Renewable Energy 201 (2022) 157–178

Contents lists available at ScienceDirect

Renewable Energy

journal homepage: www.elsevier.com/locate/renene

A hierarchical classification/regression algorithm for improving extreme wind speed events prediction

C. Peláez-Rodríguez [a,*], J. Pérez-Aracil [a], D. Fister [a], L. Prieto-Godino [b], R.C. Deo [c], S. Salcedo-Sanz [a,c]

[a] Department of Signal Processing and Communications, Universidad de Alcalá, Alcalá de Henares, 28805, Spain
[b] Department of Digitalization and O&N tools, Iberdrola, Spain
[c] School of Mathematics, Physics and Computing, University of Southern Queensland, Springfield, QLD, 4300, Australia

ARTICLE INFO

Keywords:
Wind speed extremes
Wind extremes prediction
Hierarchical classification/regression schemes
Wind energy
Machine learning

ABSTRACT

A novel method for prediction of the extreme wind speed events based on a Hierarchical Classification/Regression (HCR) approach is proposed. The idea is to improve the prediction skills of different Machine Learning approaches on extreme wind speed events, while preserving the prediction performance for steady events. The proposed HCR architecture rests on three distinctive levels: first, a data preprocessing level, where training data are divided into clusters and accordingly associated labels. At this point, balancing techniques are applied to increase the significance of clusters with poorly represented wind gusts data. At a second level of the architecture, the classification of each sample into the corresponding cluster is carried out. Finally, once we have determined the cluster a sample belongs to, the third level carries out the prediction of the wind speed value, by using the regression model associated with that particular cluster. The performance of the proposed HCR approach has been tested in a real database of hourly wind speed values in Spain, considering Reanalysis data as predictive variables. The results obtained have shown excellent prediction skill in the forecasting of extreme events, achieving a 96% extremes detection, while maintaining a reasonable performance in the non-extreme samples. The performance of the methods has also been assessed using forecast data (GFS) as predictors.

1. Introduction

Renewables are clean, inexhaustible and increasingly competitive energy resources. They are getting massive attention from the energy production authorities, countries and energy companies in the last decades, in order to reduce our dependence on fossil fuels. Among renewable energy sources, wind energy is one of the most rapidly growing and potentially useful energy worldwide [1], because of its high efficiency and resource availability together with the low pollution produced by wind farms [2]. It is also one of the most promising renewable energies in terms of penetration in the electric power system, economic impact and annual growth rate [3,4], due to its natural, cheap and clean nature. In addition, it is possible to produce energy from wind turbines at each hour of the day, and it is suitable for systems that require energy continuously [5].

As other renewable sources, its inherent disadvantages are uncertainty and intermittence [6], which induce grid instability and may lead either to lack of supply in peak hours, or wasting energy in consumption valleys. The electrical grid requires a steady, reliable and controllable electricity input source, and therefore an accurate wind power forecasting plays a key role in the integration of a large share of wind power in an electricity system [7]. Research in this area has been focused on the development of robust and reliable tools. In the literature, there are many studies on wind power forecasting using several analysis methods and over very different prediction time-horizons which have been summarized and reviewed in a number of reviews articles over the last decade [2,8–12]. Broadly speaking, these strategies can be divided into two different approaches: the first one, weather-based, relies on the study of physical phenomena to build a model (Numerical Weather Model, WNM). In the second approach, time series based, statistical algorithms are used to analyze historical wind speed data series [13,14]. A new group of methods within the time series approach, namely data mining, emerged in last decade [15] and has gained popularity among the scientific community since then [16,17], due to the good results obtained in different prediction problems.

One specific and major aspect of wind speed forecasting concerns the prediction of extreme wind speeds (EWS). These events are often

* Corresponding author.
E-mail address: cesar.pelaez@uah.es (C. Peláez-Rodríguez).

https://doi.org/10.1016/j.renene.2022.11.042
Received 27 May 2022; Received in revised form 6 September 2022; Accepted 12 November 2022
Available online 16 November 2022

responsible for the worst damages caused by wind, especially in wind farms facilities. In fact, wind farms must be restrained from operating during such events, in order to minimize the hazards involved with them. Thus, it is of crucial importance for the wind power sector, to have a proper knowledge as well as robust and reliable assessments to estimate the frequency and intensity of extreme events, not only to avoid wind turbines damage, but also to minimize cut-out events [18]. Although the majority of current approaches focus on the prediction of mean wind speed [19], in order to forecast production, several works aimed at the detection of EWS too.

A first review of classical techniques for EWS prediction was reported in [20]. More recent reviews of modern techniques, including NWM and also Machine Learning (ML) approaches have been presented in [21–23]. In [24] several ML algorithms have been applied to a problem of EWS prediction. Logistic regression, MLPs and C4.5 classification trees and CART algorithms were tested in a problem of EWS prediction at Kumeu, New Zealand. In [25] a similar problem case study was tackled for a New Zealand case, evaluating the classification trees, MLPs and Self-Organizing Maps (SOMs). In-situ measurements and data acquired between 2008 and 2012 at Kumeu site was used for this study. In [26] a problem of extreme wind prediction at the surroundings of storm cells in the USA is carried out. The problem consists in calculating the probability of extreme winds over 50 kt (25.7 m/s) in zones close to storm cells. The problem is formulated as a binary classification problem. The predictive variables considered in this case are based on radar measurements, storm motion and shape, and atmospheric soundings at the near-storm environment. Several ML models have been tested, including, logistic regression, RF, MLPs and Gradient boosting trees ensembles. In [19] an ensemble model for EWS prediction is presented. The proposed ensemble includes RF, a long–short term memory (LSTM) algorithm and Gaussian processes for regression. A comparison against each model on their own, the persistence and a gradient boosted decision tree showed the good performance of the ensemble method. Also dealing with ensemble models, in [23] a comprehensive review and comparison of eight ensemble methods based on ML for EWS forecasting is carried out. The proposed algorithms are tested in six years of data from a high-resolution ensemble prediction system of the German weather service. In [27] a SOM is proposed to analyze the meteorological origin of EWS in Australia. The SOM is used to establish the origin of Application of Self-organizing Maps to classify the meteorological origin of EWS into convective (from thunderstorms) and non-convective origin (synoptic), with different subclasses in each case. In [28], 33 year reanalysis data set is used to construct an hourly time series of nationally-aggregated wind power generation in Great Britain (GB), in order to quantify extreme wind power generation statistics, assuming a fixed and modern distribution of wind farms. In [29], the study scrutinizes future scenarios of extreme winds in Brazil by applying trend analysis techniques on a 50-year historical series of observational wind speed and meteorological parameters at 10 m height in Brazil. By applying techniques of cluster analysis it was possible to characterize six main regions with macro climatic similarities. In [30], geophysical predictors from the ERA5 reanalysis are used in conjunction with an autoregressive term in regression and ANN models with different predictors, and varying model complexity for forecasting of EWS occurrence and magnitude. Finally, in [31] a RF approach is applied to the identification of extreme wind field characteristics and associated wind-induced load effects on structures, via detection of thunderstorms. The idea is to use large databases containing high-frequency sampled continuous wind speed data, and use the shapelet transform to identify individual attributes distinctive of extreme wind events. Experiments base on the real data from 14 Mediterranean ports in Italy, Spain and France.

One of the inherent issues in forecasting the atmospheric extreme events (including EWS) resides in dealing with highly unbalanced databases, since the number of instances with extreme wind speeds often represents a minimum percentage of the total data. This problem has been mostly explored in the context of classification tasks [32]. However, the challenge we are tackling in this paper concerns a continuous predictive domain, where in addition to forecasting the presence or absence of EWS, it is also important to provide a reasonable estimation of its magnitude. The main strategy to deal with such challenge consists in the preprocessing of the datasets in order to balance the training data [33], either by performing a random undersampling of the majority classes or generating new synthetic samples for classification [34] or regression [35].

The methodology proposed in this paper for EWS prediction consists of a Hierarchical Classification/Regression (HCR) approach, where the time series training data is divided into separate subsets (or clusters) depending on the wind speed value. Each cluster of training data is employed to fit a specific regression model. Architectures with a similar idea can be found in the literature [36], in the context of ML approaches for prediction problems. In [37] it is shown that the quality of a prediction improves when an HCR model is applied, with respect to bare ML methods. Similar methods have been implemented in different fields: in [38] a joint classification-regression method has been applied to estimate remaining useful life of devices in industrial manufacturing systems. Also, in [39] device age is estimated from pictures using a hierarchical classification systems. In [40] a cluster and regression model was used to predict school dropout in higher education. In [41] HCR is implemented for predicting high performance concrete compressive strength, and in [42] a similar architecture is used to estimate the cost of manufacturing thin-film transistor liquid-crystal displays. In spite of these previous approaches, the application of HCR architectures to the detection of extremes weather events has not yet been explored in the literature to the extent of our knowledge.

Differing from the previously discussed approaches, the HCR methodology proposed in this paper consists on a three-level architecture. The first level consists of a data preprocessing step, where training data are divided into clusters and labels are added accordingly. Then, balancing techniques are applied to increase the significance of clusters with EWS, which are represented poorly in the original data. At the second level, the classification of each sample into the corresponding cluster is carried out. A variety of classifiers are trained with preprocessed labeled data after different balancing techniques are applied. Finally, this pool of classifiers is integrated into a voting classifier ensemble using a majority-voting rule. Once determined to which cluster a sample belongs to, the third level of the architecture forecasts the wind speed value, by applying the regression model that corresponds to that particular cluster. The proposed HCR approach has been implemented and tested for prediction of extreme EWS events at a wind farm in Spain. Specifically, ten years of hourly wind speed data are available at a wind farm in Western Spain, where the proposed HCR has been applied, obtaining excellent results reported in the experimental section of the paper.

The rest of the manuscript has been organized as follows: Section 2 describes some ML techniques that have been used to build the HCR approach for EWS prediction. Then, in Section 3 we describe the proposed HCR methodology, along with a brief description of the computational methods used and how to combine them to form the HCR. Section 4 presents the experimental work carried out and the results obtained in a real wind farm in Spain. Finally, Section 5 closes the paper with some final remarks and conclusions.

2. Computational methods

This section describes the details on ML balancing, classification and regression methods used in this paper to construct the proposed HCR approach. They have been summarized in Table 1.

Table 1
Summary of the implemented ML methods in data preprocessing (noted as balancing), classification and regression tasks.

Category	Method
Balancing	SMOTE Borderline-SMOTE SVM-SMOTE ADASYN K-Means SMOTE Random undersampling SMOGN
Classification	Support vector machine Random forest Gaussian Naive Bayes K-Nearest Neighbors AdaBoost Multi-layer perceptron
Regression	Linear regression Regression trees Random forest Multi-layer perceptron Extreme learning machine

2.1. SMOTE

Synthetic Minority Oversampling Technique (SMOTE) [34] is a sampling method to address classification problems with imbalanced class distribution. The key feature of this method is that it combines undersampling of the frequent classes with oversampling of the minorities. It uses an oversampling strategy that supports generating "synthetic" cases with a rare target value. For each case from the set of instances with rare values, the strategy is to randomly select one of its k-Nearest Neighbors (k-NN) from this same set. With these two instances a new example is created whose attribute values are an interpolation of the values of the two original cases.

2.2. Borderline-SMOTE

Borderline-SMOTE is a special variant of SMOTE that addresses the common drawback of most of the classification algorithms, which attempt to learn the borderline of each class as exactly as possible in the training process. The examples on the borderline and the ones nearby are more apt to be misclassified than the ones far from the borderline, and thus more important for classification. Following this rationale, the Borderline-SMOTE [43] only addresses the borderline examples of the minority class to be oversampled, as instances far from the borderline may contribute little to classification. Hence, the borderline minority examples are found out first, and then the synthetic examples are generated from them and added to the original training set.

2.3. SVM-SMOTE

SVM-SMOTE [44] focuses on generating new minority class instances near borderlines. A Support Vector Machine (SVM) classifier is trained to predict synthetic instances. This method focuses only on the minority class instances residing along the decision boundary, due to the fact that this region is the most crucial for establishing the decision boundary. Furthermore, the artificial minority instances are generated in such a way that the regions of the minority class with fewer majority class instances would be expanded by extrapolation. Otherwise the current boundary of the minority class would be consolidated by interpolation.

2.4. ADASYN

Adaptive synthetic sampling (ADASYN) [45] approach exploits a different technique, by weighting the distribution of minority class examples, according to their level of difficulty in learning. Thus, more synthetic data are generated for the minority class examples (that are harder to learn) compared to those minority examples that are easier to learn. As a result, the ADASYN approach improves learning with respect to the data distributions, reducing the bias introduced by the class imbalance and adaptively shifting the classification decision boundary toward the more difficult instances.

2.5. K-Means SMOTE

K-Means SMOTE [46,47] employs the simple and popular k-means clustering algorithm, in conjunction with SMOTE oversampling, in order to rebalance skewed datasets, avoiding the generation of noise by oversampling only in safe areas. Its focus is placed on both between-class imbalance and within-class imbalance, combating the small disjuncts problem by inflating sparse minority areas. Sample distribution is based on cluster density, generating more samples in sparse minority areas than in dense ones in order to combat within-class imbalance.

K-means SMOTE consists of three steps: clustering, filtering, and oversampling. In the clustering step, the input space is clustered into k groups using k-means clustering. The filtering step selects clusters for oversampling, retaining those with a high proportion of minority class samples. It then distributes the number of synthetic samples to generate, assigning more samples to clusters where minority samples are sparsely distributed. Finally, in the oversampling step, SMOTE is applied in each selected cluster to achieve the target ratio of minority and majority instances.

2.6. Random undersampling

Random undersampling is among the simplest strategies to correct for data imbalance. The majority class is undersampled by randomly removing instances until the data is balanced. Removing data will obviously reduce the strain on storage and also improve time complexity. However, it might lead to a loss of useful information as reported in previous study [48].

2.7. SMOGN

SMOGN [35] aims to deal with imbalanced regression problems where the most important cases to the user are poorly represented in the available data. It combines random undersampling with two oversampling techniques: SmoteR [49] and introduction of Gaussian Noise. SMOGN generates new synthetic examples with SmoteR only when the seed example and the k-NN selected are "close enough" and uses the introduction of Gaussian Noise when the two examples are "more distant".

2.8. Support vector machine

The SVM [50] is a supervised learning algorithm used for many classification problems. The objective of the SVM algorithm is to find a hyperplane that, to the best degree possible, separates data points of one class from those of another class. "Best" is defined as the hyperplane with the largest margin between the two classes. Linearly separable problems allow the algorithm to exploit a linear hyperplane, but for most practical problems a more complex hyperplanes with soft margins that allow a small number of misclassifications are exploited.

Support vectors refer to a subset of the training instances that identify the location of the separating hyperplane. The standard SVM algorithm is formulated for binary classification problems and multiclass problems are typically reduced to a series of binary ones.

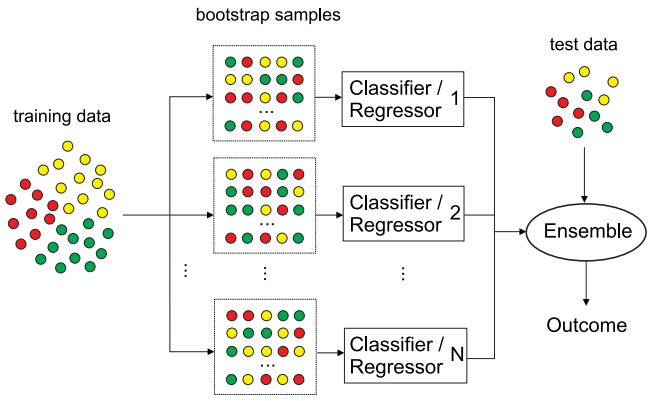


Fig. 1. Bagging technique for classification or regression problems.

Formally, given a labeled training data set $(x_i, y_i)_{i=1:n}$, and given a nonlinear mapping $\phi(\cdot)$, the SVM method solves the following problem:

$$\min_{w,\xi_i,b}\left\{\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i\right\} \tag{1}$$

constrained to:

$$y_i(\langle\phi(X_i), w\rangle + b) \geq 1 - \xi_i \forall i = 1, \ldots, n \tag{2}$$

$$\xi_i \geq 0 \forall i = 1, \ldots, n \tag{3}$$

where w and b define a linear classifier in the feature space, and ξ_i are positive slack variables enabling to deal with permitted errors (see [51]). Appropriate choice of nonlinear mapping ϕ guarantees that the transformed samples are more likely to be linearly separable in the (higher dimension) feature space. The regularization parameter C controls the generalization capability of the classifier, and it must be selected by the user.

2.9. Random forest

Random Forest (RF) [52] is among the most renowned bagging-like techniques for classification and regression problems. Bagging are the simpler ensemble technique to train multiple learners and provide an unified output. It considers an ensemble composed by learners with equal architecture, that is, with same topology, number of input–output variables and parameters (Fig. 1).

RF employs decision or regression trees as predictors in a way that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. Therefore RF differs from the pure bagging technique in the topology of the trees changes among them. The generalization error for forests converges to a limit as the number of trees in the forest becomes large. The generalization error of a forest of tree classifiers depends on the strength of the individual trees in the forest and the correlation between them.

2.10. Gaussian naive Bayes

Naive Bayes methods [53] are supervised learning algorithms based on the Bayes theorem. These methods assume the "naive", i.e. simple, condition of independence among every pair of features given the value of the class variable. The Bayes theorem states the following relationship, given class variable y and dependent feature vector x_1 through x_n:

$$P(y|x_1, \ldots, x_n) = \frac{P(x_1, \ldots, x_n|y)P(y)}{P(x_1, \ldots, x_n)} \tag{4}$$

Assuming the independence of the random variables x_i, we can express the previous equation as follows:

$$P(y|x_1, \ldots, x_n) = \frac{P(y)\prod_{i=1}^{n}P(x_i|y)}{P(x_1, \ldots, x_n)} \tag{5}$$

Naive Bayes classifier retrieves the maximum argument of the previous expression, so-called Maximum a Posteriori, or simply MAP, considering that $P(x_1, \ldots, x_n)$ is constant:

$$\hat{y} = \underset{y}{argmax}P(y|x_1, \ldots, x_n) = \underset{y}{argmax}P(y)\prod_{i=1}^{n}P(x_i|y) \tag{6}$$

Gaussian Naive Bayes (GNB) simply assumes that each $x_i|y$ is a normal random variable:

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y}}e^{-\frac{x_i-\mu_y}{2\sigma_y^2}} \tag{7}$$

2.11. K-Nearest Neighbors

The k-NN [54] method is among the top techniques for data mining that uses the well-known principle of Cicero: birds of a feather flock together or literally equals with equals easily associate. It tries to classify an unknown sample based on the known classification of its neighbors (Fig. 2).

K-NN is a non-parametric ML method which looks for a set of K instances of the training set which are the closest to the new test instance. The term "closest" to an instance x_i is measured with respect to a metric or distance $d(\cdot)$ which fulfills:

$$d(x_i, x_j) \geq 0 \ \forall \ i, \ j$$

and

$$d(x_i, x_j) = 0 \Rightarrow x_i = x_j \tag{8}$$

$$d(x_i, x_j) = d(x_j, x_i) \tag{9}$$

$$d(x_i, x_j) \leq d(x_i, x_k) + d(x_k, x_j) \tag{10}$$

Most frequently, the Euclidean distance is used within the k-NN domain.

2.12. AdaBoost

Adaptive Boosting (AB) [55] is a kind of a boosting method that proposes to train each of the ensembled learners (individual models) iteratively. Normally, each learner focuses on the data that was misclassified by its predecessor, adapting its parameters to achieve better results. As in all boosting methods, AdaBoost establishes the same structure for all of the learners involved in the ensemble, that is, same architecture, number of parameters, or input–output variables. After creating the ensemble structure, the learners are trained sequentially, in such a way that each new learner requires that the previous learner had been trained before. Fig. 3 shows an outline of the AdaBoost algorithm for multi-class classification.

2.13. Multi-Layer Perceptrons

Multi-Layer Perceptron (MLP) [56] is a class of Artificial Neural Networks (ANN) [57] consisting of an input layer, several hidden layers, and an output layer, which all are built by a number of special processing units called neurons. Layers are placed consecutively, and each neuron of a layer is connected to the other neurons of the consecutive layer by means of weighted links (Fig. 4. The values of these weights are related to the capacity of the MLP to learn the problem, and they are learnt from a sufficiently long number of examples. The process of assigning values to these weights from labeled examples is known as the training process of the perceptron. MLP training processes are based on stochastic methods, a backpropagation trial-and-error is among the more well-known learning algorithms applied to train ANNs.

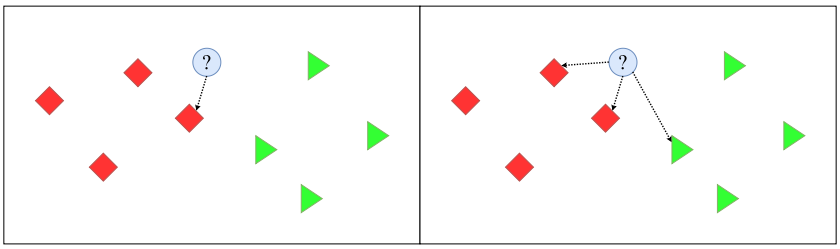


Fig. 2. On the left figure, the 1-NN decision rule: the point ? is assigned to its nearest neighbor class (red square). On the right, the k-NN decision rule, with $k = 3$: the point ? is again assigned to the red square class, because 2 out of 3 neighbors belong to that class.

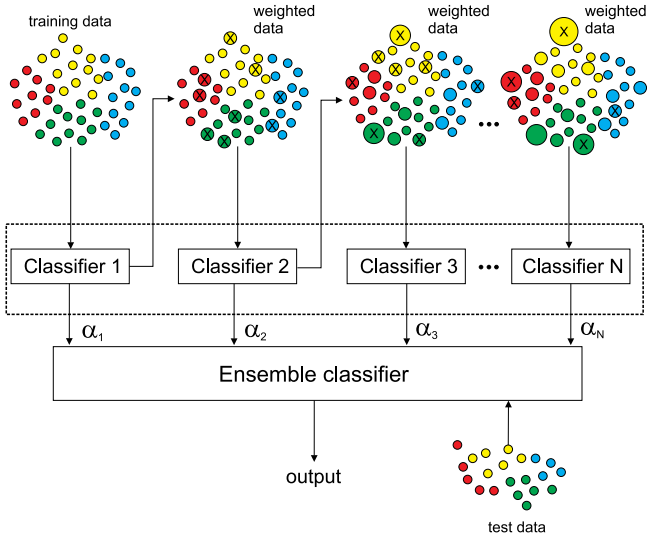


Fig. 3. Diagram of the AdaBoost algorithm exemplified for multi-class classification problems.

2.14. Linear regression

Linear regression (LR) [58] is a statistical method for modeling the linear relationship between dependent variables (predictor variables) and independent variables (target variables). The general formula for multiple regression models is:

$$Y = \beta_0 + \sum_{j=1}^{n} \beta_j X_j + \epsilon \tag{11}$$

where Y denotes the target variable; β_0 represents a constant; β_j denotes the regression coefficient ($j = 1, 2, \ldots, n$); and ϵ is an error term.

2.15. Regression trees

Classification and Regression Trees (CART) [59] are ML methods for constructing prediction models from data. The models are obtained by recursively partitioning the data space and fitting a simple prediction model within each partition. As a result, the partitioning can be represented graphically as a decision tree.

Regression trees (RT) take continuous or ordered discrete values for dependent variables, and a regression model is fitted to each node to give the predicted values of the target variable.

2.16. Extreme learning machines

Extreme Learning Machine (ELM) [60] is a type of training method for MLPs characterized by being computationally faster than traditional gradient back-propagation. In the ELM algorithm the weights between the inputs and the hidden nodes are set at random, usually by using a uniform probability distribution. Then, it establishes the output matrix of the hidden layer and computes the Moore–Penrose pseudo-inverse of this matrix. The optimal values of the weights belonging to the output layer are directly obtained by multiplying the computed pseudo-inverse matrix with the target.

3. Methodology: Hierarchical classification/regression approach

The HCR methodology proposed in this paper consists of a three-level architecture. Fig. 5 shows a schematic diagram of the training process in which the three stages of the method can be distinguished. The first level, corresponding to data preprocessing is depicted in green color. It consists of dividing the training data domain in different clusters and assigning a label to each instance in accordance to the cluster where it belongs. Afterwards, a balancing data preprocessing is performed to ensure equal representation of all classes. The different approaches for clustering the data, together with the different balancing techniques used, are discussed in detail in Section 3.1. The second level, represented in color red, corresponds to the classification step. The classifiers used, detailed in Section 3.2, are fitted with the labeled training data once the balancing process has been accomplished. Finally, the third level is represented in color blue. It relates to the regression step. In this training phase each cluster of training data is used to fit a different regression model. The specific ML regressors implemented are described in Section 3.3. Table 1 exhibits the different ML methods used at each level.

3.1. Data balancing and preprocessing

This step aims to transform the continuous domain of the target variable into discrete labels in the training data. The original target values of the training data samples are divided into n classes, in which each class represents a specific range of the continuous output values.

3.1.1. Data clustering

The clustering process consists in the following process: Let us consider a training dataset D, and let X_i be the ith data sample in D, where $X_i = (X_1, X_2, \ldots, X_m, Y_{X_i})$ contains m different input variables and Y_{X_i} is the output variable. A $n-1$ number of thresholds T_j are defined to establish the boundaries of the n corresponding subsets of training data. Then, the rules displayed in Eq. (12) are used to assign labels to each data samples of D in accordance to the belonging subset depending of the target value Y_{X_i}.

$$X_i \in \begin{cases} \text{Cluster 1}, & \text{if } Y_{X_i} < T_1 \\ \text{Cluster 2}, & \text{if } T_1 < Y_{X_i} < T_2 \\ \text{Cluster n-1}, & \text{if } T_{n-2} < Y_{X_i} < T_{n-1} \\ \text{Cluster n}, & \text{otherwise } (Y_{X_i} > T_{n-1}) \end{cases} \tag{12}$$

Two values of n have been tested in this paper (i.e. $n = 2, 4$). Additionally, different approaches for determining the threshold values in each case have been evaluated. For the 2-class scenario, where the data is divided between extreme or non-extreme event, three alternative

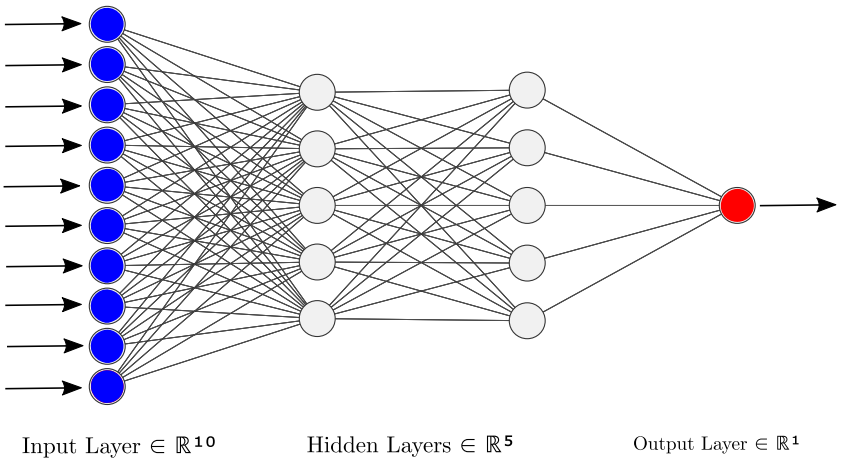


Fig. 4. Structure of a MLP, with two hidden layers.

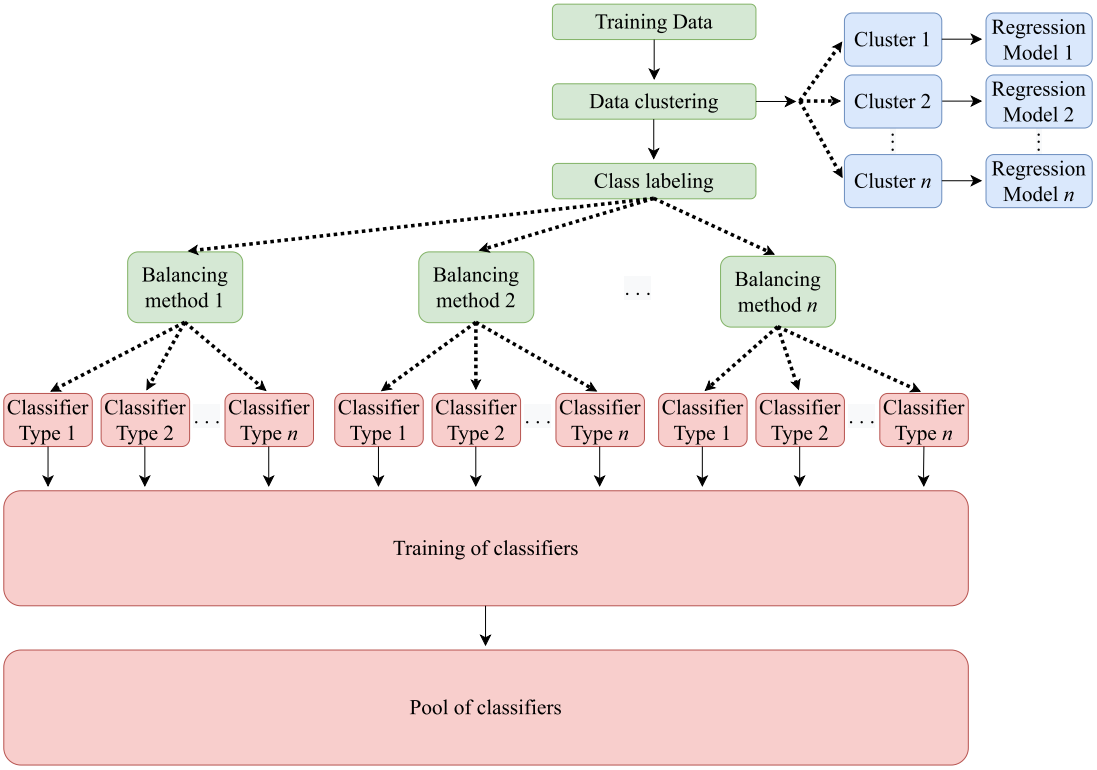


Fig. 5. Proposed HCR training architecture.

threshold values have been analyzed (i.e. $T = \mu$, $T = \mu + \sigma$, $T = \mu + 2\sigma$, where μ corresponds to the mean of the training data target value and σ to its standard deviation). The threshold value is relevant because it determines the degree of cluster imbalance.

For the 4-case scenario, two approaches have been used. First, in order to obtain four subsets with equal number of samples, thresholds have been established based on the training data quartiles, $T = (Q_1, Q_2, Q_3)$. The second approach has consisted of setting a fixed step size, such that $T = (\mu, \mu + 0.75\sigma, \mu + 1.5\sigma)$. Results for all scenarios are shown and discussed in Section 4.

3.1.2. Balancing techniques

Since clustering is not necessarily symmetrical (it depends on the way the thresholds are selected), it may result in highly unbalanced clusters. Therefore balancing techniques are required to compensate the number of instances per cluster before proceeding to the classifiers training phase.

Six different state-of-art data balancing methods have been used, combining oversampling and undersampling strategies: SMOTE, Borderline-SMOTE, SVM-SMOTE, ADASYN, K-Means SMOTE and Random Undersampling. A brief overview of each approach is provided in Section 2.

3.2. Classification

Once the training data has been clustered and balanced, a group of classifiers is trained with the labeled data. Thus, as shown in Fig. 5, each type of a classifier used is fitted with the balanced data generated by the different techniques applied, resulting in a pool of 36 classifiers (6 balancing techniques × 6 types of classifiers). A brief theoretical

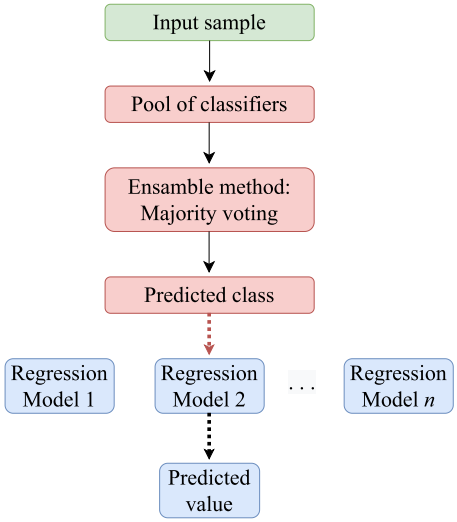


Fig. 6. HCR prediction architecture.

overview of the 6 classifiers implemented (SVM, RF, GNB, kNN, AB and MLP) is provided in Section 2.

3.3. Regression

ML regression methods are trained with each subset of training data once it has been clustered, so that each training data sample is used in the fitting of a single regression model. Five different state-of-art regressors have been evaluated (LR, RT, RF, MLP and ELM), which are also described in Section 2.

3.4. Prediction architecture

Once the different models (classifiers and regressors) involved in the HCR are fitted with its corresponding training data, they are used to perform the numeric estimation as shown in Fig. 6. The prediction procedure withholds (1) feeding the classifiers with an unknown sample, (2) ensembling the pool of classifiers using a majority voting method [61], (3) using the specific regression model to produce the final prediction value (according to the predicted class obtained as the ensemble output).

4. Experiments and results

In this section we describe the experimental work carried out in this paper to evaluate the proposed HCR approach in the prediction of EWS. We first describe the wind speed data available and the predictive variables, from ERA5 Reanalysis. The metrics considered to evaluate the proposed HCR approach are described later. The description of the experimental setup, results obtained and discussion closes this experimental section.

4.1. Initial wind speed data

A medium-size wind farm located in Spain has been selected, whose location can be consulted in Fig. 7. Time series with hourly wind speed data covering a period of 10 years (2003–2013) have been used for training and validation of the models.

Fig. 8 depicts the histograms for the wind farm studied, showing a distribution centered at 5 m/s. One may observe the long-tailed distribution for high speeds. Although a wind turbine can operate with wind speeds up to and over 30 m/s during some minutes, the turbine can also be stopped with wind speeds under the cut-off value, in the case of a hysteresis loop. Therefore, it is precisely these poorly represented extreme wind speeds that we are particularly interested in accurately forecasting so as to anticipate early enough events that may potentially lead to the stoppage of the installation or cause damage to the infrastructures. Consequently, different prediction horizons covering from the short term (1 h) to the long term (24 h) have been used.

Table 2
Predictive variables considered at each node from the ERA-5 reanalysis.

Variable name	ERA5 variable
d2m	2m dewpoint temperature
t2m	2m temperature
sp	Surface pressure
msl	Mean sea level pressure
u10	10m u-component of wind
u10n	10m u-component of neutral wind
u100	100m u-component of wind
v10	10m v-component of wind
v10n	10m v-component of neutral wind
v100	100m v-component of wind
hcc	High cloud cover
mcc	Medium cloud cover
lcc	Low cloud cover
tcc	Total cloud cover

4.2. Predictive variables

The EWS prediction presented in this paper is carried out based on meteorological data from ERA5 reanalysis [62]. ERA5 provides hourly information on meteorological variables related to temperature, pressure, precipitation and snowfall among others; with a resolution of 0.25 degrees of longitude and latitude between grids.

Aiming to tackle the EWS prediction problem, predictive variables were determined to be those related to temperature, pressure, speed of different wind components at different heights, and the proportion of the grid box covered by clouds. Table 2 lists the 14 predictive variables selected per node. For the wind speed forecasting in the wind farm, five geographical nodes have been selected, located at the corresponding farm coordinates and within 100 km to the north, south, east and west, respectively. Therefore, a total of $N = 70$ predictor variables (inputs) have been considered for the case under study.

The preliminary step in the processing of these time series entailed the elimination of the data corresponding to missing measurements of the target variable. Then, the prediction time-horizon is set, and a shift is performed accordingly on the wind speed data (target) with respect to the input variables, i.e. if the prediction horizon is set in 1 h, we will use x_t to predict y_{t+1}, so the target value will be shifted one row (1 h) in the timed dataset.

Five different prediction time-horizons have been considered in this work: 1 h, 3 h, 6 h, 12 h and 24 h. They cover short-term forecasting (30 min to 6 h ahead), medium-term forecasting (6-24 h ahead) and long-term forecasting (1 day to 1 week ahead) [6].

4.3. Evaluation metrics

In order to assess the performance of the EWS prediction models proposed, different evaluation metrics have been used. We have considered generic regression error metrics and also specific metrics that reflect the models performance in the prediction of extreme events.

4.3.1. Mean Absolute Error (MAE)

First, a common indicator for regression problems, Mean Absolute Error (MAE), has been considered:

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{13}$$

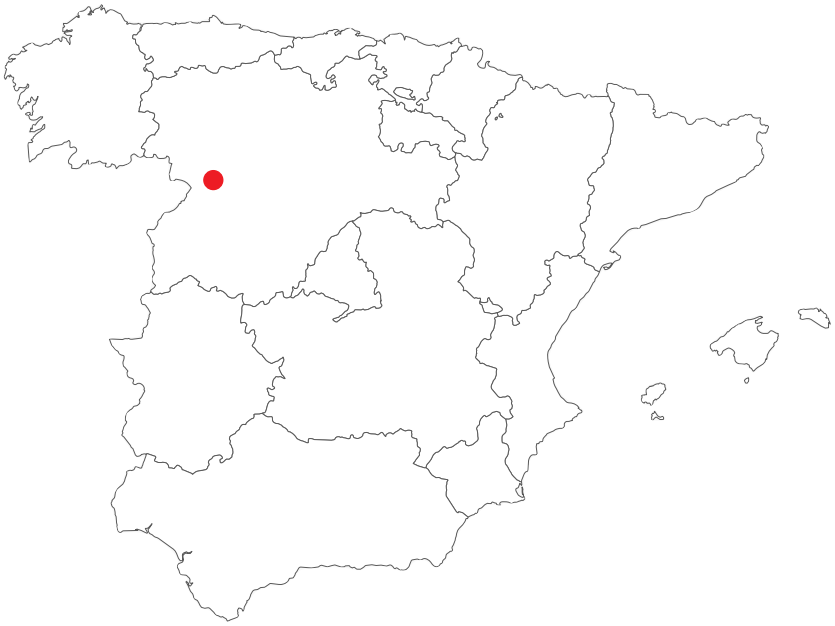


Fig. 7. Geographical location of the wind farm considered in the experiments.

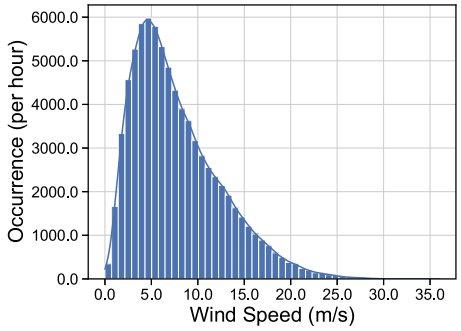


Fig. 8. Wind speed histogram.

where $\hat{y}$ represents predicted values (provided by the model) and y are the actual values. The subscript i is used to refer to a single sample $y_i = y[i]$.

4.3.2. Extreme Events Mean Absolute Error (EEMAE)

Then, the prediction only of the extreme events is evaluated, since the accurate prediction of these points is precisely the scope of this work. For this purpose, it is first necessary to establish the threshold beyond a point is considered as an extreme event. Fig. 9 depicts the wind speed time series for the wind farm location, with the data divided into train (80%) and test (20%), this figure also illustrates the threshold defined to separate extreme data (outliers) from non-extreme data (non-outliers), set at the mean + 3 times the standard deviation of the wind speed values. The extreme values account for 0.38% of the total number of instances available. In order to measure the performance of the regressors on these outlier data, a new indicator, named Extreme Events Mean Absolute Error (EEMAE), has been used, which corresponds to the MAE (Eq. (13)) calculation on these values.

4.3.3. Case-specific weighting

Standard error metrics (such as MAE or RMSE) are not the most appropriate metrics for the subclass of regression problems related with extremes we are dealing with, since they take all prediction errors equally across the domain of the target variable. Case-specifics weighting associates each sample with a relevance value related to the target value, hence the cost of a prediction is weighted by the relevance value of that sample. Therefore, training cases with a target variable value more "relevant" will have higher weights. The MAE weighted by the relevance value has been referred as Case Weighted Mean Absolute Error (CWMAE) (Eq. (14)).

$$\text{CWMAE} = \frac{\sum_{i=1}^{N} |y_i - \hat{y}_i| \cdot R_i}{\sum_{i=1}^{N} R_i} \tag{14}$$

Four different "relevance functions" dependent on the target variable have been used, depicted in Fig. 10. First, an step-wise function has been defined, which associates to each sample a relevance value bounded between 0.1 and 0.9, increasing this value 0.15 for every step of difference, defining a step as one multiple of the standard deviation (σ) (Eq. (15)).

$$R_{interval}(y) = \begin{cases} 0.1 & \text{if} \quad y < \overline{Y_{train}} \\ 0.25 & \text{if} \quad \overline{Y_{train}} \leq y < \overline{Y_{train}} + \sigma \\ 0.40 & \text{if} \quad \overline{Y_{train}} + \sigma \leq y < \overline{Y_{train}} + 2\sigma \\ 0.55 & \text{if} \quad \overline{Y_{train}} + 2\sigma \leq y < \overline{Y_{train}} + 3\sigma \\ 0.6 & \text{if} \quad \overline{Y_{train}} + 3\sigma \leq y < \overline{Y_{train}} + 4\sigma \\ 0.75 & \text{if} \quad \overline{Y_{train}} + 4\sigma \leq y < \overline{Y_{train}} + 5\sigma \\ 0.9 & \text{if} \quad \overline{Y_{train}} + 5\sigma \leq y \end{cases} \tag{15}$$

The second relevance function implemented established a linear relationship between R_i and y_i, being the slope of the straight $\frac{1}{(\max(Y_{train}) - \min(Y_{train}))}$, in such a way that $R = 1$ corresponds to the maximum value of the target variable in the training data and $R = 0$ to the minimum value (Eq. (16)).

$$R_{lineal}(y) = \frac{y}{(\max(Y_{train}) - \min(Y_{train}))} \tag{16}$$

The following establishes a cubic relationship between the value of the target variable and R, $R_i = a + b * y_i^3$. Thereby samples close to the minimum value of $Y_i rain$ have a relevance value near 0 and as the target value increases, so does the weight of these predictions (Eq. (17)).

$$R_{cubic}(y) = \left(\frac{1}{\max(Y_{train})^3 - \min(Y_{train})^3} \right) \times y^3 - \frac{\min(Y_{train})^3}{\max(Y_{train})^3 - \min(Y_{train})^3} \tag{17}$$

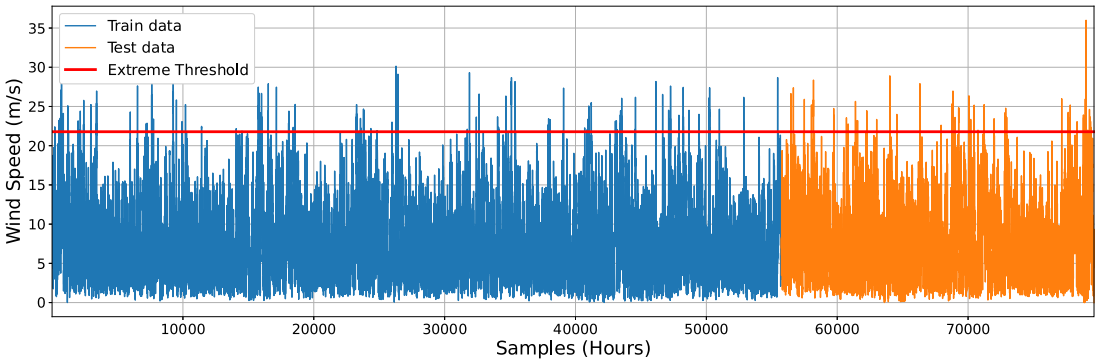


Fig. 9. Wind speed time series.

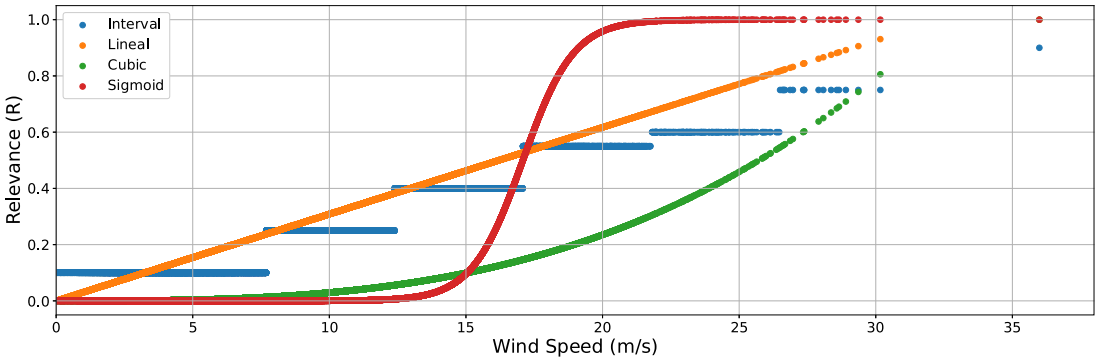


Fig. 10. Relevance functions implemented.

The last proposal consists of using a sigmoid-like relevance function. This relevance function is based on the following sigmoid (Eq. (18)):

$$R_{sigmoid}(y) = \frac{1}{1 + \exp^{-s*(y-c)}} \tag{18}$$

where c is the center of the sigmoid and s denotes its shape. The parameter c represents the value where $R(y) = 0.5$, therefore, it refers to the threshold above which the target variable start to be more relevant. In this study it has been settled to the mean of the training data target value plus two times the standard deviation (Eq. (19)), considering values above this threshold of considerable significance.

$$c = \overline{Y_{train}} + 2\sigma \tag{19}$$

With respect to the parameter s, it defines how fast the sigmoid decays to 0. It is determined by a decay factor k, defined as $k = 0.5$ and by a precision factor Δ, defined as $\Delta = 1^{-4}$. Therefore s is obtained by solving Eq. (20).

$$s = \frac{\ln(\Delta^{-1} - 1)}{|c \cdot k|} \tag{20}$$

Once these four relevance functions have been defined, a different CWMAE shall be calculated for each one and the average value will be taken to assess the performance of the regression models.

4.3.4. True Positive Rate (TPR), False Positive Rate (FPR) and G-mean (G-mean)

The main drawback in the metrics presented previously, EEMAE and CWMAE, is that they are only focused on the prediction of extremes values, but they do not penalize the situation where the model predicts an extreme being the actual value normal (False Positive). These false alarms may lead to severe economic damages, such as a disruption in energy production due to a extreme wind prediction that does not occur, or the deployment of emergency equipment to reinforce installations when it is not necessary. Three popular classification error metrics are used to account for this concern, True Positive Rate (TPR), False Positive Rate (FPR) and Geometric-mean (G-mean).

TPR, also referred to as Recall, determines the ability of a model to find all the relevant cases within a dataset. It is computed by dividing the number of relevant cases truly predicted, True Positives (TP), by the total number of relevant cases present in the data, Positives (P). In this context, since we are working with a regression model, we define a threshold, Eq. (21), above which both actual and predicted values are considered as extreme (or positive). And consequently each sample of the test data set is assigned with a boolean value of TP (1 if both the prediction and the actual value are above T, Eq. (22)) and P (1 if the actual value is above T, Eq. (23)). Therefore TPR is computed following Eq. (24), where a value of 1 indicates that all extremes are predicted correctly and 0 denotes that none extremes have been anticipated.

$$T = \overline{Y_{train}} + 3\sigma \tag{21}$$

$$\mathrm{TP}_i = \begin{cases} 0 & \text{if} \quad \hat{y}_i \leq T \quad \text{or} \quad y_i \leq T \\ 1 & \text{if} \quad \hat{y}_i > T \quad \text{and} \quad y_i > T \end{cases} \tag{22}$$

$$\mathrm{P}_i = \begin{cases} 0 & \text{if} \quad \hat{y}_i \leq T \\ 1 & \text{if} \quad \hat{y}_i > T \end{cases} \tag{23}$$

$$TPR = \frac{\sum_{i=0}^{N} TP_i}{\sum_{i=0}^{N} P_i} \tag{24}$$

Similarly, the FPR is computed by dividing the number of False Positives (FP), i.e. the number of false alarms or events falsely predicted as extreme, by the number of Negatives (N), i.e. the sum of non-extreme events. According to Eqs. (22) and (23), a boolean value of FP and N is given to each data set sample (Eqs. (25) and (26), respectively). Then FPR is calculated as indicated in Eq. (27), where a value of 0 indicates that all non-extremes events have been predicted correctly, and 1 denotes that all of them were predicted as false extremes.

$$\mathrm{FP}_i = \begin{cases} 0 & \text{if} \quad \hat{y}_i \leq T \quad \text{or} \quad y_i > T \\ 1 & \text{if} \quad \hat{y}_i > T \quad \text{and} \quad y_i \leq T \end{cases} \tag{25}$$

Table 3
Experimental setup.

Balancing methods	**SMOTE**		**Borderline-SMOTE**	
	k neighbors	5	k neighbors	5
	SVM-SMOTE		**ADASYN**	
	k neighbors	5	n neighbors	2
	out step	0.5		
	K-Means SMOTE		**SMOGN**	
	k neighbors	2	k neighbors	9
	Cluster balance threshold	0.02	Samp method	'balance'
			Pert	0.02
			rel thres	0.80
			rel method	'auto'
			rel xtrm type	'both'
			rel coef	1.50
Classification methods	**SVM**		**RF**	
	C	1.0	n estimators	100
	kernel	'rbf'	max depth	400
	GNB		**kNN**	
	var smoothing	1e−09	n neighbors	20
			Algorithm	'brute'
	AB		**MLP**	
	n estimators	50	Hidden layer sizes	500
	Learning rate	1.0	Activation	'relu'
			Solver	'adam'
			max iter	300
Regression methods	**RT**		**RF**	
	max depth	400	n estimators	400
	MLP		ELM	
	Hidden layers	2	Hidden size	500
	Neurons per layer	64		
	Activation	'relu'		
	Solver	'adam'		

$$N_i = \begin{cases} 0 & \text{if} \quad \hat{y}_i > T \\ 1 & \text{if} \quad \hat{y}_i \leq T \end{cases} \tag{26}$$

$$FPR = \frac{\sum_{i=0}^{N} FP_i}{\sum_{i=0}^{N} N_i} \tag{27}$$

Finally, G-mean is the root of the product of class-wise sensitivity. This measure tries to maximize the accuracy on each of the classes while keeping these accuracies balanced. For binary classification G-mean is the squared root of the product of the sensitivity and specificity. G-mean is a good indicators in imbalanced domains because it is independent of the distribution of examples between classes [63]. G-mean is computed following the formula shown in Eq. (28), where TPR and TNR (True Negative Rate) are calculated as indicated in Eqs. (24) and (29), respectively, with TN (True Negative) calculated as shown in Eq. (30).

$$\text{G-mean} = \sqrt{TPR \cdot TNR} \tag{28}$$

$$TNR = \frac{\sum_{i=0}^{N} TN_i}{\sum_{i=0}^{N} N_i} \tag{29}$$

$$TN_i = \begin{cases} 0 & \text{if} \quad \hat{y}_i > T \quad \text{or} \quad y_i > T \\ 1 & \text{if} \quad \hat{y}_i \leq T \quad \text{and} \quad y_i \leq T \end{cases} \tag{30}$$

4.4. Experimental setup

The experimental setup, along with the parameters set for the different ML methods, are detailed in this section. First, a preliminary dataset preparation is performed. The steps of this preprocessing are: (1) shifting of the predictor variables n instances from the target variable and removal of the last n rows, where n is the prediction time horizon expressed as the number of hours in advance at which the forecast is made; (2) Splitting the dataset into training and test (80%–20%) subsets, assuring that no test instance was seen by the ML/DL methods during the training. Since dealing with timed-series data, instead of randomly splitting the datasets, last 20% of the data were separated as test data; (3) Feature scaling, scaling the features, which is important to ensure the upper and lower limits of data in the given predefined range. Feature standardization was performed, causing data to have zero-mean and a unit-variance (Eq. (31)), as follows:

$$x' = \frac{x - \bar{x}}{\sigma} \tag{31}$$

where x is the original feature vector, $\bar{x}$ denotes the feature mean and σ its standard deviation.

Table 3 shows the parameters used for the balancing, classification and regression methods employed and detailed in Section 2. Their implementation has been performed on a Intel(R) Core(TM) i7-10700 CPU with 2.90 GHz and 16 GB RAM using the Python libraries, imblearn, sklearn and tensorflow.

4.5. Experimental results

In this section, results for the different experiments performed are presented. First, the performance of regression models using ML methods is shown (Section 4.5.1). Then, in order to evaluate the proposed methodology in comparison with a similar one, the results of applying SMOGN before training the ML models are presented (Section 4.5.2). Finally, results using the HCR methodology with different parameters are reported (Section 4.5.3). A brief discussion on the obtained results is presented in Section 4.5.4. All these experiments are performed on the basis of 1-hour ahead forecasting. A comparison of the results with different prediction horizons is presented in Appendix A.

4.5.1. ML methods

In first place, the five regression methods discussed in Section 2 are applied to the raw training data, i.e. without using any balancing technique. Table 4 shows the error metrics for the five regression models. It may be observed that MLP is overall the best model, achieving

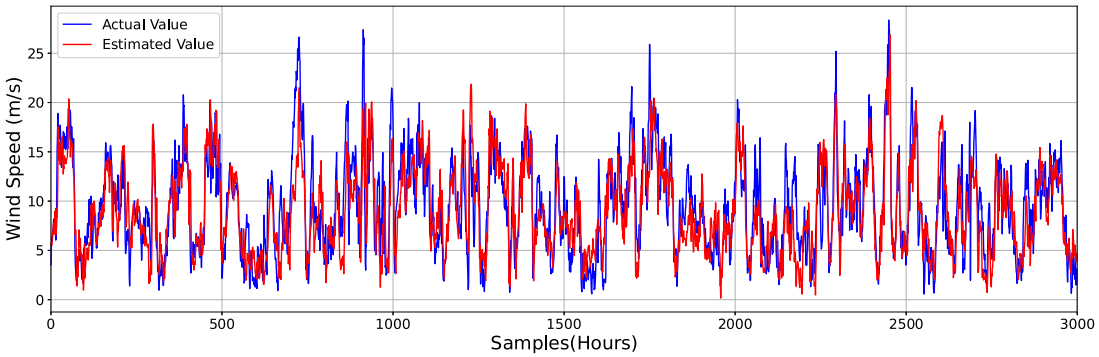


Fig. 11. Comparison of actual wind speed (blue) with forecasted wind speed (red) using MLP regression model.

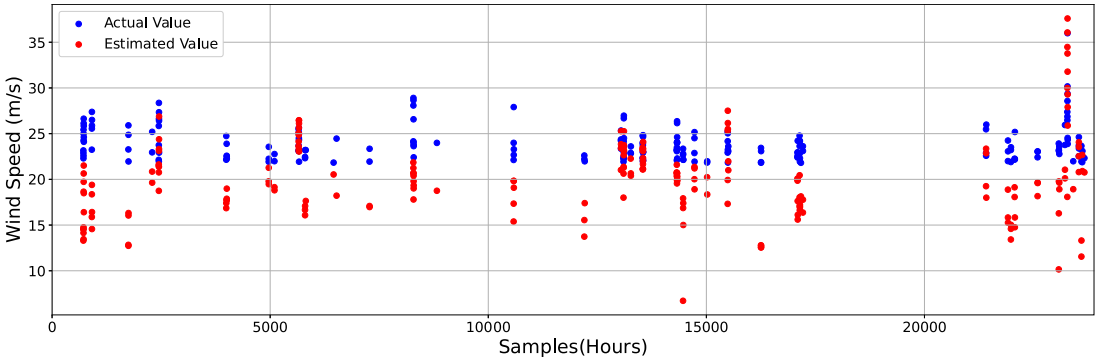


Fig. 12. Performance of the regression model in actual wind extremes using MLP regression model.

Table 4
Error metrics for ML methods applied on imbalanced training data.

		MAE	EEMAE	CWMAE	TPT	FPR	G-Mean
ML methods	RL	2.98	10.36	4.60	0.00	**0.00**	0.00
	RT	2.73	5.35	3.47	0.23	0.01	0.48
	RF	**1.96**	5.26	2.63	0.16	**0.00**	0.39
	MLP	2.03	**3.82**	**2.54**	**0.31**	**0.00**	**0.56**
	ELM	2.13	4.77	2.81	0.29	**0.00**	0.54

Table 5
Error metrics for ML methods applied after training data was balanced with SMOGN method.

		MAE	EEMAE	CWMAE	TPT	FPR	G-Mean
SMOGN + ML	RL	6.75	3.75	5.66	0.80	0.11	0.85
	RT	4.82	3.31	4.42	0.54	0.05	0.73
	RF	**4.29**	3.11	**3.73**	0.47	**0.03**	0.69
	MLP	4.67	**2.57**	4.46	0.85	0.06	**0.89**
	ELM	4.85	4.70	5.28	**0.88**	0.09	**0.89**

quite low MAEs and being able to correctly detect 37% of the extremes without causing any false alarms.

Fig. 11 shows the wind speed time series forecast using the MLP regression model for the first 3000 h of the test data, where a reasonably accurate prediction may be appreciated. However, when analyzing the model's performance in the prediction of extreme events (Fig. 12), it can be clearly noticed that the forecast at these points is far from being optimal, with a MAE at those instances of 3.82 m/s and being only able to detect 37% of the extremes.

4.5.2. SMOGN+ML

Then SMOGN method, explained in Section 2.7, has been applied for balancing training data before fitting the regressors. Results are shown in Table 5. Comparing these results with those displayed in Table 4 for the ML methods applied on the raw training data, it can be observed how the tendency for the five regressors is practically the same. The EEMAE significantly improves, however, both the MAE and CWMAE worsen substantially, indicating that performance in the extremes is improved by compromising efficiency in the non-extreme values. Similarly, the TPR improves substantially for the five regressors, reaching 88% with ELM, but at the cost of increasing the number of false alarms.

Figs. 13 and 14 shows the wind speed time series prediction using SMOGN+MLP for the first 3000 test hours and for the test extremes, respectively. It can be seen how the prediction in the extremes instances has improved significantly.

4.5.3. HCR

Next, the proposed HCR methodology proposed in Section 3 is considered. Different combinations of parameters in the HCR structure are evaluated, modifying both the number of clusters into which the training dataset is divided, as well as the thresholds that determine the separation between clusters.

Case 1: $n = 2$

First, the HCR method is evaluated with a number of clusters ($n = 2$). Three different thresholds have been established ($T = \mu$, $T = \mu + \sigma$, $T = \mu + 2\sigma$). The selection of these thresholds determines the degree of imbalance among the training subsets formed after the clustering process (Fig. 15). In this figure it can be seen how the first case does not require the application of data balancing techniques, since the minority class represents 40% of the total training data, while in the other cases it is necessary, since the degree of imbalance is severe (16.2% and 4.7%, respectively).

The next step in the proposed HCR methodology consists of applying the balancing techniques in the required cases and training the pool of classifiers. Table B.12 displays the G-mean score of each of the trained classifiers, as well as the voting classifier resulting from the ensemble

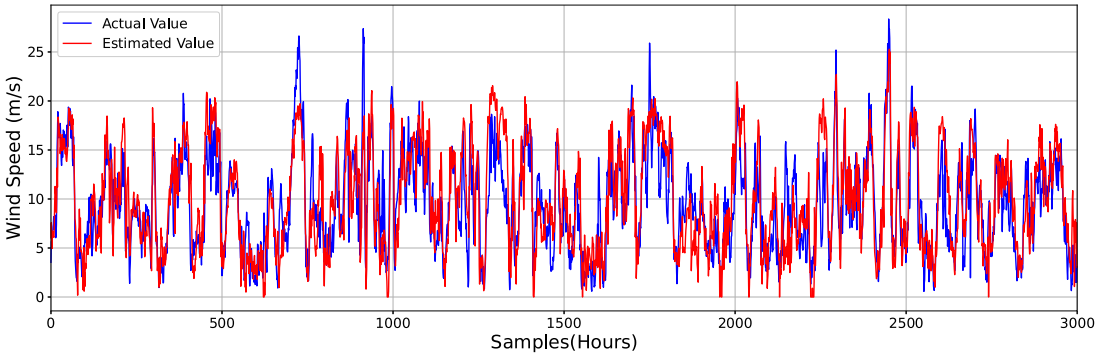


Fig. 13. Comparison of actual wind speed (blue) with forecasted wind speed (red) using SMOGN + MLP regression model.

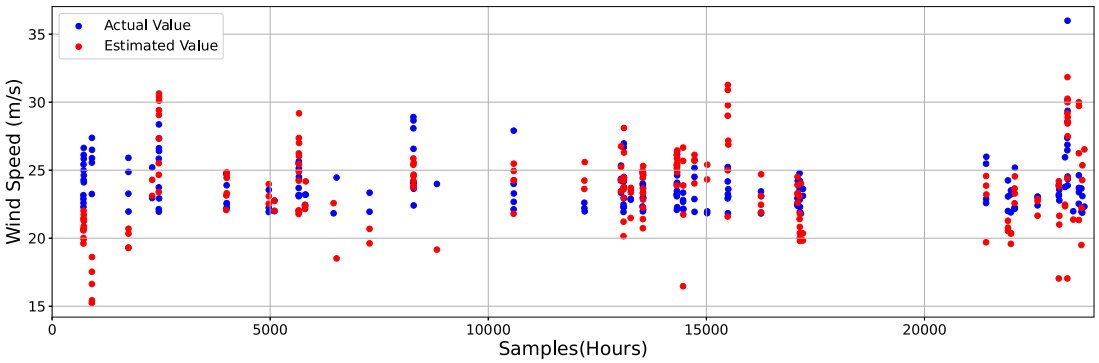


Fig. 14. Performance of the regression model in actual wind extremes using SMOGN + MLP regression model.

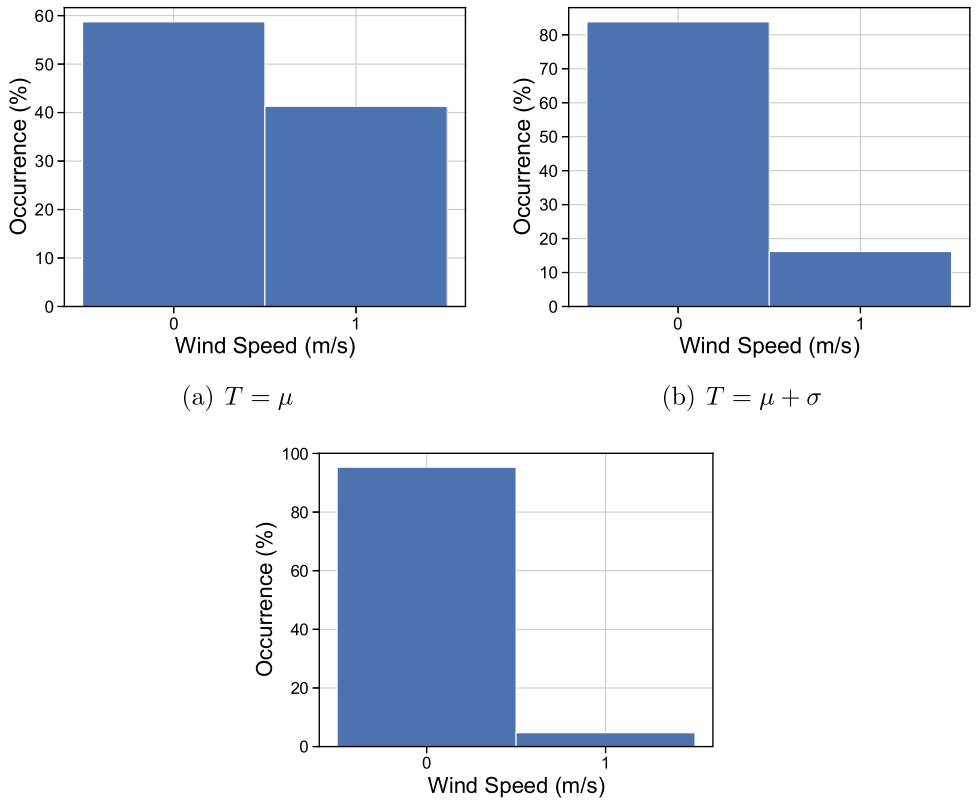


Fig. 15. Imbalance degree of the training subsets according to the threshold selected. 0 represents the no-extremes training subset and 1 denotes the extremes training subsets.

Table 6
Error metrics for HCR methodology with $n = 2$ and different threshold values.

		MAE	EEMAE	CWMAE	TPR	FPR	G-Mean
$T = \mu$	RL	2.75	6.24	3.09	0.04	**0.00**	0.20
	RT	3.27	5.47	4.04	0.52	0.04	0.70
	RF	2.72	3.83	**3.01**	0.43	0.04	0.65
	MLP	2.85	3.20	3.31	0.64	0.02	0.79
	ELM	2.77	3.07	3.08	0.73	0.02	0.85
$T = \mu + \sigma$	RL	3.05	3.73	3.38	0.26	0.01	0.51
	RT	3.37	4.96	4.31	0.41	0.07	0.62
	RF	2.87	3.37	3.75	0.50	0.05	0.69
	MLP	3.16	5.32	4.88	0.87	0.08	0.89
	ELM	3.18	4.00	4.51	0.88	0.07	0.91
$T = \mu + 2\sigma$	RL	3.12	**2.77**	4.33	0.81	0.07	0.87
	RT	3.09	4.70	4.38	0.55	0.05	0.72
	RF	**2.46**	3.02	3.88	0.79	0.07	0.86
	MLP	2.86	6.73	5.29	0.91	0.08	0.91
	ELM	2.92	5.85	5.21	**0.92**	0.08	**0.92**

of all of them. First, it can be appreciated how for the non-balanced classifiers, the G-mean score deteriorates as the threshold value and, consequently, the degree of cluster imbalance increases. This reveals the necessity of employing data balancing techniques, with which the classifiers achieve good results for all 3 cases.

Finally, each test data is classified by using the ensemble voting classifier and, based on this output, the regressor corresponding to that training subgroup is then employed to predict the wind speed value. Table 6 displays the results of the wind speed forecasting for the 3 threshold values evaluated. Some points may be inferred from this: first, MAE remains fairly constant for the 3 thresholds studied, with an average variation of 12% between maximum and minimum for each regressor. Regarding the EEMAE, two tendencies may be observed, neural networks (MLP and ELM) tend to increase the value of EEMAE as the threshold value increases, while linear and CART regressors (LR, RT and RF) experience the opposite tendency. For the CWMAE it is clearly observed how the prediction degrades while increasing the threshold, yet obtaining better results than SMOGN + ML methods. Finally TPR significantly improves when increasing T, achieving a remarkable 92% of extreme events detection, although false alarm rate increases accordingly up to a 8% value.

Figs. 16 and 17 shows the temporal predicted series compared to the actual wind speed values. The aforementioned tendency observed when increasing the value of the threshold may be noticed. Showing the existing trade-off between obtaining a high percentage of extremes detected (TPR) and increasing the number of false alarms (TNR) (Fig. 16) and worsening the prediction of the extremes by overestimating them (Fig. 17). Consequently, the choice of the threshold value will depend on the needs of each specific problem, in terms of priority and preferability of the scenario (high TPR, low TNR or a compromise between the two values). Even so, it can be appreciated how this results improve those obtained previously with the ML or SMOGN+ML methods: comparing the case of SMOGN + ELM, which gets the higher TPR (0.88) with HCR with ELM as regressor and $T = \mu + \sigma$, which gets the same TPR, it may be noticed how the others five error metrics are better with the HCR methodology.

Case 2: $n = 4$

The second configuration tested for the proposed HCR methodology consists of increasing the number of clusters to 4 ($n = 4$). In addition, two different approaches are examined for the determination of the thresholds for each cluster. The first one entails the formation of 4 training subsets with a similar number of samples, so that no data balancing techniques were necessary. For this purpose, threshold values were established based on the quartiles values ($T = (Q_1, Q_2, Q_3)$) of the training data in relation to the target variable, meaning that each cluster is composed with 25% of the training data. The second approach

Table 7
Error metrics for HCR methodology with $n = 4$ and different threshold values.

		MAE	EEMAE	CWMAE	TPR	FPR	G-Mean
$T = (Q_1, Q_2, Q_3)$	RL	2.74	4.69	**2.99**	0.13	**0.00**	0.36
	RT	3.15	4.22	4.26	0.46	0.06	0.66
	RF	2.87	3.47	3.40	0.52	0.03	0.71
	MLP	3.00	3.92	3.95	0.81	0.05	0.88
	ELM	3.01	3.45	3.82	0.82	0.05	0.89
$T = (\mu, \mu + 0.75\sigma, \mu + 1.5\sigma)$	RL	**2.63**	**2.82**	3.32	0.55	0.03	0.73
	RT	2.92	3.74	3.92	0.59	0.06	0.75
	RF	2.69	2.90	3.77	0.72	0.05	0.82
	MLP	3.08	5.99	5.11	**0.96**	0.08	**0.94**
	ELM	2.87	4.53	4.50	0.89	0.07	0.91

consists in setting a equal step size between thresholds ($T = (\mu, \mu + 0.75\sigma, \mu + 1.5\sigma)$), thus the size of the training subsets is variable and data balancing techniques are necessary. Fig. 18 shows the histograms of the four training subsets for the threshold values selected. As expected, it can be seen how for the first approach the four groups presents the same number of instances while for the second approach the higher the target value the lower the instances present.

Table B.13 reports the performance of the pool of classifiers trained with the labeled training data after applying the different data balancing techniques when necessary. Here it is observed how, firstly, the results are worse than those shown in Table B.12 for a number of clusters $n = 2$. This is explained by the fact that increasing the number of subsets consequently increases the complexity of the classification task. In addition, it can also be appreciated how the results for the fixed-step threshold values improve notably when balancing techniques are applied.

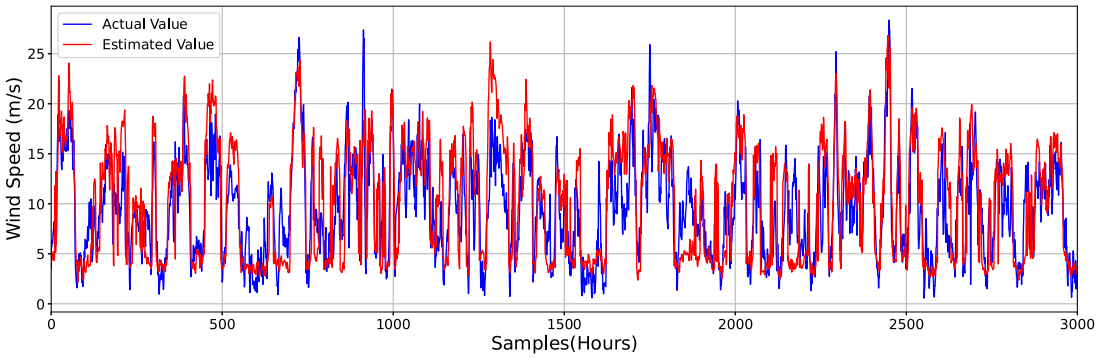
The forecasting results are shown in Table 7 for the two threshold values approaches. The LR, RT and RF regressors clearly perform better with the second approach, obtaining better results on all 6 metrics. Regarding the neural networks (MLP and ELM), more balanced predictions are obtained with the first approach (better MAE, EEMAE, CWMAE and FPR), but a higher percentage of extremes is detected with the second approach (reaching a TPR of 96%). Again, the trade-off between both performances is revealed. Comparing with the other tested methodologies, it seems that the HCR architecture with $n = 4$ is the best performing in both areas, achieving excellent TPR results without worsening excessively in the remaining metrics.

Finally, Figs. 19 and 20 depict the comparison between the actual and predicted values. The strengths of this architecture may be observed, achieving excellent results in extreme event detection without overestimating or degrading the forecasting in the rest of the values.

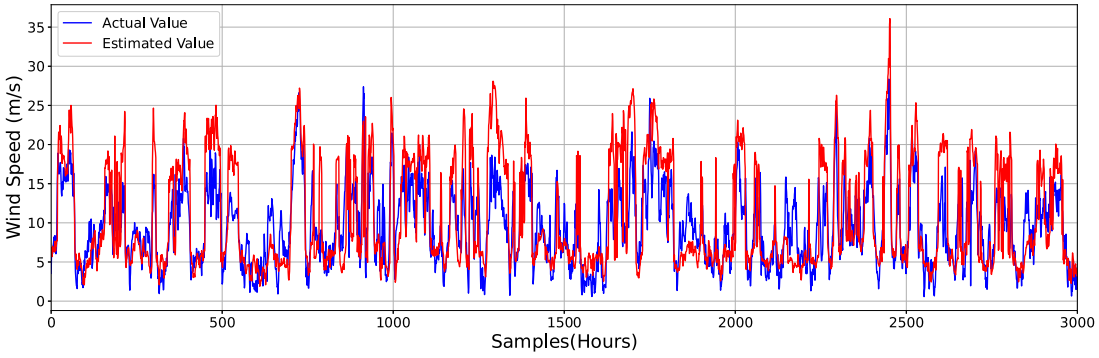
4.5.4. Discussion on the obtained results

Once the results of all evaluated methods have been obtained and presented, a comparison of the performance of each one can be carried out. For the sake of clarity in terms of this comparison, the average error metrics for each of the methods are shown in Figs. 21 and 22. Therefore, for each of the seven methodologies assessed (ML, SMOGN+ML, HCR: $n = 2$, $T = \mu$, HCR: $n = 2$, $T = \mu + \sigma$, HCR: $n = 2$, $T = \mu + 2\sigma$, HCR: $n = 4$, $T = (Q_1, Q_2, Q_3)$, HCR: $n = 4$, $T = (\mu, \mu + 0.75\sigma, \mu + 1.5\sigma)$), the error metrics average is calculated using the five different regressors.

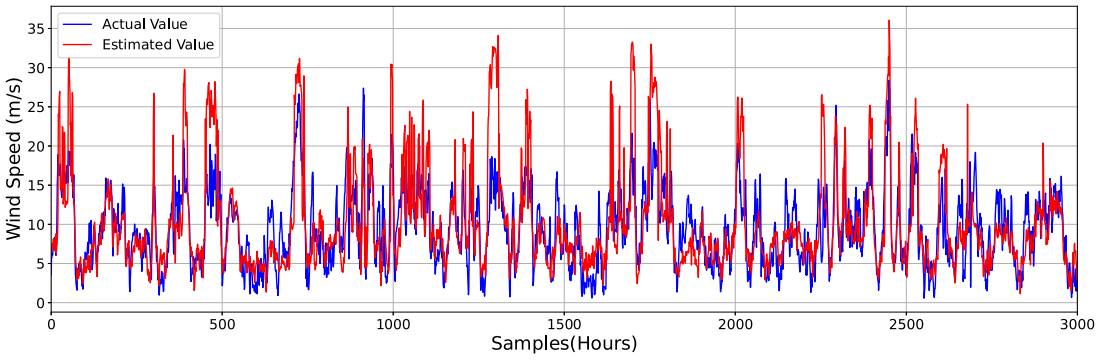
In Fig. 21 the MAE, EEMAE and CWMAE are depicted, and it can be seen how, despite obtaining the best MAE in extreme events (EEMAE) with SMOGN+ML, this implies a notable worsening of the other two metrics. Meanwhile, with the different HCR architectures, the MAE only slightly worsens with respect to the initial case, and depending on the selected architecture, different EEMAE and CWMAE indicators are obtained, but always within similar values. It can be seen that the most balanced method is the HCR with 4 clusters and $T = (Q_1, Q_2, Q_3)$. Similarly, Fig. 22 shows the TPR, FPR and G-mean average values, where a great improvement in the detection of extremes



(a) $T = \mu$



(b) $T = \mu + \sigma$



(c) $T = \mu + 2\sigma$

Fig. 16. Comparison of actual wind speed (blue) with forecasted wind speed (red) using HCR with $n = 2$, ELM as regressor and different values of threshold.

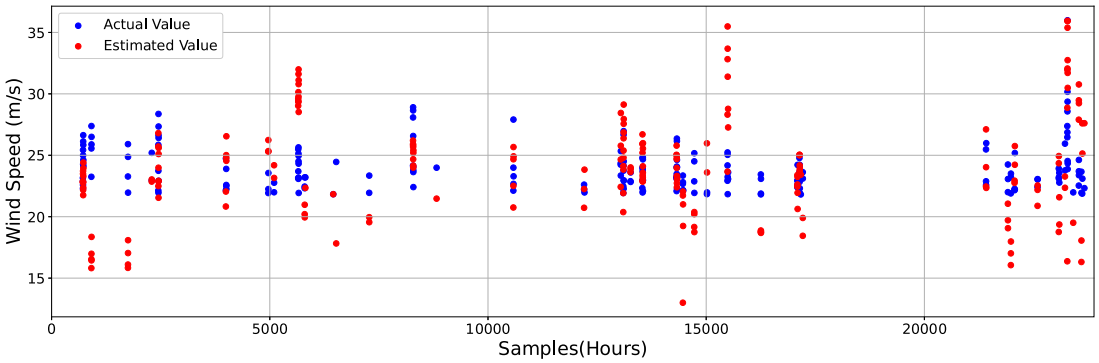
events (TPR) may be observed for all the methods compared with the initial ML scenario. Nevertheless, this enhancement also translates into an increase in the number of false alarms (FPR). In this case HCR architecture with $n = 4$ and $T = (\mu, \mu + 0.75\sigma, \mu + 1.5\sigma)$ remains the best method, achieving a remarkable FPR without increasing the FPR substantially.

Analyzing both figures as a whole, it is possible to conclude that the proposed HCR methodology is a better alternative to both bare-ML methods and to other methods aimed at solving regression problems in unbalanced databases as SMOGN. The HCR architecture choice will depend on the particular problem being tackled, given that some architectures will prioritize the detection of extreme events regardless of a small deterioration in the prediction of non-extreme values, while for others the scenario is the opposite. The architecture with the best performance on both sides is the HCR with $n = 4$ and $T = (\mu, \mu + 0.75\sigma, \mu + 1.5\sigma)$, which achieves a very high TPR (0.96 with MLP as regressor), while maintaining very competitive MAEs metrics.

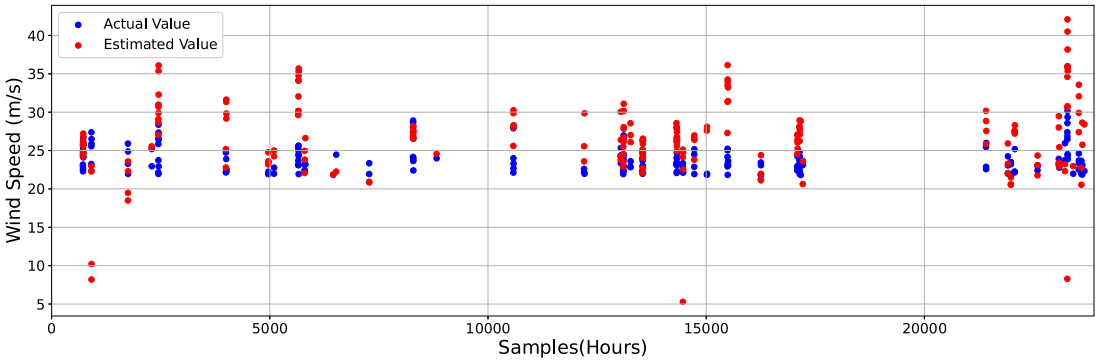
4.6. Validation using forecast data

The meteorological data used in the training of the models has been obtained from ERA5 reanalysis, though any other Reanalysis data is possible. Note that these data offer high accuracy for the ML algorithms training, and have the advantage of avoiding the intrinsic error of a prediction model. However, the implementation of the proposed methodology in a real operational environment requires the use of forecast data.

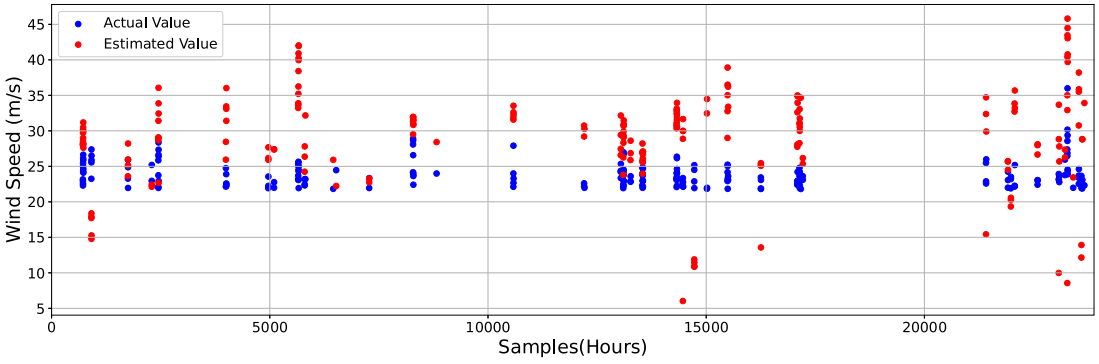
In order to evaluate the performance of the proposed approach in real operation after training the ML algorithms, forecast data from Global Forecast System (GFS) [64] is considered. The GFS is a well-known weather forecast numerical model, developed and managed by the National Center for Environmental Prediction (NCEP). It generates data for dozens of atmospheric and land–soil variables, which can be used to validate the application of the methodology presented in a real operation environment. GFS is a global model with a base horizontal



(a) $T = \mu$



(b) $T = \mu + \sigma$



(c) $T = \mu + 2\sigma$

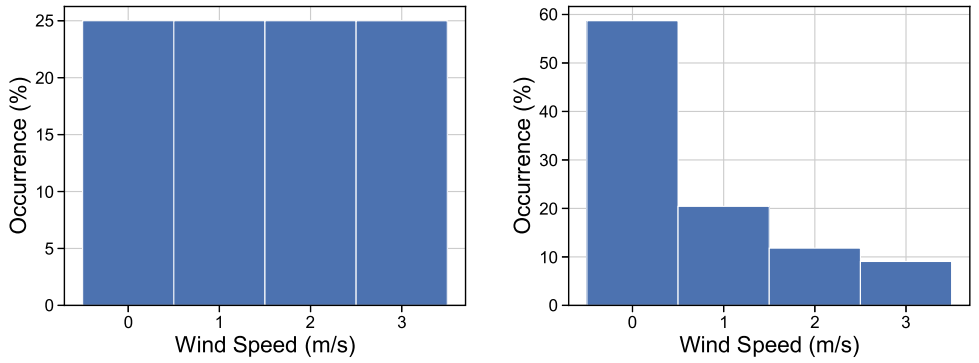
Fig. 17. Performance of the regression model in actual wind extremes using HCR with $n = 2$, ELM as regressor and different values of threshold.

resolution of 28 km between grid points. Temporal resolution covers analysis and forecasts out to 16 days.

Forecast data of the same predictive variables shown in Table 2 and from the same geographical nodes used in Section 4.5 were used to perform the validation of the methodology. A temporal horizon of 3 h for the forecast data is chosen, meaning that, after training over ERA5 Reanalysis data, we will use as predictive variables the meteorological GFS forecast data of +3 h to predict the EWS in the following hour.

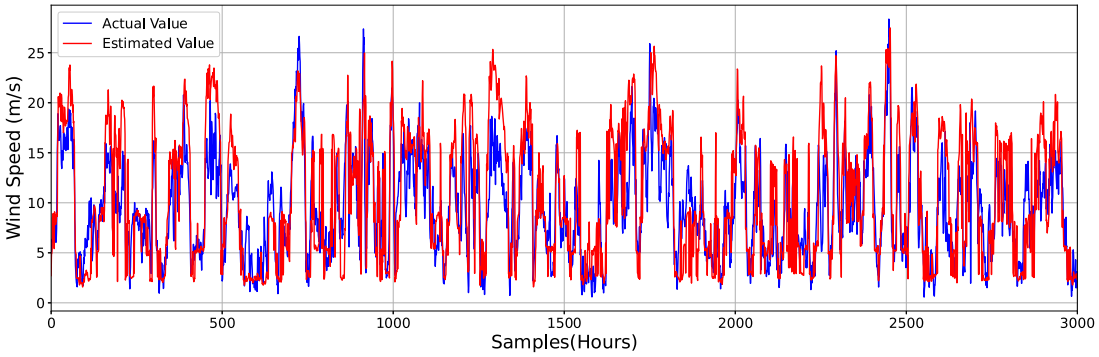
First, Fig. 23 shows that the direct GFS forecast (+3 h) data cannot be used to estimate the EWS in an accurate way. The trend of the actual value is followed, of course, but the GFS seems to underestimate in general the wind speed value, and thus the prediction of EWS cannot be carried out with the direct output of the GFS forecast.

Results for the validation data by applying ML bare approaches, SMOGN and the different HCR algorithms tested in this work, i.e. HCR with $n = 2$ and HCR with $n = 4$, are shown in Tables 8, 9, 10 and 11, respectively. These results improve the performance of the direct GFS output, in all cases. In addition, as in the results shown in previous section, the proposed methodology increases the detection of EWS without worsening the prediction in the rest of the samples, but adding the intrinsic error of the forecast data used as predictor variables. In this case, it may be observed how the neural networks regressors prefer the HCR method with $n = 2$ and T = μ, while the CART regressors (RT and RF) achieve excellent results with $n = 4$ and T = $(\mu, \mu + 0.75\sigma, \mu + 1.5\sigma)$, improving by a great margin the EWS prediction made by bare ML methods. LR remains a bit lost with no remarkable results in any case. Note that the validation data used in this section its not exactly the same as the one used in Section 4.5, since the sample rate of the GFS data employed is only of 4 samples per day. Therefore, metrics shown in this section are not directly comparable to those displayed previously.

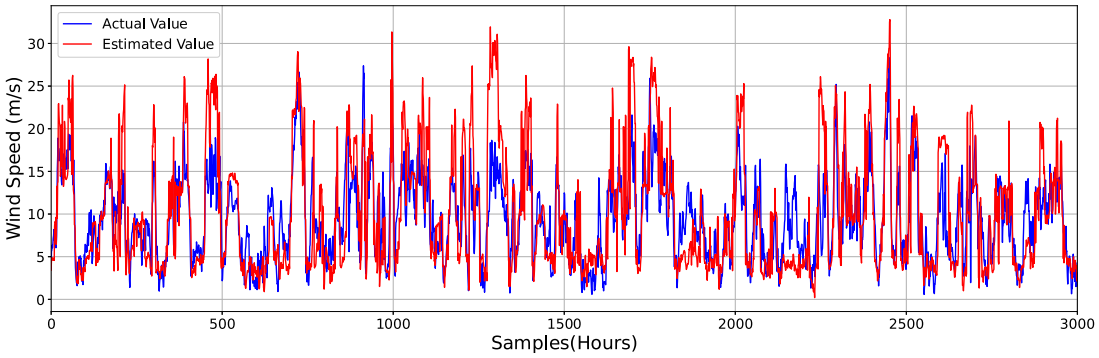


(a) $T = (Q_1, Q_2, Q_3)$ (b) $T = (\mu, \mu + 0.75\sigma, \mu + 1.5\sigma)$

Fig. 18. Imbalance degree of the training subsets according to the threshold selected.



(a) $T = (Q_1, Q_2, Q_3)$



(b) $T = (\mu, \mu + 0.75\sigma, \mu + 1.5\sigma)$

Fig. 19. Comparison of actual wind speed (blue) with forecasted wind speed (red) using HCR with $n = 4$, ELM as regressor and different values of threshold.

Table 8
Error metrics for ML methods applied on imbalanced training data.

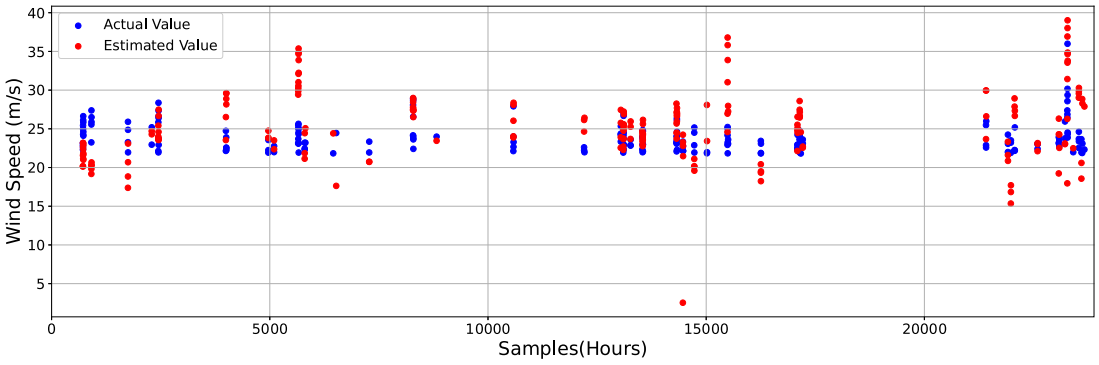
		MAE	EEMAE	CWMAE	TPT	FPR	G-Mean
ML methods	RL	5.04	18.55	9.24	0.00	0.00	0.00
	RT	3.56	8.59	4.92	0.33	0.00	0.58
	RF	2.58	8.59	4.07	0.00	0.00	0.00
	MLP	8.47	17.92	10.56	0.00	0.25	0.00
	ELM	7.47	11.53	6.54	0.33	0.08	0.55

Table 9
Error metrics for ML methods applied after training data was balanced with SMOGN method.

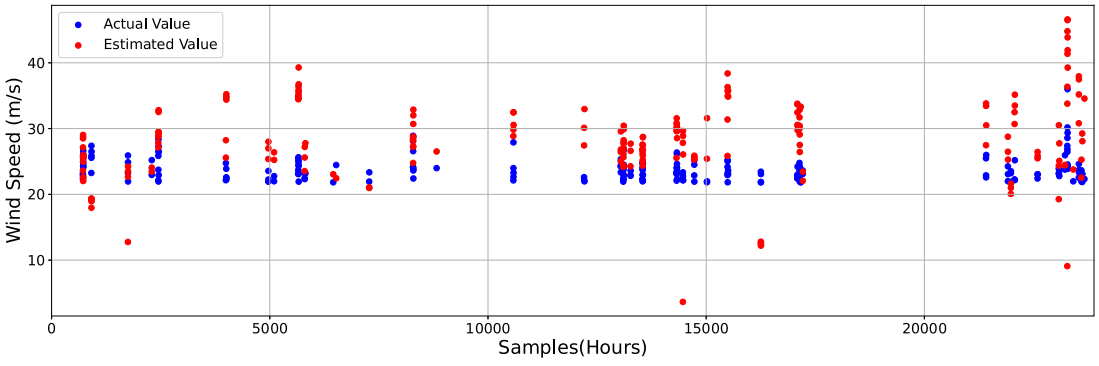
		MAE	EEMAE	CWMAE	TPT	FPR	G-Mean
SMOGN + ML	RL	7.91	15.97	9.84	0.25	0.20	0.45
	RT	4.43	5.00	4.49	0.26	0.05	0.56
	RF	3.12	5.66	3.04	0.00	0.01	0.00
	MLP	9.55	9.20	11.72	0.67	0.32	0.67
	ELM	5.50	5.52	6.62	0.33	0.01	0.57

Finally, Fig. 24 shows the time series wind speed prediction using GFS forecast data as predictive variables and applying the proposed HCR methodology, for the best performing case.

It is important to remark that the time horizon of the forecast GFS data used as predictors is independent of the prediction horizon of the model. In this paper, models to predict wind speed with 5 different time horizons are presented (+1 h, +3 h, +6 h, +12 h and +24 h) Appendix A. If the user selects the model to predict the next hour's wind speed and enters as predictive variables GFS data for +3 h, the



(a) $T = (Q_1, Q_2, Q_3)$



(b) $T = (\mu, \mu + 0.75\sigma, \mu + 1.5\sigma)$

Fig. 20. Performance of the regression model in actual wind extremes using HCR with $n = 4$, ELM as regressor and different values of threshold.

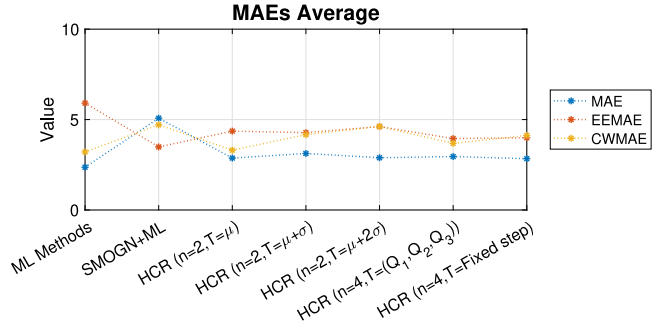


Fig. 21. MAE, EEMAE and CWMAE for different methodologies tested averaging the performance of the five regressors.

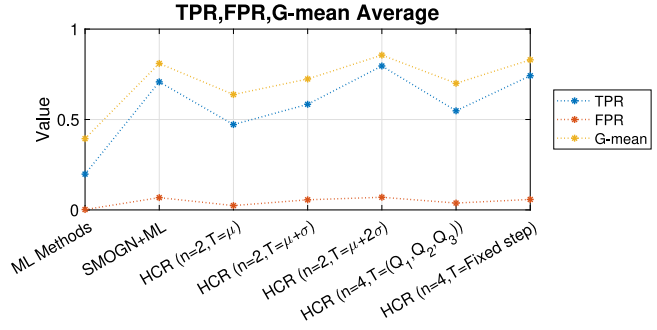


Fig. 22. TPR, FPR and G-mean for different methodologies tested averaging the performance of the five regressors.

Table 10
Error metrics for HCR methodology with $n = 2$ and different threshold values.

		MAE	EEMAE	CWMAE	TPR	FPR	G-Mean
$T = \mu$	RL	5.92	8.31	7.82	0.55	0.21	0.63
	RT	3.98	9.97	5.12	0.00	0.04	0.00
	RF	3.08	9.97	5.12	0.00	0.01	0.00
	MLP	6.17	1.73	6.74	0.67	0.21	0.73
	ELM	5.73	3.20	5.21	0.90	0.13	0.93
$T = \mu + \sigma$	RL	5.88	8.31	7.77	0.51	0.21	0.63
	RT	3.99	9.13	4.82	0.00	0.03	0.00
	RF	3.26	6.21	3.93	0.00	0.03	0.00
	MLP	6.43	4.70	7.75	0.82	0.21	0.89
	ELM	7.10	11.08	8.74	0.67	0.20	0.73
$T = \mu + 2\sigma$	RL	6.97	11.66	10.97	0.58	0.16	0.65
	RT	3.91	6.35	4.80	0.40	0.07	0.56
	RF	3.15	5.54	4.39	0.37	0.05	0.56
	MLP	6.59	11.85	10.14	0.67	0.15	0.75
	ELM	7.62	11.30	12.32	0.72	0.15	0.75

Table 11
Error metrics for HCR methodology with $n = 4$ and different threshold values.

		MAE	EEMAE	CWMAE	TPR	FPR	G-Mean
$T = (Q_1, Q_2, Q_3)$	RL	8.18	11.02	9.86	0.28	0.25	0.43
	RT	3.80	7.11	4.39	0.33	0.05	0.33
	RF	3.15	6.37	3.37	0.43	0.01	0.30
	MLP	6.07	6.76	7.45	0.45	0.26	0.50
	ELM	7.12	2.06	8.72	0.93	0.31	0.83
$T = (\mu, \mu + 0.75\sigma, \mu + 1.5\sigma)$	RL	6.65	9.89	9.12	0.25	0.14	0.46
	RT	3.18	2.53	3.22	0.95	0.03	0.99
	RF	3.49	2.88	3.87	0.73	0.05	0.80
	MLP	7.66	10.87	12.26	0.90	0.26	0.86
	ELM	6.16	11.07	8.08	0.34	0.19	0.52

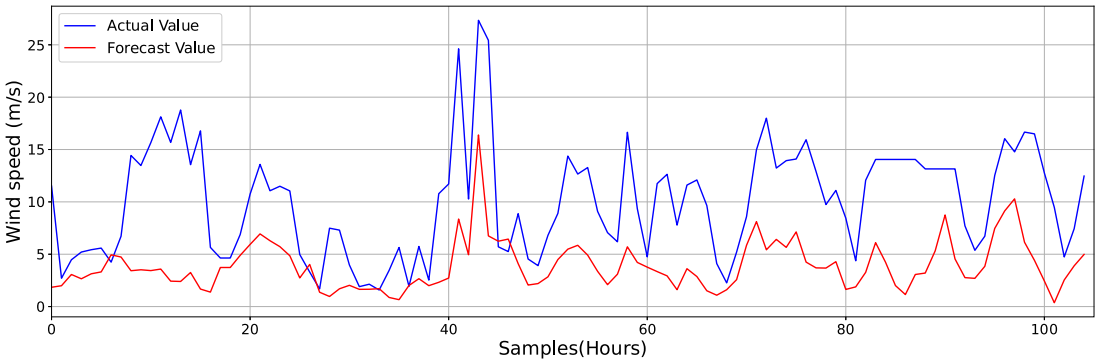


Fig. 23. Wind speed prediction with the direct output of the GFS forecast (+3 h).

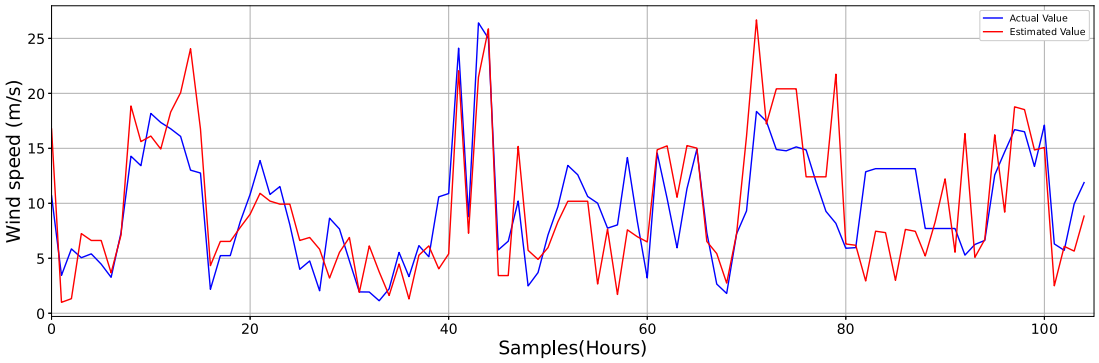


Fig. 24. Comparison of actual wind speed (blue) with forecasted wind speed (red) using GFS forecast data (+3 h) as predictors to forecast wind speed in the following hour. HCR is applied with $n = 4$, RT as regressor and T $= (\mu, \mu + 0.75\sigma, \mu + 1.5\sigma)$.

model will return the wind speed prediction in +4 h. Therefore, the selection of a specific time horizon in the forecast data is not relevant in the analysis of the algorithms comparison, the only difference would be that the intrinsic error of the prediction would be larger, as the prediction time horizon increases.

5. Conclusions

The prediction of occurrence and severity of EWS represents one of the most significant challenges in wind speed forecasting nowadays. A proper foresight of these extreme events may lead to a great economic benefit, as it prevents damages in wind farms facilities or wind turbines. Besides, an accurate estimation of the intensity of these phenomena prevents cut-outs events, thus maximizing the energy power produced in the wind farm.

One of the main problems of extreme value forecasting is the use of highly unbalanced databases, as the values that are most interesting to estimate accurately are those that are poorly represented on the dataset. This leads to the need of applying specific data processing techniques, since applying ML methods directly results in a fairly good prediction of wind speed at non-extreme values, but a very weak prediction at extreme events.

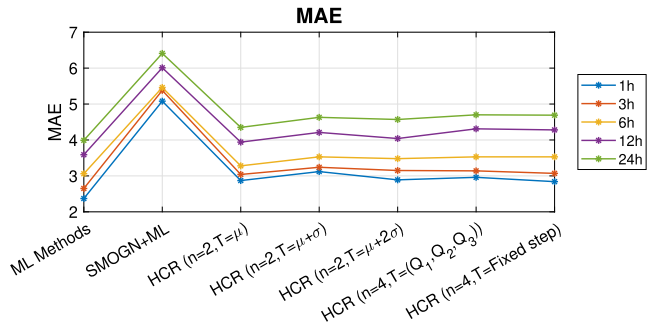
Numerous data balancing techniques exist, most of them focused on classification problems. One of the most well-known techniques of data balancing in regression, SMOGN, has been applied in this paper, achieving quite favorable results in the prediction of extreme events, but excessively worsening the quality of the prediction in the rest of the values, as well as increasing considerably the number of false alarms, which is unacceptable since it would cause large economic losses.

The methodology proposed in this paper is an HCR architecture composed of 3 levels, where the first one consists of dividing the training data into n groups according to the target variable value, labeling each group and then applying data balancing techniques to equilibrate the groups in terms of number of samples. The second level involves the training of a pool of classifiers, which will subsequently have to determine the group to which an unknown incoming test sample belongs. Finally, each training data subset is used to fit a separate regression model, each focused on a range of values within the domain of the target variable, so that some will be focused on extreme events only while others will deal with non-extreme or intermediate values. In the prediction phase, once the pool of classifiers has determined to which group the incoming test sample belongs, the regressor associated to that training group is responsible for determining the estimation value.

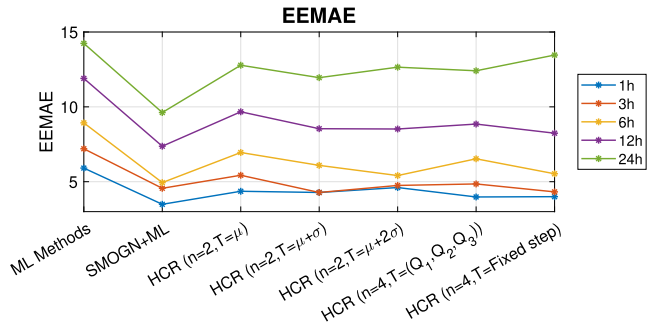
The implementation of this methodology has yielded very satisfactory results, achieving higher EWS prediction ratios than those obtained with SMOGN, while maintaining a lower false alarm ratio and a better prediction of non-extreme events.

Different architectures of the HCR methodology have been assessed, applying various numbers of clusters and different thresholds for their division. The conclusion of these tests is the existing trade-off between the correct prediction of extreme values without excessively worsening the prediction of non-extremes. When setting a number of cluster equal to 2, $n = 2$, depending on the threshold value selected, there arise more conservative architectures ($T = \mu$), that do not degrade the prediction of non-extreme values without reaching high EWS detection rates, and more aggressive architectures ($T = \mu + 2\sigma$) which obtain excellent prediction rates of extreme values (92%), with a false alarm rate of 8%, or an intermediate architectures between the two ($T = \mu + \sigma$). Therefore the selection of one model or another will depend on the specific problem tackled and what it is considered as most crucial in that case.

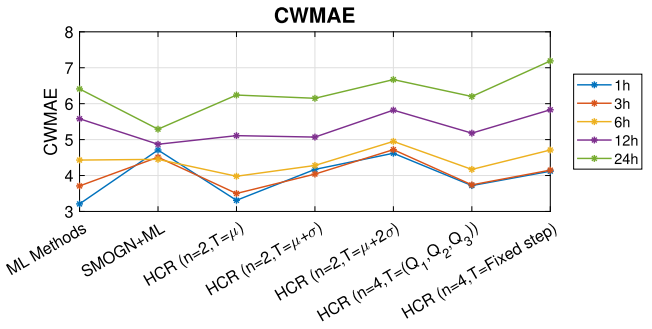
Increasing the number of clusters to 4 results in a significant overall improvement of the wind speed forecasting, providing a very strong detection of extreme events (TPR = 95%) while maintaining accurate prediction levels of non-extreme events. It can be concluded that this is the most advantageous option presented in this paper.



(a) MAE

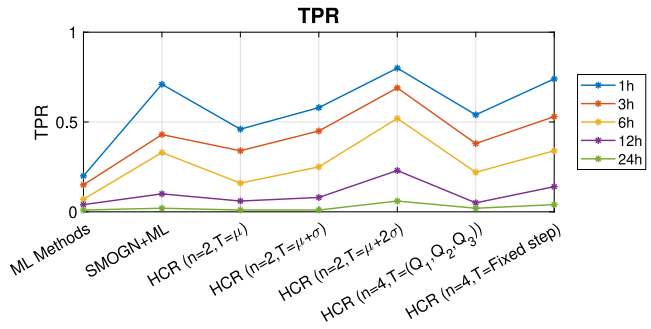


(b) EEMAE

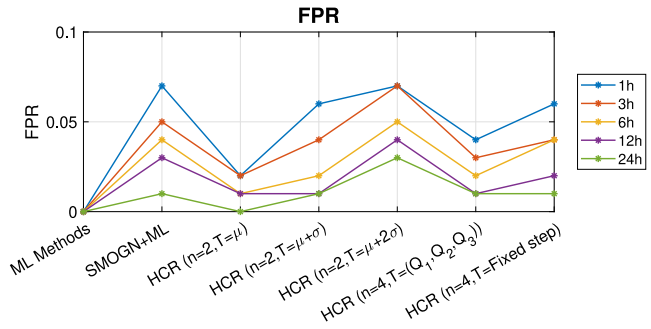


(c) CWMAE

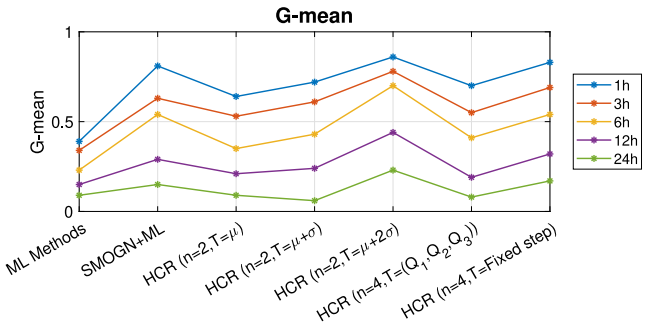
Fig. A.25. Average MAE, EEMAE and CWMAE for the five regressors using different prediction horizons.



(a) TPR



(b) FPR



(c) G-mean

Fig. A.26. Average TPR, FPR and G-mean for the five regressors using different prediction horizons.

In addition, predictions have also been evaluated with five different time horizons, ranging from short-term to long-term (1, 3, 6, 12 and 24 h). Two main conclusions can be drawn from this study: first, and as expected, all error metrics worsen as the prediction horizon increases, since the complexity of the problem consequently increases. Second, the conclusions extracted for the HCR approach on EWS presented previously are constant for all time horizons studied, i.e., applying the proposed methodology is the best option in all cases to significantly improve the prediction of extremes while maintaining an acceptable false alarm rate and admissible forecasting of non-extremes.

As a general conclusion, the proposed HCR methodology obtains very satisfactory results in the prediction of extreme wind events, while preserving a strong forecast in the remaining non-extremes values. This approach can be beneficial to avoid damages in wind farms caused by EWS and to maximize the energy produced by wind turbines.

CRediT authorship contribution statement

C. Peláez-Rodríguez: Software, Methodology, Conceptualization, Investigation, Writing – original draft. **J. Pérez-Aracil:** Conceptualization, Software, Validation, Writing – review & editing. **D. Fister:** Writing – review & editing, Investigation, Methodology. **L. Prieto-Godino:** Resources, Visualization. **R.C. Deo:** Writing – review & editing, Methodology, Supervision. **S. Salcedo-Sanz:** Writing – review & editing, Supervision, Project administration, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research has been partially supported by the project PID2020-115454GB-C21595 of the Spanish Ministry of Science and Innovation (MICINN). This research has also been partially supported by the European Union, through H2020 Project “CLIMATE INTELLIGENCE Extreme events detection, attribution and adaptation design using machine learning (CLINT)”, Ref: 101003876-CLINT.

Table B.12
G-mean of the classifiers' pool, for $n = 2$ and different values of the threshold.

Balancing method	Classifier	$T = \mu$	$T = \mu + \sigma$	$T = \mu + 2\sigma$
No balancing	SVM	0.85	0.80	0.61
	RF	0.85	0.80	0.62
	GNB	0.71	0.72	0.71
	kNN	0.81	0.72	0.52
	AB	0.80	0.75	0.59
	MLP	0.83	0.81	0.67
SMOTE	SVM	–	0.86	0.86
	RF	–	0.84	0.78
	GNB	–	0.73	0.73
	kNN	–	0.83	0.84
	AB	–	0.83	0.85
	MLP	–	0.80	0.74
Borderline-SMOTE	SVM	–	0.85	0.86
	RF	–	0.84	0.76
	GNB	–	0.75	0.75
	kNN	–	0.82	0.83
	AB	–	0.83	0.85
	MLP	–	0.80	0.74
SVM-SMOTE	SVM	–	0.87	0.87
	RF	–	0.85	0.79
	GNB	–	0.72	0.72
	kNN	–	0.83	0.83
	AB	–	0.83	0.83
	MLP	–	0.83	0.73
ADASYN	SVM	–	0.85	0.86
	RF	–	0.84	0.78
	GNB	–	0.74	0.75
	kNN	–	0.82	0.84
	AB	–	0.83	0.85
	MLP	–	0.82	0.72
K-Means SMOTE	SVM	–	0.83	0.80
	RF	–	0.81	0.64
	GNB	–	0.68	0.50
	kNN	–	0.79	0.74
	AB	–	0.79	0.72
	MLP	–	0.82	0.74
Random undersampling	SVM	–	0.87	0.89
	RF	–	0.87	0.89
	GNB	–	0.72	0.73
	kNN	–	0.84	0.86
	AB	–	0.84	0.86
	MLP	–	0.85	0.87
	Voting Clas.	0.85	0.86	0.84

Table B.13
G-mean of the classifiers' pool, for $n = 4$ and for different values of the threshold.

Balancing method	Classifier	$T = (Q_1, Q_2, Q_3)$	$T = (\mu, \mu + 0.75\sigma, \mu + 1.5\sigma)$
No balancing	SVM	0.55	0.52
	RF	0.56	0.55
	GNB	0.37	0.32
	kNN	0.50	0.44
	AB	0.49	0.41
	MLP	0.51	0.54
SMOTE	SVM	–	0.58
	RF	–	0.58
	GNB	–	0.34
	kNN	–	0.51
	AB	–	0.51
	MLP	–	0.51
Borderline-SMOTE	SVM	–	0.58
	RF	–	0.58
	GNB	–	0.33
	kNN	–	0.49
	AB	–	0.50
	MLP	–	0.54
SVM-SMOTE	SVM	–	0.57
	RF	–	0.57
	GNB	–	0.33
	kNN	–	0.50
	AB	–	0.50
	MLP	–	0.54
ADASYN	SVM	–	0.57
	RF	–	0.58
	GNB	–	0.32
	kNN	–	0.50
	AB	–	0.50
	MLP	–	0.52
K-Means SMOTE	SVM	–	0.54
	RF	–	0.54
	GNB	–	0.36
	kNN	–	0.46
	AB	–	0.42
	MLP	–	0.51
Random undersampling	SVM	–	0.51
	RF	–	0.53
	GNB	–	0.33
	kNN	–	0.43
	AB	–	0.39
	MLP	–	0.52
	Voting Clas.	0.55	0.55

Appendix A. Analysis of different prediction time horizons

In this Appendix we show an analysis of the effect of the prediction time-horizon in the performance of the proposed approach, in order to complete the description of the algorithm's performance. Five different prediction time-horizons have been considered, covering from short-term forecasting to long-term forecasting: 1 h, 3 h, 5 h, 12 h and 24 h. Average metrics for the five regressors implemented are shown in Figs. A.25 and A.26 comparing the six error metrics for the different prediction time horizons.

The trend is clearly visible in all the graphs, where it can be observed how the predictions degrade as the value of the time-horizon increases. In addition, the conclusions drawn previously for the one-hour-ahead prediction can be extrapolated to the other time horizons.

For all cases, the best MAE are obtained with ML methods (Fig. A.25(a)). The implementation of SMOGN causes these values to deteriorate drastically, while with the HCR architecture they remain in an acceptable range.

Regarding the EEMAE (Fig. A.25(b)), the values obtained with SMOGN
+ML are the lowest as in the 1h-ahead-prediction, but the HCR approaches reduce this metric with respect to the initial ML methods in all cases.

In the case of CWMAE (Fig. A.25(c)), slightly different results are obtained, with some fluctuation in the results of some methodology, obtaining better metrics for more distant horizons. Also, a different tendency is appreciated when using SMOGN + ML methods, while increasing the time horizon of the prediction, the CWMAE becomes better compared to the CWMAE of the initial ML methods.

The same trend can be found in the TPR, FPR and G-mean plots (Fig. A.26), forecasting with more hours in advance results in fewer extremes being detected, achieving the best results in all cases with HCR architectures.

Appendix B. Performance of the classifiers' pool

In this Appendix the performance of the different classifiers used is shown. These classifiers are trained with the labeled training data after applying the different data balancing techniques when necessary. Table B.12 reports the performance of the classifiers for a number of clusters $n = 2$. In this case, it can be observed how the most accurate balancing method is the random undersampling for most of the classifiers. Also, a threshold value of $T = \mu + \sigma$ is the one that provide better results.

Table B.13 shows the results when $n = 4$. Here, neither balancing method is clearly superior, but rather, for each type of classifier, one

or another method performs better. Also, it can be observed how the results are worse for a higher number of clusters compared to the previous case (Table B.12), this is explained by the fact that increasing the number of subsets consequently increases the complexity of the classification task.

References

[1] Y. Kumar, J. Ringenberg, S.S. Depuru, V.K. Devabhaktuni, J.W. Lee, E. Nikolaidis, B. Andersen, A. Afjeh, Wind energy: Trends and enabling technologies, Renew. Sustain. Energy Rev. 53 (2016) 209–224, http://dx.doi.org/10.1016/j.rser.2015.07.200.

[2] W.Y. Chang, Short-term wind power forecasting using the enhanced particle swarm optimization based hybrid method, Energies 6 (2013) 4879–4896, http://dx.doi.org/10.3390/en6094879.

[3] M. Brenna, F. Foiadelli, M. Longo, D. Zaninelli, Improvement of wind energy production through HVDC systems, Energies 10 (2017) http://dx.doi.org/10.3390/en10020157.

[4] S. Ali, S.M. Lee, C.M. Jang, Techno-economic assessment of wind energy potential at three locations in South Korea using long-term measured wind data, Energies 10 (2017) http://dx.doi.org/10.3390/en10091442.

[5] H. Demolli, A.S. Dokuz, A. Ecemis, M. Gokcek, Wind power forecasting based on daily wind speed data using machine learning algorithms, Energy Convers. Manage. 198 (2019) http://dx.doi.org/10.1016/j.enconman.2019.111823.

[6] S.S. Soman, H. Zareipour, O. Malik, P. Mandal, A review of wind power and wind speed forecasting methods with different time horizons, in: North American Power Symposium 2010, IEEE, 2010, pp. 1–8.

[7] M. Lydia, S.S. Kumar, A comprehensive overview on wind power forecasting, in: 2010 9th International Power and Energy Conference, IPEC 2010, 2010, pp. 268–273, http://dx.doi.org/10.1109/IPECON.2010.5697118.

[8] A.M. Foley, P.G. Leahy, A. Marvuglia, E.J. McKeogh, Current methods and advances in forecasting of wind power generation, Renew. Energy 37 (2012) 1–8, http://dx.doi.org/10.1016/j.renene.2011.05.033.

[9] J. Shi, J. Guo, S. Zheng, Evaluation of hybrid forecasting approaches for wind speed and power generation time series, Renew. Sustain. Energy Rev. 16 (2012) 3471–3480, http://dx.doi.org/10.1016/j.rser.2012.02.044.

[10] J. Jung, R.P. Broadwater, Current status and future advances for wind speed and power forecasting, Renew. Sustain. Energy Rev. 31 (2014) 762–777, http://dx.doi.org/10.1016/j.rser.2013.12.054.

[11] A. Tascikaraoglu, M. Uzunoglu, A review of combined approaches for prediction of short-term wind speed and power, Renew. Sustain. Energy Rev. 34 (2014) 243–254.

[12] Y. Wang, R. Zou, F. Liu, L. Zhang, Q. Liu, A review of wind speed and wind power forecasting with deep neural networks, Appl. Energy 304 (2021) 117766.

[13] W. Xiaolan, L. Hui, One-month ahead prediction of wind speed and output power based on EMD and LSSVM, in: 2009 International Conference on Energy and Environment Technology, ICEET 2009, Vol. 3, 2009, pp. 439–442, http://dx.doi.org/10.1109/ICEET.2009.571.

[14] R.G. Kavasseri, K. Seetharaman, Day-ahead wind speed forecasting using f-ARIMA models, Renew. Energy 34 (2009) 1388–1393, http://dx.doi.org/10.1016/j.renene.2008.09.006.

[15] A. Kusiak, H. Zheng, Z. Song, Short-term prediction of wind farm power: A data mining approach, IEEE Trans. Energy Convers. 24 (2009) 125–136, http://dx.doi.org/10.1109/TEC.2008.2006552.

[16] M. Pérez-Ortiz, S. Jiménez-Fernández, P.A. Gutiérrez, E. Alexandre, C. Hervás-Martínez, S. Salcedo-Sanz, A review of classification problems and algorithms in renewable energy applications, Energies 9 (8) (2016) 607.

[17] V. Bali, A. Kumar, S. Gangwar, Deep learning based wind speed forecasting-A review, in: 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence), IEEE, 2019, pp. 426–431.

[18] S. Outten, S. Sobolowski, Extreme wind projections over Europe from the Euro-CORDEX regional climate models, Weather Clim. Extremes 33 (2021) http://dx.doi.org/10.1016/j.wace.2021.100363.

[19] H. Wang, Y.-M. Zhang, J.-X. Mao, H.-P. Wan, A probabilistic approach for short-term prediction of wind gust speed using ensemble learning, J. Wind Eng. Ind. Aerodyn. 202 (2020) 104198.

[20] J.P. Palutikof, B. Brabson, D.H. Lister, S. Adcock, A review of methods to calculate extreme wind speeds, Meteorol. Appl. 6 (2) (1999) 119–132.

[21] J. Wang, S. Qin, S. Jin, J. Wu, Estimation methods review and analysis of offshore extreme wind speeds and wind energy resources, Renew. Sustain. Energy Rev. 42 (2015) 26–42.

[22] P. Sheridan, Current gust forecasting techniques, developments and challenges, Adv. Sci. Res. 15 (2018) 159–172.

[23] B. Schulz, S. Lerch, Machine learning methods for postprocessing ensemble forecasts of wind gusts: A systematic comparison, 2021, arXiv preprint arXiv:2106.09512.

[24] P.J. Sallis, W. Claster, S. Hernández, A machine-learning algorithm for wind gust prediction, Comput. Geosci. 37 (9) (2011) 1337–1344.

[25] S. Shanmuganathan, P. Sallis, Data mining methods to generate severe wind gust models, Atmosphere 5 (1) (2014) 60–80.

[26] R. Lagerquist, A. McGovern, T. Smith, Machine learning for real-time prediction of damaging straight-line convective wind, Weather Forecast. 32 (6) (2017) 2175–2193.

[27] A.C. Spassiani, M.S. Mason, Application of self-organizing maps to classify the meteorological origin of wind gusts in Australia, J. Wind Eng. Ind. Aerodyn. 210 (2021) 104529.

[28] D.J. Cannon, D.J. Brayshaw, J. Methven, P.J. Coker, D. Lenaghan, Using reanalysis data to quantify extreme wind power generation statistics: A 33 year case study in Great Britain, Renew. Energy 75 (2015) 767–778.

[29] M.P. Pes, E.B. Pereira, J.A. Marengo, F.R. Martins, D. Heinemann, M. Schmidt, Climate trends on the extreme winds in Brazil, Renew. Energy 109 (2017) 110–120.

[30] J. Coburn, S.C. Pryor, Do machine learning approaches offer skill improvement for short-term forecasting of wind gust occurrence and magnitude? Weather Forecast. 37 (5) (2022) 525–543.

[31] M. Arul, A. Kareem, M. Burlando, G. Solari, Machine learning based automated identification of thunderstorms from anemometric records using shapelet transform, J. Wind Eng. Ind. Aerodyn. 220 (2022) 104856.

[32] V. López, A. Fernández, S. García, V. Palade, F. Herrera, An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics, Inform. Sci. 250 (2013) 113–141, http://dx.doi.org/10.1016/j.ins.2013.07.007.

[33] G.E. Batista, R.C. Prati, M.C. Monard, A study of the behavior of several methods for balancing machine learning training data, ACM SIGKDD Explor. Newsl. 6 (1) (2004) 20–29.

[34] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, SMOTE: Synthetic minority over-sampling technique, J. Artificial Intelligence Res. 16 (2002) 321–357.

[35] P. Branco, R.P. Ribeiro, L. Torgo, B. Krawczyk, N. Moniz, SMOGN: a pre-processing approach for imbalanced regression, Proc. Mach. Learn. Res. 74 (2017) 36–50.

[36] Y. Ren, L. Zhang, P.N. Suganthan, Ensemble classification and regression-recent developments, applications and future directions [review article], IEEE Comput. Intell. Mag. 11 (2016) 41–53, http://dx.doi.org/10.1109/MCI.2015.2471235.

[37] S.W. Ke, W.C. Lin, C.F. Tsai, Y.H. Hu, Soft estimation by hierarchical classification and regression, Neurocomputing 234 (2017) 27–37, http://dx.doi.org/10.1016/j.neucom.2016.12.037.

[38] J.Y. Wu, M. Wu, Z. Chen, X. Li, R. Yan, A joint classification-regression method for multi-stage remaining useful life prediction, J. Manuf. Syst. 58 (2021) 109–119, http://dx.doi.org/10.1016/j.jmsy.2020.11.016.

[39] S. Kohli, S. Prakash, P. Gupta, Hierarchical age estimation with dissimilarity-based classification, Neurocomputing 120 (2013) 164–176.

[40] M.N.C.A. Lima, W.L. Soares, I.R.R. Silva, R.A.D.A. Fagundes, A combined model based on clustering and regression to predicting school dropout in higher education institution, Int. J. Comput. Appl. 176 (2020) 975–8887.

[41] J.S. Chou, C.F. Tsai, Concrete compressive strength analysis using a combined classification and regression technique, Autom. Constr. 24 (2012) 52–60, http://dx.doi.org/10.1016/j.autcon.2012.02.001.

[42] J.S. Chou, C.F. Tsai, Preliminary cost estimates for thin-film transistor liquid-crystal display inspection and repair equipment: A hybrid hierarchical approach, Comput. Ind. Eng. 62 (2012) 661–669, http://dx.doi.org/10.1016/j.cie.2011.11.037.

[43] H. Han, W.-Y. Wang, B.-H. Mao, LNCS 3644 - borderline-SMOTE: A new over-sampling method in imbalanced data sets learning, LNCS 3644 (2005) 878–887.

[44] H.M. Nguyen, E.W. Cooper, K. Kamei, Borderline over-sampling for imbalanced data classification, Int. J. Knowl. Eng. Soft Data Paradig. 3 (2011) 4–21.

[45] H. He, Y. Bai, E.A. Garcia, S. Li, ADASYN: Adaptive synthetic sampling approach for imbalanced learning, in: 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), Vol. 1, 2008, pp. 1322–1328, http://dx.doi.org/10.1109/IJCNN.2008.4633969.

[46] F. Last, G. Douzas, F. Bacao, Oversampling for imbalanced learning based on k-means and smote, 2017, arXiv preprint arXiv:1711.00837.

[47] G. Douzas, F. Bacao, F. Last, Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE, Inform. Sci. 465 (2018) 1–20, http://dx.doi.org/10.1016/j.ins.2018.06.056.

[48] T. Hasanin, T.M. Khoshgoftaar, The effects of random undersampling with simulated class imbalance for big data, in: Proceedings - 2018 IEEE 19th International Conference on Information Reuse and Integration for Data Science, IRI 2018, Institute of Electrical and Electronics Engineers Inc., 2018, pp. 70–79, http://dx.doi.org/10.1109/IRI.2018.00018.

[49] L. Torgo, R.P. Ribeiro, B. Pfahringer, P. Branco, Smote for regression, in: Portuguese Conference on Artificial Intelligence, Springer, 2013, pp. 378–389.

[50] B. Schölkopf, A.J. Smola, R.C. Williamson, P.L. Bartlett, New support vector algorithms, Neural Comput. 12 (2000) 1207–1245, http://dx.doi.org/10.1162/089976600300015565.

[51] S. Salcedo-Sanz, J.L. Rojo-Álvarez, M. Martínez-Ramón, G. Camps-Valls, Support vector machines in engineering: an overview, Wiley Interdiscip. Rev.: Data Min. Knowl. Discov. 4 (3) (2014) 234–267.

[52] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.
[53] H. Zhang, The optimality of naive Bayes, Aa 1 (2) (2004) 3, URL www.aaai.org.
[54] A. Mucherino, P.J. Papajorgji, P.M. Pardalos, K-nearest neighbor classification, in: Data Mining in Agriculture, Springer, 2009, pp. 83–106.
[55] Y. Freund, R. Schapire, Experiments with a new boosting algorithm, in: Machine Learning: Proceedings of the Thirteenth International Conference, 1996, pp. 148–156.
[56] M.W. Gardner, S.R. Dorling, Artificial neural networks (the multilayer perceptron)-A review of applications in the atmospheric sciences, Atmos. Environ. 32 (1998) 2627–2636.
[57] C.M. Bishop, et al., Neural Networks for Pattern Recognition, Oxford University Press, 1995.
[58] N.R. Draper, H. Smith, Applied Regression Analysis, Vol. 326, John Wiley & Sons, 1998.
[59] W.-Y. Loh, Classification and regression trees, Wiley Interdiscip. Rev.: Data Min. Knowl. Discov. 1 (1) (2011) 14–23.
[60] G.B. Huang, Q.Y. Zhu, C.K. Siew, Extreme learning machine: Theory and applications, Neurocomputing 70 (2006) 489–501, http://dx.doi.org/10.1016/j.neucom.2005.12.126.
[61] D. Ruta, B. Gabrys, Classifier selection for majority voting, Inf. Fusion 6 (2005) 63–81, http://dx.doi.org/10.1016/j.inffus.2004.04.008.
[62] H. Hersbach, B. Bell, P. Berrisford, G. Biavati, A. Horányi, J. Muñoz Sabater, J. Nicolas, C. Peubey, R. Radu, I. Rozum, et al., ERA5 hourly data on single levels from 1979 to present, in: Copernicus Climate Change Service (C3S) Climate Data Store (CDS), Vol. 10, ECMWF Reading, UK, 2018.
[63] R. Barandela, J.S. Sánchez, V. Garcıa, E. Rangel, Strategies for learning in class imbalance problems, Pattern Recognit. 36 (3) (2003) 849–851.
[64] F. Yang, H.-L. Pan, S.K. Krueger, S. Moorthi, S.J. Lord, Evaluation of the NCEP global forecast system at the ARM SGP site, Mon. Weather Rev. 134 (12) (2006) 3668–3690.