>	X <sup>MAD</sup>	Accuracy	X
88.341	94.738	94.313	<b>95.015</b>	95.015	88.341	94.738
Precision	84.057	91.471	90.661	<b>91.419</b>	84.057	91.471
Recall	44.861	78.602	76.874	<b>80.329</b>	44.861	78.602

(a) K = 11

Table 10: Performance of K-NN for each transformed dataset and different values of K. For each K, bold text highlights the highest performance values.

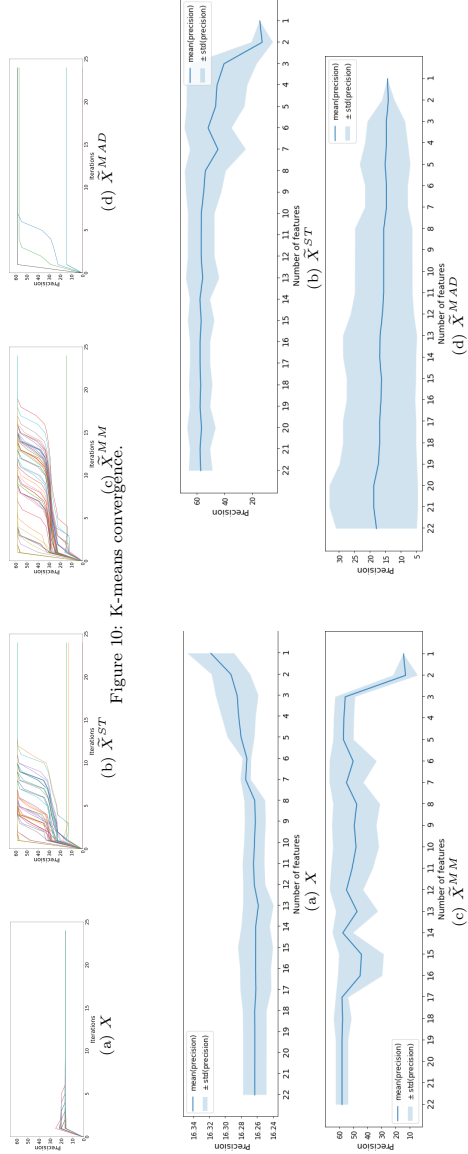


Figure 10: K-means convergence.

Figure 11: K-means results when applying Feature Selection based on the values of proportional influence (Algorithm 3).

$(K - 1)/2$  neighbours of the sample belong to class 0, and the rest to class 1. If the non-shared neighbours for each transformed dataset belong to different classes, in one case the sample is labelled as 0, while in the other method the sample's label is set as 1. In this line, Tables 8a to 8d present for different values of  $K$  the mean number of equal neighbours employed by the same sample by different FN methods. For all cases, the lowest mean values are obtained with the raw dataset. Regarding the normalised datasets, the lowest coincidence is observed by comparing  $\tilde{X}^{MM}$  and  $\tilde{X}^{MAD}$  for  $K = \{11, 19, 27, 35\}$ , followed by the pair  $\tilde{X}^{ST}$  and  $\tilde{X}^{MM}$  in Tables 8a and 8b. In Tables 8c and 8d the mean number value is the same for  $\tilde{X}^{ST}$  respect to both  $\tilde{X}^{MM}$  and  $\tilde{X}^{MAD}$ .

### 5.3.2. Analysis of FN influence on the K-NN performance

The neighbours' distribution differences derived from the FN method selection directly impacts on the samples' classification. Tables 9a to 9d present the proportion of samples labelled by K-NN as class 1 for  $j$ -th dataset that are equally labelled as class 1 when utilising  $i$ -th dataset, i.e.  $S_{ij}/|\{\hat{y}_i = 1\}|$  being  $S_{ij} = \{\hat{y}_j = 1|\hat{y}_i = 1\}$  for a given  $K$  and  $i, j \in \{X, \tilde{X}^{ST}, \tilde{X}^{MM}, \tilde{X}^{MAD}\}$ . It can be observed that up to 13.857% of samples labelled as 1 when utilising one FN method are classified as 0 when using a different one, with the associated economic and energetic consequences of the decisions on the refinery operation depending on these estimations. Actually, for  $K = 35$  only the 86.143% of the samples labelled as 1 with MAD are classified as class 1 when employing MM. Also, with  $K = 11$  between 4% and 10% of the samples labelled as class 1 by one method are labelled as 0 by other FN method.

The samples classification differences are reflected in the algorithm's performance measures results. Tables 10a to 10d collect the accuracy, precision and recall obtained by K-NN algorithm for  $X, \tilde{X}^{ST}, \tilde{X}^{MM}$  and  $\tilde{X}^{MAD}$ . In this particular case, the performance results obtained over  $X$  are improved by applying any of the selected FN methods. Except for  $K = 11$  with ST or  $K = 35$  with MM in terms of precision, MAD attains the best performance. As described in Section 5.1 the dataset is imbalanced so, the accuracy is not the most representative performance measure in this particular case.

$K$	ST-MM	MM-MAD	ST-MAD	$K$	ST-MM	MM-MAD	ST-MAD
11	0.810	0.758	0.052	11	1.728	3.455	1.727
19	1.469	1.862	0.393	19	1.727	2.822	1.095
27	0.170	0.090	0.260	27	1.306	3.286	1.980
35	0.983	0.735	0.248	35	1.180	2.527	1.347

(a) Precision

(b) Recall

Table 12: Differences between the performance reached by each FN method.

Table 12 presents the absolute differences in precision and recall, respectively, when comparing the results of the normalised datasets for different values of  $K$ . These differences are up to 1.862% and 3.455% in terms of precision and recall, respectively. Thus, even with the same mean number of equal neighbours for  $\tilde{X}^{ST}$  respect to  $\tilde{X}^{MM}$  or  $\tilde{X}^{MAD}$  (Tables 8c and 8d), the algorithm's performance can vary non trivially due to the particular influence of one single neighbour, with differences in precision up to 0.983%, and up to 1.980% in terms of recall.

Hence, for all the above mentioned, it is experimentally confirmed that FN influences the Euclidean distance calculation, which affects the neighbours' se-

lection and, consequently, the K-NN algorithm's performance. Also, it must be noticed that the reported impact of FN method selection will depend on the particularities of the engineering application.

#### 5.4. Normalisation Effect on the K-means Algorithm

In the following, the influence on the K-means convergence of the features' dominance derived from the FN method selection is analysed.

Figure 10 depicts the precision results obtained by applying Algorithm 2. Each colour of Figure 10 represents each of the 100 random initialisations. K-means algorithm for the analysed problem converges to a solution in less than 25 iterations. However, depending on the features' proportional influence distribution, the convergence is reached in less number of iterations.

With the raw dataset, highly dominated by the feature C2.4 in terms of proportional influence, the algorithm converges for every random initialisation in less than 10 iterations (Figure 10a). Similarly, K-means converges in less than 10 iterations with  $\tilde{X}^{MAD}$ , which in Figure 8 presented 2 dominant features, C2.22 and C2.10. In the case of  $\tilde{X}^{ST}$ , C2.22 and C2.10 are the dominant features (Figure 8), but the others do not present, in comparison to the maximum one, as low influences as it occurs for  $\tilde{X}^{MAD}$ . Then, in this case, K-means employs up to 15 iterations depending on the initialisation. In contrast, K-means applied to  $\tilde{X}^{MM}$  needs more than 15 iterations to converge. This is due to absence of significantly dominant features in  $\tilde{X}^{MM}$ .

Therefore, it can be concluded that the lower the number of dominant features and the higher the differences in the influence between the most influencing features and the rest of the features, the lower the number of iterations K-means needs to converge to a solution. Then, the features' *IN* can be considered as a strategy for Feature Selection.

Figures 11a to 11d depict the mean precision and standard deviation obtained from the results of 100 random initialisations of the K-means after applying FS based on the features' *IN* (Algorithm 3). The X-axis represents the number of features employed for training the K-means. As depicted in Figure 8,  $\tilde{X}^{MAD}$  presents the highest differences between the features' *IN* values. Then, in Figure 11d, the removal of features with lowest influence does not affect the mean precision reached by the K-means algorithm. Even by selecting only the most influencing feature, C2.22, the mean precision remains around 0.15. Similarly,  $X$  is mainly dominated by feature C2.4, but in contrast to  $\tilde{X}^{MAD}$ , in Figure 5 other features present a *IN* value around 0.3. Then, by removing the features with lowest influence value, in Figure 11a the mean precision and the std values almost do not change. In fact, in this particular case the employment of the two most influencing features only changes 0.001 the original precision value. In the case of  $\tilde{X}^{ST}$  dataset, most of the features present a *IN* between 40 – 50%. By selecting the eight most influencing features, in Figure 11b the mean precision only decreases from 0.6 to 0.5. It can also be observed that by using the two most influencing features of  $\tilde{X}^{ST}$  dataset, the precision obtained is lower than 0.2. In contrast to Figures 11a, 11b and 11d, Figure 11c

presents significant fluctuations in the mean precision value after removing the six features with lowest proportional influence. The reason is that the features of  $\tilde{X}^{MM}$  with lowest  $IN$  present a value higher than 0.4 (Figure 8).

The results depicted in Figures 11a to 11d show that the *Proportional influence* enables to capture the influence each feature will have on the distance-based ML algorithm calculations. However, such influence derived from the FN methods is not related to the true importance of the feature for estimating the real output of the problem. In this context, the proposed approach is focused on understanding the features dominance on the Euclidean distance calculation (Equation 8). Furthermore, by inferring each feature's influence on the algorithm training, memory and run time can be saved by removing non-contributing features that do not affect the clusters configuration.

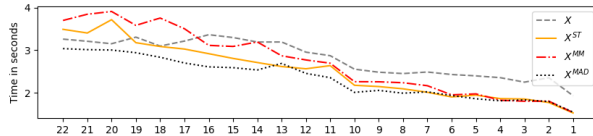


Figure 12: Time in seconds employed by K-means to do 100 initialisation for raw, ST, MM and MAD datasets, in a 16 GB RAM Dell Latitude 5580 workstation equipped with Intel Core i7-7600U CPU running at Microsoft Windows 10 Enterprise.

Figure 12 depicts the time (in seconds) employed for K-means algorithm to train the model based on the Feature Selection conducted according to the features influences of Figures 5 and 8. For the FS conducted over the raw and the normalised datasets, it is observed that the time consumed presents a decreasing tendency. Therefore, in addition to the memory-related advantages derived from the features downsizing, the computational cost decreases as the number of employed features is reduced.

## 6. Roadmap for the FN Method Selection

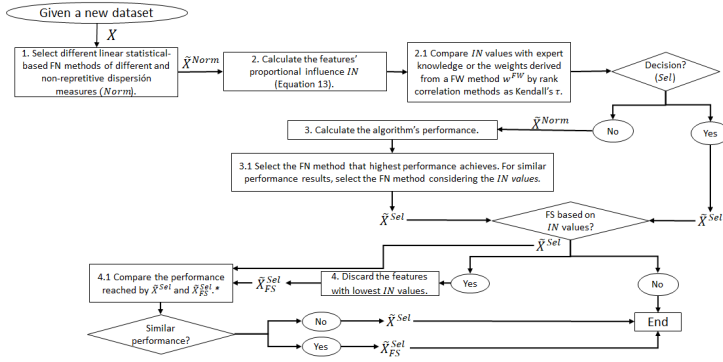


Figure 13: Roadmap for the proper FN method selection.

In Figure 13, a roadmap for the proper FN method selection is suggested. Next, two examples are presented to exemplify the application of such roadmap over a new given use case.

**Example 6.1.** *In the following, an example of the application of the presented roadmap over the IoT Botnet Attacks dataset from UCI repository is shown. It contains data about ack flooding Mirari attack (Bagui et al., 2021) consisting in 151743 samples characterised by 115 features.*

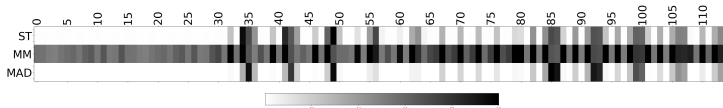


Figure 14: Proportional influence  $IN$  values.



Figure 15: Features' weight  $w^{MI}$  estimated with Mutual Information (MI)

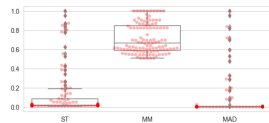


Figure 16: Distribution of the features'  $IN$  values derived from each FN method.

	ST	MM	MAD
Accuracy	99.996	100.	99.988
Precision	99.999	100.	99.999
Recall	99.995	100.	99.983

Table 13: Performance obtained by K-NN with  $K = 7$  for the normalised datasets.

In **step 1**,  $ST$ ,  $MM$  and  $MAD$  FN methods are selected. Figure 14 depicts the  $IN$  values calculated in **step 2**. In the absence of expert knowledge, in **step 2.1** the features' relevance is estimated with Mutual Information (MI) supervised FW method. The calculated weights  $w^{MI}$  are plotted in Figure 15. As it can be observed, there is no similitude between the  $IN$  values and the MI-based weights rankings. In fact, Kendall's  $\tau$  correlation coefficients between  $w^{MI}$  and  $IN(\tilde{X}^{Norm})$  for  $Norm \in \{ST, MM, MAD\}$  are  $-0.544$ ,  $-0.455$  and  $-0.567$ , respectively. Then, no FN method selection can be chosen. Since no decision about the proper FN method can be made based on the results of **step 2.1**, the algorithm's performance for each normalised dataset is computed. Table 13 collects the performance measures resulting of applying K-NN with  $K = 7$  in **step 3**. With the FN methods selected in **step 1**, K-NN obtains performance values higher than 99.9%. However, as Figure 16 depicts, in  $\tilde{X}^{ST}$  and  $\tilde{X}^{MAD}$  most of the features present  $IN$  values lower than 0.6. Then, in these two cases, high performance values are achieved with datasets dominated by few number of features. Consequently,  $\tilde{X}^{ST}$  or  $\tilde{X}^{MAD}$  are preselected in **step 3.1** to analyse if FS based on the  $IN$  values can be conducted to reduce computational time and memory without performance loss. In **step 4** the features with  $IN \geq 0.6$  are selected to train the models. Thus, 10 features remain in  $\tilde{X}_{FS}^{ST}$  after FS, which correspond to the 8.7% the features of the dataset. For  $\tilde{X}_{FS}^{MAD}$ , only 6 features present  $IN \geq 0.6$ . In **step 4.1**, K-NN for  $\tilde{X}_{FS}^{ST}$  reaches accuracy, precision, and recall values equal to 95.022%, 93.125% and 99.99%, respectively. In contrast,

for  $\tilde{X}^{MAD}$  an accuracy of 35.625% and precision and recall values equal to 0% are obtained.

680 Then, based on the performance results and the computation time and memory saving derived from applying according to **step 4** FS based on IN values,  $\tilde{X}_{FS}^{ST}$  is selected for this use case.

**Example 6.2.** Next, Air Pressure System (APS) Failure at Scania Trucks dataset from UCI repository is employed. It is composed by 591 samples described by 165 features. The goal is to identify if the registered failures are related to components of the APS. For doing so, in this example K-means is used to clustering the samples. In this case, ST, MM and Pareto scaling (PA) are selected in **step 1**. Figure 17 shows the features' IN values estimated in **step 2**.

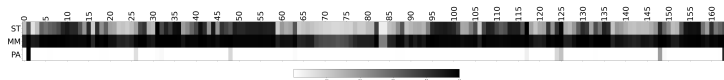


Figure 17: Features' proportional influence

690 Let's assume that expert knowledge considers, in the following order, features 67, 20, 21, 25, 137, 65 and 64 as the most important ones for modelling the problem. In this case, the main interest is to check which FN method transforms the features in such a way that the mentioned features present higher influence on the Euclidean distance-based algorithm. For doing so, instead of analysing the proportional influence of all the features, in **step 2.1**, Kendall's  $\tau$  correlation between the rank given by expert knowledge and the seven mentioned features' IN values ranking is computed. The obtained correlation equals 0.619, 0.810 and  $-0.048$  for ST, MM and PA, respectively. Then, since the dataset derived from MM FN method obtains the highest correlation value, MM is selected in **step**  
700 **2.1**. In Figure 17, the lowest IN value is 0.503. Thus, due to the high features' proportional influence on the algorithm's calculation no FS is conducted.

## 7. Conclusions

Data preprocessing is a fundamental and time-consuming part of data analysis in multiple engineering applications. Besides, the reliability of the models is dependent on the quality of the employed data. In the Industry 4.0 context, where hundreds of features are collected, the preprocessing stage is an arduous and challenging task. Then, the higher the understating of the preprocessing approaches' implications, the higher the options of automating such processes. Precisely, industrial use cases are characterised by features representing distinct physical properties captured from different process stages. Therefore, it is common to find variables of different magnitude, and consequently FN is usually applied. In this sense, this paper proposes a theoretical-experimental analysis of the normalisation effect on transforming the original dataset and its implications on the Euclidean distance-based algorithms. Regarding the theoretical analysis, it has been proven that statistical normalisation methods based on dispersion statistics do not equalise the contribution of the features. In fact, it is  
715

shown that each FN method transforms a dataset diversely and that a given FN method converts each feature differently. Consequently, Feature Normalisation can be interpreted as a particular case of Feature Weighting. In this sense, a new proposed metric referred as *Normalisation weight* for measuring the dispersion factor weight gives guidance about the impact of the normalisation method on the features. The features conversion directly influences those algorithms that employ the joint information of the features to estimate the output. This work proves the influence of the FN method selection on the Euclidean distance-based ML algorithm calculations. In order to measure such influence, a new metric referred as *Proportional influence* has been proposed.

In particular, K-NN and K-means algorithms are employed to demonstrate the normalisation's influence over the Euclidean distance calculations. K-NN is utilised to reflect the normalisation influence on the neighbours' selection and, hence, on the classification performance. With regards to K-means algorithm, the features dominance impact over the convergence of the algorithm is also demonstrated. Thus, normalisation influences not only the performance but also the computational cost, which can be considered when handling large datasets or applying the model in an online adaptive scenario. In this sense, a methodology is presented to study the neighbours' distribution similarity between pairs of transformed datasets for the K-NN algorithm and, for K-means algorithm, a guideline for reducing memory and computational cost depending on the features influence on the metric is presented.

The theoretical conclusions are experimentally validated over different well-known UCI datasets and real-life refining industrial case. Finally, this work proposes a roadmap to guide the FN method selection for a given new problem.

In order to complement the results obtained in this research, future work in this area for automatic preprocessing could include the correlation analysis between the features for enhancing the calculations of the features influence. The suggested analysis can bring light to the impact of removing the features on the algorithm performance. Besides, the employment of Feature Weighting methods to analyse each feature's relative importance for estimating the output could be employed to select among the normalisation methods the most suitable one for the problem at hand. In addition, the study about the features' preprocessing impact on the model's performance will be extended to further ML algorithms.

## References

- de Amorim, R. C. (2016). A survey on feature weighting based k-means algorithms. *Journal of Classification*, 33, 210–242.
- 755 Bagui, S., Wang, X., & Bagui, S. (2021). Machine learning based intrusion detection for iot botnet. *International Journal of Machine Learning and Computing*, 11.



- Basak, S., & Huber, M. (2020). Evolutionary feature scaling in k-nearest neighbors based on label dispersion minimization. In *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)* (pp. 928–935). IEEE.
- van den Berg, R. A., Hoefsloot, H. C., Westerhuis, J. A., Smilde, A. K., & van der Werf, M. J. (2006). Centering, scaling, and transformations: improving the biological information content of metabolomics data. *BMC genomics*, *7*, 142.
- Bergen, K. J., & Beroza, G. C. (2019). Earthquake fingerprints: Extracting waveform features for similarity-based earthquake detection. *Pure and Applied Geophysics*, *176*, 1037–1059.
- Bhanja, S., & Das, A. (2018). Impact of data normalization on deep neural network for time series forecasting. *arXiv preprint arXiv:1812.05519*, .
- Bian, Z., Vong, C. M., Wong, P. K., & Wang, S. (2020). Fuzzy knn method with adaptive nearest neighbors. *IEEE transactions on cybernetics*, .
- BOE-A-2006-2779 (2015). Real decreto 61/2006, de 31 de enero, por el que se determinan las especificaciones de gasolinas, gasóleos, fuelóleos y gases licuados del petróleo y se regula el uso de determinados biocarburantes. Spain, Ministerio de Industria, Turismo y Comercio. <https://www.boe.es/buscar/act.php?id=BOE-A-2006-2779&tn=1&p=20060928>.
- Borghesan, F., Chioua, M., & Thornhill, N. F. (2019). Forecasting of process disturbances using k-nearest neighbours, with an application in process control. *Computers & Chemical Engineering*, *128*, 188–200. doi:<https://doi.org/10.1016/j.compchemeng.2019.05.009>.
- Cavalcante, C. C., Maschio, C., Santos, A. A. S., Schiozer, D., & Rocha, A. (2019). A continuous learning algorithm for history matching. *Engineering Applications of Artificial Intelligence*, *85*, 543–568. doi:<https://doi.org/10.1016/j.engappai.2019.07.012>.
- Chandrashekar, G., & Sahin, F. (2014). A survey on feature selection methods. *Computers & Electrical Engineering*, *40*, 16–28.
- Chelmiah, E. T., McLoone, V. I., & Kavanagh, D. F. (2020). Remaining useful life estimation of rotating machines using octave spectral features. In *IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society* (pp. 3031–3036). IEEE.
- Chu, C.-W., Holliday, J. D., & Willett, P. (2009). Effect of data standardization on chemical clustering and similarity searching. *Journal of chemical information and modeling*, *49*, 155–161.
- Chu, Z., Yu, J., & Hamdulla, A. (2020). Lpg-model: A novel model for throughput prediction in stream processing, using a light gradient boosting machine, incremental principal component analysis, and deep gated recurrent unit network. *Information Sciences*, .

- 800 Cortés-Ibáñez, J. A., González, S., Valle-Alonso, J. J., Luengo, J., García, S., & Herrera, F. (2020). Preprocessing methodology for time series: An industrial world application case study. *Information Sciences*, *514*, 385–401.
- Cunningham, P., & Delany, S. J. (2020). k-nearest neighbour classifiers-. *arXiv preprint arXiv:2004.04523*, .
- 805 Curreri, F., Graziani, S., & Xibilia, M. G. (2020). Input selection methods for data-driven soft sensors design: application to an industrial process. *Information Sciences*, .
- Deng, Z., Choi, K.-S., Jiang, Y., Wang, J., & Wang, S. (2016). A survey on soft subspace clustering. *Information sciences*, *348*, 84–106.
- Di Mauro, M., Galatro, G., Fortino, G., & Liotta, A. (2021). Supervised feature selection techniques in network intrusion detection: A critical review. *Engineering Applications of Artificial Intelligence*, *101*, 104216.
- 810 Dua, D., & Graff, C. (2017). UCI machine learning repository. URL: <http://archive.ics.uci.edu/ml>.
- Famili, A., Shen, W.-M., Weber, R., & Simoudis, E. (1997). Data preprocessing and intelligent data analysis. *Intelligent data analysis*, *1*, 3–23.
- 815 García, S., Luengo, J., & Herrera, F. (2015). *Data preprocessing in data mining*. Springer.
- Gutierrez-Torre, A., Berral, J. L., Buchaca, D., Guevara, M., Soret, A., & Carrera, D. (2020). Improving maritime traffic emission estimations on missing data with crbms. *Engineering Applications of Artificial Intelligence*, *94*, 103793. doi:<https://doi.org/10.1016/j.engappai.2020.103793>.
- 820 Hassani, H., Hallaji, E., Razavi-Far, R., & Saif, M. (2021). Unsupervised concrete feature selection based on mutual information for diagnosing faults and cyber-attacks in power systems. *Engineering Applications of Artificial Intelligence*, *100*, 104150.
- 825 Hsu, F.-Y., Lee, H.-M., Chang, T.-H., & Sung, Y.-T. (2018). Automated estimation of item difficulty for multiple-choice tests: An application of word embedding techniques. *Information Processing & Management*, *54*, 969–984.
- Hurley, N. C., Haimovich, A. D., Taylor, R. A., & Mortazavi, B. J. (2019). Visualization of emergency department clinical data for interpretable patient phenotyping. *arXiv preprint arXiv:1907.11039*, .
- 830 Jain, A. K. (2010). Data clustering: 50 years beyond k-means. *Pattern recognition letters*, *31*, 651–666.
- Julian, K. D., Kochenderfer, M. J., & Owen, M. P. (2019). Deep neural network compression for aircraft collision avoidance systems. *Journal of Guidance, Control, and Dynamics*, *42*, 598–608.
- 835

- Kundu, S., & Ari, S. (2017). Score normalization of ensemble svms for brain-computer interface p300 speller. In *2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)* (pp. 1–5). IEEE.
- 840 Lapata, M. (2006). Automatic evaluation of information ordering: Kendall's tau. *Computational Linguistics*, *32*, 471–484.
- Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., & Liu, H. (2017). Feature selection: A data perspective. *ACM Computing Surveys (CSUR)*, *50*, 1–45.
- 845 Milligan, G. W., & Cooper, M. C. (1988). A study of standardization of variables in cluster analysis. *Journal of classification*, *5*, 181–204.
- Noda, I. (2008). Scaling techniques to enhance two-dimensional correlation spectra. *Journal of Molecular Structure*, *883*, 216–227.
- 850 Park, G., Baek, Y., & Lee, H.-K. (2005). Re-ranking algorithm using post-retrieval clustering for content-based image retrieval. *Information processing & management*, *41*, 177–194.
- Peng, Z., Xiao, X., Hu, G., Sangaiyah, A. K., Atiquzzaman, M., & Xia, S. (2020). Abff: An autoencoder based practical approach for software fault localization. *Information Sciences*, *510*, 108–121.
- 855 Pławiak, P., Abdar, M., Pławiak, J., Makarenkov, V., & Acharya, U. R. (2020). Dghnl: A new deep genetic hierarchical network of learners for prediction of credit scoring. *Information Sciences*, *516*, 401–418.
- 860 Qiu, X., Zhang, L., Suganthan, P. N., & Amaratunga, G. A. (2017). Oblique random forest ensemble via least square estimation for time series forecasting. *Information Sciences*, *420*, 249–262.
- Rostami, M., Berahmand, K., Nasiri, E., & Forouzande, S. (2021). Review of swarm intelligence-based feature selection methods. *Engineering Applications of Artificial Intelligence*, *100*, 104210.
- 865 Schaffer, C. M., & Green, P. E. (1996). An empirical comparison of variable standardization methods in cluster analysis. *Multivariate Behavioral Research*, *31*, 149–167.
- 870 Semenov, V., Sukhoparov, M., & Lebedev, I. (2020). Approach to the state analysis of industry 4.0 nodes based on behavioral patterns. In A. Ronzhin, G. Rigoll, & R. Meshcheryakov (Eds.), *Interactive Collaborative Robotics* (pp. 273–282). Cham: Springer International Publishing.
- Singh, D., & Singh, B. (2019). Investigating the impact of data normalization on classification performance. *Applied Soft Computing*, (p. 105524).

- 875 Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information processing & management*, *45*, 427–437.
- Thara, D., PremaSudha, B., & Xiong, F. (2019). Auto-detection of epileptic seizure events using deep neural network with different feature scaling techniques. *Pattern Recognition Letters*, *128*, 544–550.
- 880 Trim, P. J., Atkinson, S. J., Princivalle, A. P., Marshall, P. S., West, A., & Clench, M. R. (2008). Matrix-assisted laser desorption/ionisation mass spectrometry imaging of lipids in rat brain tissue with integrated unsupervised and supervised multivariate statistical analysis. *Rapid Communications in Mass Spectrometry: An International Journal Devoted to the Rapid Dissemination of Up-to-the-Minute Research in Mass Spectrometry*, *22*, 1503–1509.
- 885 Vaitheeshwari, R., & SathieshKumar, V. (2019). Performance analysis of epileptic seizure detection system using neural network approach. In *2019 International Conference on Computational Intelligence in Data Science (ICCIDS)* (pp. 1–5). IEEE.
- 890 Vanini, Z. S., Khorasani, K., & Meskin, N. (2014). Fault detection and isolation of a dual spool gas turbine engine using dynamic neural networks and multiple model approach. *Information Sciences*, *259*, 234–251.
- Walach, J., Filzmoser, P., & Hron, K. (2018). Data normalization and scaling: Consequences for the analysis in omics sciences. In *Comprehensive Analytical Chemistry* (pp. 165–196). Elsevier volume 82.
- 895 Wetschereck, D., Aha, D. W., & Mohri, T. (1997). A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. *Artificial Intelligence Review*, *11*, 273–314.
- Yao, L., & Ge, Z. (2019). Distributed parallel deep learning of hierarchical extreme learning machine for multimode quality prediction with big process data. *Engineering Applications of Artificial Intelligence*, *81*, 450–465. doi:<https://doi.org/10.1016/j.engappai.2019.03.011>.
- 900 Zhu, L., Sun, A., & Choi, B. (2011). Detecting spam blogs from blog search results. *Information processing & management*, *47*, 246–262.

### **III Normalisation Influence on ANN-Based Models Performance: A New Proposal for Features' Contribution Analysis**

**Iratxe Niño-Adan, Eva Portillo, Itziar Landa-Torres, Diana Manjarres**

Published in *IEEE Access*, June 2021, volume 9, pp. 125462–125477.  
DOI: <https://doi.org/10.1109/ACCESS.2021.3110647>.

**JCR: 3.367**

*Category: Computer Science, Information Systems, 65/161, Q2*





# Normalization Influence on ANN-Based Models Performance: A New Proposal for Features' Contribution Analysis

IRATXE NIÑO-ADAN<sup>1,2</sup>, EVA PORTILLO<sup>2</sup>, ITZIAR LANDA-TORRES<sup>3</sup>, AND DIANA MANJARRES<sup>1</sup>

<sup>1</sup>Tecnalia Research and Innovation, Basque Research and Technology Alliance (BRTA), 48160 Derio, Spain

<sup>2</sup>Department of Automatic Control and Systems Engineering, Faculty of Engineering, University of the Basque Country (UPV/EHU), 48013 Bilbao, Spain

<sup>3</sup>Petronor Innovación S.L., 48550 Muskiz, Spain

Corresponding author: Iratxe Niño-Adan (iratxe.nino@tecnalia.com)

This work was supported in part by DATA Inc. Fellowship under Grant 48-AF-W1-2019-00002, in part by Tecnalia Research and Innovation Ph.D. Scholarship, in part by the Spanish Centro para el Desarrollo Tecnológico Industrial (CDTI, Ministry of Science and Innovation) through the “Red Cervera” Programme (AI4ES Project) under Grant CER-20191029, and in part by the 3KIA Project funded by the ELKARTEK Program of the SPRI-Basque Government under Grant KK-2020/00049.

**ABSTRACT** Artificial Neural Networks (ANNs) are weighted directed graphs of interconnected neurons widely employed to model complex problems. However, the selection of the optimal ANN architecture and its training parameters is not enough to obtain reliable models. The data preprocessing stage is fundamental to improve the model’s performance. Specifically, Feature Normalisation (FN) is commonly utilised to remove the features’ magnitude aiming at equalising the features’ contribution to the model training. Nevertheless, this work demonstrates that the FN method selection affects the model performance. Also, it is well-known that ANNs are commonly considered a “black box” due to their lack of interpretability. In this sense, several works aim to analyse the features’ contribution to the network for estimating the output. However, these methods, specifically those based on network’s weights, like Garson’s or Yoon’s methods, do not consider preprocessing factors, such as *dispersion factors*, previously employed to transform the input data. This work proposes a new features’ relevance analysis method that includes the dispersion factors into the weight matrix analysis methods to infer each feature’s actual contribution to the network output more precisely. Besides, in this work, the *Proportional Dispersion Weights (PWD)* are proposed as explanatory factors of similarity between models’ performance results. The conclusions from this work improve the understanding of the features’ contribution to the model that enhances the feature selection strategy, which is fundamental for reliably modelling a given problem.

**INDEX TERMS** Artificial neural networks, explainability, feature contribution, feature normalization.

## I. INTRODUCTION

Artificial Neural Networks (ANNs) are algorithms that simulate the human brain learning behaviour, modelled by a weighted directed graph of interconnected nodes or neurons. These neurons are simple functions whose arguments are the weighted summation of the inputs to the node [1]. Due to their ability to solve challenging computational problems [2], [3], ANNs are widely applied in different fields, like industry among others [4]–[8]. However, they are still considered a “black box” since the network’s predictions cannot be directly explained. Therefore, in the last decades, there has

been a surge of interest in explainable Artificial Intelligence (xAI) approaches [9]. In this line, researchers have shown an increased claim in understating the features’ contribution for modelling the network [10]–[12]. As authors in [13] expound, the goal of feature relevance explanation techniques is to describe the functioning of a model by measuring each feature’s influence on the predicted output. Since feature relevance methods can be viewed as indirect techniques to explain a model, they have become a vibrant subject of study in the xAI field [14]–[18].

The understanding of the features’ relevance is essential not only to explain the features’ contribution to the model but also to conduct proper Feature Selection (FS) [19]–[21]. FS is traditionally considered a preprocessing technique. It is

The associate editor coordinating the review of this manuscript and approving it for publication was M. Venkateshkumar<sup>1</sup>.

well-known that in data analysis in general, and for ANN in particular, data preprocessing is one of the essential stages in the development of a solution, and the choice of preprocessing steps can often have a significant effect on the algorithm's performance [22]. In the era of digitalisation, hundreds of features from complex systems are usually monitored to extract valuable knowledge from the data [23]. In order to reduce the model complexity as well as to save memory and computational cost, features' relevance-based FS is commonly applied [24]–[26]. In some cases, the features' relevance calculation is conducted by means of network's weights-based feature importance analysis methods [27]–[29].

Along with FS, another commonly employed preprocessing approach is the linear normalisation of the input features. Feature Normalisation (FN) is often useful if the features present values that differ significantly in magnitude. Each FN method transforms a given dataset differently. The impact of the FN method's selection on the algorithm's performance has been experimentally studied by some researchers [30]–[33] to estimate the most appropriate one for a given problem. However, it remains the extended approach of employing the min-max normalisation method before the use of an ANN [34]–[38]. Despite the importance of data normalisation, no works are found that include the influence of data normalisation when analysing the features' contribution to the resultant ANN model.

Thus, this work advances the state-of-the-art by theoretically examining the impact of data normalisation on the relative contribution of the input features to the ANN and, ultimately, the algorithm's performance. For that purpose, this work presents a new proposal for feature's contribution analysis that extends Garson's and Yoon's methods to include the normalisation influence when estimating the features' contribution to the ANN. The theoretical conclusions are also experimentally validated.

Section II describes the ANN-based models and Section III presents the formulation of FN. In Section IV the Garson's and Yoon's traditional features' relevance analysis methods based on weight matrix analysis (Section IV-A) are presented, and Section IV-B describes a new proposal for the adaptation of these methods to include the dispersion factors in the computation of the feature's contribution. Section V describes the employed well-known datasets from UCI repository [39] and argues the proposed methods of this work. The experimental results of the analysis are collected in Section VI; and a discussion and proposal of future work are described in Section VII. Finally, Section VIII collects the conclusion of the work.

## II. ARTIFICIAL NEURAL NETWORKS

An Artificial Neural Network is a weighted directed graph of interconnected neurons that propagates data from the input layer to the output layer by transforming such data to obtain valuable information for modelling a problem. A neuron receives the weighted values from the neurons of the previous layer. In the neuron, the sum of the weighted values

is computed and employed as the argument of an activation function  $\varphi : \mathbb{R} \rightarrow \mathbb{R}$ ; being the identity  $\varphi(x) = x$  the simplest one. The ANN architecture is flexible in the number of hidden layers and neurons per layer. The higher the number of hidden layers and the neurons that compose them, the higher the model complexity.

In this work, the network's layers are represented by  $h \in \{0, 1, \dots, H, H + 1\}$ , where  $H$  is the number of hidden layers, and  $h = 0$  and  $h = H + 1$  symbolise the input and output layers, respectively. The number of neurons in the  $h$ -th hidden layer is denoted by  $n_h$ . Note that  $n_0 = m$  is equal to the number of features, and for the single output problems,  $n_{H+1} = 1$ . The matrix weight of the edges that connect the neurons of the  $h - 1$  layer with the neurons of the  $h$ -th layer is  $W^h \in \mathbb{R}^{n_{h-1} \times n_h}$ , and  $b_h$  represents the bias of the  $h$ -th layer. Then, the mathematical formulation of an ANN-based model is:

$$Y = \varphi \left( \dots \varphi \left( X \cdot W^{(1)} + b_1 \right) \dots W^{(H+1)} + b_{H+1} \right) \quad (1)$$

For  $\varphi(x) = x$ , (1) can be rewritten as

$$\hat{Y} = X \cdot \left( \prod_{h=1}^{H+1} W^h \right) + cte = X \cdot \mathbb{W} + cte. \quad (2)$$

For the single output problem,  $\mathbb{W}$  is a vector of length  $m$ , where the entry  $j \in \{1, \dots, m\}$  represents the total weight the network assigns to the  $j$ -th feature.

From (1), and especially, when the activation function is the identity as in (2), the ANN's weights are the fundamental parameters that relate the input data with the estimated output. The ANN weights, along with the bias, are iteratively updated during the training phase of the model to obtain  $\hat{Y} \approx Y$ . However, for reaching so, not only the parameters training is determinant but also the quality of the input data. As authors in [40] remark, input data must be provided in the amount, structure and format that suits the data mining task. Besides, in order to avoid that the measurement unit affects the data mining task, all the features should be expressed in the same measurement units with a common scale or range. Feature Normalisation (FN) attempts to equalise the features' magnitude, and it is also employed to speed up the learning process in ANNs, helping the weights converge faster.

## III. FEATURE NORMALIZATION

FN is a preprocessing technique widely employed to avoid the magnitude differences between the features of a given dataset. Any statistical-based FN method can be expressed as

$$\tilde{X} = \frac{X - pos(X)}{dis(X)} \quad (3)$$

Equation (3) transforms a given dataset  $X$  into a normalised one  $\tilde{X}$  based on  $pos(X)$  and  $dis(X)$ ;  $pos(X)$  refers to the position or central tendency statistic vector,<sup>1</sup> whereas

<sup>1</sup>For the sake of brevity, the vector composed by position or central tendency statistic is referred as position statistic from now on.



$dis(X)$  is the dispersion statistic vector which scales the features.

Equation (4) defines the decimal notation proposed to highlight the magnitude factors of each feature.

$$x_{ij} = sign(x_{ij}) 0.d_1d_2d_3 \dots \cdot 10^{n_j} = \widehat{x}_{ij} \cdot 10^{n_j} \quad (4)$$

In (4),  $d_1, d_2, d_3, \dots \in \{0, 1, \dots, 9\}$  and  $n_j \in \mathbb{Z}$  is fixed in such a way that  $\forall j, |n_j|$  is the minimum number which fulfils:  $X_j = \widehat{X}_j \cdot 10^{n_j}$ , and  $max|\widehat{X}_j| < 1$ . Then,  $\forall i, j, |\widehat{x}_{ij}| < 1$ , and  $10^{n_j}$  represents the magnitude factor of each feature. With the defined decimal notation, and since the statistical factors are estimated by linear operations,  $pos(X_j) = pos(\widehat{X}_j) \cdot 10^{n_j}$  and  $dis(X_j) = dis(\widehat{X}_j) \cdot 10^{n_j}$ ; FN can be re-written as

$$\widetilde{X}_j = \frac{\widehat{X}_j \cdot 10^{n_j} - pos(\widehat{X}_j) \cdot 10^{n_j}}{dis(\widehat{X}_j) \cdot 10^{n_j}} = \frac{\widehat{X}_j - pos(\widehat{X}_j)}{dis(\widehat{X}_j)} \quad (5)$$

Equation (5) shows that the magnitude factors in the normalised dataset disappear. This is the main reason why, as aforementioned in Section II, FN is widely employed to equalise the magnitude of the features. However, as a consequence of FN, each feature  $j$  is scaled by a dispersion factor  $dis(\widehat{X}_j)$  dependant on its values distribution.

Note that the normalised features present a dispersion equal to 1 in terms of the dispersion factor employed to transform the dataset. But, in order to fulfil  $dis(\widetilde{X}_j) = 1$  each feature  $\widehat{X}_j$  is differently expanded or compressed. Thus, the higher the value of  $dis(\widehat{X}_j)$ , the higher the level of compression a feature undergoes, and consequently, the lower the contribution weight on the ML algorithm. Analogously, the lower the value of  $dis(\widehat{X}_j)$ , the higher the expansion of  $\widehat{X}_j$ , and the higher the expected contribution to the model. Thus, the inverse of the dispersion factors can be viewed as unsupervised feature weights. In fact, in the ANN's first layer, the network's weights are multiplied by the normalisation weights, so the first layer's resulting weights are  $dis(\widehat{X}_j)^{-1} \cdot W_j^0 \forall j \in \{1, \dots, m\}$ . Then, since the dispersion factors act as weights along with the network's weights, it conditions the model performance and the features' contribution to the model.

#### IV. FEATURE RELEVANCE ANALYSIS METHODS

ANN-based models are considered a "black box" since the network's predictions cannot be directly explained. Therefore, several approaches to understand the features' contribution for modelling the network have been proposed. Some features' relevance analyses for ANN-based problems rely on the network's Weights Matrix Analysis (WMA) to estimate the features' contribution to the model. In this Section, first, the well-known Garson's and Yoon's methods are described. Next, a novel approach that considers the network's weights and the dispersion factors is proposed.

##### A. FEATURE RELEVANCE ANALYSIS METHODS BASED ON NETWORK'S WEIGHT MATRIX

In order to understand the features' contribution to the model, WMA methods are usually employed. These methods, which

belong to the features' relevance explanation techniques, calculate the features' contribution based on the network's weights related to each feature. Among the WMA methods, Garson's [41], and Yoon's methods [42] are well-known. They compute the features' contribution values as defined in (6) and (7), respectively.

$$Garson_j = \frac{|\prod_{h=1}^{H+1} W^h|_j}{\sum_{j=1}^m |\prod_{h=1}^{H+1} W^h|} = \frac{|\mathbb{W}_j|}{\sum_{j=1}^m |\mathbb{W}_j|} \in [0, 1] \quad (6)$$

$$Yoon_j = \frac{(\prod_{h=1}^{H+1} W^h)_j}{\sum_{j=1}^m |\prod_{h=1}^{H+1} W^h|} = \frac{\mathbb{W}_j}{\sum_{j=1}^m |\mathbb{W}_j|} \in [-1, 1] \quad (7)$$

Similarly to other features' relevance explanation techniques, it is considered that the higher the  $Garson_j$  or  $Yoon_j$  value is, the higher the features' contribution to the network.

Note that the preprocessed features implicitly influence the contribution values estimated by Garson's and Yoon's methods in the sense that the weights have been obtained from the training process with the preprocessed features (and not the raw features). In order to calculate more precisely the real feature's contribution to the model, a novel method that explicitly and formally considers the dispersion factors in addition to the network's weights is presented.

##### B. FEATURE RELEVANCE ANALYSIS METHODS BASED ON NETWORK'S WEIGHT MATRIX AND DISPERSION FACTORS: A NEW PROPOSAL FOR THE ADAPTATION OF GARSON'S AND YOON'S METHODS

Despite data preprocessing –and hence FN– is considered essential to obtain quality results, until the date, no works that analyse the preprocessing stage impact for estimating the features' influence on the ANN-based model are found. This work aims to advance the state-of-the-art by including the dispersion factors in the features' contribution estimation.

Equation (2) can be viewed as the formula for  $\hat{Y}$  estimation given a dataset  $X$ . However, before ANN employment,  $\forall j \in \{1, \dots, m\} X_j$  is usually transformed by a statistical-based normalisation method. Then, from (2) and (5), the mathematical formulation of an ANN-based model trained with a normalised dataset  $\widetilde{X}$  can be re-defined as:

$$\hat{Y} = \widetilde{X} \cdot \left( \prod_{h=1}^H W^h \right) + cte = X \cdot \mathbb{D} \cdot \left( \prod_{h=1}^H W^h \right) + cte \quad (8)$$

where  $\mathbb{D} = \text{diag } dis(X_1)^{-1}, \dots, dis(X_m)^{-1}$  is the diagonal matrix, and the elements  $\mathbb{D}_{jj}$  correspond to the inverse of the dispersion factor of the  $j$ -th feature. Equation (8) illustrates that the dispersion factors, in addition to the weight matrix, influence the features' contribution to the model. Consequently, in order to estimate the true impact of a given feature on the model, this work proposes to include the dispersion factors in Garson's and Yoon's methods for the features' influence calculations as follows:

$$\widehat{Garson}_j = \frac{(\mathbb{D} \cdot |\mathbb{W}|)_j}{\sum_{j=1}^m |(\mathbb{D} \cdot \mathbb{W})_j|} \in [0, 1] \quad (9)$$

$$\widehat{Yoon}_j = \frac{(\mathbb{D} \cdot \mathbb{W})_j}{\sum_{j=1}^m |(\mathbb{D} \cdot \mathbb{W})_j|} \in [-1, 1] \quad (10)$$

As described in Section IV-A, the higher the value of  $\mathbb{W}_j$ , the higher the  $j$ -th feature's contribution to the network. Similarly, by interpreting the inverse of the dispersion as unsupervised weights, as stated in Section III, the higher the value of  $dis(X_j)^{-1}$ , the higher the contribution of such feature to the model. Thus, the same rationale can be applied to  $dis(X_j)^{-1} \cdot |\mathbb{W}_j|$ .

**V. MATERIALS AND METHODS**

This Section describes the procedure employed to experimentally analyse and validate that the FN method selection influences the ANN-based model performance and hence, justify the inclusion of the dispersion factors in the features' contribution estimation. Given the experimental analysis and validation, Fig. 1 shows the high-level diagram of 1) the data split and preprocessing and 2) the ANN-based model training and evaluation conducted in this work. The elements employed in the following Sections to evaluate the FN influence on the ANN-based models are highlighted with a magnifying glass symbol. Note that Section III demonstrates that the magnitude factors disappear when normalising the features. Consequently, from now on  $\widehat{X}$  (4) is employed.

**A. DATASETS**

In order to validate the hypothesis presented in Section IV-B, four public available real use cases from UCI repository [39] are employed.

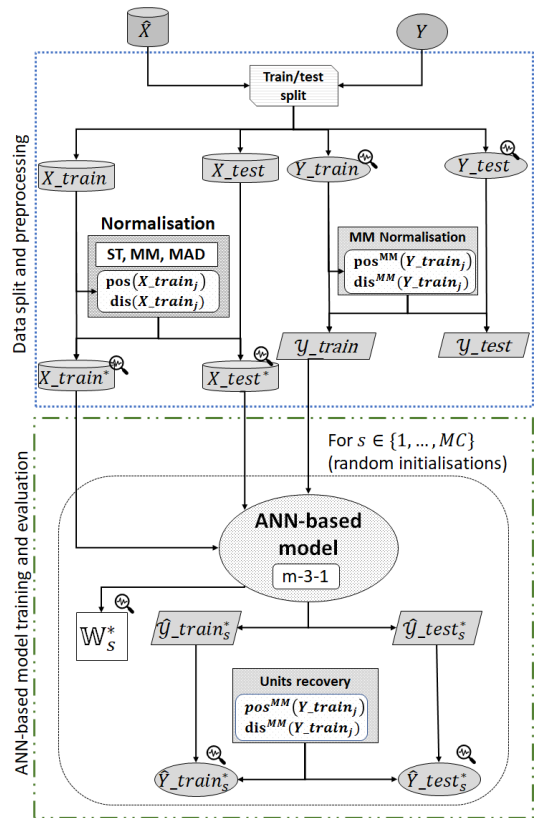
**TABLE 1.** Description of datasets from UCI repository utilized in this work.

Dataset	Features	Samples	Y	Use case
CBM [43]	14	11934	[0.975, 1]	Naval propulsion plants-compressor decay state.
NOX [44]	9	7384	[25.905, 119.68]	Gas turbine CO and NOx emission.
CO [44]			[0.2128, 41.097]	Data from 2015
News [45]	59	39644	[8, 731]	News online popularity

Table 1 summarises the utilised datasets. This work is focused on regression problems; then, the four datasets have continuous output values. Both NOX and CO datasets utilise the same input data. However, NOX dataset aims at estimating the Nitrogen oxides (NOx) emission from a gas turbine, while for CO the Carbon monoxide (CO) emission of the same gas turbine is registered.

**B. DATA PREPARATION, ANN-BASED MODEL TRAINING, AND EVALUATION METRICS**

The first step consists in preparing a given dataset to obtain the train/test subsets and normalise the data. Then, the ANN-based model is trained with the preprocessed data. Each step of Fig. 1 and the primary metrics employed to analyse the obtained results are next described.



**FIGURE 1.** High-level diagram of the proposed method for data preprocessing, and ANN-based model train and evaluation.

**1) DATASET SPLIT INTO TRAIN AND TEST SETS**

A given dataset  $X \in \mathbb{R}^{n \times m}$  composed by  $n$  samples described by  $m$  features and the associated real labels  $Y \in \mathbb{R}^n$  are split into train ( $X_{train}, Y_{train}$ ) and test ( $X_{test}, Y_{test}$ ) disjoint sets of  $n_{train} = 0.7 \cdot n$ , and  $n_{test} = n - n_{train}$  samples, respectively. The training set is employed to adjust the model's parameters (weights and bias), while the test set is utilised to validate the model's performance.

**2) NORMALIZATION METHODS**

In order to validate the impact of the FN method selection on the network, three well-known normalisation methods are employed in this work. Table 2 presents the selected normalisation methods and the statistical position and dispersion factors utilised to transform the features.

Each normalisation method from Table 2 utilises different position and dispersion statistics to transform the features of a given dataset. More concretely, ST, MM and MAD compress or expand each feature based on its standard deviation  $\sigma$ , range, and median absolute deviation  $mad$  dispersion statistics, respectively. Thus, ST and MAD calculate

**TABLE 2. Normalization methods selected in this work for the analysis and validation of the proposal.**

Normalisation method	$\text{pos}(X_j)$	$\text{dis}(X_j)$
Standardisation (ST)	$X_j$	$\sigma_j$
Min-max (MM)	$\min(X_j)$	$\text{range}(X_j) = \max(X_j) - \min(X_j)$
Median Absolute Deviation (MAD)	$\text{Me}(X_j)$	$\text{mad}(X_j) = \text{Me}( X_j - \text{Me}(X_j) )$

the dispersion of the features' samples around the mean and median values, respectively. In contrast, MM computes the statistical factors considering the extreme values of the features.

The FN methods from Table 2 are employed as follows: for each normalisation method  $* \in \{ST, MM, MAD\} = Norm$  the statistical factors are calculated from  $X_{train}$  as described in (3). Then, they are applied to  $X_{train}$  and  $X_{test}$  to create train  $X_{train}^*$  and test  $X_{test}^*$  datasets. Thus, from a given dataset  $X$ , a normalised dataset  $\tilde{X}^*$  is obtained for each  $* \in Norm$ .

Independently of the FN method utilised to transform the input features, the output label is normalised with MM. The only difference between the analysed models is the normalisation method utilised for the input data transformation. Then,  $\min(Y_{train})$  and  $\text{range}(Y_{train})$  values are utilised in (3) to calculate  $\mathcal{Y}_{train}$ ,  $\mathcal{Y}_{test}$ .

### 3) ANN TRAINING STRATEGY FOR THE ANALYSIS OF THE NORMALISATION INFLUENCE

In this work, for each normalised dataset, an ANN with one hidden layer composed of three hidden neurons ( $[m-3-1]$ ) is utilised. The neurons of the hidden and output layers are activated with the identity function. A maximum of 300 iterations is set, and the training stops if no improvement is observed for 10 iterations. The network's weights are initialised with Xavier's method [46], and  $MC = 50$  random initialisations are utilised for each normalised dataset. In this way, for each initialisation,  $s \in \{1, \dots, MC\}$ , the networks trained with each  $\tilde{X}^*$  employ the same initial network's weights.

The ANN is trained with  $X_{train}^*$  searching for the optimal weights and bias values that obtain, for each initialisation,  $\hat{\mathcal{Y}}_{train_s}^* \approx \mathcal{Y}_{train}$ . From each trained network, the estimated outputs  $\hat{\mathcal{Y}}_{train_s}^*$  and  $\hat{\mathcal{Y}}_{test_s}^*$  are calculated and re-scaled with the statistical factors of  $Y_{train}$  into the original units, obtaining  $\hat{Y}_{train_s}^*$  and  $\hat{Y}_{test_s}^*$ . Besides,  $\forall s$ , the network's weight vector  $\mathbb{W}_s$  is saved for further analyses.

### 4) METRICS FOR INFERRING THE NORMALISATION INFLUENCE

The main goal of this work is to validate the impact of the FN method selection influence on the model's performance and the adequacy of employing dispersion factors, in addition to the network's weight vector, to infer the features' contribution appropriately. For doing so, (1) the estimated outputs  $\hat{Y}_{train_s}^*$  and  $\hat{Y}_{test_s}^*$ ; (2) the statistical dispersion factors  $\text{dis}(X_j)$ ; and (3) the weight vector  $\mathbb{W}_s^*$  obtained from the trained models are analysed primarily based on the following metrics.

- Kendall's  $\tau$  correlation coefficient [47] measures the degree of similarity between two ranks assigned to the same set of objects, i.e. paired rankings. Kendall's  $\tau$  ranges from -1 to 1. A  $\tau = 0$  indicates the non-relationship between the two rankings. If  $\tau = -1$ , a ranking is the inverse of the other, while  $\tau = 1$  when both rankings are the same. Then, the higher the value of  $\tau$ , the higher the ranks similarity.

- Distance is a key concept in many statistical and pattern recognition methods which measures the closeness or similarity between two objects. The Euclidean distance  $E_d$  between two vectors  $\mathbf{a}$ ,  $\mathbf{b}$  is defined as  $E_d(\mathbf{a}, \mathbf{b}) = \sqrt{\sum_{j=1}^m (a_j - b_j)^2}$ , and it is equal to 0 if the components of both vectors are the same. The higher  $E_d$ , the higher the dissimilarity between the components of the vectors. Although  $E_d$  is scale dependant, in this work, it is applied to vectors with components ranging from 0 to 1.

- Performance measures are utilised to analyse the error or the similarity between two output features  $\mathbf{y}_a$ ,  $\mathbf{y}_b$  of length  $n$ . In this work, the mean absolute error (MAE), the root mean squared error (RMSE) and the coefficient of determination ( $R^2$ ) regression performance measures are employed.  $\text{MAE}(\mathbf{y}_a, \mathbf{y}_b) = (1/n) \sum_{i=1}^n |y_{a_i}, y_{b_i}|$  and  $\text{RMSE}(\mathbf{y}_a, \mathbf{y}_b) = (1/n) \sqrt{\sum_{i=1}^n (y_{a_i}, y_{b_i})^2}$  measures the error as the mean absolute and the mean square quadratic differences between the elements of both output features, respectively. Thus, the lower the MAE and RMSE values, the higher the similarity between  $\mathbf{y}_a$  and  $\mathbf{y}_b$ . In contrast,  $R^2(\mathbf{y}_a, \mathbf{y}_b) = 1 - (\sum_{i=1}^m (y_{a_i} - y_{b_i})^2 / \sum_{i=1}^m (y_{a_i} - \bar{y}_a)^2)$  is a statistical measure of how well the regression predictions  $y_b$  approximate points of  $y_a$ .  $R^2$  takes values up to 1. Values of  $R^2$  lower than 0 appear when the model fits the data worse than a horizontal hyper-plane, while  $R^2 = 1$  indicates that the regression predictions perfectly fit the data. Then, the higher the  $R^2$ , the higher the similarity between  $\mathbf{y}_a$  and  $\mathbf{y}_b$ .

### C. ANALYSIS OF THE NORMALIZATION INFLUENCE ON THE MODEL'S PERFORMANCE

The first analysis aims to verify that the model's performance varies depending on the selected FN method. In particular, the analysis lies in 1) studying the differences between the outputs predicted by the models trained with the differently normalised datasets and 2) comparing the ANN-based models' performance depending on the FN method.

#### 1) DIFFERENCE BETWEEN THE PREDICTIONS ESTIMATED BY THE DIFFERENT MODELS

In order to analyse the difference between the predictions estimated by the different models, for each  $s \in \{1, \dots, MC\}$  the MAE, RMSE and  $R^2$  between  $\hat{Y}_{train_s}^*$  and  $\hat{Y}_{train_s}^+$  and between  $\hat{Y}_{test_s}^*$  and  $\hat{Y}_{test_s}^+$  for  $* \neq + \in Norm$  are calculated, and the maximum, mean, minimum and standard deviation (std) values are computed for  $MC$  initialisations. If the selection of the FN method does not influence the model's performance, then  $\text{MAE} = \text{RMSE} = 0$  and  $R^2 = 1$ . Otherwise, differences between the estimated outputs would

demonstrate the influence on the model's performance of the FN methods.

## 2) ANN PREDICTION PERFORMANCE DEPENDING ON THE FN METHOD

Complementary, in order to analyse the ANN prediction performance depending on the FN method, for each random initialisation, MAE, RMSE and  $R^2$  between  $\hat{Y}_{train}^*$  and  $\hat{Y}_{train}$ , and  $\hat{Y}_{test}^*$  and  $\hat{Y}_{test}$  are calculated together with the maximum, mean, minimum and std values estimated for the MC initialisations. Similarly, if the normalisation method selection does not affect the model's performance, the same MAE, RMSE and  $R^2$  statistical values are expected independently from the FN method employed when transforming the data. In contrast, differences in the estimated performance would also demonstrate the hypothesis of this work.

Complementary, the non-parametric Wilcoxon signed-rank test [48] is employed to check the existence of statistical differences between  $RMSE(Y_{train}, Y_{train}^*)$  and  $RMSE(Y_{train}, Y_{train}^+)$ , or between  $RMSE(Y_{test}, Y_{test}^*)$  and  $RMSE(Y_{test}, Y_{test}^+)$  for  $* \neq + \in Norm$ . In Wilcoxon signed-rank test, the same subjects are evaluated under two different conditions. In this case, each subject is the model with the  $s$ -th random initialisation of the weights, and the different conditions are the FN methods  $* \neq + \in Norm$  utilised to transform the network's input features. The null hypothesis  $H_0$  of Wilcoxon signed-rank test assumes that the related samples  $[RMSE(Y_{train}, Y_{train}_1^*), \dots, RMSE(Y_{train}, Y_{train}_{MC}^*)]$  and  $[RMSE(Y_{train}, Y_{train}_1^+), \dots, RMSE(Y_{train}, Y_{train}_{MC}^+)]$  come from the same population, i.e., the distribution of differences has a median of zero. The test's p-values are calculated and, if p-value < 0.05,  $H_0$  is rejected with a significant level of 5%.

## D. ANALYSIS OF THE DISPERSION FACTORS AS EXPLANATORY FACTORS OF THE VARIATIONS IN MODEL'S PERFORMANCE

Each FN method collected in Table 2 employs different dispersion statistics or factors to transform the input features. Then, as stated in Section III, it is expected that each FN method  $* \in Norm$  transforms differently a given dataset, which ultimately conditions the features' contribution values and, consequently, the ANN-based model's performance. Once verified that FN methods impact the model's performance, the dispersion factors are analysed as explanatory factors of such variations. It is assumed that a relationship exists between the results in the ANN-based model performance and the dispersion factors. Thus, the higher the differences between the dispersion factors, the higher the difference between the output estimations and the weight vector of the models trained with different  $\tilde{X}^*$ . The analysis of the dispersion factors as explanatory factors is conducted as follows:

### 1) ANALYSIS OF PROPORTIONAL DISPERSION FACTORS

In this work, first,  $\forall * \in Norm$  the scaling dispersion factors  $w_j^* = 1/dis^*(X_j)$ , specifically, their Proportional Dispersion Weight (PDW) estimated as  $\hat{w}_j^* = w_j^* / \sum_{j=1}^m w_j^*$  are analysed. In order to infer the expected similarity between  $\tilde{X}^*$  and  $\tilde{X}^+$  for  $* \neq + \in Norm$ , the Kendall's  $\tau$  correlation and the Euclidean distance between  $\hat{w}_j^*$  and  $\hat{w}_j^+$  are calculated to evaluate the similarity between the PDWs employed to create the different normalised datasets.

### 2) SIMILARITY BETWEEN PDW AND PERFORMANCE RESULTS

Once estimated the PDWs for each normalisation method, the level of similarity between the dispersion factors are compared accordingly with the level of similarity between the model's performance reached from Section V-C2 by differently normalised datasets. The coherence between both will allow setting the dispersion factors as explanatory factors of the variations in the model's performance.

## E. ANALYSIS OF THE NORMALISATION INFLUENCE ON THE FEATURES' CONTRIBUTION

As described in Section II, Garson's and Yoon's methods are based on the network's weights to estimate the contribution of each feature in the model. From (6) and (7) it is observed that the main difference in the resulting features' contribution values is due to the direction, so, for the sake of brevity, from now, only Garson's method is considered.

Thus,  $G^*$  and  $\hat{G}^*$  represent the features' contribution values calculated with the traditional and the adapted Garson's methods, respectively.  $* \in Norm$  refers to the FN method employed to obtain the  $X_{train}^*$ , so as the weight vectors  $\mathbb{W}^*$  from (6) and  $\mathbb{D}^* \cdot \mathbb{W}^*$  from (9) can be computed. Then, for each  $*$ , MC networks with different initial weights are trained, and  $G_s^*$  and  $\hat{G}_s^*$  are finally computed.

Once analysed the FN method selection influence on the model's performance and the relationship between the PDWs and the estimated outputs, this Section studies the impact of FN on the features' contribution values and the adequacy of the proposed adapted Garson's method to calculate the real features' influence. For doing so, first, an analysis of the features' contribution values in terms of the traditional and the adapted Garson's method is conducted. Then, a comparison with the results from Sections V-C and V-D is performed. Finally, a Feature Selection strategy is applied in order to demonstrate the superiority of the proposed adapted Garson's method for estimating the real features' contribution.

### 1) MEAN FEATURES CONTRIBUTION

In order to analyse the FN method selection impact on the features' influence on the network, the mean features' contribution values resulting from the MC random initialisation based on the traditional  $\bar{G}^* = (1/MC) \sum_{s=1}^{MC} G_s^*$  and on the proposed adapted Garson's method  $\hat{\bar{G}}$  are calculated and analysed considering the steps described below.

a: TRADITIONAL GARSON'S METHOD

First, the differences between the weight matrix-based features contribution derived from the selection of the FN method is analysed. In order to inspect the  $\overline{G}_j^*$  values distribution and the discriminative influence of the  $j$ -th feature: 1) the difference between the maximum and the minimum, and 2) the standard deviation of the features' contribution values are calculated. Then, aiming at examining the effect of FN in the features' influence on the model, a pairwise comparison between  $\overline{G}^*$  and  $\overline{G}^+$  with  $* \neq +$  is conducted in terms of Kendall's  $\tau$  correlation coefficient and Euclidean distance.

b: PROPOSED ADAPTED GARSON'S METHOD

The same analysis is performed over  $\hat{G}^*$  to inspect the features' contribution computed with the adapted Garson's method.

c: COMPARISON BETWEEN THE TRADITIONAL AND THE PROPOSED ADAPTED GARSON'S METHOD

Finally, with the aim of inferring the validity of the proposed adapted Garson's method to estimate the real features' contribution, first, a comparison between  $\overline{G}^*$  and  $\hat{G}^*$  is performed. Then, the results from Sections V-C2 and V-D are here utilised to infer from the correspondence between the models' performance, the dispersion factors and the features' relevance analysis methods the superiority of the proposed adapted Garson's method.

2) FEATURE SELECTION BASED ON THE FEATURES' CONTRIBUTION

In order to demonstrate the superiority of the proposed adapted Garson's method, a FS strategy is conducted to analyse the effect of removing features considering the traditional Garson's method versus the proposed adapted one. The estimated features' contribution values from the models that obtain the lowest RMSE are employed for this strategy. Then, for each  $* \in Norm$ , the feature' influence values calculated with the traditional Garson's method are denoted as  $\underline{G}^*$ , while the estimated with the proposed one are referred to as  $\hat{G}^*$ . The FS based on the features' contribution values computed with the traditional or the proposed Garson's methods ( $f_C \in \{\underline{G}^*, \hat{G}^*\}$ ) is applied as described in Algorithm 1.

VI. EXPERIMENTAL VALIDATION

This Section shows the experimental results obtained from training and testing the ANN architecture presented in Section V-B3. More concretely, first, the influence of the FN methods on the models' performance is studied. Next, an analysis of the proportional dispersion weights as explanatory factors of the estimated outputs is presented. Finally, the impact of FN on the features' contribution is demonstrated, and the superiority of the proposed adapted Garson's method is validated.

Algorithm 1 Feature Selection Strategy

```

1: for  $f_C \in \{\underline{G}^*, \hat{G}^*\}$  do
2:   for  $ite \in \{1, \dots, m - 1\}$  do
3:     Remove the  $ite$  features with lowest  $f_C$  value.
4:     Train the network, estimate the output and re-scale it to the original units.
5:     Estimate the RMSE between the real labels and the estimated ones.
6:   end for
7: end for
8: Plot the RMSE values estimated based on  $\underline{G}^*$ , and  $\hat{G}^*$  jointly with the RMSE estimated with all the features to analyse the effect of the feature removal.
    
```

A. ANALYSIS OF THE NORMALISATION INFLUENCE ON THE MODEL'S PERFORMANCE

As described in Section V-C, an analysis of the dissimilarity between the outputs estimated by the models trained with differently normalised datasets is conducted. Note that  $\forall * \in Norm$ , the same 50 random initialisations establish the initial weights of the ANN. Thus, the only differences when training the models are the FN methods utilised to transform the input features.

1) DIFFERENCE BETWEEN THE PREDICTIONS ESTIMATED BY THE DIFFERENT MODELS

First, the comparison between the estimated outputs obtained from the differently normalised datasets is conducted.

Table 3 collects for each dataset the maximum, mean, minimum and standard deviation of MAE, RMSE and  $R^2$  values from comparing the estimated  $\hat{Y}_{train}^*$  with  $\hat{Y}_{train}^+$  and  $\hat{Y}_{test}^*$  with  $\hat{Y}_{test}^+$  for  $* \neq + \in Norm$ . Given that similar results are obtained from train and test sets, for sake of brevity only the results from the training set are described. From the calculated scores presented in Tables 3a to 3d variations in the estimated outputs derived from the FN method selection can be inferred. In News, NOX and CO datasets the mean MAE is up to 1021.831, 0.363 and 1.547, respectively. Similarly, the mean  $RMSE(\hat{Y}_{train}^*, \hat{Y}_{train}^+)$  values obtained are higher than 0.28, 1.21, and 86.41, respectively. In the case of CBM dataset, the RMSE vales are close to zero. However, in Table 3a, the mean  $R^2$  values are lower than 0.605 when comparing  $Y_{train}^{MM}$  with  $Y_{train}^{ST}$  or  $Y_{train}^{MAD}$ , respectively. Then, from Table 3 it is concluded that the predictions considerably vary depending on the FN method selected for the feature preprocessing phase.

2) COMPARISON BETWEEN THE MODELS' PERFORMANCE SCORES

As demonstrated above, different outputs are obtained from the models trained with differently normalised data. As an example, Fig. 2 depicts the  $Y_{test}$  and  $Y_{test}^*$  obtained for  $* \in Norm$  from the NOX dataset.

**TABLE 3. Comparison between the estimated outputs  $\hat{Y}^*$ ,  $\hat{Y}^+$  for  $* \neq + \in Norm$  in terms of MAE, RMSE and  $R^2$ .**

		Train			Test		
		ST vs MM	ST vs MAD	MM vs MAD	ST vs MM	ST vs MAD	MM vs MAD
MAE	max	0.005	0.003	0.005	0.005	0.003	0.005
	mean	<b>0.003</b>	0.000	<b>0.003</b>	<b>0.003</b>	0.000	<b>0.003</b>
	min	0.001	0.000	0.001	0.001	0.000	0.001
RMSE	max	0.004	0.001	0.001	0.001	0.001	0.001
	mean	0.004	0.003	0.004	0.004	0.003	0.004
	min	<b>0.002</b>	0.000	<b>0.002</b>	<b>0.002</b>	0.000	<b>0.002</b>
$R^2$	max	0.001	0.000	0.001	0.001	0.000	0.001
	mean	0.001	0.000	0.001	0.001	0.000	0.001
	min	0.874	0.999	0.822	0.873	0.999	0.821
max	max	0.605	0.967	<b>-0.324</b>	<b>0.604</b>	0.967	<b>-0.345</b>
	mean	-0.040	-0.018	-7.594	-0.066	-0.053	-8.070
	std	0.163	0.142	1.320	0.165	0.147	1.382

(a) CBM turbine

		Train			Test		
		ST vs MM	ST vs MAD	MM vs MAD	ST vs MM	ST vs MAD	MM vs MAD
MAE	max	1032.526	3199.007	3226.765	900.803	845.512	986.802
	mean	70.406	<b>1021.831</b>	1017.157	60.408	<b>106.492</b>	91.797
	min	12.986	17.183	18.070	12.979	13.049	12.178
RMSE	max	185.677	832.802	830.036	165.592	150.734	136.101
	mean	831.934	643.061	715.432	828.236	629.392	709.561
	min	48.247	<b>86.412</b>	74.341	48.018	<b>80.288</b>	68.274
$R^2$	max	0.997	0.999	0.912	0.974	1.029	0.987
	mean	141.181	114.295	98.760	140.604	112.828	97.985
	min	0.993	0.988	0.986	0.993	0.993	0.994
max	max	0.872	<b>-64.589</b>	<b>-71.387</b>	0.831	<b>0.597</b>	<b>-0.176</b>
	mean	-1.687	-430.938	-441.726	-4.272	-3.645	-41.970
	std	0.433	99.112	102.614	0.764	0.761	5.992

(b) News

		Train			Test		
		ST vs MM	ST vs MAD	MM vs MAD	ST vs MM	ST vs MAD	MM vs MAD
MAE	max	3.308	1.532	3.319	3.291	1.541	3.359
	mean	1.428	0.428	<b>1.547</b>	1.439	0.429	<b>1.559</b>
	min	0.482	0.140	0.475	0.479	0.139	0.478
RMSE	max	0.701	0.224	0.741	0.706	0.227	0.746
	mean	2.614	1.232	2.633	2.614	1.229	2.668
	min	1.117	0.339	<b>1.213</b>	1.123	0.339	<b>1.220</b>
$R^2$	max	0.382	0.116	0.383	0.379	0.115	0.380
	mean	0.547	0.179	0.581	0.551	0.180	0.585
	min	0.997	1.000	0.997	0.997	1.000	0.997
max	max	0.966	0.997	0.957	0.964	0.997	0.956
	mean	0.852	0.969	0.826	0.850	0.968	0.824
	std	0.035	0.005	0.042	0.036	0.005	0.043

(c) NOX

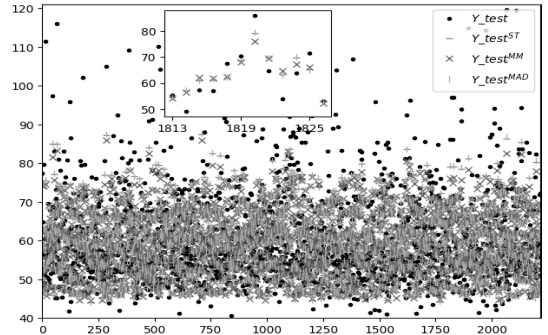
		Train			Test		
		ST vs MM	ST vs MAD	MM vs MAD	ST vs MM	ST vs MAD	MM vs MAD
MAE	max	0.859	0.372	1.068	0.873	0.363	1.079
	mean	0.327	0.110	<b>0.363</b>	0.327	0.110	<b>0.364</b>
	min	0.084	0.029	0.080	0.082	0.029	0.077
RMSE	max	0.177	0.060	0.191	0.176	0.060	0.190
	mean	0.657	0.290	0.833	0.666	0.279	0.833
	min	0.255	0.086	<b>0.284</b>	0.254	0.086	<b>0.284</b>
$R^2$	max	0.065	0.023	0.061	0.065	0.023	0.060
	mean	0.138	0.047	0.150	0.136	0.046	0.148
	min	0.998	1.000	0.998	0.998	1.000	0.998
max	max	0.959	0.995	0.946	0.958	0.995	0.944
	mean	0.802	0.966	0.622	0.803	0.966	0.600
	std	0.044	0.005	0.066	0.044	0.005	0.067

(d) CO

As Fig. 2 shows, the  $Y_{test}^*$  values do not match the real labels, and their values considerably differ depending on the FN method. For instance, in the zoomed subplot for sample number 1813, the estimated output for MM is more than 3 units lower than the estimated with ST and MAD; so differences in the performance of the models trained with the different normalised sets are expected.

Next, the model's performance of each selected dataset is analysed as aforementioned in Section V-C. Table 4 collects for  $* \in Norm$  the maximum, mean, minimum and standard deviation of MAE, RMSE and  $R^2$  values calculated for  $\hat{Y}_{train}^*$  with respect to  $Y_{train}$ , and for  $\hat{Y}_{test}^*$  with respect to  $Y_{test}$ . Note that the models obtain similar performance results for train and test sets, and since this work does not aim to analyse the models' generalisation ability, only the results over the train set are described.

As inferred from Table 3, and as Table 4 shows, FN method selection affects the model's performance. For instance, for News dataset (Table 4b), depending on  $* \in Norm$ , there



**FIGURE 2. NOX.**

is a difference up to 29.928 and 40.225 in terms of mean MAE and RMSE, respectively. For the rest of datasets regarding  $*$ , the differences in terms of mean MAE or RMSE are lower than 0.1. However, in the case of the CBM dataset, the 0.002 of increment in the error depending the FN method corresponds to 8% of the original range of the real output (Table 1). Nevertheless, although the models' performance differences in Tables 4a, 4c and 4d may not seem significant, notice that Table 3 shows considerable differences between the models' outputs. Then, in order to complement the conclusions derived from Table 4, Table 5 collects the p-values obtained with the Wilcoxon signed-rank test for assessing significant differences in the model's performance regarding the FN method with which the training and test sets are normalised.

The null hypothesis  $H_0$ , which states no statistical differences in the model's performance –in terms of RMSE– derived from the FN method selection, can be rejected in 17 out of 24 performed tests with a significance level of 5%. These 17 p-values, that represent the 70.833% of the p-values collected in Table 5, are remarked with bold text. In the rest of the cases (ST with respect to MM for News in the test set, and in both train and test sets for CBM in ST with respect to MAD, and MM with respect to ST and MAD for CO datasets), there is no evidence for rejecting  $H_0$ . However, Table 3b for the News dataset shows that the mean±std values of  $RMSE(\hat{Y}_{test}^{ST}, \hat{Y}_{test}^{MM})$  estimated from the 50 random initialisation is  $48.018 \pm 140.604$  (more than 6% of the range of the real labels of the dataset in Table 1). Similarly, for train and test sets, the mean±std values depicted in Table 3d when comparing the RMSE of the outputs estimated for CO dataset normalised with MM with respect to ST or MAD are  $0.25 \pm 0.14$  and  $0.28 \pm 0.15$ , respectively (around 1% of the range of the real labels in Table 1). Then, although in the mentioned cases there is no evidence for rejecting  $H_0$ , with the calculated statistics, significant differences are inferred when the estimated outputs obtained by different  $*$  are straightly compared.

All in all, it can be concluded that the selection of a normalisation method for the preprocessing phase results in significant differences in the model's performance.

**TABLE 4.** Maximum, mean, minimum and standard deviation of MAE, RMSE and R<sup>2</sup> values for comparing the real label  $Y$  and the estimated one  $\hat{Y}^*$  for each of the 50 random initializations and for each  $* \in Norm$ .

		Train				Test			
		raw	ST	MM	MAD	raw	ST	MM	MAD
MAE	max	0.007	0.005	0.007	0.005	0.007	0.005	0.007	0.005
	mean	0.006	0.004	0.006	0.004	0.006	0.004	0.006	0.004
	min	0.006	0.002	0.004	0.002	0.006	0.002	0.004	0.002
	std	0.000	0.001	0.000	0.001	0.000	0.001	0.000	0.001
RMSE	max	0.008	0.006	0.008	0.006	0.008	0.006	0.008	0.006
	mean	0.007	0.005	0.007	0.005	0.007	0.005	0.007	0.005
	min	0.007	0.003	0.005	0.003	0.007	0.003	0.005	0.003
	std	0.000	0.001	0.000	0.001	0.000	0.001	0.000	0.001
R <sup>2</sup>	max	0.079	0.843	0.494	0.850	0.082	0.849	0.498	0.857
	mean	0.025	0.583	0.235	0.589	0.026	0.588	0.235	0.594
	min	-0.016	0.352	-0.010	0.347	-0.016	0.354	-0.020	0.348
	std	0.023	0.120	0.080	0.132	0.025	0.122	0.082	0.134

(a) CBM turbine

		Train				Test			
		raw	ST	MM	MAD	raw	ST	MM	MAD
MAE	max	127.948	831.368	126.486	726.563	127.543	829.684	126.386	721.834
	mean	126.346	153.708	126.027	155.955	126.241	153.447	126.042	150.022
	min	125.809	124.746	125.769	125.405	125.749	124.436	125.731	124.715
	std	0.362	117.345	0.166	84.068	0.297	117.045	0.181	83.712
RMSE	max	150.145	1039.799	149.107	3230.031	149.874	914.020	149.214	1000.008
	mean	148.641	188.782	148.527	1045.089	148.768	182.990	148.779	184.921
	min	148.311	148.387	148.272	148.611	148.400	147.293	148.383	147.824
	std	0.361	159.569	0.172	808.528	0.288	138.917	0.177	119.457
R <sup>2</sup>	max	0.522	0.521	0.522	0.520	0.517	0.524	0.517	0.521
	mean	0.520	-0.329	0.520	-36.969	0.515	-0.158	0.514	-0.063
	min	0.510	-22.513	0.516	-225.889	0.507	-17.326	0.512	-20.939
	std	0.002	3.686	0.001	53.142	0.002	2.895	0.001	2.992

(b) News

		Train				Test			
		raw	ST	MM	MAD	raw	ST	MM	MAD
MAE	max	5.900	6.072	6.038	5.985	5.857	6.037	6.039	5.971
	mean	5.562	5.493	5.535	5.512	5.542	5.419	5.474	5.441
	min	5.476	5.348	5.407	5.351	5.453	5.281	5.332	5.252
	std	0.074	0.115	0.124	0.124	0.073	0.120	0.141	0.136
RMSE	max	7.836	8.043	8.105	7.875	7.728	8.044	8.099	7.819
	mean	7.637	7.333	7.428	7.356	7.555	7.260	7.363	7.282
	min	7.520	7.131	7.241	7.136	7.455	7.062	7.171	7.063
	std	0.065	0.152	0.191	0.156	0.057	0.159	0.197	0.163
R <sup>2</sup>	max	0.554	0.599	0.586	0.598	0.526	0.575	0.562	0.575
	mean	0.540	0.575	0.564	0.573	0.513	0.550	0.538	0.548
	min	0.515	0.489	0.482	0.511	0.491	0.448	0.441	0.479
	std	0.008	0.018	0.023	0.018	0.007	0.020	0.026	0.021

(c) NOX

		Train				Test			
		raw	ST	MM	MAD	raw	ST	MM	MAD
MAE	max	1.026	1.063	0.915	1.178	1.009	1.047	0.903	1.156
	mean	0.956	0.783	0.772	0.797	0.939	0.777	0.766	0.791
	min	0.876	0.736	0.738	0.737	0.864	0.730	0.731	0.731
	std	0.034	0.052	0.040	0.069	0.032	0.051	0.040	0.068
RMSE	max	1.676	1.593	1.480	1.703	1.876	1.820	1.710	1.913
	mean	1.596	1.294	1.291	1.306	1.808	1.546	1.543	1.557
	min	1.500	1.256	1.257	1.256	1.729	1.512	1.514	1.512
	std	0.042	0.052	0.049	0.069	0.036	0.047	0.043	0.061
R <sup>2</sup>	max	0.533	0.672	0.672	0.673	0.447	0.577	0.576	0.577
	mean	0.471	0.652	0.654	0.645	0.395	0.558	0.559	0.551
	min	0.417	0.473	0.545	0.398	0.349	0.387	0.459	0.323
	std	0.028	0.030	0.027	0.041	0.024	0.028	0.025	0.038

(d) CO

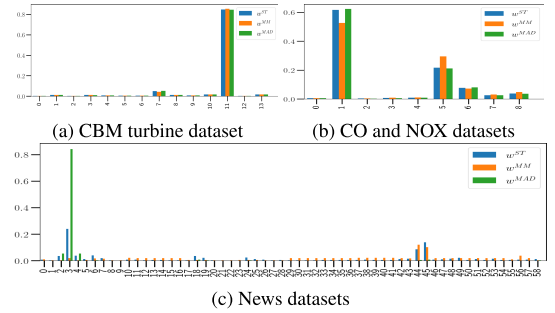
**TABLE 5.** P-values obtained from Wilcoxon signed-rank test.

	Train			Test		
	ST vs MM	ST vs MAD	MM vs MAD	ST vs MM	ST vs MAD	MM vs MAD
CBM	0.000	0.124	0.000	0.000	0.124	0.000
News	0.000	0.000	0.000	0.075	0.000	0.000
NOX	0.001	0.005	0.021	0.001	0.010	0.017
CO	0.559	0.000	0.110	0.633	0.000	0.121

**B. ANALYSIS OF THE DISPERSION FACTORS AS EXPLANATORY FACTORS**

As explained in Section II the network’s weights adjust the features’ contribution in order to create a model that estimates  $\hat{Y} \approx Y$ . Nevertheless, the hypothesis of this work is that the dispersion factors influence the model’s training, and consequently, the model’s performance. The former hypothesis has been validated in Section VI-A. In order to study the influence of the FN method selection, first, an analysis and comparison of the proportional dispersion weights estimated by different

FN methods are conducted. Then, an analysis of the similarity of these factors and the output estimations over the differently normalised datasets is performed.



**FIGURE 3.** Proportional dispersion weights (PDW)  $\hat{w}^*$  for  $* \in Norm$ .

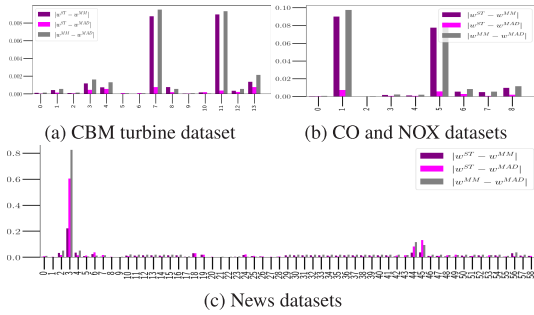
1) ANALYSIS OF THE PROPORTIONAL DISPERSION FACTORS Fig. 3 shows for each dataset and for  $* \in Norm$ , the proportional dispersion weights  $\hat{w}_j^*$  estimated for each feature. In Fig. 3c for News dataset, especially in features 2, 3, 4, 44 and 45, it is observed that  $\hat{w}_j^*$  significantly differs depending on the normalisation method employed. In fact, in News dataset,  $\forall * \in Norm$ , feature 3 obtains the highest PDW, but  $\hat{w}_3^{ST}$  takes values closer to  $\hat{w}_3^{MM}$  than to  $\hat{w}_3^{MAD}$ . In contrast, for CBM, NOX and CO datasets,  $\hat{w}_j^{ST}$  and  $\hat{w}_j^{MAD}$  present the most similar values.

In order to conduct the pairwise comparison between the dispersion factors, Fig. 4 depicts the absolute difference between  $\hat{w}_j^*$  and  $\hat{w}_j^+$  for  $* \neq + \in Norm$ .

Fig. 4c clearly shows that in News dataset  $|\hat{w}_j^{ST} - \hat{w}_j^{MM}| < |\hat{w}_j^{ST} - \hat{w}_j^{MAD}|$  for  $j \in \{3, 44, 45\}$ ; while for  $j \in \{6, 18, 19, 24, 58\}$  the minimum  $|\hat{w}_j^* - \hat{w}_j^+|$  is reached with MM and MAD. In contrast,  $\hat{w}_j^{ST}$  and  $\hat{w}_j^{MAD}$  present the lowest absolute differences in Figs. 4a and 4b.

Table 6 describes the similarity between  $\hat{w}_j^*$  and  $\hat{w}_j^+$  for  $* \neq + \in Norm$  in terms of Kendall’s  $\tau$  correlation and Euclidean distance.

For News dataset,  $\tau(\hat{w}_j^{ST}, \hat{w}_j^{MM})$  and  $\tau(\hat{w}_j^{MM}, \hat{w}_j^{MAD})$  values from Table 6a are far from 1, demonstrating that each feature’s position in the rank derived from  $\hat{w}_j^*$  significantly varies depending on  $* \in Norm$ . When comparing  $\hat{w}_j^{MM}$  with  $\hat{w}_j^{ST}$  or  $\hat{w}_j^{MAD}$  in CBM dataset, or  $\hat{w}_j^{ST}$  with  $\hat{w}_j^{MAD}$  in News dataset,  $\tau$  ranges between 0.818 and 0.889. So, minor PDWs’ rank variations can be found depending on  $* \in Norm$ . The only case in which the proportional dispersion weights’ ranking does not vary depending on  $*$  is observed in NOX or CO datasets. However, if the Euclidean distance between PDW values is analysed, it can be concluded that there are differences between the proportional estimated values with ST or MAD respect to the obtained with MM, from which differences in the network’s performance can be foreseen.



**FIGURE 4.** Absolute differences between the proportional dispersion weights  $|\hat{w}^* - \hat{w}^+|$  for  $* \neq + \in Norm$ .

**TABLE 6.** Similitude analysis between the proportional dispersion weights  $\hat{w}^*$ ,  $\hat{w}^+$  for  $* \neq + \in Norm$ .

dataset	ST vs MM	ST vs MAD	MM vs MAD
CBM	0.889	0.978	0.807
News	0.116	0.818	-0.046
NOX-CO	1	1	1

(a) Kendall's  $\tau(\hat{w}^*, \hat{w}^+)$ .

dataset	ST vs MM	ST vs MAD	MM vs MAD
CBM	0.02	0.01	0.029
News	0.147	0.042	0.173
NOX-CO	0.023	0.009	0.031

(b)  $E_d(\hat{w}^*, \hat{w}^+)$ .

## 2) SIMILARITY BETWEEN PDW AND PERFORMANCE RESULTS

Next, an analysis of  $\hat{w}^*$  as explanatory factors of the similarities between the model's performance obtained by the dataset normalised by different FN methods is conducted.

According to Fig. 3c and Table 6, for News dataset,  $\hat{w}^{ST}$  and  $\hat{w}^{MM}$  are the most similar PDWs. These results match with those from Table 3b where the lowest MAE and RMSE and highest  $R^2$  result from juxtaposing  $\hat{Y}_{train}^{ST}$  with  $\hat{Y}_{train}^{MM}$ ; while the mean  $R^2$  value obtained when comparing MAD with ST or MM is lower than  $-64.589$ . Besides, Tables 6a and 6b show that the lowest  $\tau$  and the highest  $E_d$  values are obtained when examining the PDW of News datasets. Similarly, Table 4b presents the highest differences between the model's performance resulting from the different FN methods (up to 40.225 and 34.211 in terms of mean RMSE).

Similarly, for CBM dataset, the Kendall's  $\tau$  in Table 6 is lower than 0.9 when comparing the ranks of  $\hat{w}^{ST}$  or  $\hat{w}^{MAD}$  respect to  $\hat{w}^{MM}$ . Consequently, in Table 3a the mean  $R^2$  is 0.6 and  $-0.324$  for the mentioned cases, respectively.

In contrast,  $\hat{w}^{ST}$  and  $\hat{w}^{MAD}$  are the most similar PDWs for CBM, NOX and CO datasets (Fig. 3 and Table 6). Similarly, in Tables 3a, 3c and 3d the lowest mean MAE and RMSE values are obtained when comparing  $\hat{Y}_{train}^{ST}$  with  $\hat{Y}_{train}^{MAD}$ . In fact, in NOX and CO datasets, the mean MAE and RMSE errors between the outputs estimated with MM respect to the calculated ones with ST or MAD are more than 3.2 times higher than the resulting from comparing ST and MAD. Besides, for these two datasets,  $\tau(\hat{w}^*, \hat{w}^+) = 1$  (see Table 6), which explains that in Tables 3c and 3d the mean  $R^2(\hat{Y}_{train}^*, \hat{Y}_{train}^+)$  are higher than 0.94 for  $* \neq + \in Norm$ .

All in all, it is demonstrated that the higher the similarity between  $\hat{w}^*$  and  $\hat{w}^+$  for  $* \neq + \in Norm$ , the lower the

difference expected between the output estimations resulting from the dataset normalised with  $*$  and  $+$ . Thus, in order to select among different FN methods the suitable one for the problem at hand, by knowing in advance the similarity between  $\hat{w}^*$  and  $\hat{w}^+$ , the expected similarity between  $Y_{train}^*$  and  $Y_{train}^+$  can be inferred.

## C. ANALYSIS OF THE NORMALIZATION INFLUENCE ON THE FEATURES' CONTRIBUTION

After demonstrating in previous Sections the influence of FN method selection on the model's performance, next, as detailed in Section V-E1, an analysis of the features' contribution values estimated from the differently normalised datasets is conducted based on the traditional and the proposed adapted Garson's method. In addition, in order to demonstrate the superiority of the adapted Garson's method to truly infer the real features' contribution to the model, the FS strategy described in Section V-E2 is applied.

### 1) MEAN FEATURES' CONTRIBUTION

This Section analyses the dissimilarities between the features' contribution to the models trained with different FN methods, and the differences in the contribution values estimated with the traditional Garson's method and the proposed adapted one. As described in Section V-E, this inspection is conducted over the mean contribution values  $\bar{G}$  and  $\hat{G}$  estimated from all the initialisation.

#### a: TRADITIONAL GARSON'S METHOD

Fig. 5 depicts the values of  $\bar{G}^*$  for  $* \in Norm$ . In addition, Table 7a collects for each dataset, the difference between the highest and the lowest features' contribution values, and the std of each  $\bar{G}^*$  are collected in Table 7b.  $\tau(\bar{G}^*, \bar{G}^+)$  and  $E_d(\bar{G}^*, \bar{G}^+)$  for  $* \neq + \in Norm$  are shown in Tables 7c and 7d, respectively.

In Figs. 5a-5d it is observed that  $\bar{G}^*$  values considerably varies depending on the FN method. In fact, Table 7a shows that the differences between the most extreme values are greater than 82% for the CBM dataset normalised with ST or MAD methods and for the News dataset normalised with MM. Contrary, in the other cases, the features with the lowest influence in the network present at least 60% the contribution value of the most influencing one. So, in these cases, the features' contribution to the network is more uniform than the observed in the former datasets. Finally, regarding the features' influence ranking, since 9 out of 12 Kendall's  $\tau$  values are lower than 0.72 in Table 7c, it can be concluded that the selection of the FN method considerably alters the network's weight rank. The only cases with  $\tau \geq 0.944$  are obtained when comparing  $\bar{G}^{ST}$  and  $\bar{G}^{MAD}$  for CBM, NOX and CO datasets. These results may be justified by the low difference between  $\hat{w}^{ST}$  and  $\hat{w}^{MAD}$  estimated for CBM, NOX and CO datasets observed in Figs. 4a and 4b, respectively.



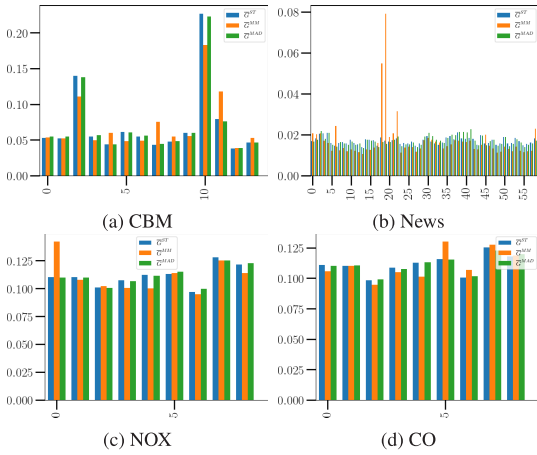


FIGURE 5.  $\hat{G}^*$ .

TABLE 7. Similitude analysis between the features' contribution estimated with the traditional Garson's method.

dataset	ST	MM	MAD
CBM	83.366	79.050	82.704
News	34.5	86.7	40.719
NOX	24.488	33.317	20.469
CO	21.605	27.399	19.233

(a)  $\max(\hat{G}^*) - \min(\hat{G}^*)$ .

dataset	ST	MM	MAD
CBM	0.051	0.04	0.05
News	0.002	0.01	0.002
NOX	0.01	0.015	0.009
CO	0.008	0.012	0.008

(b)  $std(\hat{G}^*)$ .

dataset	ST vs MM	ST vs MAD	MM vs MAD
CBM	0.187	0.956	0.231
News	0.385	0.716	0.405
NOX	0.444	1	0.444
CO	0.5	0.944	0.556

(c) Kendall's  $\tau(\hat{G}^*, \hat{G}^+)$ .

dataset	ST vs MM	ST vs MAD	MM vs MAD
CBM	0.077	0.006	0.075
News	0.078	0.01	0.08
NOX	0.036	0.005	0.037
CO	0.021	0.004	0.022

(d)  $E_d(\hat{G}^*, \hat{G}^+)$ .

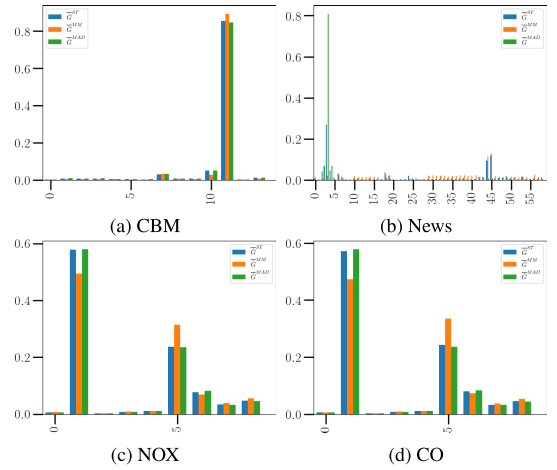


FIGURE 6.  $\hat{G}^*$ .

TABLE 8. Similitude analysis between the features' contribution estimated with the adapted Garson's method.

dataset	ST	MM	MAD
CBM	99.789	99.863	99.759
News	99.749	98.711	99.904
NOX	99.457	99.318	99.492
CO	99.473	99.390	99.503

(a)  $\max(\hat{G}^*) - \min(\hat{G}^*)$ .

dataset	ST	MM	MAD
CBM	0.225	0.237	0.223
News	0.039	0.021	0.106
NOX	0.19	0.173	0.191
CO	0.189	0.171	0.19

(b)  $std(\hat{G}^*)$ .

dataset	ST vs MM	ST vs MAD	MM vs MAD
CBM	0.824	0.978	0.846
News	0.204	0.799	0.092
NOX	1	1	1
CO	1	1	1

(c) Kendall's  $\tau(\hat{G}^*, \hat{G}^+)$ .

dataset	ST vs MM	ST vs MAD	MM vs MAD
CBM	0.046	0.008	0.053
News	0.268	0.563	0.818
NOX	0.114	0.006	0.118
CO	0.137	0.009	0.146

(d)  $E_d(\hat{G}^*, \hat{G}^+)$ .

b: PROPOSED ADAPTED GARSON'S METHOD

In the following, the same analysis is performed over  $\hat{G}^*$  for  $* \in Norm$ .

Fig. 6 illustrates significant differences between  $\max\{\hat{G}^*\}$  and  $\min\{\hat{G}^*\}$  values. In fact, as Table 8a shows, the proportional differences between the most extreme contribution values are higher than 99%. This means that, in comparison with the most influencing feature, the feature with the lowest contribution value affects less than 1% the network's calculations. Regarding the Kendall's  $\tau$  correlation coefficients collected in Table 6a, 5 out of 12 values are lower than 0.85, which means that, in those cases, the rank of  $\hat{G}^*$  varies depending on the FN method. In contrast, for NOX and CO datasets, the features' contribution ranks are the same independently from the normalisation method. This is coherent with the results observed in Table 6a, wherein  $\hat{w}^*$  presented the same rank for  $* \in Norm$ .

c: COMPARISON BETWEEN THE TRADITIONAL AND THE PROPOSED ADAPTED GARSON'S METHOD

Significant differences derived from the inclusion of dispersion factors in the features' relevance calculation are clear when examining Figs. 5 and 6. From the traditional Garson's

method, the features present more uniformly distributed contribution values that the calculated ones with the proposed approach. In fact, all the std values from  $\hat{G}^*$  are lower than 0.052 (Tables 7b); while, in Table 8b, the std values of  $\hat{G}^*$  are higher than 0.1 in most of the cases. Besides, in terms of Kendall's  $\tau(\hat{G}^*, \hat{G}^*)$  the importance rankings obtained with the traditional or the proposed Garson's methods are extremely different, as Table 9 illustrates.

In the following the results from Sections VI-C1.a and VI-C1.b are compared with those from Section VI-A.

Regarding the features' contribution values estimated with the proposed adapted Garson's method, the high  $\tau$  and low  $E_d$  values for CBM dataset from Tables 8c and 8d may explain the mean MAE and RMSE differences close to 0 in Tables 3 and 4. In NOX, where the features' relevance rankings do not vary, the low  $E_d(\hat{G}^*, \hat{G}^*)$  agree with the low mean MAE and RMSE differences from Table 3c for the corresponding case, while  $E_d(\hat{G}^*, \hat{G}^*) = 0.114$  and  $E_d(\hat{G}^*, \hat{G}^*) = 0.118$  reflect the increment in the mean errors when comparing the resulting outputs. The same rationale is applied to the results obtained for

CO dataset. In contrast, for the mentioned cases, with the traditional Garson’s method, the rank dissimilarities collected in Table 7c, and  $E_d(\overline{G}^*, \overline{G}^+) < 0.1$  from Table 7d does not seem enough to explain the performance differences from Tables 3 and 4.

TABLE 9.  $\tau(\overline{G}^*, \widehat{G}^*)$ .

	ST	MM	MAD
CBM	0.165	0.495	0.187
News	0.161	0.316	0.003
NOX	0.222	-0.111	0.222
CO	0.222	0.5	0.278

Furthermore, contrary to the observed in Table 3b for News dataset, according to Tables 7c and 7d, the lowest mean MAE and RMSE errors would be expected from the dataset normalised with ST and MAD. However, the calculated errors in Table 3b agree with the trade-off between  $\tau$  coefficients and Euclidean distance values from Tables 8c and 8d derived from the proposed adapted Garson’s method.

Then, it can be concluded that different features’ contribution values are derived from the FN method selection and that higher correspondence exists between the results from Sections VI-A and VI-B respect to the features’ contribution values estimated with the proposed adapted Garson’s method compared to the observed with the traditional one.

2) FEATURE SELECTION BASED ON FEATURES’ CONTRIBUTION

This Section applies the FS strategy described in Section V-E2 to demonstrate the superiority of the adapted Garson’s method for estimating the true features’ contribution to the model. For doing so, since multiple initialisations have been employed to train the models, for each  $* \in Norm$ , the model that reaches the lowest RMSE value is selected. Then, from such model, the features’ contribution values computed with the traditional  $G^*$  and the proposed adapted Garson’s method  $\widehat{G}^*$  are employed.

Figs. 7 to 10 depict the  $\overline{G}^*$  and  $\widehat{G}^*$  values estimated for each  $* \in Norm$  and each dataset, respectively.

In Figs. 7b, 8b, 9b and 10b it is observed that the feature with lowest influence presents a contribution value lower than 1% the value of the highest contribution. Thus, respect to the most influencing one, the contribution to the model of at least one feature is insignificant. In fact, according to Figs. 7b and 8b,  $\widehat{G}_j^* < 2 \cdot \max\{\overline{G}_j^*\}$  for most of the features. In contrast, the features’ contribution values estimated with the traditional Garson’s method do not show as high disparity between the highest and lowest features contribution values. Besides, Table 10 collects the Kendall’s  $\tau$  coefficients from comparing the features’ contribution rank estimated for  $* \neq + \in Norm$  estimated with the traditional and the proposed Garson’s method.

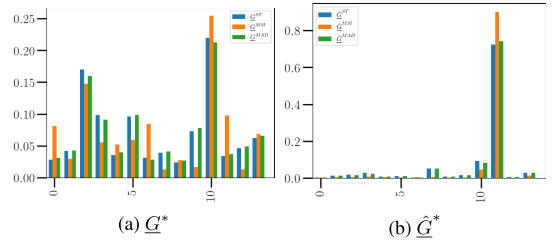


FIGURE 7. CBM.

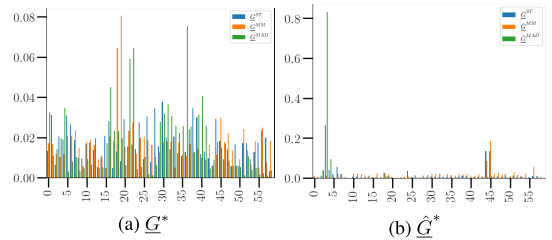


FIGURE 8. News.

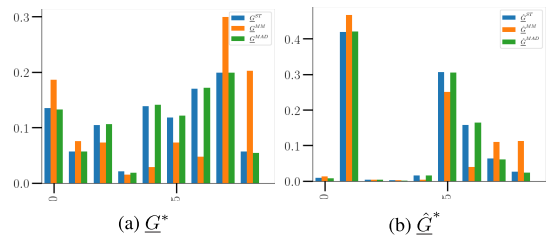


FIGURE 9. NOX.

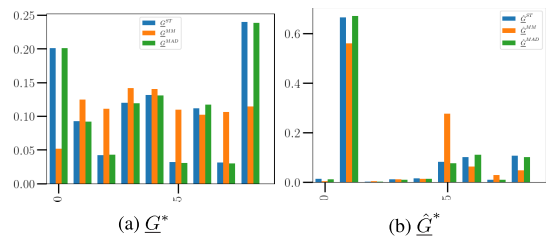


FIGURE 10. CO.

When comparing the results from Tables 10a and 10b it is observed that the features’ contribution rank significantly varies depending on  $*$  according to the traditional Garson’s method. In contrast, highest  $\tau(\widehat{G}^*, \widehat{G}^+)$  values are obtained when comparing the features’ rank estimated with the proposed approach.

Aiming at contrasting the features’ contribution rank similarity estimated by the traditional and the proposed Garson’s methods, Kendall’s  $\tau(\overline{G}^*, \widehat{G}^*)$  correlation coefficients are depicted in Table 11.

**TABLE 10.** For  $* \neq + \in Norm$ , similarity between the features' contribution rank estimated by the traditional or the proposed Garson's method.

dataset	ST vs MM	ST vs MAD	MM vs MAD
CBM	0.165	0.956	0.165
News	0.065	0.084	0.056
NOX	0.167	0.944	0.111
CO	0.167	1	0.167

(a) Kendall's  $\tau(\hat{G}^*, \hat{G}^+)$ .

dataset	ST vs MM	ST vs MAD	MM vs MAD
CBM	0.56	0.978	0.582
News	0.244	0.341	0.218
NOX	0.778	1	0.778
CO	0.556	0.889	0.667

(b) Kendall's  $\tau(\hat{G}^*, \hat{G}^+)$ .

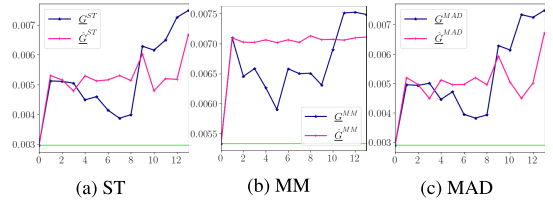
As Table 11 shows, since the  $\tau$  values are lower than 0.5, there are significant differences in the feature's influence rankings when comparing both feature relevance analysis methods. In order to demonstrate the superiority of the proposed adapted Garson's method for the estimation of the real features' contribution values, the FS strategy (Algorithm 1) is applied.

FS is the strategy of removing disturbing or non-contributing features to improve the model's performance and reduce the computational cost and the memory requirements. As explained in Section IV-B and proven in Section VI, this work states and demonstrates the influence of the FN method selection in the model's performance and in the features' contribution to the model. Thus, in this Section the features removal is conducted as described in Section V-E2 based on  $\hat{G}^*$  and  $\hat{G}^+$  for  $* \in Norm$ . Every time a feature is discarded, the model is retrained, and the RMSE between the estimated output and the real one is calculated. This experiment aims to compare the validity of the adapted Garson's method, against the traditional Garson's method, for estimating the real features' contribution. For each dataset and each  $*$ , the random initialisation that reaches the lowest RMSE when employing the whole dataset is utilised. Note that given a dataset, the lowest RMSE value is obtained with different random initialisations for the different FN methods.

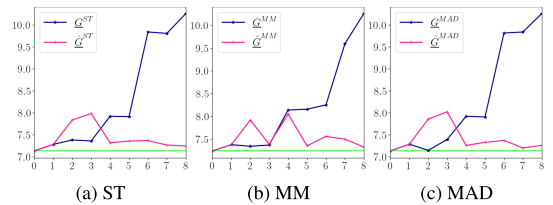
**TABLE 11.** Kendall's  $\tau$  correlation between  $\tau(\hat{G}^*$  and  $\hat{G}^+)$  for each  $* \in Norm$ .

	ST	MM	MAD
CBM	0.363	0.495	0.297
News	0.37	0.404	0.433
NOX	0.167	0.444	0.222
CO	0.222	0.056	0.111

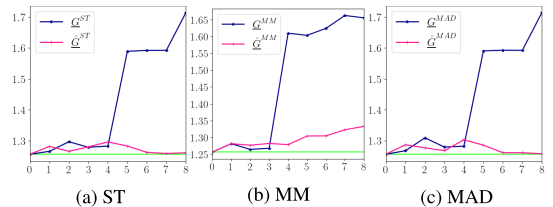
Figs. 11 to 13 depict for each dataset and each FN method the RMSE value obtained for each iteration of the FS strategy. The X-axis refers to the number of features removed at each stage of the procedure. Thus, 0 refers to the employment of the whole dataset. The Y-axis collects the RMSE value between the real and the estimated output for the training set. The blue stars depict the results obtained when the features are discarded according to  $\hat{G}^*$ ; and the pink vertical lines, the RMSE value resulting from the feature selection strategy based on  $\hat{G}^+$ . The horizontal green line represents the RMSE value reached with the complete dataset. Note that the features are removed one by one, and since the rank similarity between the contributions estimated by the traditional and the



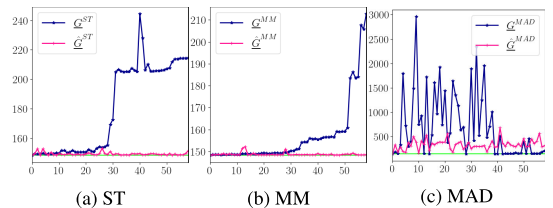
**FIGURE 11.** CBM dataset.



**FIGURE 12.** NOX dataset.



**FIGURE 13.** CO dataset.



**FIGURE 14.** News dataset.

adapted Garson's methods differs, the removed feature may not coincide at each stage of the algorithm.

When comparing CBM, NOX and CO datasets it is observed that the RMSE values resultant from the features removal based on  $\hat{G}^{ST}$  and  $\hat{G}^{MAD}$  and based on  $\hat{G}^{ST}$  and  $\hat{G}^{MAD}$  are approximately the same. This was expected from the results of Tables 10a and 10b. Nevertheless, in these cases, especially for NOX and CO datasets, the RMSE values obtained from the FS based on the adapted Garson's method are closer to the performance reached with the whole dataset, especially when increasing the number of removed features. In fact, as observed in Figs. 7a, 9a and 10a, and in Table 10b, there are significant differences between the contribution value estimated for the most influencing features and the resting ones. Consequently, in Figs. 12a to 12c and 13a to 13c it is observed that the RMSE obtained with all the features

and the RMSE obtained when utilising uniquely the most influencing features is almost the same. Furthermore, in News dataset, in Fig. 4c significant differences between  $\hat{w}^*$  for  $* \in Norm$  were observed. Consequently, in Figs. 14a to 14c by comparing the results from the FS strategy according to the traditional or the adapted Garson's methods, it is clear that each feature contribution in the model differs depending on the normalisation method employed to transform News dataset. Besides, in Figs. 14a and 14b it is observed that the RMSE error estimated at each stage of the FS strategy based on the  $\hat{C}^*$  remains closer to the RMSE obtained with all the features than the RMSE resulting from FS according to Garson's traditional method.

All in all, it is demonstrated that the dispersion factors inclusion in the features' contribution calculation significantly improves the estimation of the real features' influence on the model, as observed through the FS strategy.

## VII. DISCUSSION

As stated and demonstrated in this work, the FN method selection significantly affects the ANN-based model's performance and the inclusion of dispersion factors when estimating the features' contribution improves the understanding of the features' influence on the model.

The former point emphasises the influence of the FN method selection; however, it remains open the question about which FN to employ to transform a given dataset in order to reach the best model's performance; or even if it is recommendable the application of FN or discard the magnitude of the features by removing the  $10^{nj}$  factors from (4). As stated in Sections III and IV-B, FN imposes a dispersion weight to compress or expand the features. Thus, FN can be viewed as a Feature Weighting method that estimates the features' weights in an unsupervised manner since the dispersion factors are calculated based on the features' statistical characteristics. A weight that does not correspond to the real relative importance of a given feature can result in a performance loss. In fact, in Table 4b it is observed that for the test set, lower mean RMSE and higher  $R^2$  scores are obtained from the raw dataset with the magnitude factors removed than from any normalised dataset. Thus, further research about the suitability of the FN method selection would be interesting given the properties of a given dataset. Moreover, since this work demonstrates the influence of the FN on the network, it is evident that other preprocessing techniques may also condition the model's performance. Hence, the impact of supervised FW preprocessing methods to improve the model's performance should be investigated. Furthermore, a conjoint comparison between the supervised weights calculated with a given FW method and their similitude with the dispersion factors estimated with different FN may guide the selection of a given normalisation method to preprocess the input data.

In addition, this work analyses the weight matrix analysis-based methods to understand the features' contribution to the model. However, it would be interesting to extend the analysis to other explainability analysis approaches. The presented

results are obtained from networks with the identity activation function in the hidden and output layers. Then, further studies for network's with different activation functions are needed.

Another interesting research topic until the date in the ANN branch is the search of the optimal weights initialisation to maintain the fair initial features' contribution to the solution search space. However, in the same way that FN influences the features' contribution to the model's performance, it may be suspected that it may also condition the suitability of the initial weight configuration for a fair weights adjustment. As aforementioned, the lowest RMSE values reached by each normalised dataset are obtained with different random initialisations. Moreover, note that despite the significant differences in terms of mean MAE, RMSE and  $R^2$  based on  $*$ ; the minimum estimated errors (reached with different initialisations) in Table 4 are almost the same for each normalised dataset. Further studies about the network initialisation based on the conjoint influence of the dispersion factors and the initial weight matrix may be of great interest, which may result in a new weight initialisation strategy.

## VIII. CONCLUSION

Due to the high ability of ANN to model complex systems, these algorithms are being widely employed to solve complex problems. Simultaneously, because of the lack of explainability of the ANN, state-of-the-art focuses on bringing some understanding about the network functioning. In this field, several works aim at analysing the features' contribution to the model via weight matrices study. However, in such works, the preprocessing phase is not considered when estimating the features' contribution. This work has been theoretically proven and later experimentally validated that the dispersion factors employed to transform the input features' influence the final features' contribution to the model and the model's performance. In fact, as shown in this work, the presented proportional dispersion weights are explanatory factors of the similarity between the performance obtained by models trained with different FN methods. Then, as a conclusion of this work, it is recommended to include information about the dispersion factors to analyse the features' real contribution. In this line, this work proposes adapted Garson's and Yoon's methods that include features' dispersion factors for a more precise estimation of the features' influence on the model. Besides, a feature selection strategy is employed to analyse in terms of RMSE variations the effect of removing features according to Garson's method or the proposed adapted Garson's method. These experiments demonstrate that the RMSE results obtained when removing features according to the adapted Garson's method match the conclusions obtained from the features' contribution values. Then, the knowledge extracted from this proposal improves the understanding of the features' contribution to the model and enhances the feature selection strategy, which is fundamental in real use cases to model the problem at hand reliably.

Future work will focus on considering the conjoint comparison between the features' weights derived from supervised

FW and their similitude with the dispersion factors to guide the optimal FN method selection. Besides, the impact of FW as preprocessing technique for performance improvement and the influence of preprocessing techniques on ANNs with different activation functions will be considered in future works. Moreover, new network's initialisation approaches based on the conjoint influence of the preprocessing factors and the initial weight matrix may be an exciting future research topic.

## REFERENCES

- [1] M. H. Hassoun, *Fundamentals of Artificial Neural Networks*. Cambridge, MA, USA: MIT Press, 1995.
- [2] A. K. Jain, J. Mao, and K. M. Mohiuddin, "Artificial neural networks: A tutorial," *Computer*, vol. 29, no. 3, pp. 31–44, Mar. 1996.
- [3] O. I. Abiodun, M. U. Kiru, A. Jantan, A. E. Omolara, K. V. Dada, A. M. Umar, O. U. Linus, H. Arshad, A. A. Kazaure, and U. Gana, "Comprehensive review of artificial neural network applications to pattern recognition," *IEEE Access*, vol. 7, pp. 158820–158846, 2019.
- [4] C. M. Agu, M. C. Menkiti, E. B. Ekwe, and A. C. Agulanna, "Modeling and optimization of Terminalia catappa L. Kernel oil extraction using response surface methodology and artificial neural network," *Artif. Intell. Agricult.*, vol. 4, pp. 1–11, Jan. 2020.
- [5] M. Jalanko, Y. Sanchez, V. Mahalec, and P. Mhaskar, "Adaptive system identification of industrial ethylene splitter: A comparison of subspace identification and artificial neural networks," *Comput. Chem. Eng.*, vol. 147, Apr. 2021, Art. no. 107240.
- [6] J. Lee, Y. C. Lee, and J. T. Kim, "Migration from the traditional to the smart factory in the die-casting industry: Novel process data acquisition and fault detection based on artificial neural network," *J. Mater. Process. Technol.*, vol. 290, Apr. 2021, Art. no. 116972.
- [7] B. Li, C. Li, J. Huang, and C. Li, "Application of artificial neural network for prediction of key indexes of corn industrial drying by considering the ambient conditions," *Appl. Sci.*, vol. 10, no. 16, p. 5659, Aug. 2020.
- [8] Y. Park, M. Choi, K. Kim, X. Li, C. Jung, S. Na, and G. Choi, "Prediction of operating characteristics for industrial gas turbine combustor using an optimized artificial neural network," *Energy*, vol. 213, Dec. 2020, Art. no. 118769.
- [9] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (XAI)," *IEEE Access*, vol. 6, pp. 52138–52160, 2018.
- [10] A. Fischer, "How to determine the unique contributions of input-variables to the nonlinear regression function of a multilayer perceptron," *Ecol. Model.*, vols. 309–310, pp. 60–63, Aug. 2015.
- [11] J. D. Olden and D. A. Jackson, "Illuminating the 'black box': A randomization approach for understanding variable contributions in artificial neural networks," *Ecol. Model.*, vol. 154, nos. 1–2, pp. 135–150, 2002.
- [12] C. R. D. Sá, "Variance-based feature importance in neural networks," in *Proc. Int. Conf. Discovery Sci.* Cham, Switzerland: Springer, 2019, pp. 306–315.
- [13] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, and F. Herrera, "Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI," *Inf. Fusion*, vol. 58, pp. 82–115, Jun. 2020.
- [14] A. Eck, L. M. Zintgraf, E. F. J. de Groot, T. G. J. de Meij, T. S. Cohen, P. H. M. Savelkoul, M. Welling, and A. E. Budding, "Interpretation of microbiota-based diagnostics by explaining individual classifier decisions," *BMC Bioinf.*, vol. 18, no. 1, pp. 1–13, Dec. 2017.
- [15] K. Amarasinghe and M. Manic, "Explaining what a neural network has learned: Toward transparent classification," in *Proc. IEEE Int. Conf. Fuzzy Syst. (FUZZ-IEEE)*, Jun. 2019, pp. 1–6.
- [16] D. Janzing, L. Minorics, and P. Blöbaum, "Feature relevance quantification in explainable AI: A causal problem," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2020, pp. 2907–2916.
- [17] J. Jiménez-Luna, F. Grisoni, and G. Schneider, "Drug discovery with explainable artificial intelligence," *Nature Mach. Intell.*, vol. 2, no. 10, pp. 573–584, Oct. 2020.
- [18] J. Wang, J. Wiens, and S. Lundberg, "Shapley flow: A graph-based approach to interpreting model predictions," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2021, pp. 721–729.
- [19] B. Venkatesh and J. Anuradha, "A review of feature selection and its methods," *Cybern. Inf. Technol.*, vol. 19, no. 1, pp. 3–26, Mar. 2019.
- [20] V. Bolón-Canedo and A. Alonso-Betanzos, "Ensembles for feature selection: A review and future trends," *Inf. Fusion*, vol. 52, pp. 1–12, Dec. 2019.
- [21] Q. Al-Tashi, S. J. Abdulkadir, H. M. Rais, S. Mirjalili, and H. Alhussian, "Approaches to multi-objective feature selection: A systematic literature review," *IEEE Access*, vol. 8, pp. 125076–125096, 2020.
- [22] C. M. Bishop, *Neural Networks for Pattern Recognition*. London, U.K.: Oxford Univ. Press, 1995.
- [23] K. D. Pramanik, M. Mukhopadhyay, and S. Pal, "Big data classification: Applications and challenges," in *Artificial Intelligence and IoT: Smart Convergence for Eco-Friendly Topography*, vol. 85. Singapore: Springer, 2021, p. 53. [Online]. Available: <https://www.springer.com/gp/book/9789813363991>
- [24] Y. Kong and T. Yu, "A graph-embedded deep feedforward network for disease outcome classification and feature selection using gene expression data," *Bioinformatics*, vol. 34, no. 21, pp. 3727–3737, Nov. 2018.
- [25] O. Yucel, E. S. Aydın, and H. Sadikoglu, "Comparison of the different artificial neural networks in prediction of biomass gasification products," *Int. J. Energy Res.*, vol. 43, no. 11, pp. 5992–6003, Sep. 2019.
- [26] H. Turabieh, M. Mafarja, and X. Li, "Iterated feature selection algorithms with layered recurrent neural network for software fault prediction," *Expert Syst. Appl.*, vol. 122, pp. 27–42, May 2019.
- [27] F. Curri, G. Fiumara, and M. G. Xibilia, "Input selection methods for soft sensor design: A survey," *Future Internet*, vol. 12, no. 6, p. 97, Jun. 2020.
- [28] B. Jeong and H. Cho, "Feature selection techniques and comparative studies for large-scale manufacturing processes," *Int. J. Adv. Manuf. Technol.*, vol. 28, nos. 9–10, pp. 1006–1011, Jul. 2006.
- [29] N. L. da Costa, M. D. de Lima, and R. Barbosa, "Evaluation of feature selection methods based on artificial neural network weights," *Expert Syst. Appl.*, vol. 168, Apr. 2021, Art. no. 114312.
- [30] T. A. Folorunso, A. M. Aibinu, J. G. Kolo, S. O. Sadiku, and A. M. Orire, "Effects of data normalization on water quality model in a recirculatory aquaculture system using artificial neural network," *i-Manager's J. Pattern Recognit.*, vol. 5, no. 3, p. 21, 2018.
- [31] A. Gökhan, C. O. Güzeller, and M. T. Eser, "The effect of the normalization method used in different sample sizes on the success of artificial neural network model," *Int. J. Assessment Tools Educ.*, vol. 6, no. 2, pp. 170–192, Apr. 2019.
- [32] B. K. Singh, K. Verma, and A. Thoke, "Investigations on impact of feature normalization techniques on classifier's performance in breast tumor classification," *Int. J. Comput. Appl.*, vol. 116, no. 19, pp. 11–15, Apr. 2015.
- [33] X. A. Larriva-Novo, M. Vega-Barbas, V. A. Villagrà, and M. S. Rodrigo, "Evaluation of cybersecurity data set characteristics for their applicability to neural networks algorithms detecting cybersecurity anomalies," *IEEE Access*, vol. 8, pp. 9005–9014, 2020.
- [34] C. E. Choong, S. Ibrahim, and A. El-Shafie, "Artificial neural network (ANN) model development for predicting just suspension speed in solid-liquid mixing system," *Flow Meas. Instrum.*, vol. 71, Mar. 2020, Art. no. 101689.
- [35] N. Khare, P. Devan, C. Chowdhary, S. Bhattacharya, G. Singh, S. Singh, and B. Yoon, "SMO-DNN: Spider monkey optimization and deep neural network hybrid classifier model for intrusion detection," *Electronics*, vol. 9, no. 4, p. 692, Apr. 2020.
- [36] R. Ramani, K. V. Devi, and K. R. Soundar, "MapReduce-based big data framework using modified artificial neural network classifier for diabetic chronic disease prediction," *Soft Comput.*, vol. 24, no. 21, pp. 16335–16345, Nov. 2020.
- [37] W. Zhang, Q. M. J. Wu, Y. Yang, and T. Akilan, "Multimodel feature reinforcement framework using Moore–Penrose inverse for big data analysis," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Oct. 6, 2020, doi: 10.1109/TNNLS.2020.3026621.
- [38] Z. Shi, W. Zheng, and W. Yin, "Improving the reliability of the prediction of terrestrial water storage in Yunnan using the artificial neural network selective joint prediction model," *IEEE Access*, vol. 9, pp. 31865–31879, 2021.
- [39] D. Dua and C. Graff, "UCI machine learning repository," School Inf. Comput. Sci., Univ. California, Irvine, CA, USA, 2017. [Online]. Available: <http://archive.ics.uci.edu/ml>
- [40] S. García, J. Luengo, and F. Herrera, *Data Preprocessing in Data Mining*, vol. 72. Cham, Switzerland: Springer, 2015.

- [41] G. D. Garson, "Interpreting neural-network connection weights," *AI Expert*, vol. 6, no. 4, pp. 46–51, Apr. 1991, doi: 10.5555/129449.129452.
- [42] Y. Yoon, G. Swales, Jr., and T. M. Margavio, "A comparison of discriminant analysis versus artificial neural networks," *J. Oper. Res. Soc.*, vol. 44, no. 1, pp. 51–60, Jan. 1993.
- [43] A. Coraddu, L. Oneto, A. Ghio, S. Savio, D. Anguita, and A. Figari, "Machine learning approaches for improving condition-based maintenance of naval propulsion plants," *Proc. Inst. Mech. Eng., M, J. Eng. Maritime Environ.*, vol. 230, no. 1, pp. 136–153, 2016.
- [44] H. Kaya, P. Tüfekci, and E. Uzun, "Predicting CO and NO<sub>x</sub> emissions from gas turbines: Novel data and abenchmark PEMS," *TURKISH J. Electr. Eng. Comput. Sci.*, vol. 27, no. 6, pp. 4783–4796, Nov. 2019.
- [45] K. Fernandes, P. Vinagre, and P. Cortez, "A proactive intelligent decision support system for predicting the popularity of online news," in *Proc. Portuguese Conf. Artif. Intell.* Cham, Switzerland: Springer, 2015, pp. 535–546.
- [46] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.
- [47] L. Puka, *Kendall's Tau*. Berlin, Germany: Springer, 2011, pp. 713–715.
- [48] R. Woolson, "Wilcoxon signed-rank test," in *Wiley Encyclopedia of Clinical Trials*. Hoboken, NJ, USA: Wiley, 2007, pp. 1–3. [Online]. Available: <https://www.springer.com/gp/book/9789813363991>



**IRATXE NIÑO-ADAN** was born in Bilbao, Basque Country, in 1990. She received the bachelor's degree in mathematics and the master's degree in mathematical modeling and research, statistics, and computing from the University of the Basque Country (UPV/EHU), in 2015 and 2017, respectively. She is currently pursuing the Ph.D. degree with Tecnalia Research and Innovation. Her research interests include developing advanced techniques with regards to data analytics and Industry 4.0.



**EVA PORTILLO** received the Ph.D. degree in engineering, in 2007. She is currently an Associate Professor with the Department of Automatic Control and Systems Engineering, University of the Basque Country (UPV/EHU). She has several awards at national conferences. In 2016, she was a Visiting Professor with the Knowledge Engineering and Discovery Research Institute (KEDRI), Auckland University of Technology, Auckland, New Zealand. From 2017 to 2019, she has been the Vice-Dean of the master's and Doctoral School of the University of the Basque Country, where she is currently an Academic Secretary of the Doctoral School. She received the Prize for Outstanding Ph.D. thesis awarded by UPV/EHU, in 2010.



**ITZIAR LANDA-TORRES** received the Ph.D. degree in telecommunication engineering from the University of Deusto and the Ph.D. degree in information technology from the University of Alcalá de Henares (UAH). She has been a Researcher at Tecnalia Research and Innovation, working in artificial intelligence, data-mining, pattern analysis, neural networks, clustering, and grouping problems related to different fields of knowledge. During her ten-year-long research career, she has coauthored more than 25 scientific publications in various international journals *Applied Soft Computing*, *Engineering Applications of Artificial Intelligence*, and *Expert Systems with Applications*. She is currently the Innovation Project Manager at Petronor Innovación S.L. (Repsol).



**DIANA MANJARRES** received the Ph.D. degree in telecommunication engineering from the University of the Basque Country (UPV/EHU) and the Ph.D. degree in information technology from the University of Alcalá (UAH). She is currently a Scientific Researcher at Tecnalia Research and Innovation, working in artificial intelligence and optimization algorithms. During her ten-year-long research career, she has coauthored more than 25 scientific publications in various international journals and conferences. Her research interests include heuristic techniques for NP-hard optimization problems, multi-objective optimization, data-mining, pattern analysis, neural networks, clustering, and grouping problems related to different fields of knowledge.

...

## **IV Soft-sensor design for vacuum distillation bottom product penetration classification**

**Iratxe Niño-Adan, Itziar Landa-Torres, Eva Portillo, Diana Manjarres**

Published in *Applied soft Computing*, December 2020, volume 102,  
107072.

DOI: <https://doi.org/10.1016/j.asoc.2020.107072>.

**JCR: 6.725**

*Categories: Computer Science, Interdisciplinary Applications, 11/112,  
Q1, Computer Science, Artificial Intelligence, 24/139, Q1*





# Soft-sensor Design for Vacuum Distillation Bottom Product Penetration Classification Based on a Novel Normalization and Feature Weighting Methodology

Iratxe Niño-Adan<sup>a,b,1</sup>, Itziar Landa-Torres<sup>c</sup>, Diana Manjarres<sup>a</sup>, Eva Portillo<sup>b</sup>

<sup>a</sup>*TECNALIA, Basque Research and Technology Alliance (BRTA), 48160 Derio, Spain*

<sup>b</sup>*Department of Automatic Control and System Engineering, Faculty of Engineering, University of the Basque Country UPV/EHU, Bilbao, Spain*

<sup>c</sup>*Petronor Innovación S.L, 48550 Muskiz, Spain*

---

## Abstract

Petroleum oil refineries are complex systems that convert crude into subproducts of value. The profit of the refinery depends on the quality of the resultant subproducts, which are usually determined by a laboratory analysis called “Needle penetration”. Normally, this laboratory analysis is costly and time-consuming since entails around four hours for its accomplishment. In order to solve this limitation, this paper proposes a novel soft-sensor design for online vacuum distillation bottom product penetration classification. The design of the soft-sensor is based on a new approach, the two-stage methodology, that considers the joint effect of both Normalization and Supervised Filter Feature Weighting methods to transform the features. This methodology stands on the analysis of the real impact of applying normalization methods on the contribution of each feature, providing results that significantly differ from the traditional premises of the state-of-the-art. The analysis include the impact of normalization on distance metrics such as the Euclidean. Also, a new adaptation of Pearson correlation for the estimation of the feature weights respect to categorical labels is proposed in this work. Once the features are transformed, five well-known Machine Learning (ML) algorithms (K-means, K-NN, RFc, SVC and MLP) are considered for the design of the soft-sensor. The final soft-sensor design is selected based on the feature space transformation strategy and the ML algorithm that achieves the best results in terms of: accuracy, precision, generalization and explicability. In order to validate the proposal, real monitored data from a petroleum refinery plant sited in The Basque Country is employed. Results show that the proposed two-stage methodology improves the results obtained by the Normalization methods.

*Keywords:* Petroleum Refinery, Needle Penetration, Classification, Soft-sensor, Feature Normalization, Feature Weighting

---

*Email address:* [iratxe.nino@tecnalia.com](mailto:iratxe.nino@tecnalia.com) (Iratxe Niño-Adan)

*Preprint submitted to Applied Soft Computing*

*December 28, 2021*

## 1. Introduction

In recent years, the Industry 4.0 era refers to the current trend of high automation, monitorization and data exchange of the systems involved in manufacturing technologies. In this context, petroleum oil refineries, composed of chemical unit processes that convert crude into products of value, are also being monitored with the aim at optimizing their operational conditions and improving the quality of the final fractional distillation products. In fact, the benefit for the refinery industry from commercializing them depends on a highly complex system in which the properties of the crude and the operational variables are of outstanding importance along the distillation process.

In particular, a widely extended method for controlling the quality of the vacuum distillation unit bottom product is the “Needle penetration”, a laboratory analysis conducted over a sample according to the norm UNE-EN 1426:2015 (bitumen and bituminous binders. Determination of Needle penetration) [1]. In practice, it is not performed following a predefined schedule and the results are available after four hours, which entails a significant delay in knowing the penetration quality of the product with the consequent economic loss for the refinery. This drawback would be solved with the deployment of a soft-sensor. However, for our sake of knowledge no soft-sensors for the penetration quality estimation have been proposed in the literature until the date.

In the last decade, several researches based on Machine Learning algorithms (ML) have been conducted for solving different open challenges in the refinery industry. Some related examples are: the optimization of the total refinery profit [2], the optimal scheduling for crude oil loading and unloading [3] or steal injection [4], multi-level production planning [5], fault [6] or leakage detection system [7] and maintenance decision of petrochemical plant [8].

Nevertheless, most of the petroleum oil refinery process related works delve into the development of soft-sensors approaches capable of inferring relevant magnitudes at different parts of the distillation process based on some measured data and important operational variables. Among them, works focused on soft-sensors design for the estimation of butane concentration can be found in the literature [9, 10, 11]. Other works deal with the design of soft-sensors for the estimation of the toluene content [12], oxygen content prediction in an industrial coke furnace [13], or  $H_2S$  and  $SO_2$  acid gases concentration in a Sulfur Recovery Unit (SRU) [14].

Regarding soft-sensors for the analysis of the quality of the obtained subproducts, several works have been proposed in the last years. The work presented in [15] aims at inferring the results of the laboratory analysis of the quality of fractional distillation products, such as kerosene and light and heavy diesel fraction according to the international ASTM D86 standard. It employs a Neural Network (NN), with a Genetic Algorithm (GA) for the optimization of the weights and the NN structure. Authors in [16] propose an online prediction method for some quality parameters of the distillation process, such as: the temperature of heavy naphtha, kerosene and gas oil (ASTM D96) and the Abbel

inflammability analysis factor through a technique named eXtended Evolving Fuzzy Takagi-Sugeno (exTS). Similarly, in [17] the authors propose a soft-sensor based on Partial Least Square (PLS) regression in combination with a GA-based Feature Selection strategy to select the most relevant process variables for operation and quality control of the ASTM 90% distillation temperature (D90) in a crude distillation unit. Likewise, [18] presents a Bootstrap Aggregated model for the estimation of the kerosene dry point.

However, a common problem reported along the works that employ real industrial data is that the collected datasets are “data rich but information poor” [19]. Then, special attention to the representation of the input space has been considered in different works of the literature. In order to extract relevant information from the collected data and ensure the ML model reliability, a preprocessing step is commonly needed. However, in case redundant or noisy features are included, the model cannot be able to reflect the industrial process reality. Consequently, Feature Selection (FS) techniques [20] are usually utilized as a preprocessing step to discard the non-informative features.

Regarding the application of FS methods in the refinery process, authors in [21] obtain an improved gasoline dry point prediction accuracy by means of eliminating the influence of the widespread outliers of the operation data. Likewise, in order to extract the relevant features, the work presented in [22] employs a Recurrent Denoising Autoencoder (RDAE) and a Cumulative Percent Variance (CPV). Then, the authors propose a Weighted AutoRegressive Long Short Term Memory (WAR-LSTM) structure which is applied to a Fluid Catalytic Cracking unit to estimate the flow rate and the yield of some resultant subproducts: gasoline, diesel oil, coke and liquefied petroleum gas. In the preprocessing step of [23], authors employ a Robust Partial Least Square (RPLS) method to identify the multivariate anomalies and analyze the model performance based on different sampling intervals. Then, a Dynamic Partial Least Square model (DPLS) selects and optimizes the dynamic soft-sensing input variables that best estimate the naphtha dry point from an atmospheric-vacuum distillation tower. Finally, aiming at predicting the kerosene D95 in a crude Distillation Unit (CDU), authors in [24] present a double LASSO algorithm integrated into a MLP model that selects the most sensitive features respect to the output value, avoiding redundancy problems with some correlated features.

Apart from FS techniques, Feature Weighting (FW) methods are a widely employed preprocessing step to transform the input space and reflect the real importance of each feature. Indeed, it obtains more accurate results than selecting the features with highest relative importance [25]. However, within the scope of the refinery process, the only work found in the literature that considers FW is [9]. The aim of the authors is to estimate the butane concentration at the bottom of a debutanizer column by identifying first the relevant features by correlation analysis and then, by assigning weights to the features accordingly to this correlation output.

In addition to FS and FW, Normalization is also a widely employed preprocessing technique that aims at eliminating magnitude differences among

the features in order to equalize their contribution on the ML algorithm calculations. In the state of the art, plenty normalization methods are available, and diverse experimental analysis have been conducted to select the most suitable one for different scenarios [26, 27, 28] but consensus has not been found. In particular, the work presented in [28] experimentally investigates the impact of 14 normalization methods considering FS and FW. The authors employ Ant Colony algorithm for selecting the relevant features and finding the optimal weights in a wrapper manner in order to increase the K-NN classification accuracy over 14 synthetic and real datasets from UCI repository. Based on the accuracy results obtained over the datasets it is concluded that mean and standard deviation are more suitable than min-max and median measures to normalize the data and that, in conjunction with FS and FW, data normalization affects the outcomes in terms of accuracy. However, in [28] it is not analyzed the transformed normalized space, the normalized weighted space or the features influence on the classification of the samples, so as the conclusions are based only on the performance results. In this regard, in contrast to [28], this research establishes and validates conceptual hypotheses about the effect of the normalization on transforming the input space and over the Supervised Filter Feature Weighting methods application with special attention to the resulting transformed space and its influence on the modeling. This way this work aims at strengthening a general understanding about the influence of applying normalization methods. More concretely, in the ML scope this work:

- Analyzes the effect of normalization and feature weighting methods on transforming the dataset based on the following hypotheses: 1) Each normalization method transforms differently a given dataset, 2) a given normalization method transforms differently the features of a given dataset, and then, 3) the normalization method modifies the final applied relevance of the features estimated by Feature Weighting methods.
- The verification of these hypotheses not only affects the way the features of a dataset are transformed, but also their impact on the ML algorithm. Then, based on the obtained conclusions, a new two-stage methodology that combines normalization and feature weighting is proposed in order to intelligently transform the input space.
- In addition, a new Feature Weighting method called “Adapted Pearson correlation” is presented, specifically designed to estimate the relative importance of real-valued features respect to categorical labels in terms of class separability.

From the application perspective, this work presents a real use case of a petroleum refinery plant sited in The Basque Country, utilized to experimentally:

- Validate the conclusions from the presented hypotheses.
- Advance over the state of the art by proposing a soft-sensor for the online penetration quality standard classification. In order to design, develop

135 and validate the soft-sensor, real-time monitored data regarding crude  
 properties and operational variables are employed.

- Demonstrate the validity of the proposed two-stage methodology  
 which achieves the best accuracy, precision, generalization ability and  
 interpretability for the above-mentioned penetration quality classification  
 140 problem.

The remaining of the paper is structured as follows: Section 2 presents an  
 outline of the crude refining process. Section 3 depicts the hypotheses about  
 the normalization influence on transforming the input space and over the FW  
 methods application, along with the proposed adaptation of Pearson correlation  
 145 for FW. Besides, the steps followed for the design and development of the  
 soft-sensor for the penetration quality standard classification of the vacuum  
 distillation unit bottom product are also described in Section 3. Classification  
 results and further analysis of the normalization effect over the input feature  
 space are shown in Section 4. Finally, Section 5 depicts the conclusions and the  
 150 future work.

## 2. Crude Refining Process Description

As it is remarked in Section 1, industrial petroleum refineries are complex  
 distillation systems, composed by chain units where chemical reactions aim at  
 converting crude in high quality subproducts.

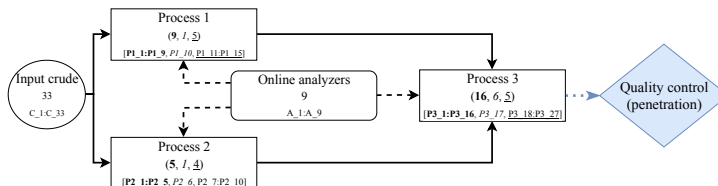


Figure 1: Flow diagram of the crude refining process. At each process unit, the features related to flow, pressure and temperature are remarked with bold, italic and underlined text, respectively.

155 Figure 1 depicts a high-level flow diagram of the crude refining process.  
 Firstly, the input crude properties, described in the dataset by 33 features  
 (C.1:C.33), determined by experts in the field and updated every six months  
 – or with a higher frequency if major changes regarding the use of new crude  
 occurs – is injected into Process 1 and Process 2 where the chemical and physical  
 160 reactions start the refining process. From Process 1 and 2 the processed crude  
 is pumped into the Process 3, where the refining process continues. The behavior  
 of the Process 1, 2 and 3 is shown through 15 (P1.1:P1.15), 10 (P2.1:P2.10)  
 and 27 (P3.1:P3.27) features, respectively, that are continuously monitoring the  
 development of the industrial system, providing information of flow, pressure or

165 temperature. Along all the units of the chain system, 9 online chemical quality  
analyzers (A1.1:A.9) monitor the process development.

The information of the crude refining process is collected in a dataset,  
consisting of 33485 samples described by the mentioned 94 features recorded  
every 15 minutes during 349 days.

170 The interest of the refinery is to maintain a high penetration quality of the  
bottom product of the vacuum distillation unit located at the end of Process 3.  
In order to control the quality specifications, some samples are analyze in the  
laboratory according to [1]. Depending on the result, the samples are classified  
into two groups: the group 0 represents the samples that fulfill penetration  
175 quality standards, and the group 1 refers to the samples that do not meet the  
constraints. In practice, if a sample does not meet the designed quality standard,  
the plant operators adjust the operational variables aiming at correcting the  
outcoming quality at the end of the process.

### 180 3. Design and Development of the Vacuum Distillation Bottom Product Penetration Soft-sensor

As stated above, the aim of the soft-sensor is to estimate the quality of  
the vacuum distillation bottom product resultant from Process 3 in terms of  
penetration, considering the 94 features collected. In particular, this work  
proposes a new two-stage methodology based on Normalization and Supervised  
185 Filter FW methods with specific contributions respect to (a) the analysis of the  
impact of normalization methods, (b) the adaptation of Pearson correlation for  
the estimation of the feature weights respect to categorical labels, and (c) the  
soft-sensor design, which integrates the proposed two-stage methodology and  
ML algorithms. With that purpose, first the hypotheses about the impact  
190 of normalization methods are presented in Subsection 3.1. The proposed  
methodology for the design of the soft-sensor is described in Subsection 3.2.

#### *3.1. Hypotheses about the Impact of Linear Normalization Methods*

As mentioned in Section 1, Normalization is widely employed in the  
preprocessing step in order to equalize the magnitude of the features that  
195 compose the dataset. However, this work demonstrates that the normalization  
methods do not totally equalize the contribution of each feature and that the  
selection of the normalization technique affects the final algorithm performance.

In this work, three different normalization methods are applied:  
Standardization (ST), Min-max normalization (MM) and MAD normalization  
200 (MAD). All of them are linear transformations of the features based on standard  
deviation  $\sigma$ , range or MAD dispersion statistic, respectively. The mentioned  
normalization methods are further described in Section 3.2.

In the following points, the hypotheses and foundations to justify the impact  
of the application of Linear Normalization methods are described below.

- 205 1. Different normalization methods transform differently the same dataset:  
In general, normalization methods are thought to equalize the contribution

of the features in the model. However, the selected normalization methods transform each feature of the dataset in order to obtain a dispersion statistic equal to one. This means that each normalization method transforms the dataset to equalize a dispersion statistic along the features of the dataset, and, as each normalization method works with a different dispersion statistic, each normalization method transforms differently the same dataset.

2. The same normalization method transforms differently the features of the same dataset: As remarked in Hypothesis 1, each normalization method equalizes a dispersion statistic along all the features of the dataset. However, in this transformation, the features that originally present high dispersion statistic value are compressed while the features with originally low dispersion statistic are expanded. This means that, given the equalization of a dispersion statistic along the features of the dataset, each feature is affected differently.

3. Supervised Filter Feature Weighting methods estimate the relative importance of each feature respect to the labels. In general, given  $m$  features, the calculated weights  $w_j$  are presented in such way that  $\sum_{j=1}^m w_j = 1$ . Since normalization methods are commonly applied in ML algorithms and, from Hypotheses 1 and 2, they affect the initial magnitudes of the features, the selected normalization method influences the applied relevance of the features. This means that the factors applied to the ML algorithm, resultant of the joint employment of normalization and FW, can be significantly different from those independently computed by the Supervised Filter FW method.

In Figure 2, an example of the influence of the normalization factor is depicted.

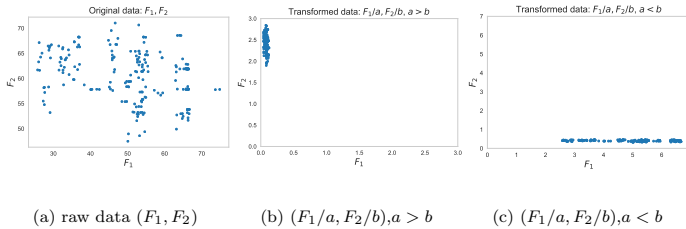


Figure 2: Linear transformations effect over the raw features  $F_1, F_2$ .

Given a dataset composed by the features  $F_1$  and  $F_2$  represented in Figure 2a, the application of the above-mentioned normalization techniques involves to compress or to expand each feature by the factors  $a, b$ , calculated for  $F_1$  and  $F_2$ , respectively. Then, Figure 2b depicts the transformed features after applying such factors when the dispersion statistic of  $F_1$  is much lower than the one

calculated for  $F_2$ . Therefore, if  $a > b$ , the values of feature  $F_1$  are compressed around a point in the  $X$  axis, while the values in  $F_2$  presents more expansion along the  $Y$  axis. Contrary, if  $a < b$ , feature  $F_2$  present more compression along the  $X$  axis than the presented by  $F_2$  in  $Y$  axis (Figure 2c).

Furthermore, in order to illustrate the impact of the normalization, the Euclidean distance  $d_E$ , commonly employed in several ML algorithms, is considered. Given  $\mathbf{x}_i, \mathbf{x}_h \in (F_1, F_2)$  two samples of the original dataset, and  $\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_h \in (\tilde{F}_1, \tilde{F}_2)$  the same two samples in the transformed dataset  $(\tilde{F}_1, \tilde{F}_2)$ , the Euclidean distance between  $\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_h$  is expressed as:

$$\begin{aligned} d_E^2(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_h) &= (\tilde{x}_i^{F_1} - \tilde{x}_h^{F_1})^2 + (\tilde{x}_i^{F_2} - \tilde{x}_h^{F_2})^2 = \\ &= (1/a^2) \cdot (x_i^{F_1} - x_h^{F_1})^2 + (1/b^2) \cdot (x_i^{F_2} - x_h^{F_2})^2 \end{aligned} \quad (1)$$

Therefore, if  $a < b \rightarrow 1/a > 1/b$  and, consequently,  $F_1$  presents more influence on calculating the distance between the samples. Contrary, if  $a > b \rightarrow 1/a < 1/b$  and consequently  $F_2$  dominates the calculations in this case.

Definitively, all the explained transformations in Hypotheses 1 and 2 over the features can lead to features contributing more than others, and without a prior knowledge of the feature relevance, no conclusions of which method is better can be taken. Furthermore, from Equation 1 it can be concluded that feature normalization can be considered as an unsupervised feature weighting method in which each feature is transformed based on a feature's statistic, instead of its true relative importance on estimating the output.

Regarding the Hypothesis 3, the application of a filter FW method in the previous example transforms Equation 1 to:

$$w_1^2 \cdot (1/a^2) \cdot (x_i^{F_1} - x_h^{F_1})^2 + w_2^2 \cdot (1/b^2) \cdot (x_i^{F_2} - x_h^{F_2})^2 \quad (2)$$

where  $w_1, w_2$  are the estimated weights for the features  $F_1, F_2$  respectively.

In the case that  $w_1 > w_2$ , it would be desirable the first component to present higher contribution in the Euclidean distance calculation. However, if  $w_1/a < w_2/b$ , the second component would predominate the calculations in Equation 2. Henceforth, the selection of the normalization method influences the applicability of the FW method and its impact on the ML algorithm, as described in Hypothesis 3.

### 3.2. Soft-sensor Design and Development based on the Proposed Normalization and Feature Weighting Methodology

In order to create a reliable soft-sensor for the penetration quality classification, this work analyzes according to the hypotheses presented in Section 3.1 the implications of the normalization methods for transforming the original feature space, their impact on the application of the FW methods and their effect over the ML algorithms calculations. This way, it will be possible to select the best transformation of the original data that obtains the highest



275 accuracy in the developed soft-sensor with special attention to the features  
 280 influence on the created model.

With that goal, Figure 3 depicts the proposed two-stage methodology for  
 intelligently transforming the original dataset aiming at capturing the relative  
 importance of the features on estimating the output and thus, creating the  
 soft-sensor.

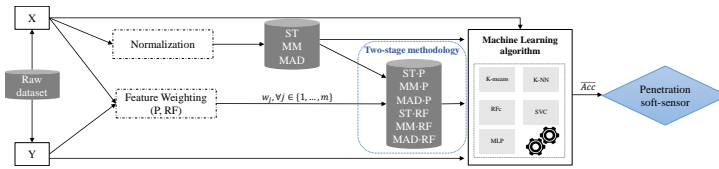


Figure 3: Flow chart of the soft-sensor design based on the proposed two-stage methodology.

### 3.2.1. Normalization

Firstly, in order to analyze the normalization effect and to validate  
 hypotheses 1 and 2 from Section 3.1, three widely employed Normalization  
 methods [26, 29] are selected for performing the features normalization:

- 285 • Standardization:  $ST_j = (X_j - \bar{X}_j) / \sigma_j \quad \forall j \in \{1, \dots, 94\}$ . The resultant  
 features are centered around the mean with standard deviation equal to  
 1. Although this method is affected by the presence of outliers, some  
 soft-sensor proposals [24] apply it in the preprocessing of the features.
- 290 • Min-max:  $MM_j = (X_j - \min(X_j)) / (\max(X_j) - \min(X_j)), \quad \forall j \in$   
 $\{1, \dots, 94\}$ . The samples of the resulting features take values between  $[0, 1]$ .  
 Despite the technique is highly influenced by the presence of outliers, this  
 method is claimed as the most recommended by authors in [26]. It has  
 also been employed in the field of soft-sensors [15].
- 295 • MAD:  $MAD_j = (X_j - \text{Me}(X_j)) / \text{Me}(|X_j - \text{Me}(X_j)|), \quad \forall j \in \{1, \dots, 94\}$ .  
 In contrast to the other techniques, MAD normalization is considered a  
 robust transformation [30] of the features because the statistics employed  
 for the transformation are not affected by the presence of outliers.

### 3.2.2. Feature Weighting (P, RF)

300 Secondly, in order to estimate the relative importance of each feature on  
 estimating the output and to validate the hypothesis 3 stated in Section 3.1 two  
 different Filter Feature Weighting methods are utilized for the calculation of  
 the weights  $w_j, \forall j \in \{1, \dots, 94\}$ : the proposed adaptation of Pearson correlation  
 and the well-known Random Forest algorithm.

- 305 • Novel Adapted Pearson Correlation (P): In order to calculate the Pearson  
 correlation coefficient between a continuous and a categorical feature,  
 where the categories have no meaning beyond the representation of the

distinct group membership, an adaptation of the Pearson correlation method is proposed in this paper. Given  $K$  the number of classes,  $Y \in \{0, \dots, K-1\}$  the original labels assigned to the samples  $\mathbf{x}_i$  310  $\forall i \in \{1, \dots, n\} = I$  of the dataset  $X$ . The centroids of the different classes can be encoded as  $Y' \in \{\{\mathbf{x}_i | y_i = 0\}, \dots, \{\mathbf{x}_i | y_i = K-1\}\} = \{\mathbf{c}_0, \dots, \mathbf{c}_{K-1}\}$  with  $X, Y' \subset \mathbb{R}^{n \times m}$ . With the described transformation, the Pearson correlation coefficient between each feature of the dataset and the corresponding component of the centroids results on:

$$\rho(X_j, Y'_j) = \frac{\sum_{i=1}^n (x_{ij} - \bar{X}_j)(y'_{ij} - \bar{Y}'_j)}{\sqrt{\sum_{i=1}^n (x_{ij} - \bar{X}_j)^2} \sqrt{\sum_{i=1}^n (y'_{ij} - \bar{Y}'_j)^2}} \quad (3)$$

Based on the transformation of the labels, 315

$$\bar{X}_j = \frac{\sum_{i=1}^n x_{ij}}{n} = \frac{\sum_{k=0}^{K-1} n_k \sum_{i \in I, y_i=k} x_{ij} / n_k}{n} = \frac{n_0 c_{0j} + \dots + n_{K-1} c_{K-1j}}{n} = \bar{Y}'_j \quad (4)$$

where  $n_k$  is the number of samples belonging to the class  $k$ , and, both,  $\bar{X}_j, \bar{Y}'_j$ , can be interpreted as the pondered mean of the centroids, weighted by the proportion of samples belonging to each cluster. Then, employing Equation 4 and adding by classes, the numerator can be rewritten as:

$$\begin{aligned} & \sum_{k=0}^{K-1} \sum_{i \in I, y_i=k} (x_{ij} - \bar{X}_j)(y'_{ij} - \bar{Y}'_j) = \\ & = \sum_{k=0}^{K-1} (c_{kj} - \bar{X}_j) \sum_{i \in I, y_i=k} (x_{ij} - \bar{X}_j) = \\ & = \sum_{k=0}^{K-1} (c_{kj} - \bar{X}_j) (\sum_{i \in I, y_i=k} x_{ij}) - n_k \bar{X}_j = \\ & = \sum_{k=0}^{K-1} n_k (c_{kj} - \bar{X}_j)^2 \end{aligned} \quad (5)$$

Similarly, in the denominator 320

$$\sqrt{\sum_{i=1}^n (y'_{ij} - \bar{Y}'_j)^2} = \sqrt{\sum_{k=0}^{K-1} n_k (c_{kj} - \bar{X}_j)^2} \quad (6)$$

Applying Equations 5 and 6, Equation 3 results on:

$$\begin{aligned} \rho(X_j, Y'_j) & = \frac{\sum_{k=0}^{K-1} n_k (c_{kj} - \bar{X}_j)^2}{\sqrt{\sum_{i=1}^n (x_{ij} - \bar{X}_j)^2} \sqrt{\sum_{k=0}^{K-1} n_k (c_{kj} - \bar{X}_j)^2}} = \\ & = \sqrt{\frac{\sum_{k=0}^{K-1} n_k (c_{kj} - \bar{X}_j)^2}{\sum_{i=1}^n (x_{ij} - \bar{X}_j)^2}} = \sqrt{\frac{\sum_{k=0}^{K-1} n_k (c_{kj} - \bar{X}_j)^2}{\frac{n}{n} \sum_{i=1}^n (x_{ij} - \bar{X}_j)^2}} = \\ & = \frac{\sqrt{\sum_{k=0}^{K-1} \frac{n_k}{n} (c_{kj} - \bar{X}_j)^2}}{\sigma_j} = \sqrt{\sum_{k=0}^{K-1} \frac{n_k}{n} \left( \frac{c_{kj}}{\sigma_j} - \frac{\bar{X}_j}{\sigma_j} \right)^2} \end{aligned} \quad (7)$$

In view of Equation 7, for each feature  $j \in \{1, \dots, m\}$ , the Novel Adapted Pearson Correlation coefficient can be interpreted as the weighted Euclidean distance, weighted by the proportion of samples of each class  $n_k/n$ , between the vector of the centroids  $[c_{0j}, \dots, c_{K-1j}]$  and a vector of length  $K$  composed by the pondered mean of the centroids  $[\bar{X}_j, \dots, \bar{X}_j]$ , 325 both calculated over the standardized feature  $X_j$ . Hence, the higher the

absolute correlation coefficient value, the higher the separation between the centroids of the K classes and the more informative the feature is for classifying the samples. Thus, the proposed encoding of the labels allows to employ the Pearson correlation coefficient as a measure of the separability of the classes. In particular, if  $K=2$ , given that  $n = n_0 + n_1$  and  $n\bar{X}_j = n_0c_{0j} + n_1c_{1j}$ , Equation 7 can be rewritten as:

$$\begin{aligned}
\rho(X_j, Y'_j)_{K=2} &= \frac{n}{n} \sqrt{\frac{\sum_{k=0}^1 \frac{n_k}{n} (c_{kj} - \bar{X}_j)^2}{\sigma_j^2}} = \sqrt{\frac{\sum_{k=0}^1 n \cdot n_k (c_{kj} - \bar{X}_j)^2}{n^2 \sigma_j^2}} = \\
&= \sqrt{\frac{n_0(c_{0j} - \bar{X}_j)(nc_{0j} - n\bar{X}_j) + n_1(c_{1j} - \bar{X}_j)(nc_{1j} - n\bar{X}_j)}{n^2 \sigma_j^2}} = \\
&= \sqrt{\frac{n_0(c_{0j} - \bar{X}_j)(n_1c_{0j} - n_1c_{1j}) + n_1(c_{1j} - \bar{X}_j)(n_0c_{1j} - n_0c_{0j})}{n^2 \sigma_j^2}} = \\
&= \sqrt{\frac{n_0 \cdot n_1 (c_{0j} - c_{1j}) ((c_{0j} - \bar{X}_j) - (c_{1j} - \bar{X}_j))}{n^2 \sigma_j^2}} = \\
&= \sqrt{\frac{n_0 \cdot n_1 (c_{0j} - c_{1j})^2}{n^2 \sigma_j^2}} = \sqrt{\frac{n_0 \cdot n_1}{n^2} \frac{(c_{0j} - c_{1j})}{\sigma_j}} \quad (8)
\end{aligned}$$

Then, it can be noticed that the point-biserial correlation [31] can be taken as a particular case of the proposed Adapted Pearson Correlation for the case  $K = 2$ . The final weight for each feature is computed as its absolute value divided by the sum of all the features' weights. Hence, the sum of the resulting weights is equal to one.

- Random Forest classifier (RF) [32]: The features' relevance is calculated as the mean of the relevance, in terms of Mean Decrease Gini, obtained from 30 MonteCarlo simulations where a naïve Random Forest – with 100 estimators and without limit in the depth of the trees – classifies the training samples.

### 3.2.3. Application of the Two-stage Methodology

As depicted in Figure 3, the proposed two-stage methodology consists in multiplying each normalized feature by the weight factor estimated by the FW method. Consequently, six final datasets are calculated as  $ST \cdot P = w_j^P \cdot ST_j, \dots, MAD \cdot RF = w_j^{RF} \cdot MAD_j$ , respectively, for  $j \in \{1, \dots, m\}$ .

### 3.2.4. Soft-sensor Development: Machine Learning Algorithms

As shown in Figure 3, once the 10 datasets are obtained (raw, ST, MM, MAD, ST-P, MM-P, MAD-P, ST-RF, MM-RF and MAD-RF), five of the most commonly ML algorithms are employed to develop the soft-sensor, i.e. K-means, K-Nearest Neighbors (K-NN), Random Forest (RFc), Support Vector Classification (SVC) and MultiLayer Perceptron (MLP) with one hidden layer. In contrast to their counterparts, K-means is a clustering algorithm that splits the samples in different groups. In this work, each group is understood as a resultant penetration quality class and a label (0 or 1) is assigned considering

the class membership of the samples of each group. That being so, the accuracy of the model for comparing the group membership resulting from K-means and  
360 the real class of the samples is maximized.

Finally, the accuracies ( $\overline{Acc}$ ) statistics are obtained and the final soft-sensor is selected among the obtained results.

#### 4. Results

The design and selection of the soft-sensor based on the proposed two-stage  
365 methodology described in Figure 3 is conducted in Subsection 4.1. In addition, an analysis of the normalization effect on the features transformation and on the application of the weights is conducted for the selected soft-sensor in Subsection 4.2. This way, the hypotheses of Subsection 3.1 and the applicability of the two-stage methodology for increasing the accuracy of the final soft-sensor are  
370 validated. Finally, the knowledge extracted from the selected soft-sensor is illustrated in Subsection 4.3.

##### 4.1. Analysis of the Results and Selection of the Soft-sensor

As described in Section 1, the main purpose of this work is to develop a  
375 real-time soft-sensor able to classify the penetration quality based on 94 features of the refinery process. The samples are categorized as 0 if the obtained quality meets the designed standard; otherwise, the sample is labeled as 1. It must be remarked that several laboratory analysis are done whenever a change in the process can result on not meeting the designed standard quality.

The raw dataset  $X$  contains 268 labeled samples from which 160 correspond  
380 to the designed standard quality samples labeled as 0. Aiming at simulating the practice of the off-line model training and the on-line quality prediction procedure, the original chronological order of data is considered for the training and test sets selection. Thus, the first 90% of these samples are selected as the training set, while the remaining 10% are employed as test set. Following the  
385 two-stage methodology presented in Subsection 3.2, both the normalization and the FW are applied. For statistical significance, both the labeled and unlabeled samples of the training set are employed in the normalization step, while for the supervised weights calculation, only the labeled ones are utilized.

Then, the proposed ML algorithms are applied to the obtained 10  
390 transformed datasets. In order to maximize the accuracy, the GridSearch (GS) algorithm is applied to obtain the best hyper-parameters among the following values. In the case of K-NN algorithm, the number of neighbors varies between [5, 6, 7, ..., 40] (35 options). For RfC, the GS searches the best performance among the number of estimators = [5, 10, 15, 20, 25, 30, 40, 50, 60, 70, 80, 90, 100]  
395 (13 possibilities). In the case of MLP, the GS searches among the combinations of the following activation functions  $\in$  {identity, logistic, relu} and the number of neurons in the hidden layer equal to  $\in$  {1, 2, 3, ..., 10} (30 combinations). Finally, for SVC, the GS algorithm combines the following values:  $C \in$  [10, 1, 0.1, 0.01, 0.001],  $\gamma \in$  [10, 1, 0.1, 0.001, 0.0001] and the kernel functions

400 [linear, rbf, sigmoid] (75 combinations for SVC). Obviously, K-means does not require any hyper-parameter optimization since  $K$  is equal to 2.

Table 1 presents the hyper-parameters selected by the GS algorithm for the proposed ML algorithms.

		Hyper-Parameters						
		K-NN		RFc	SVC			MLP
BD		n neighbors	n estimators	C	$\gamma$	kernel	activation	neurons
raw		13	70	10	10	linear	logistic	8
ST		37	70	0.01	10	linear	logistic	1
MM		5	70	10	0.01	rbf	logistic	10
MAD		10	70	1	0.01	sigmoid	logistic	4
ST-P		21	90	10	10	linear	logistic	10
MM-P		15	90	10	10	rbf	logistic	7
MAD-P		11	70	10	10	rbf	logistic	2
ST-RF		17	70	10	10	linear	logistic	10
MM-RF		23	70	10	10	rbf	logistic	10
MAD-RF		5	70	10	10	linear	logistic	3

Table 1: Best hyper-parameters obtained by the GridSearch algorithm for the proposed ML algorithms.

405 Table 2 presents the accuracy statistics results (%) obtained for the different ML algorithms with the selected hyper-parameters for each dataset (Table 1).

		raw	Normalization			Two-stage methodology					
			ST	MM	MAD	ST-P	MM-P	MAD-P	ST-RF	MM-RF	MAD-RF
K-means	mean	51.852	70.37	70.37	40.741	70.37	<b>81.407</b>	40.741	70.37	77.778	40.741
	std	0	0	0	0	0	0.519	0	0	0	0
	max	51.852	70.37	70.37	40.741	70.37	81.481	40.741	70.37	77.778	40.741
	min	51.852	70.37	70.37	40.741	70.37	77.778	40.741	70.37	77.778	40.741
K-NN	acc	70.37	77.778	77.778	77.778	<b>81.481</b>	<b>81.481</b>	77.778	<b>81.481</b>	<b>81.481</b>	77.778
RFc	mean	52.185	52.185	52.185	53.334	52.741	74.37	52.778	52.926	<b>74.445</b>	52.778
	std	3.535	3.535	3.535	4.709	3.652	6.132	4.479	4.188	5.803	4.479
	max	59.259	59.259	59.259	77.778	66.667	77.778	70.37	66.667	77.778	70.37
	min	44.444	44.444	44.444	44.444	44.444	55.556	44.444	44.444	55.556	44.444
SVC	acc	51.852	<b>81.481</b>	<b>81.481</b>	77.778	77.778	77.778	70.37	<b>81.481</b>	<b>81.481</b>	74.074
MLP	mean	51.667	67.259	65.444	64.519	74.815	<b>77.37</b>	66.852	74.852	67.555	63.889
	std	14.372	8.5	11.018	4.971	6.317	12.839	3.261	7.884	10.622	5.918
	max	81.481	81.481	85.185	74.074	85.185	85.185	74.074	88.889	85.185	70.37
	min	29.63	44.444	40.741	44.444	59.259	40.741	40.741	51.852	55.556	37.037

Table 2: Accuracy statistics (mean, std, max, min) obtained in 100 MonteCarlo simulations for each dataset and ML algorithm. Since K-NN and SVC are deterministic only the accuracy value (acc) is shown.

410 As observed in Table 2, the proposed two-stage methodology outperforms or at least equalizes the maximum and mean accuracy results obtained for the raw and the normalized datasets for each ML algorithm. Comparing the best mean accuracy values obtained by the proposed two-stage methodology and a normalization method, there is an improvement approximately of 11% in the case of K-means, 4% in K-NN, 21% in RFc and 10% in the MLP algorithm. For the SVC the two-stage methodology equalizes the obtained mean accuracy by the normalization techniques. Similarly, when focusing on the maximum accuracy, the two-stage methodology outperforms in approximately 11%, 4%, and 3% the maximum accuracy obtained by the normalized datasets for K-means, K-NN and MLP, respectively. RFc and SVC present the same

maximum accuracy by the normalization techniques and by the proposed two-stage methodology. In overall, the convenience of the employment of the two-stage methodology is validated with this real case problem.

Likewise, the mean accuracy values obtained by K-means with MM-P, K-NN with ST-P, MM-P, ST-RF and MM-RF, and SVC with ST-RF and MM-RF is around 81.4%.

Once validated that the proposed two-stage methodology equalizes or outperforms the results obtained by the normalized datasets for all the presented ML algorithms, the soft-sensor is selected according to: (a) the stability of the soft-sensor, (b) the principle of the parsimony, (c) the explicability and interpretability of the soft-sensor, and (d) the precision of the soft-sensor:

(a) As depicted in Table 2, the maximum accuracy reached by K-means applied over MM-P is 81.481% and the standard deviation (std) is 0.519%, much lower than those obtained by MLP or RFc algorithms. Besides, such std value is low enough to maintain K-means as an option for the soft-sensor, along with K-NN and SVC that also achieve similar accuracy values.

(b) The principle of the parsimony or Occam's razor says that "given two explanations about the data, in equal conditions, the simplest one is preferable" [33]. The advantages of applying this criterion are specially important in terms of generalization capability and computation and memory resources requirements. First, it can be noticed that SVC depends on the configuration of more hyper-parameters than K-NN and K-means. In the case of K-NN, the number of neighbors selected by GS for the different transformed datasets varies from 15 to 23. In addition, K-NN algorithm needs higher memory resources to keep all the historical samples and search the nearest neighbors to classify a new sample. In contrast, K-means assigns the new sample to the closest cluster based only on the Euclidean distance to the centroids.

(c) Additionally to the generalization capabilities and the computational and memory cost, the ML algorithms in Table 2 are sorted based on their explicability and interpretability [34], being K-means the most interpretable algorithm and the MLP the less one. Moreover, in K-means, the assignment of a new sample to any of both penetration quality groups is done based on the sample's distance to the centroids. Therefore, the soft-sensor user would be able to interpret in which features the sample is distant to the normal group, thus, obtaining information about which part of the system is affecting the process in order to correct it.

(d) In addition to the accuracy reached by the soft-sensor, it must be considered that, if the resulting subproducts present a desired quality but are wrongly classified, the change of the operational conditions may result on a lost of quality and economical profit. So, as important as the accuracy of the model is the precision for classifying correctly the samples that do not meet the designed standard quality in order to change the process only when there is certainty that a correction is needed. Thus, taking the group '1' as the positive samples, K-means for MM-P dataset shows a precision =  $\text{True positives}/(\text{True positives} + \text{False positives})$  of 0.938 in contrast with the 0.875 reached by MM-P in K-NN, the 0.812 obtained by ST-P, ST-RF

and MM-RF in K-NN algorithm and ST-P, ST-RF and MM-RF in SVC or  
 465 the 0.750 in the case of MM-P with SVC. Consequently, with the selection of  
 the soft-sensor based on K-means with MM-P, the operator would have high  
 confidence in the classification of the new sample into the ‘1’ group, and only  
 then, re-adjust the refining process.

470 *4.2. Analysis of the Impact of the Normalization and FW on the Selected  
 Soft-sensor*

In this subsection, an analysis about the normalization methods effect over  
 the dataset transformation and the obtained accuracy is performed. The  
 analysis is based on the hypotheses presented in Subsection 3.1.

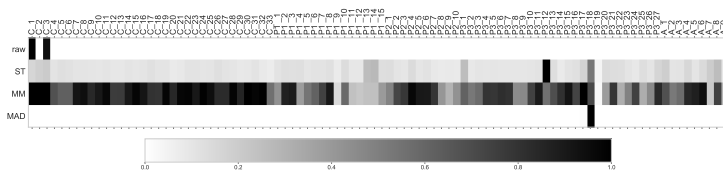


Figure 4: Color-map of the values of the normalized ranges of the 94 features that compose the raw and the normalized datasets (ST, MM, MAD).

475 In Figure 4, each row represents the normalized ranges of the labeled training  
 samples of the raw and the normalized datasets (ST, MM, MAD). The cells in  
 Figure 4 arrange from black color, corresponding to the maximum range value,  
 to white color for the insignificant ones.

As it can be observed in the first row of Figure 4, features C.1 and C.3  
 in the raw dataset present a range much higher than the rest of features.  
 480 However, ranges behavior for ST, MM and MAD varies significantly. Features  
 P3\_12 and P3\_18 in ST present the highest ranges values, and all the features’  
 ranges in MAD are insignificant compared to the range of feature P3\_18. In  
 the case of MM, contrary to the other normalization methods, several features  
 referred to input crude properties present a range value close to the highest  
 485 one. Consequently, the same feature is differently compressed or expanded  
 based on the normalization strategy employed. Besides, the application of the  
 same normalization method to diverse features of the raw dataset differently  
 transforms each feature, i.e., some of them are expanded while others are  
 compressed. Definitely, Hypotheses 1 and 2 are validated.

490 The second step described in Section 3 is the application of the proposed  
 two-stage methodology in order to intelligently transform the dataset to be  
 representative of the features relative importance, i.e. for each normalized  
 dataset, each normalized feature is multiplied by its corresponding weight.

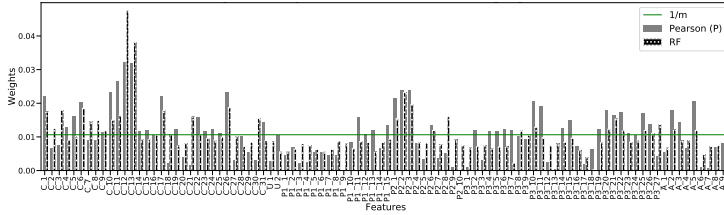


Figure 5: Weights obtained for the 94 features that compose the raw dataset by P and RF Feature Weighting methods.

Figure 5 renders the weights for the 94 features obtained by the novel Adapted Pearson and RF FW methods described in Section 3. The horizontal line represents the weight ( $1/m$ ) each feature would have if all of them were equally relative important for estimating the output. As it can be observed, both Pearson and RF select the same features, C\_12 and C\_13, as the most relevant ones to estimate the output. However, the degree of relevance each FW method assigns is different. RF presents higher difference between the maximum and the minimum estimated weights but Pearson is more discriminative than RF, i.e.  $\sum_{j=1}^m |w_j^P - 1/m| > \sum_{j=1}^m |w_j^{RF} - 1/m|$ . Besides, Pearson calculates weights almost equal to zero for some features such as, P1\_9, P2\_10 and P3\_13, among others.

Figure 6 presents the normalized ranges of the features for the six datasets resultant from the two-stage methodology, i.e. after multiplying the normalized features by the estimated weights.

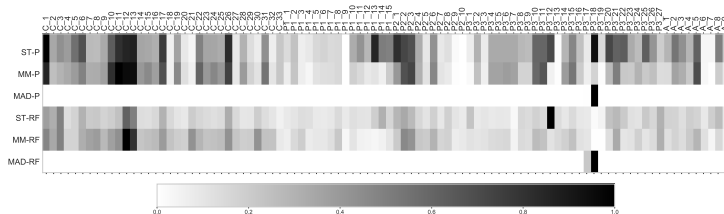


Figure 6: Color-map of the normalized ranges of the 94 features that compose the six datasets resultant from the two-stage methodology.

As it can be noticed in Figure 6, the normalized ranges of the above-mentioned features P1\_9, P2\_10 and P3\_13 are near to zero for ST-P, MM-P and MAD-P datasets, which means that their relative importance have significantly decreased respect to those assigned by the normalization methods. In fact, comparing the ranges calculated for ST and MM from Figure 4 with ST-P and MM-P or ST-RF and MM-RF from Figure 6, respectively, the effect of the weights on transforming the dominance by the feature weighting methods of some features is appreciated.



On the contrary, the ranges for MAD, MAD-P, MAD-RF in Figure 4 and Figure 6 present the same behavior. This is because the range of the feature P3\_18 transformed by MAD normalization method is too high with respect to the other features. Hence, the application of the weights by P and RF is not enough to reduce the dominance of the mentioned feature. These results validate the Hypothesis 3 and justify the need of employing an adequate normalization technique.

In this context, an analysis of the impact of an inappropriate selection of the normalization method for the preferred K-means based soft-sensor is conducted: The selected soft-sensor based on K-means with MM-P achieves a mean accuracy of 81.407%. However, as depicted in Table 2, the accuracy obtained for MAD, MAD-P and MAD-RF with K-means is 40.741%. Figure 4 and 6 show that the feature P3\_18 presents the maximum range value in the three cases, while the range of the remaining features is equal to zero. The higher the range of the feature, the higher the expected distance between the centroids. Thus, due to this high feature dominance for MAD, MAD-P and MAD-RF, K-means achieves the same classification accuracy in these cases.

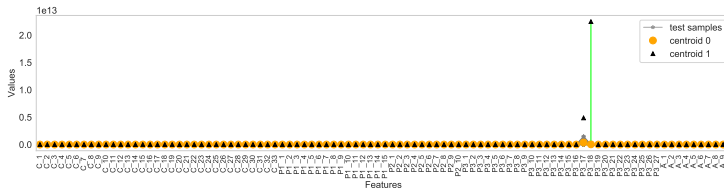


Figure 7: Test samples values and centroids calculated by K-means from MAD-RF dataset.

In Figure 7 the circle and the triangle depict the centroids of the training groups ‘0’ and ‘1’, respectively, while the dashed lines refers to the test samples. As expected from the ranges for MAD, MAD-P, and MAD-RF in Figure 4 and 6, the difference between the centroids of the feature P3\_18, remarked with the vertical green line, is higher than the sum of all the distances between the centroids of the rest of features. Thus, given the test sample  $\mathbf{x}_i$  and the centroids  $\mathbf{c}_g$ ,  $g \in \{0, 1\}$ , if  $|c_{0F} - c_{1F}| \gg |c_{0j} - c_{1j}|$ ,  $F \in \{1, \dots, 94\}$ ,  $F \neq j \in \{1, \dots, 94\}$ , then,  $\sum_{j=1}^{94} (x_{ij} - c_{0j})^2 - \sum_{j=1}^{94} (x_{ij} - c_{1j})^2 \approx (x_{iF} - c_{0F})^2 - (x_{iF} - c_{1F})^2$ , i.e., the sum of the distances of the samples to the centroids is equivalent to the distance of the centroids to the imposing feature F. Consequently, the test sample belongs to the group ‘0’ if  $(x_{iF} - c_{0F})^2 - (x_{iF} - c_{1F})^2 < 0$ .

As depicted in Figure 7, all the test samples stay closer to the centroid ‘0’ for the feature P3\_18. Therefore, based on such feature, K-means algorithm classifies all the test samples as normal samples (group ‘0’) with only a 40.741% of classification accuracy.

#### 4.3. Industrial Application: Knowledge Extracted from the Soft-sensor

This section illustrates the knowledge extraction from the selected soft-sensor, K-means with MM-P, during the real operation of the industrial

Basque petroleum refinery plant.

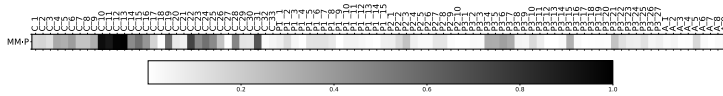


Figure 8: Normalized absolute difference between the centroids ‘0’ and ‘1’ of each feature for the dataset MM-P.

With that purpose, Figure 8 depicts the normalized absolute differences  $ndif_{c_j} = |c_{0j} - c_{1j}|/\max(|c_0 - c_1|)$  between the centroid ‘0’ and ‘1’ of each feature  $j \in \{1, \dots, 94\}$ . Note that the features with high  $ndifc$  are the most contributing ones in the Euclidean distance calculations. In this case, the darkest cells in Figure 8 correspond to features [C\_13, C\_12, C\_10, C\_11], which present a normalized distance between the centroids higher than 0.9 respect to the maximum one. Furthermore, it can be noticed that the mentioned features correspond to the second, first, sixth and third most relative important features, respectively, according to Pearson FW method (Figure 5). Given that (a) the selected soft-sensor provides the highest accuracy and precision among all the K-means based soft-sensors, and (b) the most relevant features emerged from both MM and MM-P methods, it can be concluded that the features [C\_13, C\_12, C\_10, C\_11] impact significantly on the outgoing quality.

From the process perspective, C\_13, C\_12, C\_10 and C\_11 correspond to two compositional properties from the previous analysis of the input crude properties shown in Section 2. More specifically, they are parameters corresponding to the distilled streams of atmospheric residue and vacuum residue, respectively, which, at the same time, are relevant measures to infer the penetration quality standard of the bottom product of the vacuum distillation unit. Thus, the predominance of these features in the soft-sensor resulting from the two-stage methodology and the high precision reached by the model demonstrate the reliability of the soft-sensor for its applicability.

## 5. Conclusion

In this paper, a soft-sensor able to classify the quality of the bottom product of the vacuum distillation unit based on the input crude properties and the operational conditions of the system is presented. In the preprocessing step, a novel two-stage methodology which intelligently transforms the input space to a most representative one that accounts for the relative importance of each feature for estimating the output value is proposed. Then, the transformations yielded by the application of this methodology are analyzed with different ML algorithms: K-means, K-NN, RFc, SVC and MLP. In order to select the final soft-sensor, the performance of the ML algorithms is evaluated in terms of accuracy, generalization capability, memory and computational resources, interpretability and precision scores.

The novelties and advantages presented in this research are: (1) An analysis of the effect of both, normalization and feature weighting methods, and their implication in the Euclidean distance calculations; (2) The successful application of the proposed two-stage methodology respect to the traditional normalization methods for improving the accuracy of the algorithms; (3) the proposed adapted Pearson correlation coefficient as a novel supervised FW method to estimate the relative importance of the features respect to categorical labels, (4) the soft-sensor ability to classify the samples according to the penetration quality standard, based on the crude properties and the operational conditions, with high accuracy and precision, and (5) the validity of the proposed methodology over a real petrochemical industry case study.

Experiments show that the proposed K-means with MM·P soft-sensor approach outperforms its ML counterparts for this case study. In addition, the explainability of the K-means algorithm is preferred for the soft-sensor deployment. Moreover, its precision of 0.938 proves a high reliability about the correct classification of the non-desired quality.

Future work will delve into including a confidence-based learning system to the proposed soft-sensor. Moreover, further analysis about the temporal information retrieval of both labeled and unlabeled samples for improving the classification accuracy and providing a high accurate long-term forecasting will be performed.

## 6. Acknowledgements

This work has been supported by a DATAinc fellowship (48-AF-W1-2019-00002) and a TECNALIA Research and Innovation PhD Scholarship. Besides, this work is part of the 3KIA project (KK-2020/00049), funded by the ELKARTEK program of the SPRI-Basque Government.

## References

- [1] Aenor, Une-en 1426 : Bitumen and bituminous binders—determination of needle penetration (2015) 1–16.
- [2] L. d. P. A. Sales, F. M. T. d. Luna, B. d. A. Prata, An integrated optimization and simulation model for refinery planning including external loads and product evaluation, *Brazilian Journal of Chemical Engineering* 35 (2018) 199–215.
- [3] G. K. Saharidis, M. Minoux, Y. Dallery, Scheduling of loading and unloading of crude oil in a refinery using event-based discrete time formulation, *Computers & Chemical Engineering* 33 (2009) 1413–1426.
- [4] L. Sheremetov, J. Martínez-Muñoz, M. Chi-Chim, Two-stage genetic algorithm for parallel machines scheduling problem: Cyclic steam stimulation of high viscosity oil reservoirs, *Applied Soft Computing* 64 (2018) 317–330.

- [5] S. S. Chauhan, P. Kotecha, An efficient multi-unit production planning strategy based on continuous variables, *Applied Soft Computing* 68 (2018) 458–477.
- [6] P. Gil, H. Martins, F. Januário, A. Santos, L. Palma, A. Cardoso, J. Henriques, Data anomaly detection in wireless sensor networks with application to an oil refinery, in: 2018 13th APCA International Conference on Automatic Control and Soft Computing (CONTROLO), IEEE, pp. 425–429.
- [7] P. Xu, R. Du, Z. Zhang, Predicting pipeline leakage in petrochemical system through gan and lstm, *Knowledge-Based Systems* 175 (2019) 50–61.
- [8] B. Zhao, S. Chen, Y.-x. Wang, J.-h. Li, Maintenance decision methodology of petrochemical plant based on fuzzy curvelet neural network, *Applied Soft Computing* 69 (2018) 203–212.
- [9] X. Yuan, B. Huang, Y. Wang, C. Yang, W. Gui, Deep learning-based feature representation and its application for soft sensor modeling with variable-wise weighted sae, *IEEE Transactions on Industrial Informatics* 14 (2018) 3235–3243.
- [10] K. Siddharth, A. Pathak, A. K. Pani, Real-time quality monitoring in debutanizer column with regression tree and anfis, *Journal of Industrial Engineering International* 15 (2019) 41–51.
- [11] K. Wang, C. Shang, F. Yang, Y. Jiang, D. Huang, Automatic hyper-parameter tuning for soft sensor modeling based on dynamic deep neural network, in: 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC), IEEE, pp. 989–994.
- [12] I. Mohler, Ž. Ujević Andrijić, N. Bolf, Soft sensors model optimization and application for the refinery real-time prediction of toluene content, *Chemical Engineering Communications* 205 (2018) 411–421.
- [13] R. Zhang, Q. Jin, Design and implementation of hybrid modeling and pfc for oxygen content regulation in a coke furnace, *IEEE Transactions on Industrial Informatics* 14 (2018) 2335–2342.
- [14] R. Parvizi Moghadam, Online monitoring for industrial processes quality control using time varying parameter model, *International Journal of Engineering* 31 (2018) 524–532.
- [15] L. Pater, Application of artificial neural networks and genetic algorithms for crude fractional distillation process modeling, *arXiv preprint arXiv:1605.00097* (2016).
- [16] J. J. Macias, P. Angelov, X. Zhou, A method for predicting quality of the crude oil distillation, in: 2006 International Symposium on Evolving Fuzzy Systems, IEEE, pp. 214–220.

- 665 [17] D. Wang, J. Liu, R. Srinivasan, Data-driven soft sensor approach for quality prediction in a refining process, *IEEE Transactions on Industrial Informatics* 6 (2009) 11–17.
- [18] C. Zhou, Q. Liu, D. Huang, J. Zhang, Inferential estimation of kerosene dry point in refineries with varying crudes, *Journal of Process Control* 22 (2012) 1122–1126.
- 670 [19] P. Bernus, O. Noran, Data rich—but information poor, in: *Working Conference on Virtual Enterprises*, Springer, pp. 206–214.
- [20] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, H. Liu, Feature selection: A data perspective, *ACM Computing Surveys (CSUR)* 50 (2018) 94.
- 675 [21] X. Xu, Q. Liu, J. Ding, Gasoline dry point prediction of fractionation processes using dynamic inner partial least squares, in: *2017 11th Asian Control Conference (ASCC)*, IEEE, pp. 1438–1442.
- [22] X. Zhang, Y. Zou, S. Li, S. Xu, A weighted auto regressive lstm based approach for chemical processes modeling, *Neurocomputing* 367 (2019) 64–74.
- 680 [23] C. Li, D. Zhao, Y. Liu, J. Li, C. Wang, X. Gao, Research on the soft-sensing modeling method for the naphtha dry point of an atmospheric tower, in: *2018 37th Chinese Control Conference (CCC)*, IEEE, pp. 8060–8066.
- 685 [24] Y. Fan, B. Tao, Y. Zheng, S.-S. Jang, A data-driven soft sensor based on multilayer perceptron neural network with a double lasso approach, *IEEE Transactions on Instrumentation and Measurement* (2019).
- [25] M. Komosiński, K. Krawiec, Evolutionary weighting of image features for diagnosing of cns tumors, *Artificial Intelligence in Medicine* 19 (2000) 25–38.
- 690 [26] G. W. Milligan, M. C. Cooper, A study of standardization of variables in cluster analysis, *Journal of classification* 5 (1988) 181–204.
- [27] S. Bhanja, A. Das, Impact of data normalization on deep neural network for time series forecasting, *arXiv preprint arXiv:1812.05519* (2018).
- 695 [28] D. Singh, B. Singh, Investigating the impact of data normalization on classification performance, *Applied Soft Computing* (2019) 105524.
- [29] J. Pan, Y. Zhuang, S. Fong, The impact of data normalization on stock market prediction: using svm and technical indicators, in: *International Conference on Soft Computing in Data Science*, Springer, pp. 72–88.
- 700 [30] C. Leys, C. Ley, O. Klein, P. Bernard, L. Licata, Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median, *Journal of Experimental Social Psychology* 49 (2013) 764–766.

- 705 [31] R. F. Tate, Correlation between a discrete and a continuous variable. point-biserial correlation, *The Annals of mathematical statistics* 25 (1954) 603–607.
- [32] L. Breiman, Random forests, *Machine learning* 45 (2001) 5–32.
- [33] P. Domingos, The role of occam’s razor in knowledge discovery, *Data mining and knowledge discovery* 3 (1999) 409–425.
- 710 [34] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins, et al., Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai, *Information Fusion* 58 (2020) 82–115.

## **V Soft-Sensor for Class Prediction of the Percentage of Pentanes in Butane at a Debutanizer Column**

**Iratxe Niño-Adan, Itziar Landa-Torres, Eva Portillo, Diana Manjarres,  
Lucía Orbe**

Published in Special issue “Robotics, Sensors and Industry 4.0” *Sensors*,  
June 2021, volume 21, issue 12, 3991.

DOI: <https://doi.org/10.3390/s21123991>.

**JCR: 3.576**


*Category: Instruments & Instrumentation, 14/64, Q1*





## Article

# Soft-Sensor for Class Prediction of the Percentage of Pentanes in Butane at a Debutanizer Column

Iratxe Niño-Adan <sup>1,2,\*</sup>, Itziar Landa-Torres <sup>3</sup>, Diana Manjarres <sup>1</sup> , Eva Portillo <sup>2</sup> and Lucía Orbe <sup>3</sup>

<sup>1</sup> TECNALIA, Basque Research and Technology Alliance (BRTA), 48160 Derio, Spain; diana.manjarres@tecnalia.com

<sup>2</sup> Department of Automatic Control and Systems Engineering, Faculty of Engineering, University of the Basque Country UPV/EHU, 48013 Bilbao, Spain; eva.portillo@ehu.es

<sup>3</sup> Petronor Innovación S.L, 48550 Muskiz, Spain; itziar.landa@repsol.com (I.L.-T.); lucia.orbe@repsol.com (L.O.)

\* Correspondence: iratxe.nino@tecnalia.com

**Abstract:** Refineries are complex industrial systems that transform crude oil into more valuable subproducts. Due to the advances in sensors, easily measurable variables are continuously monitored and several data-driven soft-sensors are proposed to control the distillation process and the quality of the resultant subproducts. However, data preprocessing and soft-sensor modelling are still complex and time-consuming tasks that are expected to be automatised in the context of Industry 4.0. Although recently several automated learning (autoML) approaches have been proposed, these rely on model configuration and hyper-parameters optimisation. This paper advances the state-of-the-art by proposing an autoML approach that selects, among different normalisation and feature weighting preprocessing techniques and various well-known Machine Learning (ML) algorithms, the best configuration to create a reliable soft-sensor for the problem at hand. As proven in this research, each normalisation method transforms a given dataset differently, which ultimately affects the ML algorithm performance. The presented autoML approach considers the features preprocessing importance, including it, and the algorithm selection and configuration, as a fundamental stage of the methodology. The proposed autoML approach is applied to real data from a refinery in the Basque Country to create a soft-sensor in order to complement the operators' decision-making that, based on the operational variables of a distillation process, detects 400 min in advance with 98.925% precision if the resultant product does not reach the quality standards.

**Keywords:** pentanes; classification; autoML; soft-sensor; normalisation; feature weighting



**Citation:** Niño-Adan, I.; Landa-Torres, I.; Manjarres, D.; Portillo, E.; Orbe, L. Soft-Sensor for Class Prediction of the Percentage of Pentanes in Butane at a Debutanizer Column. *Sensors* **2021**, *21*, 3991. <https://doi.org/10.3390/s21123991>

Academic Editor: Ki H. Chon

Received: 18 May 2021

Accepted: 4 June 2021

Published: 9 June 2021

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Refineries are complex industrial systems that transform crude oil into more valuable subproducts, i.e., Liquefied Petroleum Gas (LPG), gasoline or petrol, kerosene, jet fuel, diesel oil and fuels oils. One of their primary concerns is to ensure high-quality final subproducts that meet the rigorous government regulations to achieve the maximum profit for commercialising them. In this context, due to the advances in sensing, easy-to-measure variables are continuously monitored, and several data-driven soft-sensors are proposed to control the distillation process and the quality of the resultant subproducts. In this research line, there are several works for monitoring and controlling different processes of the refinery. Among them is the work proposed in [1] for estimating oxygen content in a coke furnace, and the soft-sensor for predicting MAE and SWA acid gases in a sulphur recovery unit or for butane concentration in a debutanizer column [2,3].

Nevertheless, a common issue reported in real industrial applications is that the datasets are generally “data rich but information poor” [4]. Therefore, there is a considerable need to devise intelligent strategies for selecting informative data that extract valuable knowledge. Some researches include preprocessing methods for identifying or even discarding samples that may worsen the model output. For instance, the authors

of [5] improve the gasoline dry point prediction accuracy by removing the influence of outliers of the operation data. In the research presented by [6], a Robust Partial Least Square (PLS) method is employed to identify multivariate anomalies, and a Dynamic PLS selects and optimises the input samples that best estimate the naphtha dry-point in an atmospheric vacuum distillation tower. In line with the data selection, authors of [7] include a Gaussian process-based samples selection strategy in order to add informative samples for a dynamical adaptation of the model to present an online adaptive model that infers different propane and ethane quality measurements in the top of a depropanizer column.

Following the same research line, other related works take a step forward by utilising Feature Selection (FS) approaches. In this way, only the relevant input features are selected for creating the model. Authors of [8] include a Genetic Algorithm strategy into their PLS-based soft-sensor to select the most relevant variables for operation and quality control of the ASTM 90% distillation temperature (D90) in a crude distillation unit. Similarly, in order to extract relevant features to estimate the flow rate and the yield of some resultant sub-products (gasoline, diesel oil, coke and LPG) in a Fluid Catalytic Cracking unit, the authors of [9] employ a Recurrent Denoising Auto Encoder and a Cumulative Percent Variance. Another example can be found in [10] where, aiming at selecting the most sensitive features concerning the output value avoiding redundancy problems with correlated features, a double LASSO algorithm integrated into an MLP model is presented to predict the kerosene D95 in a crude distillation unit. Feature selection based on correlation analysis is employed by the authors of [11] for estimating H<sub>2</sub>S and SO<sub>2</sub> acid gases concentration in a Sulphur Recovery Unit and also in the soft-sensor presented in [12] for toluene content estimation.

In the above-mentioned FS approaches, a weight equal to one is assigned to the selected features and zero to the discarded ones. As widely known, a further step is done by the employment of feature weighting (FW) approaches in which a weight between zero and one is assigned in order to represent the degree of relative importance each feature gathers concerning the output label or class. This approach is applied in [13], where the authors include feature weights calculated as the correlation between each feature and the output variable to estimate the butane concentration at the bottom of a debutanizer column.

As observed in the state-of-the-art highlighted above, recent works rely on the context of Industry 4.0, digitising industrial processes [14,15] by proposing soft-sensors that integrate feature preprocessing and ML algorithms. Another hot topic in both industry and academia is automated Machine Learning (autoML) [16–20], which aims at enabling domain experts to build ML applications automatically [21]. As stated in [22], the ideal autoML approach involves data preprocessing, model generation, and model evaluation. Despite data preprocessing being the first task typically in ML approaches, autoML systems have focused on model selection and hyper-parameter searches [23], while data preprocessing still requires considerable human intervention [24]. Aiming at advancing in the automation of learning systems in the context of Industry 4.0, this research presents an autoML approach that searches for the best configuration among well-known normalisation and FW preprocessing techniques as well as among popular ML algorithms in order to create a soft-sensor that complements and supports the operators' decision-making by classifying the percentage of pentanes in the butane obtained at the end of a debutanizer column, according to the product specifications. The quality of butane is dependent on the percentage of pentanes present in the gas. If the rate exceeds a certain threshold, the product must be reprocessed, and additional costs are incurred. Therefore, several works are devoted to solving this open challenge. Ito et al. [25], based on data obtained from a gas processing plant simulation, infer in an online fashion the concentration of pentanes in a debutanizer column by combining a physical model with heuristic rules. Similarly, the authors of [26] present a NARMAX-based soft-sensor for estimating the pentanes content in butane. The authors utilise data from a real refinery plant where the time tag difference between the input and the output lies in a range of 20–60 min approximately.

In contrast to [25,26], this work aims at predicting 400 min in advance if the percentage of pentanes in butane at the end of the debutanizer column will fulfil the quality standards.

Thus, the operators can adjust the process immediately and avoid distilling a product that will not meet the specifications for more than six hours. With that purpose, this work utilises real data from a refinery of the Basque Country, and process variables of the top of the stabilising naphtha towers are employed to create the soft-sensor. The autoML preprocessing phase design and development are based on a novel two-stage methodology that combines normalisation and feature weighting to intelligently transform the input data to reflect each feature's relative importance for classifying the resulting quality. In order to configure the soft-sensor from the two-stage methodology, in the model generation and evaluation phase, seven well-known classification algorithms are considered [27]: Quadratic Discriminant Analysis (QDA), K-Nearest Neighbours (KNN), Support Vector Classification (SVC), Ridge Regression (RID), Logistic regression (LOG), MultiLayer Perceptron (MLP) and Stochastic Gradient Descent (SGD). Since the purpose of the resulting soft-sensor is to complement the operator decision-making, the model that maximises the classification performance in terms of precision is selected, aiming at maximising the operator's reliability in the model's results when performing operational changes in the system.

The remainder of the paper is structured as follows: Section 2 presents an outline of the analysed distilling process. Section 3 describes the methodology proposed for the design and development of the autoML approach that searches for the best configuration of the soft-sensor for class prediction of the percentage of pentanes in butane at the end of the debutanizer unit. Classification results, analysis of the developed soft-sensor and the profit obtained by applying the proposed approach are shown in Section 4. Finally, Section 5 depicts the conclusions and future work.

## 2. Problem Description

In this work, real data from a refinery allocated in the Basque Country are utilised. Figure 1 depicts a high-level diagram of the analysed unit chain, in which crude oil is converted into high-quality gas subproducts.

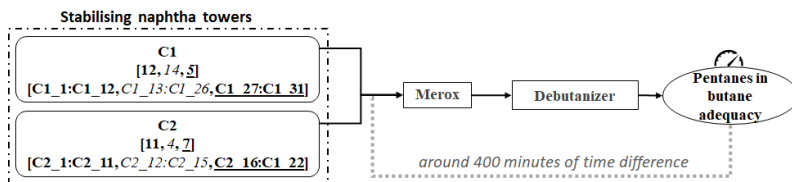


Figure 1. High-level diagram of the analysed process.

Columns C1 and C2 in Figure 1 represent two different stabilising naphtha towers. After a refining process of the raw crude, stabilised naphtha and Liquefied Petroleum Gas (LPG) are obtained at the bottom and the top of the columns C1 and C2, respectively. The resulting LPG is then pumped from the top of columns C1 and C2 to Mercox, a gas sweetening unit in which the sulphur is removed. Finally, the sweetened gases pass to the debutanizer column, where propane and butane are separated. The estimated duration of the described unit chain, from stabilising naphtha columns to the end of the debutanizer column, is 400 min.

In order to fulfil the specification standards [28], the resultant butane must not exceed a certain threshold of the percentage of pentanes (1.5%). According to the mentioned threshold, the refinery's interest is to classify the percentage of pentanes in butane as adequate (class 0) or improvable (class 1). Currently, a Dynamic Matrix Control (DMC) control algorithm optimises the distillation process in columns C1 and C2 by adjusting temperature and reflux from the top of the columns according to an estimation of the number of pentanes in the feed. Furthermore, currently, the percentage of pentanes in butane at the end of the debutanizer column is measured online. However, despite the suitability of the DMC, there are some episodes in which pentanes escape from the top of

columns C1 and C2. In such scenarios, the deviation from the requirements is detected at the end of the debutanizer column. Then, aiming at estimating the percentage of pentanes in butane 400 min in advance, in this work, the classification is conducted based on the refining process of C1 and C2. With that purpose, from the top of C1 and C2, 31 (C1\_1:C1\_31) and 22 (C2\_1:C2\_22) features are collected, respectively. These features at each column gather information about flow, temperature and pressure from operational variables and DMC. The number of features of each column regarding each of these properties are presented in Figure 1 with bold, italic and underlined text, respectively. The process variables information and the pentanes percentage output were recorded every ten minutes for 465 days, from 24 October 2017 to 31 January 2019. Thus, the dataset consists of 66,847 samples described by the 53 features described above.

### 3. Methodology

As mentioned above, this research proposes an autoML approach that selects the best configuration among well-known preprocessing techniques and different ML algorithms. The final objective is to create, based on the process condition in the top of the naphtha stabiliser columns, a reliable soft-sensor that performs the offline model training and the posterior online validation for classifying the percentage of pentanes in butane at the end of the debutanizer column 400 min in advance. This section describes the methodology employed to analyse the dataset and the stages of the autoML approach.

#### 3.1. Dataset Evaluation

Figure 2 depicts the procedure proposed for evaluating the dataset and extracting the key information from the data. This section thoughtfully details such procedure and, hence, the mathematical tests proposed for determining the optimal train and test sets for modelling the algorithm. Finally, the analysis of the relationship between the input features and the real labels is detailed, and how the result of such study determines the latter ML algorithms application is explained.

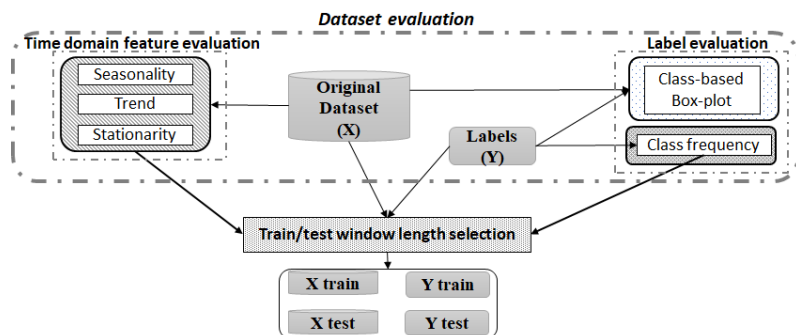


Figure 2. Dataset evaluation diagram.

##### 3.1.1. Time Domain Feature Evaluation

In order to extract information from the historical data to predict the future, it is desirable that the data collected over time is representative of current conditions, i.e., it reflects stable equilibrium. This property is called stationary, and this work proposes to check it by means of considering the features of the dataset as time series.

A time series [29,30] is an ordered sequence of observations, and it is defined as a function of three major components: seasonality, trend and random noise. Seasonality and trend are sources of non-stationarity, which means their identification and removal from the time series can result in a clearer relationship between input features  $X$  and output  $Y$ . In addition, if the random noise component of the time series is not stationary, the statistical properties of the time series evolve over time. In such setting, a shorter sampling interval

may be needed to capture key characteristics of the population. With the aim of checking the time series stationarity to select the optimal set ( $X$  train) to train the model, seasonality, trends and stationarity of the time series are analysed. The time domain feature evaluation is completed by the analysis of the rolling statistics of the time series.

- Seasonality refers to the repeating variation at regular intervals of time. The data are considered seasonal if a significant autocorrelation [31] coefficient exists at a given lag [32]. The autocorrelation function measures the linear correlation between a time series and a delayed version of itself searching for repeating patterns. Autocorrelation values range between  $[-1, 1]$ , where 1 and  $-1$  values represent total positive or negative correlations between two time series, respectively.
- Trend refers to the general tendency of the features (upward or downward); the Mann–Kendall (MK) test [33] is utilised to ascertain the absence of a trend in the time series. Thus, the null hypothesis  $H_0$  assumes that there is no trend in the time series and the MK test analyses the sign differences between samples of different moments to discard increasing or decreasing measurements in the time series.
- Stationarity analyses the random development around a constant average of the time series. Augmented Dickey–Fuller (ADF) [34] and Kwiatkowski–Phillips–Schmidt–Shin (KPSS) [35] non-parametric tests were applied with the aim of examining if the time series is stationary and the statistics are consistent over time. The null hypothesis  $H_0$  of the ADF test states the presence of a unit root, i.e., the series is non-stationary, while the alternative hypothesis assumes the weak stationary. Concretely, ADF states that if a unit root exists, the lagged version of the time series does not provide information for predicting changes in the current value of such time series. In contrast, KPSS test's  $H_0$ : (1) assumes that the time series is stationary around the trend, and (2) it expresses the time series as the sum of the deterministic trend, random walk and stationary error. Since the possible source of non-stationarity in this expression is the random walk, KPSS checks that the random walk has zero variance, i.e., it does not evolve over time.
- Rolling statistics, such as mean and standard deviation, were also analysed in order to check the stability of the time series over time, as well as to detect changes in the statistical properties of the time series. Thus, if changes are detected with the rolling statistics analysis, the window size selection is conducted considering the frequency of such changes.

### 3.1.2. Label Evaluation

Considering the real labels  $Y$  of the dataset, two analysis are conducted: (1) the frequency of each class occurrence over time is computed in order to select a representative sample population of each class for training ( $X$  train,  $Y$  train) and testing ( $X$  test,  $Y$  test) the model, and (2) the discriminative ability of the features can be adverted in the box-plot and hence, the existence/absence of a linear relationship between the input features and the output label will determine the selection of the most appropriate ML algorithm to create the soft-sensor.

### 3.2. Optimal Dataset Split Selection

In order to create a reliable model and to reproduce the offline training and online quality prediction practice, the dataset split is done respecting the temporal order of the data. As determined in Section 3.1.1, if the time series is not stationary, employing all available historical data can disturb the prediction ability of the model due to the evolving time series statistical properties over time. In such scenario, the window length for the training set must be selected in order to capture the variability of the input data and, thus, the current properties of the time series. Likewise, as described in Section 3.1.2, the training set ( $X$  train,  $Y$  train) is selected in such a way that all classes are represented.

Furthermore, note that in online environments, the moment in which the statistical properties will change with respect to the current condition is not known in advance. Con-

sequently, for the train/test window size selection, a conservative approach is conducted, which considers the possibility of statistical properties drift in the test set.

### 3.3. AutoML Approach Description

This Section describes all the stages of the autoML approach that selects the best configuration among different well-known preprocessing methods and ML algorithms in order to create the soft-sensor for supporting the decision-making. Figure 3 illustrates a high-level diagram of the proposed autoML approach.

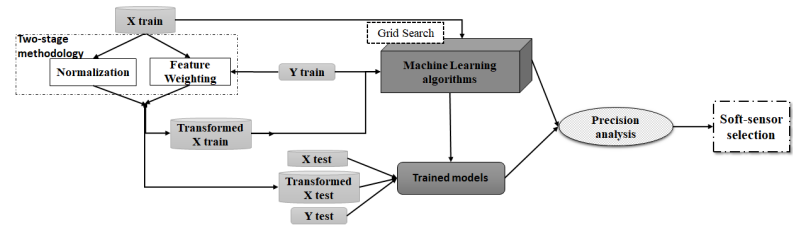


Figure 3. High-level diagram of the proposed autoML approach.

#### 3.3.1. Two-Stage Methodology

The Two-stage methodology is applied in order to transform the original raw dataset  $X \subset \mathbb{R}^{n \times m}$  into a new transformed one, denoted by  $\tilde{X}_{FW}^{Norm} \subset \mathbb{R}^{n \times m}$ , based on normalisation and feature weighting, for representing the relative importance, each feature  $j = \{1, \dots, m\}$  gathers for classifying the samples  $i = \{1, \dots, n\}$  among the different classes.

(1) Normalisation. It is thought that normalisation equalises the contribution of each feature in the ML algorithm calculations [36]. This is why normalisation methods are commonly applied during the preprocessing step in order to avoid the over-contribution of a set of features due to the magnitudes difference. However, each normalisation method transforms the dataset differently. In addition, each feature is compressed or expanded depending on the normalisation method and its statistical values [37], which ultimately can condition the features' influence on the ML algorithm calculations and its performance. Since there is no specific normalisation method suitable for all the problems, three of the most commonly employed approaches are selected in this work. All of them are linear transformations based on position and dispersion statistics.

- Standardisation (ST):  $\tilde{X}_j^{ST} = (X_j - \bar{X}_j) / \sigma_j \forall j \in \{1, \dots, m\}$ . The resultant features are centred around the mean with a standard deviation equal to 1.
- Min–max normalisation (MM):  $\tilde{X}_j^{MM} = (X_j - \min(X_j)) / \text{range}(X_j), \forall j \in \{1, \dots, m\}$  where  $\text{range}(X_j) = \max(X_j) - \min(X_j)$ . The samples of the resulting features take values between  $[0, 1]$ .
- Median Absolute Deviation normalisation (MAD):  $\tilde{X}_j^{MAD} = (X_j - \text{Me}(X_j)) / \text{MAD}(X_j), \forall j \in \{1, \dots, m\}$ . In contrast to the other techniques, MAD normalisation is considered a robust transformation [38] as the calculation of the median  $\text{Me}$  is not affected by the presence of outliers.

Since each normalisation method (ST, MM and MAD) depends on different statistics, and given that each feature is transformed differently depending on its statistical characteristics, it is expected that a different subset of features predominate for each normalised dataset ( $\tilde{X}^{ST}, \tilde{X}^{MM}$  and  $\tilde{X}^{MAD}$ ). Then, in this work, the range of the features is employed as an indicator of the influence of each feature in the algorithm performance [39]. To facilitate the comparison between features of the same dataset, the ranges are divided by the maximum range of the dataset  $\text{range}(\tilde{X}_j^{Norm}) / \max(\{\text{range}(\tilde{X}_j^{Norm}) | j = \{1, \dots, m\}\})$ . This way, the most influencing features, i.e., with a range close to the maximum one, present a normalised range close to 1. In contrast, features that present range value much lower than

the maximum one, being, in comparison, insignificant ranges, will present a normalised range close to 0.

(2) Feature Weighting. Feature weighting methods transform the features of the dataset to be representative of the relative information each gathers for estimating the output. This transformation is conducted by a vector  $\mathbf{w}$  of feature weights, where the components represent the relative importance of each feature. Note that each weight has a value in the range from 0 to 1, so the sum of  $\mathbf{w}_j$  for all  $j = \{1, \dots, m\}$  is 1. Among the FW methods, the filter approaches calculate the feature weights independently from the ML algorithm. If information on the labels is employed for computing the weights, the FW approach is considered supervised. Three supervised filter FW methods are applied in this research: two well-known methods, Random Forest and Mutual Information, both based on Information Theory and the Adapted Pearson correlation [37], a statistical-based method previously proposed by these authors. Random Forest calculates the feature weights conjointly, while Mutual Information and Adapted Pearson correlation estimate the weights for each feature in an independent manner, i.e., without considering the remaining ones. The three FW methods are briefly described below:

- Adapted Pearson correlation (P): this statistical-based FW method is an adaptation of the Pearson correlation coefficient for handling categorical and continuous features. It aims at estimating the relative importance of each feature for separating the classes in classification problems. With that purpose, the proposal presented in [37] utilises the labels of the dataset to separate the samples according to the class. Thus, the labels are encoded as the centroid of the samples that correspond to such label. Then, for each component of the vector of weights, the absolute value of the Pearson correlation coefficient is estimated between each feature and the corresponding component of the encoded label. Finally, the weights vector is divided by the sum of their components to obtain the vector of weights  $\mathbf{w}^P$ , so  $\sum_{j=1}^m w_j^P = 1$ .
- Random Forest classifier (RF): Random Forest [40] is a decision tree-based ensemble-learning ML algorithm utilised for different tasks, such as classification or regression problems. In addition, it is also widely employed for calculating the relevance of the features for estimating the output, according to their contribution in the trees employed for creating the forest. Each tree in the ensemble employs bootstrapping, which, together with an elevated number of trees and the tree splitting strategy, are randomness sources that decrease the variance of the estimations. Thus, in this work, the RF employed as the FW method is constructed by 100 decision trees. The final feature weight vector  $\mathbf{w}^{RF}$  is calculated as the mean of the features importance of 30 RF-based models. Thus, in total, 3000 decision trees are considered. Each decision tree is constructed from a random subset of features of length equal to the square root of the total number of features of the dataset. A leaf node requires a minimum of one sample, while all the nodes with more than one sample are considered internal nodes. The sub-sample set employed for training each tree presents the same size as the original dataset, but, with bootstrap, this set is drawn with replacement. Once the algorithm is trained, the relative importance of each feature is calculated by the Mean Decrease Gini [41], which computes the mean of the weighted impurity decrease of all the nodes of all the trees where the feature is used. In this work, the scikit-learn package [42] of python has been used for the estimation of  $\mathbf{w}^{RF}$ .
- Mutual Information (MI): this FW method measures the degree of mutual relatedness between a feature and the labels, which can be interpreted as the amount of shared information between them. MI employs joint and marginal probability distributions to compute the calculations, which are generally unknown in real problems. Again, the scikit-learn package of python is utilised [42], which adds a small amount of noise to continuous variables in order to remove repeated values, and employs a nearest neighbour method [43,44] for estimating the MI. In this work, the number of neighbours  $k$  is set to 3, since small  $k$  reduces systematic errors [43,44]. For each feature, the weight ranges from 0 to 1, and, the higher the values, the higher the relationship

between the feature and the labels. In order to be the sum of the components of the vector of weights  $\mathbf{w}^{MI}$  equal to 1, the estimated weights are divided by the sum of the feature weights.

The feature weights  $\mathbf{w}^P, \mathbf{w}^{RF}$  and  $\mathbf{w}^{MI}$  along with the normalisation approaches above-described are employed for creating the transformed dataset  $\tilde{X}_{FW}^{Norm}$ , as it will be explained next.

(3) Transformed Dataset Calculation. As depicted in Figure 3, the two-stage methodology lies in combining both normalisation and feature weighting to intelligently transform the raw dataset. By this means, normalisation acts over the magnitude differences among the features in order to extol the resulting importance of the representativity of the features.

Then, the two-stage based transformation results from multiplying each normalised feature by the corresponding feature weight,  $(\tilde{X}_P^{ST})_j = w_j^P \cdot (\tilde{X}^{ST})_j, \dots, (\tilde{X}_{MI}^{MAD})_j = w_j^{MI} \cdot (\tilde{X}^{MAD})_j$ , respectively, for  $j \in \{1, \dots, m\}$ .

This transformation can be expressed in matrix notation as follows: given the normalised dataset  $\tilde{X}^{Norm}$  and the diagonal matrix  $diag(w_1^*, \dots, w_m^*)$  formed by the elements of the vector of weights  $\mathbf{w}^*$ , the transformed dataset is calculated as,

$$\tilde{X}_{FW}^{Norm} = \tilde{X}^{Norm} \cdot diag(w_1^*, \dots, w_m^*) = \begin{pmatrix} \tilde{x}_{11} & \tilde{x}_{12} & \dots & \tilde{x}_{1m} \\ \tilde{x}_{21} & \tilde{x}_{22} & \dots & \tilde{x}_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ \tilde{x}_{n1} & \tilde{x}_{n2} & \dots & \tilde{x}_{nm} \end{pmatrix} \cdot \begin{pmatrix} w_1^* & 0 & \dots & 0 \\ 0 & w_2^* & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & w_m^* \end{pmatrix} \quad (1)$$

The matrix resultant from Equation (1) contains in each column the normalised weighted feature. Thus, in this work, from the combination of each of the three selected normalisation methods (ST, MM, MAD) represented by  $\tilde{X}^{Norm}$  in Equation (1), with each of the feature weights vectors  $\mathbf{w}^* \in \{\mathbf{w}^P, \mathbf{w}^{RF}, \mathbf{w}^{MI}\}$  generated by the three FW approaches (P, RF, MI), a total of nine transformed datasets are obtained:  $\tilde{X}_P^{ST}, \tilde{X}_P^{MM}, \tilde{X}_P^{MAD}, \tilde{X}_{RF}^{ST}, \tilde{X}_{RF}^{MM}, \tilde{X}_{RF}^{MAD}, \tilde{X}_{MI}^{ST}, \tilde{X}_{MI}^{MM}$  and  $\tilde{X}_{MI}^{MAD}$ .

### 3.3.2. Machine Learning Algorithms

Once the original data have been intelligently transformed by Equation (1) and the datasets with features representative of their relative importance for discriminating the class labels have been obtained, different ML classification algorithms are applied.

Specifically, seven ML classification algorithms [27] from scikit-learn [42] are employed: Quadratic Discriminant Analysis (QDA), K-Nearest Neighbours (KNN), Support Vector Classification (SVC), Ridge Regression (RID), Logistic Regression (LOG), MultiLayer Perceptron (MLP) with one hidden layer and Stochastic Gradient Descent (SGD).

QDA utilises a quadratic decision surface to separate the classes assuming that each class density function follows a multivariate Gaussian distribution. It calculates different covariance matrices for each class, which are regularised by the hyper-parameter *reg\_param*. The algorithm KNN classifies each sample based on the class membership of its *k* neighbours, i.e., the *k* closest samples measured in terms of Euclidean distance. In contrast, SVC creates a hyper-plane, allocated between the supporting vectors, for separating the samples of both classes. It includes a soft-margin hyper-parameter *C* for controlling the misclassification cost. In addition, the SVC relies on the kernel trick, which allows operating in a higher dimension through inner product between pairs of data, and its hyper-parameter  $\gamma$  regulates the influence of samples selected by the model as support vectors. The RID algorithm is a regularised version of the Ordinary Least Squares regression model, where  $\alpha$  is a regularisation hyper-parameter for controlling the regression parameters. In the case of the LOG algorithm, it employs the logistic function to classify the samples, and like SVC, it includes a hyper-parameter *C*. The MLP employed in this work is a feedforward artificial neural network with a hidden layer composed of a user-defined number of *neurons*. Each neuron applies an *activation* function to a weighted linear summation of



the input values, and the final output is a weighted sum. Finally, SGD is an optimisation algorithm for minimising a *loss* function implemented to regularise linear models, where the hyper-parameter  $\alpha$  controls the strength of the regularisation.

Once the employed algorithms and their hyper-parameters have been described, a grid search (GS) algorithm is employed to select the hyper-parameters that maximise the score in terms of the selected performance metric described in Section 3.3.3. Table 1 collects the hyper-parameters employed in the GS for each ML algorithm and the total number of possible combinations.

**Table 1.** Parameters employed in the grid search for each ML algorithm and the corresponding total number of combinations (Comb) considered in the grid search.

ML	Hyper-Parameters	Comb
QDA	$\text{reg\_param} \in \{\{1, 5\} \times 10^{-5}, \{1, 5\} \times 10^{-4}, 0.005, 0.001, 0.05, 0.01, 0.5, 0.1, 1\}$	11
KNN	$\text{neighbours} \in \{5, 6, 7, \dots, 60\}$	55
SVC	$C \in \{0.0001, 0.005, 0.001, 0.05, 0.01, 0.5, 0.1, 5, 1, 10\}$ $\gamma \in \{0.0001, 0.001, 0.01, 0.1, 1, 10\}$ $\text{kernel} \in \{\text{linear, rbf, sigmoid}\}$	180
RID	$\alpha \in \{0.0001, 0.005, 0.001, 0.05, 0.01, 0.5, 0.1, 1, 2, 4, 5, 7, 10\}$	13
LOG	$C \in \{0.0001, 0.005, 0.001, 0.5, 0.01, 0.5, 0.1, 1, 5, 10\}$	10
MLP	$\text{activation} \in \{\text{identity, logistic, relu}\}$ $\text{neurons} \in \{1, 2, 3, \dots, 10\}$	30
SGD	$\text{loss} \in \{\text{modified\_huber, hinge, squared\_hinge, perceptron}\}$ $\alpha \in \{0.00005, 0.00001, 0.0005, 0.0001, 0.005, 0.001, 0.05, 0.01, 0.5, 0.1, 1\}$	44

Some of the selected algorithms, i.e., MLP and SGD, are stochastic and depend on the initialisation. Hence, 10 random initialisations are launched per combination of hyper-parameters, and the mean performance measure value is calculated. Then, the hyper-parameters with highest mean value are selected for the ML algorithm configuration. Similarly, once the optimal hyper-parameters have been selected, the results for the ML algorithms are given by the maximum, mean, standard deviation and minimum performance values from 30 random initialisations.

In order to validate the suitability of the two-stage methodology, the classification results of the nine transformed datasets resulting from the two-stage methodology ( $\tilde{X}_p^{ST}$ ,  $\tilde{X}_p^{MM}$ ,  $\tilde{X}_p^{MAD}$ ,  $\tilde{X}_{RF}^{ST}$ ,  $\tilde{X}_{RF}^{MM}$ ,  $\tilde{X}_{RF}^{MAD}$ ,  $\tilde{X}_{MI}^{ST}$ ,  $\tilde{X}_{MI}^{MM}$  and  $\tilde{X}_{MI}^{MAD}$ ) are compared with those from the original and the normalised datasets ( $X$ ,  $\tilde{X}^{ST}$ ,  $\tilde{X}^{MM}$  and  $\tilde{X}^{MAD}$ ).

### 3.3.3. Precision Analysis

After the application of the ML algorithms described in Section 3.3.2 to the nine resultant datasets from the two-stage methodology and to the raw and the normalised ones, the performance of the algorithms over each dataset is evaluated for comparison purposes. The classification ability of each model can be visualised through the confusion matrix; it reflects the relationship between the predicted classes and the real ones. Thus, in the diagonal, the number of samples correctly predicted as class 0 or 1 are collected, which are also known as true negative (TN) and true positive (TP), respectively. In contrast, the elements out of the diagonal represent the samples wrongly classified. More concretely, the cell (0,1) collects the number of samples classified as 1 with 0 their real label, known as false positive (FP) samples; and the cell (1,0) presents the false negative (FN) cases, those erroneously classified as 0 when they really belong to the class 1. The sum of the elements of the confusion matrix  $TP+TF+FP+FN=N$  is the total number of classified samples.

From the elements of the confusion matrix, different metrics are utilised for performance evaluation. A commonly employed performance metric is the accuracy  $(TP + TN)/N$ , defined as the proportion of samples correctly classified. However, it is not recommended

for imbalanced datasets, since a high overall accuracy can be reached by compromising the minority class. Thus, there are other metrics especially designed for measuring the classification performance in terms of class 1, such as precision =  $TP/(TP + FP)$  and recall =  $TP/(TP + FN)$ . Precision measures the proportion of samples the model predicts with label 1 that really correspond to such class. Thus, the higher the precision value, the lower the number of samples belonging to class 0 that are misclassified as 1. In contrast, recall represents the proportion of samples of class 1 detected by the model. Then, a low recall value corresponds to a model with poor ability for recognising the samples of class 1.

The main interest of the refinery is to complement the operators decision making with a highly-reliable predictor that detects when an improvable quality subproduct (class 1) is resulting from the process, with the minimum false alarms, so high-cost operational changes are avoided. Then, for the automatic soft-sensor creation, in the autoML approach, the precision is selected as the principal performance measure.

#### 4. Results

In this section, the results obtained from the inspection of the dataset and the application of the methodology described in Section 3 are collected. Thereafter, an analysis of the profit the refinery would obtain from the application of the developed soft-sensor is presented.

##### 4.1. Dataset Evaluation

An analysis of the dataset was conducted based on points remarked in Section 3.1.

##### 4.1.1. Time Domain Feature Evaluation

The obtained properties that characterise the temporal behaviour of the features are analysed below:

- Seasonality: Figure 4 depicts the auto-correlation of each feature with respect to itself considering a maximum lag of 6480 samples (45 days, determined by expert knowledge). As it can be observed in Figure 4a–ba, the auto-correlation values decrease with the lag increment. In most (50 out of 53) of the time series, the auto-correlation coefficient rapidly decreases to values lower than 0.4, except for Figure 4i,z,aj. In the latter cases, the auto-correlation coefficients decrease slowly with the lag increase, but the values are lower than 0.8. Then, from the auto-correlation plots no seasonality is observed.
- Trend: the  $p$ -values obtained from applying the non-parametric Mann–Kendall test are shown in Table 2. It can be observed that 17 time series (C1\_1:C1\_4,C1\_6:C1\_9,C1\_11,C1\_17, C1\_24, C2\_3, C2\_9:C2\_10,C2\_13:C2\_14) present  $p$ -values lower than 0.05; for those features,  $H_0$  can be rejected. In the rest of the cases, there is no evidence for rejecting the hypothesis of no tendency. Thus, in 36 out of 53 time series, no trend is observed.
- Stationarity: stationarity is checked through the non-parametric ADF and KPSS tests, respectively. The obtained  $p$ -values for the ADF test are shown in Table 3. In 30 out of the 53 time series, the  $p$ -values marked with italic text in Table 3 range between [0.05, 0.488], so in these cases (and considering a significance level of 0.05) the null hypothesis can not be rejected and, consequently, they are non-stationary. Accordingly,  $H_0$  can be rejected for the rest of the cases that have obtained  $p$ -values between [0, 0.048].

Moreover, the  $p$ -values resulting from the KPSS test are depicted in Table 4.

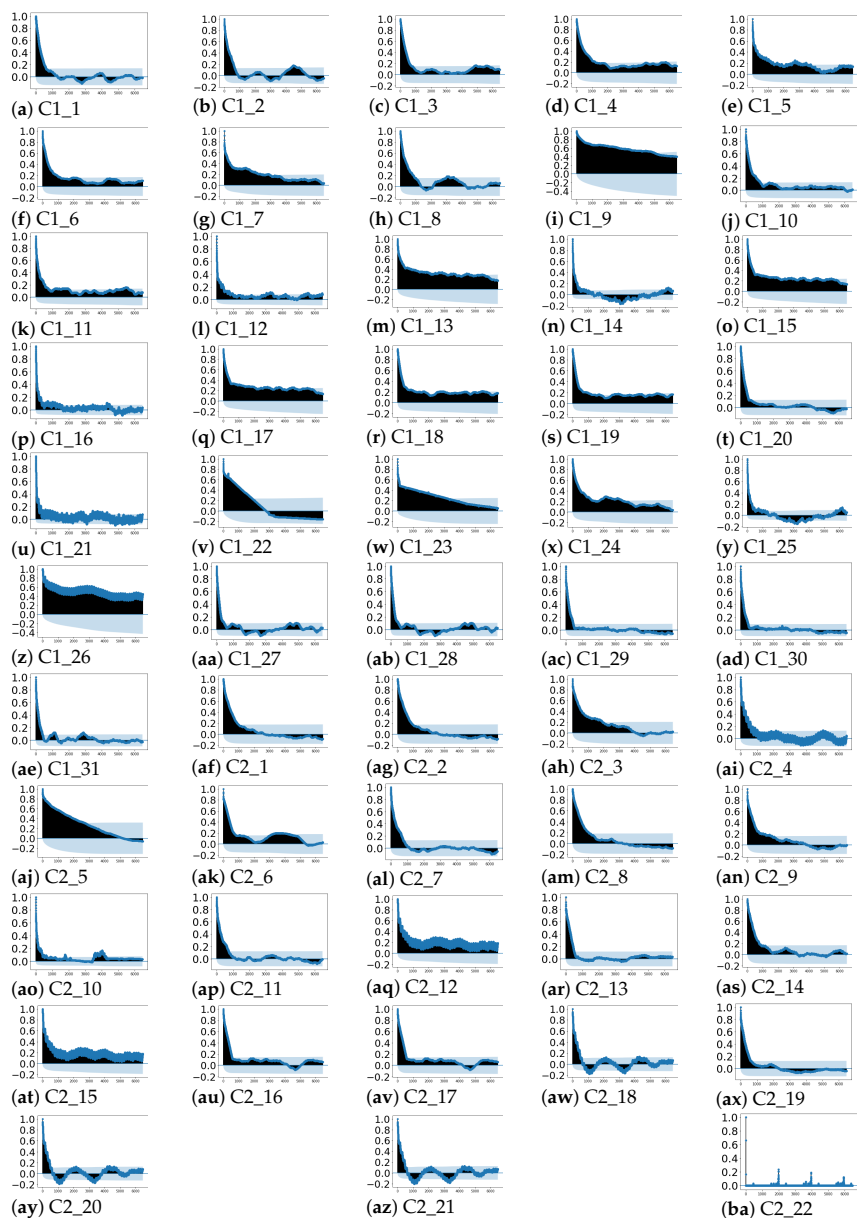


Figure 4. Auto-correlation plots of each feature of the dataset with lags up to 45 days.

**Table 2.** *p*-values obtained for the Mann–Kendall test.

	C1_1	C1_2	C1_3	C1_4	C1_5	C1_6	C1_7	C1_8	C1_9	C1_10	C1_11	C1_12	C1_13	C1_14
<i>p</i> -value	<b>0.000</b>	<b>0.000</b>	<b>0.002</b>	<b>0.001</b>	0.328	<b>0.016</b>	<b>0.007</b>	<b>0.006</b>	<b>0.000</b>	0.464	<b>0.000</b>	0.0513	<b>0.004</b>	0.373
	C1_15	C1_16	C1_17	C1_18	C1_19	C1_20	C1_21	C1_22	C1_23	C1_24	C1_25	C1_26	C1_27	C1_28
<i>p</i> -value	<b>0.003</b>	0.296	<b>0.001</b>	0.529	0.880	0.716	0.192	0.445	0.0661	<b>0.001</b>	0.220	0.686	0.201	0.196
	C1_29	C1_30	C1_31	C2_1	C2_2	C2_3	C2_4	C2_5	C2_6	C2_7	C2_8	C2_9	C2_10	C2_11
<i>p</i> -value	0.597	0.619	0.351	0.123	0.123	<b>0.000</b>	0.996	0.702	0.656	0.109	0.127	<b>0.003</b>	<b>0.024</b>	0.501
	C2_12	C2_13	C2_14	C2_15	C2_16	C2_17	C2_18	C2_19	C2_20	C2_21	C2_22			
<i>p</i> -value	0.390	<b>0.009</b>	<b>0.0001</b>	0.364	0.448	0.443	0.728	0.725	0.983	0.445	0.952			

**Table 3.** *p*-values obtained for the Augmented Dickey–Fuller test.

	C1_1	C1_2	C1_3	C1_4	C1_5	C1_6	C1_7	C1_8	C1_9	C1_10	C1_11	C1_12	C1_13	C1_14
<i>p</i> -value	<b>0.000</b>	<b>0.008</b>	<b>0.028</b>	0.241	0.185	0.100	0.087	<b>0.006</b>	0.340	0.083	0.176	0.275	0.397	<b>0.000</b>
	C1_15	C1_16	C1_17	C1_18	C1_19	C1_20	C1_21	C1_22	C1_23	C1_24	C1_25	C1_26	C1_27	C1_28
<i>p</i> -value	0.346	0.936	0.341	0.318	0.227	<b>0.032</b>	0.455	<b>0.013</b>	0.180	<b>0.006</b>	<b>0.000</b>	0.631	<b>0.018</b>	<b>0.018</b>
	C1_29	C1_30	C1_31	C2_1	C2_2	C2_3	C2_4	C2_5	C2_6	C2_7	C2_8	C2_9	C2_10	C2_11
<i>p</i> -value	<b>0.011</b>	<b>0.011</b>	<b>0.014</b>	<b>0.023</b>	<b>0.023</b>	<b>0.015</b>	<b>0.014</b>	0.050	0.488	<b>0.019</b>	<b>0.021</b>	<b>0.012</b>	0.080	<b>0.028</b>
	C2_12	C2_13	C2_14	C2_15	C2_16	C2_17	C2_18	C2_19	C2_20	C2_21	C2_22			
<i>p</i> -value	0.327	<b>0.015</b>	<b>0.048</b>	0.311	<b>0.032</b>	<b>0.030</b>	<b>0.010</b>	<b>0.002</b>	<b>0.010</b>	<b>0.001</b>	<b>0.002</b>			

**Table 4.** *p*-values obtained for the KPSS test.

	C1_1	C1_2	C1_3	C1_4	C1_5	C1_6	C1_7	C1_8	C1_9	C1_10	C1_11	C1_12	C1_13	C1_14
<i>p</i> -value	< <b>0.01</b>	< <b>0.01</b>	< <b>0.010</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	0.100
	C1_15	C1_16	C1_17	C1_18	C1_19	C1_20	C1_21	C1_22	C1_23	C1_24	C1_25	C1_26	C1_27	C1_28
<i>p</i> -value	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	<b>0.047</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>
	C1_29	C1_30	C1_31	C2_1	C2_2	C2_3	C2_4	C2_5	C2_6	C2_7	C2_8	C2_9	C2_10	C2_11
<i>p</i> -value	<b>0.049</b>	<b>0.033</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>
	C2_12	C2_13	C2_14	C2_15	C2_16	C2_17	C2_18	C2_19	C2_20	C2_21	C2_22			
<i>p</i> -value	< <b>0.01</b>	< <b>0.01</b>	<b>0.010</b>	< <b>0.01</b>	< <b>0.01</b>	<b>0.010</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	< <b>0.01</b>	0.100			

Based upon the significance level of 0.05, there is evidence for rejecting  $H_0$ , and hence defining as non-stationary 51 out of the 53 time series—marked with bold text in Table 4. C1\_14 and C2\_2 are the only ones with  $p$ -value = 0.1 > 0.05.

All in all, it can be concluded from the  $p$ -values collected in Tables 3 and 4 that the time series are non-stationary, and from Figure 4 and Tables 2 and 4 that such non-stationarity is not caused by seasonal or tendency components.

- Rolling statistics: the evolving behaviour of the series over time is depicted in Figures 5 and 6 where rolling mean and standard deviation are estimated, based on expert recommendation, with a window of length 24, i.e., 4 h. (Aiming at preserving the confidentiality of the data, Figures 5 and 6 have been scaled.)

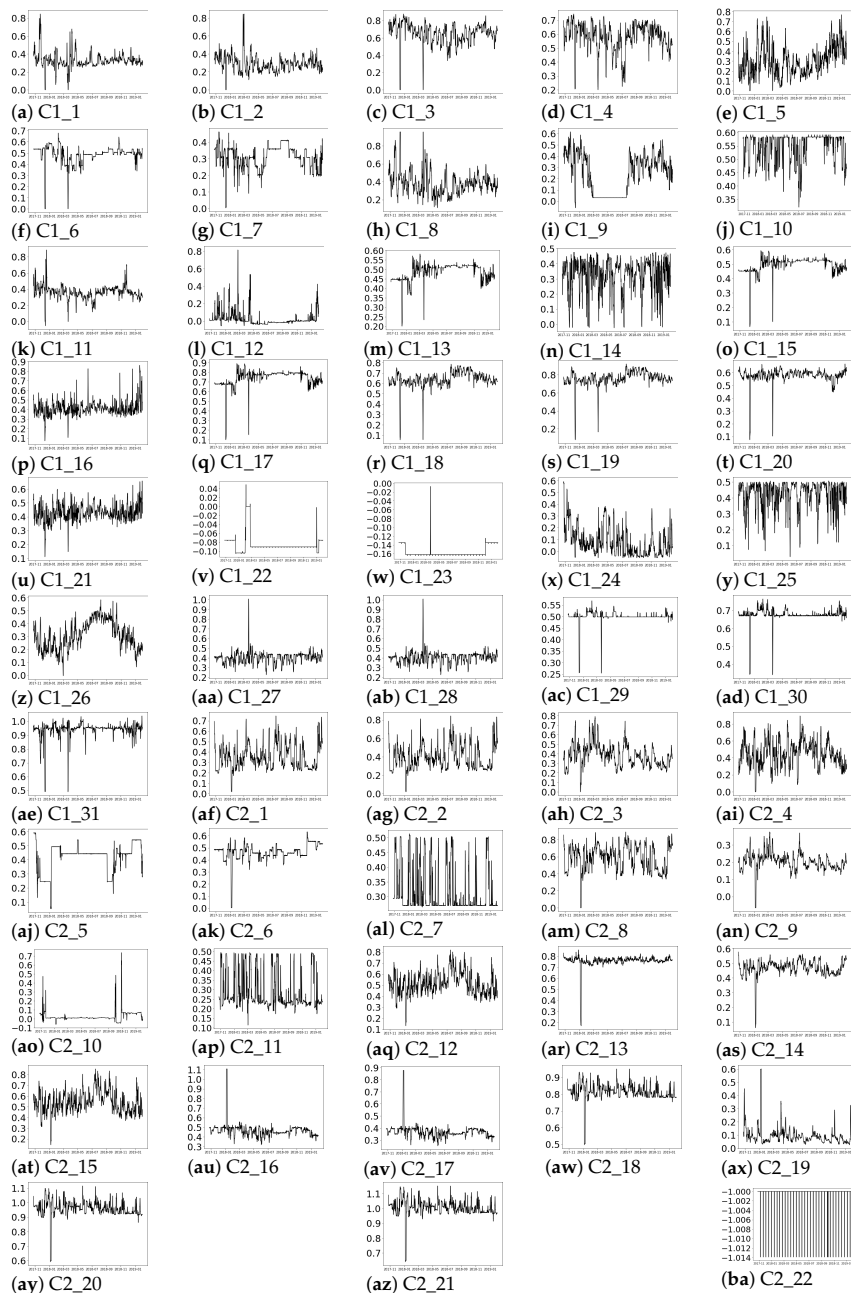


Figure 5. Rolling mean with a window of size 24.

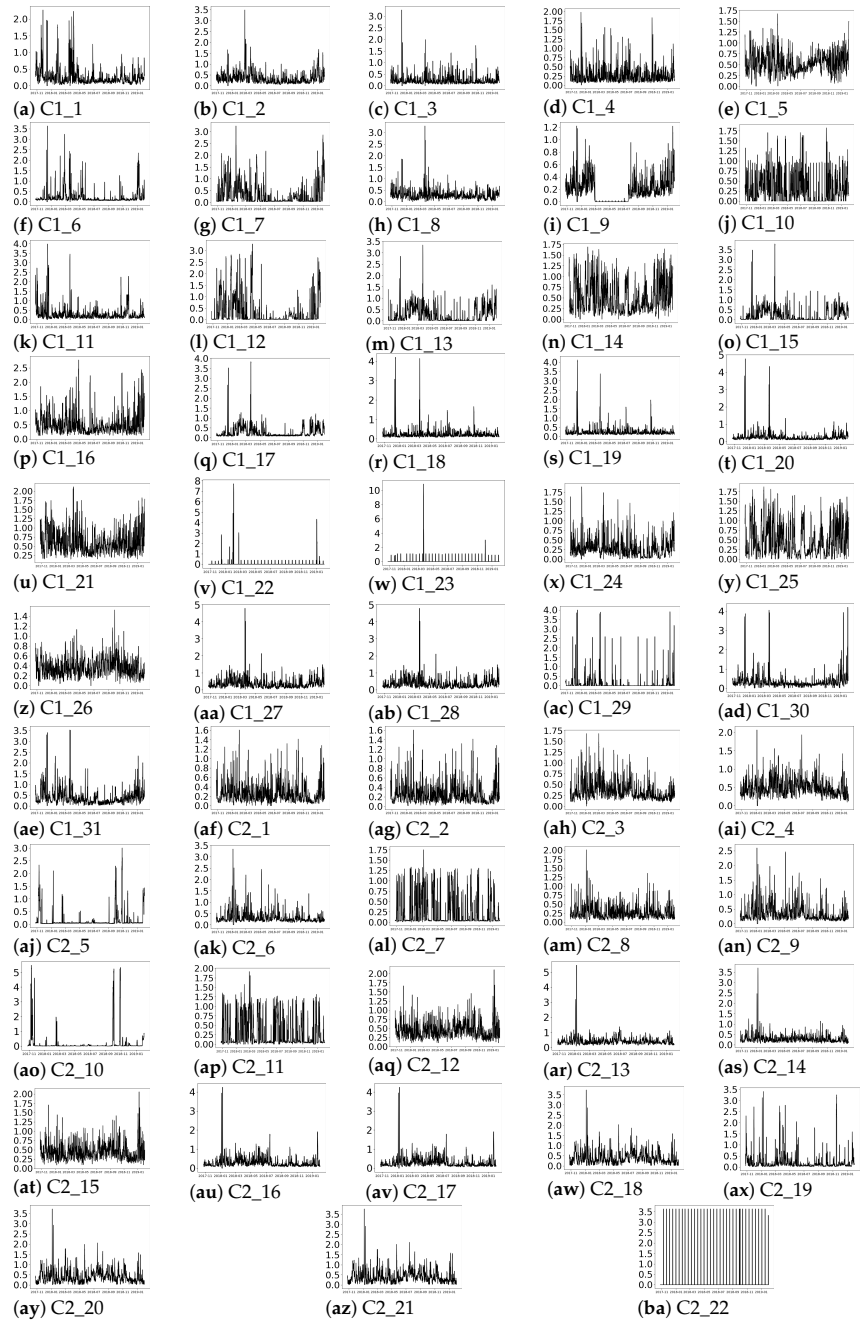


Figure 6. Rolling standard deviation with a window of size 24.

Figure 5a,ba reflect that no seasonality—in terms of repeating patterns—are observed over time, which is in accordance with the results shown in the auto-correlation plots in Figure 4. In terms of trend, as concluded from values depicted in Table 2, there was no evidence for rejecting the tendency test except for 17 time series. In the presence of trend,

the mean values of the time series would decrease or increase with time. However, in Figure 5, no stable decay or increase in the mean value is observed, except in Figure 5e,z for the period of September 2018 to 2019. Therefore, despite in Table 2, the trend hypothesis can not be rejected according to the MK test for 36 of the time series as the rolling mean does not show such trend in 34 out of those 36 time series.

Finally, Figure 6 collects the rolling standard deviation of the time series. As explained in Section 3.1, a stationary time series is developed around a constant point over time, presenting stable statistics, i.e., constant mean and standard deviation values over the time series arise. However, in Figure 6a,ba, it is observed that the rolling standard deviation presents significant peaks over time. In cases like Figure 6v,w, most of the peaks are of similar height and they appear at almost constant periods of time, but in the rest of the time series, the peaks in the rolling standard deviation are not so uniform. The variations detected over time in the rolling standard deviations values from Figure 6 reinforce the conclusions about the non-stationarity of the time series obtained with ADF and KPSS tests. Furthermore, in Figure 6c,q,r,s,ac,aj,ao, it can be observed that the values with the highest standard deviation values are found with a varying time separation of 1.5 to 4 months.

Therefore, considering (1) the results in Tables 3 and 4 regarding the non-stationarity of the time series, (2) the aforementioned non-uniformity of the rolling standard deviation along the time series, (3) expert knowledge advice, and (4) the conservative strategy, the conclusion obtained is that the window for selecting the optimal train and test set must be 3 months.

#### 4.1.2. Label Evaluation

Regarding the class samples distribution, the analysis determines that the dataset is highly imbalanced as just 15% of the samples belong to class 1. In addition, the distribution of the classes varies over time, as it can be observed in Figure 7, where up to four consecutive months with less than 1.255% of samples belonging to class 1 are found. Consequently, such periods do not fulfil the class distribution required for training the model; the samples selected for both training and testing must be representative of both classes. As stated in Section 4.1.1, the optimal window comprises 3 consecutive months of historical data. Consequently, the train/test set are obtained from consecutive periods, where approximately 15% of the samples belong to class 1.

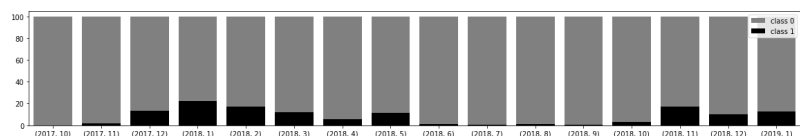


Figure 7. Percentage of samples belonging to each class by month in the recorded time.

Conversely, Figure 8 depicts a class-based box-plot. The bottom of the lower whisker and top of the upper whisker represent the minimum and maximum values, respectively and the points above or below the whisker are outliers. The top, medium and bottom of the box depict 75, 50 and 25 percentiles (Q3, Q2 and Q1), respectively. In Figure 8, it can be observed that, for each feature, if comparing the interquartile range, the box representing the samples of class 0 overlaps with the box of class 1. Q1 and Q3 estimated for class 0, and the values of class 1 are the same in features C1\_22, C1\_23 and C2\_22. The box representing class 1 is totally contained in the values that the box representing class 0 takes in features C1\_4 and C1\_11. Similarly, in C1\_6, C1\_9, C1\_10, C1\_14, C1\_16, C1\_21, C1\_25, C1\_29, C1\_31, C2\_3, C2\_6, C2\_7, C2\_9, C2\_10, C2\_19, the box representing class 0 is totally overlapped with the box of class 1. Thus, the overlapping level between the samples from Q1 to Q3 from one class with respect to the other class is up to 100%. The lowest overlapping proportion of the interquartile range of class 1 contained in the interquartile range of class 0, 12.095%, is found in C2\_5, but in this feature, the 69.494% of the interquartile range of class 0 overlaps the interquartile range of class 1. Then, the classes

are not linearly separable in any of the features. Based on these results, ML algorithms that handle the non-linearity of the features are considered in the autoML approach for creating the soft-sensor.

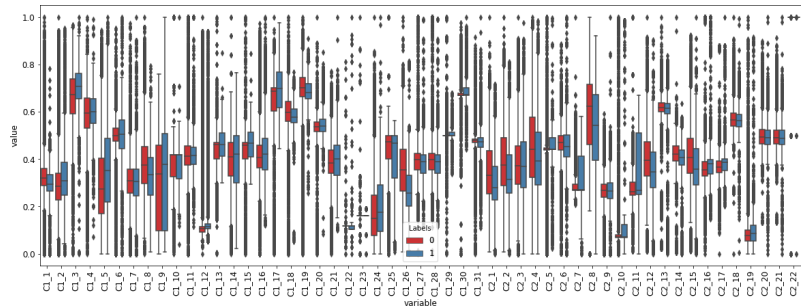


Figure 8. Box-plot of the features distinguishing between class membership of the samples.

4.2. Optimal Dataset Split

As explained in Section 3.1, if the time series are not stationary, the employment of the whole historical data can disturb the ability of the model for predicting the next time steps. Thus, a chronologically ordered subset of samples from the dataset is selected for modelling the problem. Then, based on (1) the results obtained in Section 4.1.1 about the evolution of the statistical properties of the time series over time, and (2) the results in Section 4.1.2 about the class distribution over time, the conservative period from 17 December 2017 to 15 March 2018 is selected. For the offline model training, the first two months are employed as X train and Y train, while the following one is utilised for the posterior online validation (X test, Y test).

4.3. AutoML Approach Results

In the following, the results of the application of the two-stage methodology described in Section 3.3.1 are presented.

4.3.1. Normalisation

As described in Section 3.3.1, three different normalisation methods are employed in this work. Aiming at comparing the effect of each normalisation method, the normalised ranges of training sets of the raw X and the normalised datasets  $\bar{X}^{ST}$ ,  $\bar{X}^{MM}$  and  $\bar{X}^{MAD}$ , respectively, are depicted in each row of Figure 9.

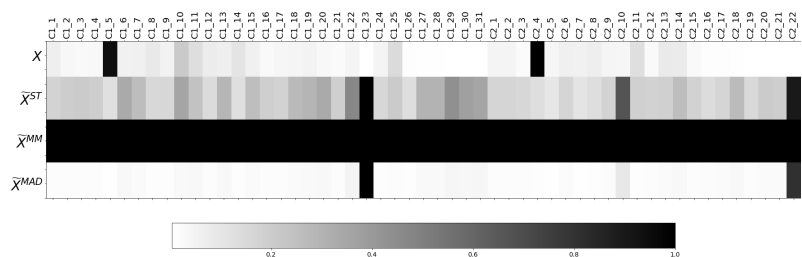


Figure 9. Normalised ranges of the features of the raw and the normalised dataset ( $X$ ,  $\bar{X}^{ST}$ ,  $\bar{X}^{MM}$  and  $\bar{X}^{MAD}$ ).

As Figure 9 shows, the dominating feature for each dataset is different depending on the normalisation method employed, and they do not match with those from the raw dataset X. The two dominant features in  $\bar{X}^{ST}$  and  $\bar{X}^{MAD}$  are C1\_23 and C2\_22, but in the  $\bar{X}^{ST}$  dataset, other features also take values higher than 0.4, while, in  $\bar{X}^{MAD}$ , the contri-



bution of the remaining features, in terms of range, are insignificant in comparison with C1\_23 or C2\_22. Therefore, from Figure 9, it is observed that the features’ dominance varies depending on the selected normalisation method. That being so, the normalisation method selection affects the features’ contribution and, therefore, conditions the ML algorithm performance.

4.3.2. Feature Weighting

Figure 10 depicts the weights  $w^P$ ,  $w^{RF}$  and  $w^{MI}$  estimated for each feature with respect to the label output by the three FW methods P, RF and MI, respectively, as described in Section 3.3.1.

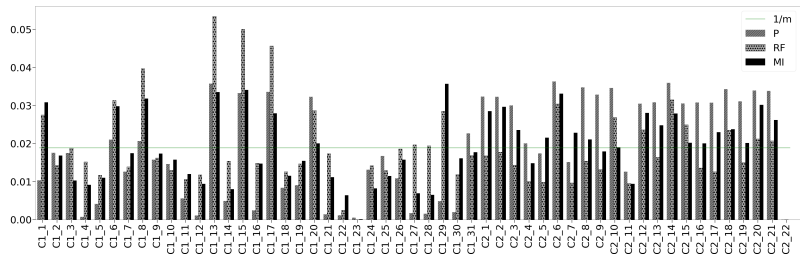


Figure 10. Feature weights estimated by each feature weighting method.

The horizontal line in Figure 10 refers to the weight each feature would have if all of them presented the same relative importance ( $1/m$ ) for estimating the output. In contrast, it is observed that the relative importance value assigned to each feature by each FW method varies. The standard deviation measures, with respect the mean, the general deviation of the weights assigned by a FW method. In the case of equal weights, the standard deviation of the weight values is zero. However, the standard deviation of the proportional weight values—the estimated weights divided by the maximum one’s value—estimated for each FW method is higher than 0.2. Thus, from the dispersion of each FW method and Figure 10, it is observed that each feature is assigned a different weight value. In addition, as Figure 10 depicts, for a given feature, the computed feature weights vary depending on the FW method employed for their calculation, such as in feature C2\_17, where the weight assigned by RF is 55.734% of the weight value assigned by P.

Table 5 collects the most relevant features according to each FW method, whereas Table 6 gathers the features with weight values lower than  $1/m$  for any FW method. Note that in Figure 10, the weight value of the most influencing features estimated by P and MI FW methods are closely followed by the weight values of other features. In contrast, the weights estimated by RF present a higher difference between the most influencing features and the rest.

Table 5. Most relevant features according to each FW method.

FW	Most Relevant Features
P	C2_6, C2_14, C1_13
RF	C1_13, C1_15, C1_16, C1_8
MI	C1_29, C1_15, C1_13, C2_6

Table 6. Features with weight value  $<1/m$  despite the FW method.

C1_2:C1_5, C1_7
C1_9:C1_12, C1_14
C1_15, C1_18, C1_19
C1_21:C1_26, C1_30, C2_11, C2_22

This work proposes to quantitatively measure the distribution of the weights as the difference between the maximum weight calculated with respect to the mean of the weights  $max(w_j^*) - \bar{w}^*$ . In addition, the cumulative absolute difference (CAD)  $\sum_{j=1}^m |w_j^* - 1/m|$  between the weights and the ideal weight ( $1/m$ ) is presented as a measure of the discriminant power of each FW method.

In the first row of Table 7, RF presents the highest difference between the central tendency weights value and the maximum one. In contrast, the obtained values for P and MI are similar. In terms of CAD, it is observed that the weights obtained with the Pearson FW method are those that most differ from the ideal ones ( $1/m$ ). Therefore, according to this method for the analysed dataset, the Pearson FW method is the most discriminant one for assigning weights to the features.

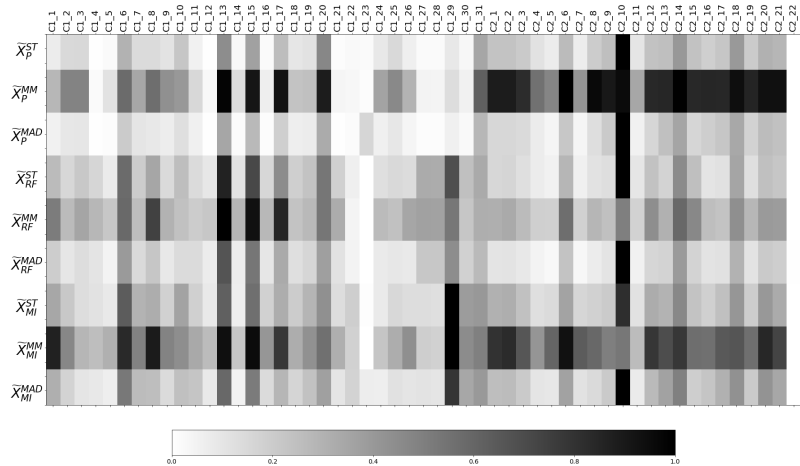
**Table 7.** Difference between the maximum and the mean weight values for each FW method.

	P	RF	MI
$max(w_j^*) - \bar{w}^*$	0.0175	0.0346	0.0168
$\sum_{j=1}^m  w_j^* - 1/m $	0.608	0.416	0.395

### 4.3.3. Two-Stage Methodology

Figure 9 shows in each row the normalised ranges of the training sets of the resulting transformed datasets from the two-stage methodology described in Section 3.3.1.

The proposed two-stage methodology states that the influence of the features for each transformed dataset is computed from the combination of each normalisation method with the weights estimated by each FW method. Such influence can be observed in Figure 11, and it is clearly proven that it varies considerably for each transformed dataset as both the normalisation method and the weight estimation clearly determines the dataset transformation.



**Figure 11.** Normalised ranges of the features of the resulting dataset after applying the proposed two-stage methodology.

Regarding the application of the weights  $w^P$  estimated with Pearson, similarly to Figure 10 where the features of column C2 present higher values than C1, in the three first rows of Figure 11, the normalised ranges of various features regarding C1 (C1\_4, C1\_12, C1\_21, C1\_22, C1\_27, C1\_28, C1\_30) are close to zero. In contrast, the cells corresponding to the transformation conducted by RF and MI methods show that, except for C1\_23, the features of C1 present a range value of at least 10% of the value of the maximum feature of the dataset.

Furthermore, in Figure 10, the weights estimated by the three FW methods for C2\_22 and C1\_23 are approximately zero. In Figure 11, C2\_22 presents a normalised range of zero, while, due to the high normalised range  $\tilde{X}^{MAD}$  in C1\_23, this feature presents a higher or equal normalised range than features that presented higher weight in Figure 10, such as C1\_27, C1\_28 or C2\_11. Similarly, recall that in Table 5, the most important features according to each FW method are collected. However, in Figure 9, it can be observed that, after the joint employment of weights estimated with P or RF and ST or MAD normalisation methods, or MI combined with MAD normalisation, the feature with the highest contribution in terms of the range is C2\_10—which does not appear in Table 5—but in Figure 9, it represents the third highest value in terms of ranges in ST and MAD. Thus, it can be concluded that the FW weights can be significantly disturbed by the normalisation methods.

#### 4.3.4. Machine Learning Algorithms and Performance Analysis

Once the transformed datasets have been obtained through the proposed two-stage methodology, the ML algorithm is applied, as described in Section 3.3.2. The precision results obtained over the entire month that comprise the test sets by each ML algorithm with the optimal hyper-parameters selected by the GS are collected in Table 8.

**Table 8.** Precision reached by each ML algorithm over the raw, normalised and transformed datasets.

Algorithm	Raw	Normalisation				Proposed Methodology								
	X	$\tilde{X}^{ST}$	$\tilde{X}^{MM}$	$\tilde{X}^{MAD}$	$\tilde{X}_P^{ST}$	$\tilde{X}_P^{MM}$	$\tilde{X}_P^{MAD}$	$\tilde{X}_{RF}^{ST}$	$\tilde{X}_{RF}^{MM}$	$\tilde{X}_{RF}^{MAD}$	$\tilde{X}_{MI}^{ST}$	$\tilde{X}_{MI}^{MM}$	$\tilde{X}_{MI}^{MAD}$	
QDA	24.414	62.304	64.286	61.340	0.000	38.506	40.909	53.548	72.973	47.689	90.000	0.000	<b>100</b>	
KNN	27.551	23.192	40.554	26.359	41.429	39.370	42.529	24.724	38.998	<b>43.416</b>	35.057	32.113	37.956	
SVC	56.897	0.000	7.368	0.000	22.562	16.068	20.564	52.250	16.333	<b>65.079</b>	21.914	16.071	24.145	
RID	81.507	38.517	86.957	51.598	22.938	98.734	54.028	20.511	<b>100</b>	51.598	21.807	96.000	51.835	
LOG	90.164	92.029	0.000	<b>100</b>	97.872	0.000	85.714	80.000	<b>100</b>	90.698	92.381	0.000	75.000	
MLP	Max	100	82.178	84.647	83.974	88.587	93.878	77.500	78.599	100	75.646	85.976	91.509	75.954
	Mean	34.595	51.622	68.631	58.136	80.558	82.055	73.460	71.364	<b>95.180</b>	71.675	76.591	73.593	72.384
	std	36.386	20.673	11.569	19.943	4.064	5.892	2.066	2.014	3.809	2.049	3.601	6.334	2.106
	Min	0.000	18.171	38.836	18.825	74.717	72.549	69.283	68.910	87.500	65.549	72.852	66.997	68.506
SGD	Max	26.606	46.868	18.929	75.862	71.795	0.000	81.022	41.640	0.000	44.660	23.343	0.000	30.334
	Mean	8.013	42.328	13.236	41.045	34.554	0.000	<b>42.824</b>	37.842	0.000	41.662	18.144	0.000	26.709
	std	6.171	2.028	1.850	13.979	13.019	0.000	10.755	1.365	0.000	1.465	2.359	0.000	1.245
	Min	0.000	38.636	10.304	17.804	15.139	0.000	24.967	35.431	0.000	39.130	12.405	0.000	24.194

As described throughout this paper, each normalisation method transforms a given dataset differently. In addition, the application of weights calculated by a particular FW method is affected by the normalisation factors employed to normalise the dataset. Then, in order to experimentally validate it, Table 8 collects the precision reached by different ML algorithms from the raw and the normalised datasets as well as from the application of the two-stage methodology. As depicted in Table 8 and remarked with bold text, the proposed two-stage methodology outperforms, in every selected ML algorithm, the obtained results by the raw and normalised datasets. For example, in QDA the precision increases from 64.286% to 90% and 100% with  $\tilde{X}_{MI}^{ST}$  and  $\tilde{X}_{MI}^{MAD}$ , respectively. In the case of MLP, from a mean precision value of 68.631% in  $\tilde{X}^{MM}$ , the two-stage methodology obtains mean precision values higher than 71% for all the combinations, reaching 95.180% of the mean precision value with the  $\tilde{X}_{RF}^{MAD}$  dataset. Regarding the results obtained from applying the two-stage methodology by different FW methods, the RF method obtains the best precision results for KNN, SVC, RID, LOG and MLP algorithms. These obtained values are closely followed by the reached ones with the P FW method in KNN, RID and LOG ML algorithms. In contrast, the P FW method reaches the maximum precision value

only for the SGD ML algorithm, and the MI FW method uniquely outperforms in the QDA ML algorithm compared to the results obtained by the other two FW methods. As described in Section 3.3.1, RF is the only FW method included in this work that considers all the features conjointly, while P and MI independently calculate each feature’s relative importance. Furthermore, note that the FW methods that obtain better and worst results for this problem are RF and MI, respectively, being both information theory-based methods. In contrast, the statistical-based method P reaches similar precision values to RF. That being so, due to the intrinsic characteristics of the FW methods formulation, P, RF and MI are considered the most suitable ones to include in the autoML approach.

The autoML approach presented in this paper selects the best configuration among different well-known normalisation and FW preprocessing methods and various commonly used ML algorithms to create a reliable soft-sensor in terms of precision. More concretely, the models with precision values higher than 95% are preselected for further analysis. Thus, QDA with  $\tilde{X}_{MI}^{MAD}$ , RID with  $\tilde{X}_P^{MM}$ ,  $\tilde{X}_{RF}^{MM}$  and  $\tilde{X}_{MI}^{MM}$  datasets, and LOG with  $\tilde{X}_P^{ST}$  and  $\tilde{X}_{RF}^{MM}$  datasets are chosen as possible soft-sensors. Table 9 collects the percentage of precision and recall reached by each selected model for different time horizons from the month that comprises the test set.

**Table 9.** Percentage of precision and recall obtained by each preselected approach for increasing temporal horizons.

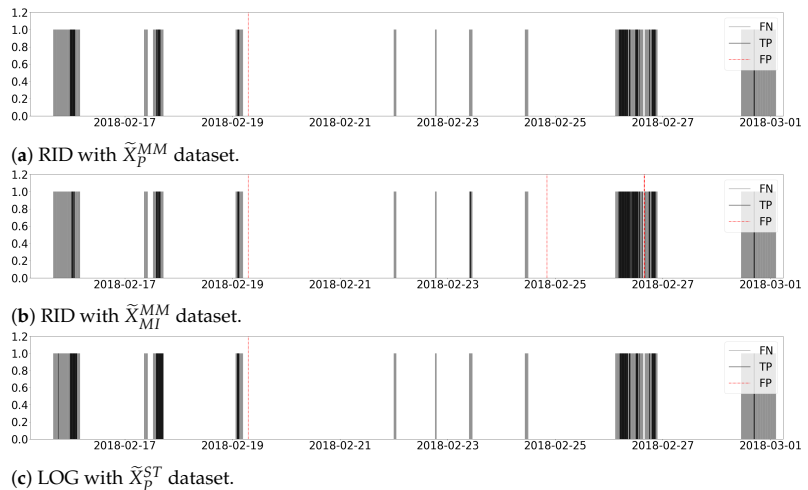
		Prediction Horizon				Prediction Horizon			
		1 W	2 W	3 W	4 W	1 W	2 W	3 W	4 W
Precision	0	100	100	100	100	Precision 96.667	98.734	98.734	98.734
Recall	0	0.277	0.161	0.154	0.154	Recall 21.168	21.607	12.52	11.982
(a) QDA for $X_{MI}^{MAD}$					(b) RID for $X_P^{MM}$				
		Prediction Horizon				Prediction Horizon			
		1 W	2 W	3 W	4 W	1 W	2 W	3 W	4 W
Precision	0	100	100	100	100	Precision 95.833	96	96	96
Recall	0	3.601	2.087	1.997	1.997	Recall 16.788	26.593	15.409	14.747
(c) RID for $X_{RF}^{MM}$					(d) RID for $X_{MI}^{MM}$				
		Prediction Horizon				Prediction Horizon			
		1 W	2 W	3 W	4 W	1 W	2 W	3 W	4 W
Precision	97.872	98.925	98.925	97.872	97.872	Precision 100	100	100	100
Recall	33.577	25.485	14.767	14.132	14.132	Recall 1.46	11.634	6.742	6.452
(e) LOG for $X_P^{ST}$					(f) LOG for $X_{RF}^{MM}$				

Table 9 shows that, for the different time horizons, the preselected approaches reliably predict the samples that do not fulfil the specification requirements. Hence, these approaches provide a high degree of confidence to the operator when changing the process operation. However, from the second week of test sets, Table 9 displays a significant decay in the percentage of recall estimated by the preselected approaches. In fact, after two weeks, they all start failing to detect more than 85% of the samples of class 1. The time series non-stationarity stated in Section 4.1 justifies the performance loss along the time and the need of adaptive methods that retrain the model with respect to the drifts in the process. Then, despite the conservative strategy described in Section 4.2 for the train/test set window length selection of a maximum of three months, the following analyses focus on the first two weeks of the test set before the drift.

Figure 12 depicts the graphical representation of the models with recall higher than 20% from Table 9. The grey vertical lines represent the False Negative (FN) samples, and the black vertical lines represent the True Positives (TP) samples. Finally, the vertical red dash-dotted lines, with a length of 1.2, are the False Positive (FP) samples that the

soft-sensor expects to minimise. Thus, Figure 12 displays the reliability of the model for correctly classifying samples from class 1, in spite of not being able to detect all the improvable samples.

The refinery's main interest is to complement the operators' decision-making with a soft-sensor that reliably detects samples of class 1 to adjust the process if necessary, minimising the operational changes when the refinery is correctly processing samples that fulfil the specifications requirements. Then, from Table 9 and Figure 12, the model resulting from the LOG ML algorithm over the  $\tilde{X}_p^{ST}$  transformed dataset is selected for creating the soft-sensor as it has a good trade-off between precision and recall.



**Figure 12.** Classification results of the selected models.

Notice that the soft-sensor estimates a new virtual measurement every 10 min given the dynamics of the change of the percentage of pentanes from adequate (class 0) to improvable (class 1). However, once the percentage of pentanes transits to class 1, from the domain expert's perspective, the required operational changes would be applied (1) under an improvable regime persisting during a significant period of time, as next detailed in Section 4.4; or (2) under the operator's consideration based on the operational variables information analysis after the first alarm from the soft-sensor. This is consistent with the decision of using the precision metric in the training process of the proposed autoML approach.

In Figure 12, the resultant subproduct that does not meet the constraints (class 1) regime in 15 February 2018 persists for 11 h and 50 min. In this case, the selected soft-sensor creates the first alarm 130 min after the first improvable level occurs, which results in an improvement of 270 min with respect to the systems that currently operate in the refinery. Similarly, on 17, 19 and 26 February 2018 it takes only 80, 40 and 120 min, respectively, for the mentioned soft-sensor to detect the subproduct quality deviation. Then, given the high reliability of the presented soft-sensor, the operator can confidently apply high-cost operational changes in order to reduce the disturbances due to an improvable percentage of pentanes.

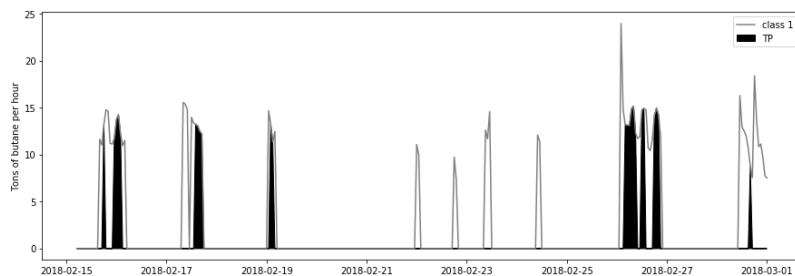
#### 4.4. Profit per Hour Provided by the Soft-Sensor

The logistic regression algorithm applied to the data set transformed by the two-stage methodology based on ST and P is selected for the soft sensor. Although the recall obtained for a two-week time horizon is only 25.485%, the precision is 98.925%. Therefore, the soft sensor is highly reliable in reporting that a sample does not meet the specification requirements. The soft-sensor is created based on the operational information from the

top of columns C1 and C2, as depicted in Figure 1, recorded 400 min before the refining process ends. Thus, the operators can react early by adjusting the process at Merox or at the debutanizer column in order to recover the resultant subproduct quality.

The refineries operate with a high quantity of material. Thereupon, even a deviation of the requirements for a short time involves a high impact on the refinery profit. Next, the economic profit derived from the soft-sensor detection of butane that does not fulfil the specification requirements is calculated.

The grey line in Figure 13 depicts the total amount of butane per hour resulting from the distilling process described in Section 3 that does not fulfil the specification requirements. The black area of Figure 13 represents the amount of butane that does not meet the specifications correctly detected by the soft-sensor. The quantity of butane resulting from the distilling process is calculated based on data from the refinery. For the units conversion, from the  $\text{m}^3/\text{l}$  of butane flow measured at the end of the unit chain to the tons of butane (Figure 13) utilised to calculate the final profit, a product density value equal to 0.575 kg/L is employed according to the refinery's laboratory analysis conducted on real data from February of 2018. As observed in Figure 13, in some hours, up to 14.56 tons of butane does not meet the specification requirements, which forces the refinery to re-inject such subproduct in the distillation process until fulfilling the specification, which ultimately results in a decrease in the amount of butane to sell. However, a prompt prediction of the butane quality in terms of percentage of pentanes allows to readjust the process and reduce the profit losses.



**Figure 13.** Tons of butane per hour that do not fulfil the specification requirements.

Due to the time-frame needed to reach the new operation point, and considering a conservative approach, only the benefit over 80% of the correctly detected improvable butane is calculated. Thus, in the analysed period and discarding 20% of the detected improvable butane, 258.22 tons of butane that do not fulfil the specification requirements are correctly detected by the proposed soft-sensor. Furthermore, each refinery sets its own sale price for each subproduct. In the refinery from where the data come, the sale price of a ton of butane in February of 2018 was 459.74\$. Thus, in the studied two weeks, the profit derived from the online prediction of the subproduct quality with the proposed soft-sensor would be a total of 111,939.35\$.

## 5. Conclusions

This work employs real data from a refinery of the Basque Country, and it proposes a soft-sensor to complement the operators' decision making model by classifying the percentage of pentanes in butane in the bottom of the debutanizer column 400 min in advance based on process information from the top of two naphtha stabilisers columns. The analysis of the different configurations of preprocessing and modelling methods to create a soft-sensor is difficult and time-consuming. Thus, this work proposes an autoML approach that automatically searches for the best configuration among different normalisation, FW preprocessing methods, and commonly employed ML algorithms to select the best configuration among different combinations of methods for a given dataset.

The autoML approach's preprocessing step employs a novel two-stage methodology that combines normalisation and feature weighting to transform the input space intelligently. The two-stage methodology aims at avoiding features dominance through normalisation methods. FW methods account for the relative importance each feature presents at estimating the real label for improving the classification performance. As proven through this work, the selection of a normalisation method conditions the feature weighting values' impact at transforming the features, which ultimately conditions the ML algorithm results. Then, three widely utilised normalisation methods, standardisation (ST), min–max normalisation (MM) and median absolute deviation normalisation (MAD), are considered for the two-stage methodology. Two information-theory-based approaches, Random Forest (RF) and Mutual Information (MI), and one statistical method, the Adapted Pearson correlation (P), are applied regarding the feature weighting methods.

As “no free lunch theorem” states, there is no one model that works best for every problem. Thus, for the modelling stage of the autoML approach, seven well-known classification algorithms (QDA, KNN, SVC, RID, LOG, MLP and SGD) are included, among which we select the most appropriate one for the problem at hand.

The autoML approach presented in this work selects the configuration among different preprocessing techniques and ML algorithms that create the most reliable model. For the analysed industrial case, the ST normalisation method with Adapted Pearson correlation-based feature weights and the Logistic regression ML algorithm is selected by the autoML approach as the best configuration to create the soft-sensor. With such configuration, the soft-sensor obtains a precision of 98.925% at predicting the resultant product of improvable quality.

In addition to the classification results obtained at testing the model over two chronologically followed weeks, the estimated profit from applying the developed soft-sensor is presented. Thus, a saving of 111,939.35\$ would have resulted from the next two weeks' classification results.

Along with the promising results obtained by the interpretable proposed approach of combining the two-stage methodology with shallow ML algorithms, in the future, adaptive techniques for online concept drift detection and automatic adaptation of the classification model will be investigated. Furthermore, for the problem at hand, due to the non-stationarity of the time series and the need of selecting a subset of data for the training set, there is no need to remove trend and seasonality. However, as future work, in the case of stationary time series, trend and seasonality removal will be included in the autoML approach. In addition, the proposed autoML has been designed for supervised scenarios. However, in some industrial problems, the labels are difficult to obtain. Thus, in future works, the authors aim to investigate a new approach to handle processes with scarce labels.

**Author Contributions:** Conceptualisation, I.N.-A., I.L.-T., D.M. and E.P.; methodology, I.N.-A., I.L.-T. and E.P.; software, I.N.-A.; formal analysis, I.N.-A.; investigation, I.N.-A.; resources, I.L.-T. and L.O.; writing—original draft preparation, I.N.-A.; writing—review and editing, I.N.-A., I.L.-T., D.M., E.P. and L.O.; visualisation, I.N.-A.; supervision, I.L.-T. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Acknowledgments:** This work has been supported by a DATAinc fellowship (48-AF-W1-2019-00002) and a TECNALIA Research and Innovation PhD Scholarship. Furthermore, this work is part of the 3KIA project (KK-2020/00049), funded by the ELKARTEK program of the SPRI-Basque Government.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Zhang, R.; Jin, Q. Design and Implementation of hybrid modeling and PFC for oxygen content regulation in a coke furnace. *IEEE Trans. Ind. Inform.* **2018**, *14*, 2335–2342. [CrossRef]
2. Wang, K.; Shang, C.; Yang, F.; Jiang, Y.; Huang, D. Automatic hyper-parameter tuning for soft sensor modeling based on dynamic deep neural network. In Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC), Banff, AB, Canada, 5–8 October 2017; pp. 989–994.
3. Siddharth, K.; Pathak, A.; Pani, A.K. Real-time quality monitoring in debutanizer column with regression tree and ANFIS. *J. Ind. Eng. Int.* **2019**, *15*, 41–51. [CrossRef]
4. Bernus, P.; Noran, O. *Data Rich-but Information Poor*; Springer: Cham, Switzerland, 2017; pp. 206–214.
5. Xu, X.; Liu, Q.; Ding, J. Gasoline dry point prediction of fractionation processes using dynamic inner partial least squares. In Proceedings of the 11th Asian Control Conference (ASCC), Gold Coast, Australia, 17–20 December 2017; pp. 1438–1442.
6. Li, C.; Zhao, D.; Liu, Y.; Li, J.; Wang, C.; Gao, X. Research on the Soft-sensing Modeling Method for the Naphtha Dry Point of an Atmospheric Tower. In Proceedings of the 37th Chinese Control Conference (CCC), Wuhan, China, 25–27 July 2018; pp. 8060–8066.
7. Chan, L.L.T.; Wu, Q.Y.; Chen, J. Dynamic soft sensors with active forward-update learning for selection of useful data from historical big database. *Chemom. Intell. Lab. Syst.* **2018**, *175*, 87–103. [CrossRef]
8. Wang, D.; Liu, J.; Srinivasan, R. Data-driven soft sensor approach for quality prediction in a refining process. *IEEE Trans. Ind. Inform.* **2009**, *6*, 11–17. [CrossRef]
9. Zhang, X.; Zou, Y.; Li, S.; Xu, S. A weighted auto regressive LSTM based approach for chemical processes modeling. *Neurocomputing* **2019**, *367*, 64–74. [CrossRef]
10. Fan, Y.; Tao, B.; Zheng, Y.; Jang, S.S. A Data-Driven Soft Sensor Based on Multilayer Perceptron Neural Network with a Double LASSO Approach. In *IEEE Transactions on Instrumentation and Measurement*; IEEE: Piscataway, NJ, USA, 2019.
11. Parvizi Moghadam, R. Online Monitoring for Industrial Processes Quality Control Using Time Varying Parameter Model. *Int. J. Eng.* **2018**, *31*, 524–532.
12. Mohler, I.; Ujević Andrijić, Ž.; Bolf, N. Soft sensors model optimization and application for the refinery real-time prediction of toluene content. *Chem. Eng. Commun.* **2018**, *205*, 411–421. [CrossRef]
13. Yuan, X.; Huang, B.; Wang, Y.; Yang, C.; Gui, W. Deep learning-based feature representation and its application for soft sensor modeling with variable-wise weighted SAE. *IEEE Trans. Ind. Inform.* **2018**, *14*, 3235–3243. [CrossRef]
14. Lasi, H.; Fettke, P.; Kemper, H.G.; Feld, T.; Hoffmann, M. Industry 4.0. *Bus. Inf. Syst. Eng.* **2014**, *6*, 239–242. [CrossRef]
15. Angelopoulos, A.; Michailidis, E.T.; Nomikos, N.; Trakadas, P.; Hatziefremidis, A.; Voliotis, S.; Zahariadis, T. Tackling faults in the industry 4.0 era—A survey of machine-learning solutions and key aspects. *Sensors* **2020**, *20*, 109. [CrossRef]
16. Lima, F.S.; Alves, V.M.C.; Araujo, A.C.B. Metacontrol: A Python based application for self-optimizing control using metamodels. *Comput. Chem. Eng.* **2020**, *140*, 106979. [CrossRef]
17. Preuveeneers, D.; Tsingenopoulos, I.; Joosen, W. Resource usage and performance trade-offs for machine learning models in smart environments. *Sensors* **2020**, *20*, 1176. [CrossRef]
18. Sun, W.; Braatz, R.D. Smart process analytics for predictive modeling. *Comput. Chem. Eng.* **2021**, *144*, 107134. [CrossRef]
19. Xin, D.; Wu, E.Y.; Lee, D.J.L.; Salehi, N.; Parameswaran, A. Whither AutoML? Understanding the Role of Automation in Machine Learning Workflows. *arXiv* **2021**, arXiv:2101.04834.
20. Aquino-Britez, D.; Ortiz, A.; Ortega, J.; León, J.; Formoso, M.; Gan, J.Q.; Escobar, J.J. Optimization of Deep Architectures for EEG Signal Classification: An AutoML Approach Using Evolutionary Algorithms. *Sensors* **2021**, *21*, 2096. [CrossRef]
21. Zöllner, M.A.; Huber, M.F. Benchmark and Survey of Automated Machine Learning Frameworks. *J. Artif. Intell. Res.* **2021**, *70*, 409–472. [CrossRef]
22. He, X.; Zhao, K.; Chu, X. AutoML: A Survey of the State-of-the-Art. *Knowl. Based Syst.* **2021**, *212*, 106622. [CrossRef]
23. Lee, D.J.L.; Macker, S.; Xin, D.; Lee, A.; Huang, S.; Parameswaran, A.G. A Human-in-the-loop Perspective on AutoML: Milestones and the Road Ahead. *IEEE Data Eng. Bull.* **2019**, *42*, 59–70.
24. Truong, A.; Walters, A.; Goodsitt, J.; Hines, K.; Bruss, C.B.; Farivar, R. Towards automated machine learning: Evaluation and comparison of AutoML approaches and tools. In Proceedings of the IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI), Portland, OR, USA, 4–6 November 2019; pp. 1471–1479.
25. Ito, E.H.; Secchi, A.R.; Gomes, M.V.; Paiva, C.R. Development of a gas composition soft sensor for distillation columns: A simplified model based and robust approach. In *Computer Aided Chemical Engineering*; Elsevier: Amsterdam, The Netherlands, 2018; Volume 44, pp. 661–666.
26. Fortuna, L.; Graziani, S.; Xibilia, M.G. Soft sensors for product quality monitoring in debutanizer distillation columns. *Control Eng. Pract.* **2005**, *13*, 499–508. [CrossRef]
27. Bishop, C.M. *Pattern Recognition and Machine Learning*; Springer Science and Business Media LLC: New York, NY, USA, 2006.
28. BOE-A-2006-2779. Real Decreto 61/2006, de 31 de Enero, por el que se Determinan las Especificaciones de Gasolinas, Gasóleos, Fuelóleos y Gases Licuados del Petróleo y se Regula el uso de Determinados Biocarburantes. Minist. Ind., Turismo Comer. (Spain). 2015. Available online: <https://www.boe.es/buscar/act.php?id=BOE-A-2006-2779&tn=1&p=20060928> (accessed on 7 June 2021).



29. Brockwell, P.J.; Brockwell, P.J.; Davis, R.A.; Davis, R.A. *Introduction to Time Series and Forecasting*; Springer International Publishing: Cham, Switzerland, 2016.
30. Parmezan, A.R.S.; Souza, V.M.; Batista, G.E. Evaluation of statistical and machine learning models for time series prediction: Identifying the state-of-the-art and the best conditions for the use of each model. *Inf. Sci.* **2019**, *484*, 302–337. [[CrossRef](#)]
31. Vecchia, A.; Ballerini, R. Testing for periodic autocorrelations in seasonal time series data. *Biometrika* **1991**, *78*, 53–63. [[CrossRef](#)]
32. Makridakis, S.; Spiliotis, E.; Assimakopoulos, V. Statistical and Machine Learning forecasting methods: Concerns and ways forward. *PLoS ONE* **2018**, *13*, e0194889. [[CrossRef](#)] [[PubMed](#)]
33. Hamed, K.H.; Rao, A.R. A modified Mann-Kendall trend test for autocorrelated data. *J. Hydrol.* **1998**, *204*, 182–196. [[CrossRef](#)]
34. Mushtaq, R. Augmented Dickey Fuller Test. 2011. Available online: <http://dx.doi.org/10.2139/ssrn.1911068> (accessed on 8 June 2021).
35. Kwiatkowski, D.; Phillips, P.C.; Schmidt, P.; Shin, Y. Testing the null hypothesis of stationarity against the alternative of a unit root. *J. Econom.* **1992**, *54*, 159–178. [[CrossRef](#)]
36. García, S.; Luengo, J.; Herrera, F. *Data Preprocessing in Data Mining*; Springer International Publishing: Cham, Switzerland, 2015.
37. Niño-Adan, I.; Landa-Torres, I.; Portillo, E.; Manjarres, D. Analysis and Application of Normalization Methods with Supervised Feature Weighting to Improve K-means Accuracy. In *International Workshop on Soft Computing Models in Industrial and Environmental Applications*; Springer Nature: Cham, Switzerland, 2019; pp. 14–24.
38. Leys, C.; Ley, C.; Klein, O.; Bernard, P.; Licata, L. Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median. *J. Exp. Soc. Psychol.* **2013**, *49*, 764–766. [[CrossRef](#)]
39. Singh, D.; Singh, B. Investigating the impact of data normalization on classification performance. *Appl. Soft Comput.* **2020**, *97*, 105524. [[CrossRef](#)]
40. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [[CrossRef](#)]
41. Louppe, G.; Wehenkel, L.; Sutter, A.; Geurts, P. Understanding variable importances in forests of randomized trees. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 5–10 December 2013; pp. 431–439.
42. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
43. Kraskov, A.; Stögbauer, H.; Grassberger, P. Estimating mutual information. *Phys. Rev. E* **2004**, *69*, 066138. [[CrossRef](#)]
44. Ross, B.C. Mutual information between discrete and continuous data sets. *PLoS ONE* **2014**, *9*, e87357.

