Atmospheric Research 272 (2022) 106157

Contents lists available at ScienceDirect

Atmospheric Research

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

Machine learning regression and classification methods for fog events prediction

C. Castillo-Botón [a], D. Casillas-Pérez [b,*], C. Casanova-Mateo [c], S. Ghimire [d], E. Cerro-Prada [e], P.A. Gutierrez [f], R.C. Deo [d], S. Salcedo-Sanz [a]

[a] Department of Signal Processing and Communication, Universidad de Alcalá (UAH), Ctra. Madrid-Barcelona, km 33, Alcalá de Henares 28805, Madrid, Spain
[b] Department of Signal Processing and Communication, Universidad Rey Juan Carlos (URJC), Camino del Molino, 5, Fuenlabrada 28942, Madrid, Spain
[c] Department of Computer Systems, Universidad Politécnica de Madrid (UPM), Campus Sur, 28942, Madrid, Spain
[d] School of Sciences, University of Southern Queensland (USQ), Queensland, QLD 4300, Australia
[e] Department of Electrical Engineering, Electronic, Automation and Applied Physics, Universidad Politécnica de Madrid (UPM), 28012, Madrid, Spain
[f] Department of Computer Science and Numerical Analysis, Universidad de Córdoba (UCO), 14014 Córdoba, Spain

ARTICLE INFO

2000 MSC:
0000
1111
PACS:
0000
1111
Keywords:
Low-visibility events
Orographic and hill-fogs
Classification problems
Regression problems
Machine Learning algorithms

ABSTRACT

Atmospheric low-visibility events are usually associated with fog formation. Extreme low-visibility events deeply affect the air and ground transportation, airports and motor-road facilities causing accidents and traffic problems every year. Machine Learning (ML) algorithms have been successfully applied to many fog formation and low-visibility prediction problems. The associated problem can be formulated either as a regression or as a classification task, which has an impact on the type of ML approach to be used and on the quality of the predictions obtained. In this paper we carry out a complete analysis of low-visibility events prediction problems, formulated as both regression and classification problems. We discuss the performance of a large number of ML approaches in each type of problem, and evaluate their performance under a common comparison framework. According to the obtained results, we will provide indications on what the most efficient formulation is to tackle low-visibility predictions and the best performing ML approaches for low-visibility events prediction.

1. Introduction

Fog plays a crucial role in different natural and human systems, such as agriculture (Shrestha et al., 2018; Baldocchi and Waller, 2014), ecosystems management (Anber et al., 2015) and transportation (Fabbian et al., 2007; Bartoková et al., 2015; Peng et al., 2018), among others. In areas affected by drought or water deficits, fogs may also be used to improve the water availability (Klemm et al., 2012; Montecinos et al., 2018). In turn, low-visibility events, usually associated with such fog accumulations deeply affect the transportation and facilities such as airports (Fabbian et al., 2007; Miao et al., 2012; Guerreiro et al., 2020) and roads (Peng et al., 2018; Bartok et al., 2012; Wu et al., 2018). Accurate prediction of fog events, and their associated low-visibility episodes, is therefore a very important problem, with underlying consequences on different aspects of meteorological and aviation applications, algorithmic development for accurate predictions and economic impact assessment based on accidents and other weather events resulting from increased fog and reduced atmospheric visibility.

Fog events prediction has been the objective task in a number of research works in literature, representing an area of research that has been more intense in recent years. There are research works dealing with the study of fog event prediction and its relationship with the persistence, usually circumscribed to specific, yet isolated, events (Belo-Pereira, 2016), on region-dependent orographic characteristics (Bendix, 2002), studying physical-chemist features affecting fog events duration (Stolaki et al., 2015) and, more recently, dealing with fog persistence from a statistical point of view (Räsänen et al., 2018; Cornejo-Bueno et al., 2020; Salcedo-Sanz et al., 2021). There are also different works that have focused on fog events prediction in specific areas and cities (da Rocha et al., 2015; Dey, 2018) with recent publications dealing with

* Corresponding author.
E-mail addresses: carlos.castillob@uah.es (C. Castillo-Botón), david.casillas@urjc.es (D. Casillas-Pérez), carlos.casanova@upm.es (C. Casanova-Mateo), sujan.ghimire@usq.edu.au (S. Ghimire), elena.cerro@upm.es (E. Cerro-Prada), pagutierrez@uco.es (P.A. Gutierrez), ravinesh.deo@usq.edu.au (R.C. Deo), sancho.salcedo@uah.es (S. Salcedo-Sanz).

https://doi.org/10.1016/j.atmosres.2022.106157
Received 14 December 2021; Received in revised form 19 March 2022; Accepted 22 March 2022
Available online 28 March 2022

alternative approaches for this problem. Examples of this are works involving Numerical Weather Prediction (NWP) for fog prediction, usually within meso-scale numerical models. In (Bergot et al., 2007) an intercomparison of six numerical model simulations of fog is carried out for the Paris-Charles de Gaulle airport. The main goal of this intercomparison is to identify the capabilities of different NWM models to forecast fog accurately. This work recognizes that the prediction of fog with NWM is still a difficult problem, in which NWM need to be improved. In (van der Velde et al., 2010) two NWM, i.e. the WRF and HIRLAM models, are evaluated against detailed observations in a problem of fog forecasting in the Netherlands. It is shown that both NWM considered have problems to correctly modelling the fog event. In (Zhou et al., 2011) the performance of WRF-NMM NWM for low-visibility events due to fog is evaluated in North America. This works shows that the performance of the low visibility/fog forecasts from NWM models was still poor in comparison to those of precipitation forecasts from the same models. In (Román-Cascón et al., 2012) is another work that states that the prediction of fogs is one of the processes not well reproduced by the NWP models. In that particular paper observational analysis of three different periods with fogs at the Spanish Northern Plateau is carried out. The WRF numerical model is used, and comparison with different parameterizations is detailed. In (Steeneveld et al., 2015) two NWM (HARMONIE and WRF) are compared in the prediction of two warm fog events in the Netherlands. The work concludes that the NWP of radiation fog is challenging, since many NWM show large biases for the timing of the onset and dispersal of the fog, and also fail predicting the depth and liquid water content. In (Román-Cascón et al., 2016) fog simulation by the WRF model in terms of liquid water content is carry out for two different sites, Valladolid (Spain) and Cabauw (The Netherland). The WRF model was able to simulate radiation fog when properly configured at a high resolution. However, it failed in simulating several properties of the fog such as the onset, dissipation and its vertical extent. In (Román-Cascón et al., 2019) the fog-forecasting skill of two mesoscale models (WRF and HARMONIE) is evaluated for the Spanish Northern plateau, specifically in Valladolid, where radiation fog is usual in autumn and winter months. Finally, in (Fernández-González et al., 2019) the HARMONIE-AROME model is tested for the prediction of poor-visibility episodes in Tenerife island, Spain. This work reports a good performance of the NWM in the prediction of fog, with false alarm rate around 35%. Note, however, that all the previous works discussed show that the forecasting of fog events by NWM is particularly difficult, due to the extremely local characteristic of some events, and also because these phenomenon are extremely sensitive to small-scale variations in atmospheric variables (Tapiador et al., 2019).

In view of the difficulties of NWM for fog events prediction, in this study we focus on Machine Learning (ML) prediction of fog events. Specifically, we present new results and discuss the predictive approaches that deal with statistical and ML methods in fog events prediction. Historically, the very first attempts in this area backdates to the 80′ where researchers proposed the use of a linear regression algorithm to this prediction task albeit with a limited amount of success (Koziara et al., 1983). To address the limitations in conventional (e.g., linear methods), ML-based algorithms have also been successfully applied in the last decade to many of the fog prediction problems. In (Fabbian et al., 2007), a multi-layer perceptron (MLP) method was tested in a problem of fog events prediction at Canberra International Airport in Australia using meteorological observations. The data from Australian Bureau of Meteorology were used to train and test a neural networks model, obtaining promising results. In (Miao et al., 2012) a fog prediction system formed by fuzzy logic-based predictors was proposed and analyzed at Perth airport (Australia). The fuzzy logic predictor works on the outputs of a meso-scale numerical model (LAPS125) outputs, with the objective of refining the predictions obtained by the numerical model. This fog prediction model was operational at the airport by averaging the outcomes of two other fog forecasting methods by means of a majority voting approach. In (Bartoková et al., 2015) a decision tree induction ML algorithm was proposed for fog events prediction in Dubai, improving the results previously obtained by numerical model approaches. In (Colabone et al., 2015) the performance of MLPs with back-propagation training procedure in a fog event prediction problem at Academia da Força Aérea (Brasil) is analyzed. In (Boneh et al., 2015) a Bayesian network is applied to a fog prediction problem at Melbourne airport. In this case the problem is tackled as a prediction time horizon of 8-h, and 34 years of data have been used to train the network. This fog prediction system has obtained better results than the previous system becoming operational for fog prediction at the Melbourne airport. In (Cornejo-Bueno et al., 2017) different ML regression techniques have also been tested in the fog prediction problem at Valladolid airport in Spain. In this case, radiation-type fog events were the most common the zone, so the prediction problem was restricted to winter months. The authors reported successful results in fog event prediction using Support Vector Regressions (SVR) and Extreme Learning Machines (ELM) approach. In (Durán-Rosal et al., 2018) a evolutionary neural network was considered for a problem of fog events classification from meteorological input variables. Several types of evolutionary neural networks were considered by selecting different basic activaton functions such as sigmoidal, product and radial functions. A multi-target training procedure was considered, obtaining excellent results in the fog event classification problem considered. In (Guijo-Rubio et al., 2018) a problem of low-visibility events due to fog was tackled by applying ordinal classification methods. Three classes were considered (i.e., fog, mist and no-fog), and different ordinal classifiers successfully tested in this problem of fog event prediction. Recently, in (Miao et al., 2020) the performance of an Long Short-Term Memory network was evaluated in a problem of fog prediction particularly using the classification task approach for the study region in Anhui, China.

In this paper we discuss some relevant ML models, and their performance in a fog event prediction problem. We consider the visibility values at the ground level in order to characterize these fog events, and from these data we consider different tasks to define the associated prediction problem as: (i) exact prediction of visibility (regression problem), (ii) thresholding visibility values, (iii) separation into their classes as a classification problem, and (iv) a classification problem considering the order among the classes considering ordinal classifications. For each prediction task we evaluate the performance of a number of ML techniques and then carry out a comprehensive inter-comparison. To the best the the author's knowledge this is the first work dealing with fog events (or its associated low-visibility effect) prediction that compares the performance of several ML techniques when the problem is defined in terms of these different tasks, providing s clarification on what is the best way of defining a fog prediction problem, and what are the best ML techniques for each of these cases. More specifically, the novelty and important contributions of the research paper are the following:

- We provide an exhaustive benchmarking for both the regression and the classification problem definitions in low-visibility events prediction due to fog, considering a large set of ML methods.
- By defining the low-visibility prediction problem as a regression task we study the sensitivity of each ML model to the standardization and normalization of the input or predictor variables and a feature reduction analysis trough Principal Component Analysis (PCA).
- Considering classification tasks, we include a robust study of the most common balancing techniques applied to solve the highly unbalance characteristic of the problem. We also discuss whether the ordinal classification, or otherwise, is a robust way to define and tackle the problem in this case.
- According to the results of the present study, we give further insights on the best combination of normalization and standardization methods in our regression methods and we obtain significantly accurate predictions with this problem formulation by applying neural

computation approaches. In the classification of fog events, we discuss the optimal techniques that should be used for facing the problem as a classification task. We have found that the ordinal classification algorithms do not provide good results and therefore warrants discarding this way of defining the problem to obtain acceptable and quality solution.

The rest of the paper has been structured as follows: the next section describes the fog events database made available to this study, focused on Mondoñedo weather station, Galicia, Spain. Section 3 summarizes the most important characteristic of the regression, classification and ordinal classification methods compared in this paper. Section 4 presents the experimental part of the study with comprehensive experiments and comparisons utilising the different ML techniques considered in this problem of fog events prediction. Section 5 closes the paper with some final conclusions and remarks on the research carried out. Appendix A shows a list of acronyms to facilitate the reading of the article.

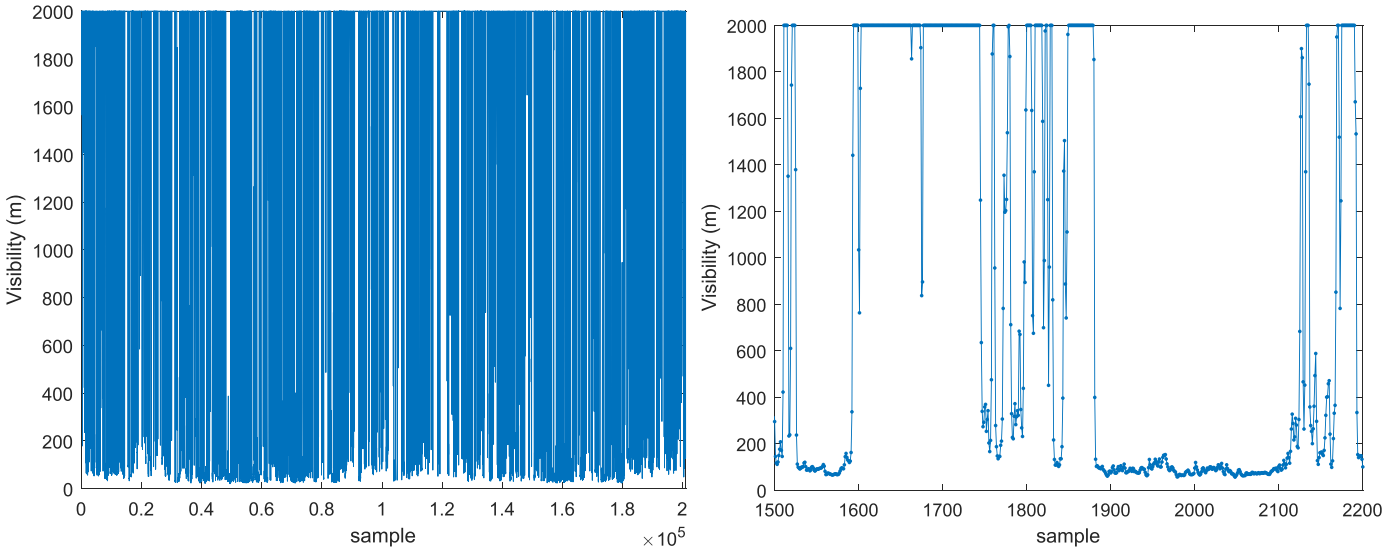
2. Fog events database

We consider real visibility data that were acquired at the Mondoñedo weather station, Galicia, Spain (43.3841°N, 7.3692°W). The Mondoñedo area is known to experience frequent low-visibility events caused by orografic fog coming from the Cantabrian sea. Fog events in this area are so deep and intense that they usually cause partial or total motor-roads closures providing consequential economic and social impacts for the entire zone. A statistical analysis of the visibility in this zone has also been recently presented in (Cornejo-Bueno et al., 2021).

We use the visibility data acquired by a Biral WS-100 visibility sensor at the Mondoñedo weather station to set the fog-even prediction as a regression and a classification problem. The sensor operates between 0 to 2000 m, hence, the limit for a fog event detection using this sensor has been set to 2000m. The sample period has been set to 30 minutes. Fig. 1 shows an example of the data acquired by the sensor in a window step. For the regression problem we directly use as the target variable (i.e., ground truth) which is the visibility series obtained by this sensor. We considered a time series of 23 months of these data from 1st January 2018 to 30^{th} November 2019. In this study we also include the exogenous meteorological variables registered by the same weather station that collects information about the atmospheric states. In order to consider the problem as a classification task, a number of classes have been introduced by considering a statistical analysis (Abdel-Aty et al., 2015) [the table with the International classification of visibility developed by the the Meteorological Office published in 1969] to determine which were the best class thresholds to classify the low-visibility events. Note that we have modified these thresholds slightly to accord with the frequency in the data and intensity values in the present zone of study. Specifically, the classes for this work have been obtained applying the following piece-wise function where *vs* stands for the visibility in meters:

$$C_i = \begin{cases} 0, & vs < 40 & i = 0\bullet \\ 1, & 40£vs < 200 & i = 1\bullet \\ 2, & 200£vs < 500 & i = 2\bullet \\ 3, & 500£vs < 2000 & i = 3\bullet \\ 4, & vs^3 2000 & i = 4\bullet \end{cases} \quad (1)$$

Fig. 2 represents the correspondence between the visibility values (measured in meters) obtained from the sensor and the five classes used in the classification problem. To obtain this representation the raw visibility values have been sorted in ascending order and then plotted. The x-axis shows the ordering of each visibility value inside the whole dataset. The y-axis shows the whole range of the visibility variable, from 0 meters to 2000 meters. By presenting it this way, we can see how the visibility ranges can translate into the classes (see Eq. (1)). From this figure we can obtain important information related to the problem at hand. First, if we were to project each color into the x-axis, we would obtain the number of instances for each class. As it can be seen, we have a highly imbalanced problem, where more than half of the instances belong to a single class and the rest of the classes represent a minority of the data. The great majority of the time the visibility is excellent, with a visibility of at least 2 kilometers. On the contrary, if we observe the amount of instances of class 0 we see that they are minuscule. If we add classes 1 and 2 then we have a bigger share of the data, but it is still much less than half. As it usually happens with imbalanced data, we are interested in the underrepresented values, which is the event of our interest, i.e., low visibility. We want to learn when and how fog appears, and to predict these rare events and further take preventive measures on the roads and airports. In Subsection 3.5 we discuss about how we dealt with the data imbalance issues. Second, if we project the colors on the y-axis, depending on how long along the y-axis a class appears in the graph, the more range of the visibility is included in that class. From this we can see that we are not dealing with a linear correspondence, which



(a) Complete available series (b) Zoom

Fig. 1. Visibility time series (ground truth) at the Mondoñedo measuring station.

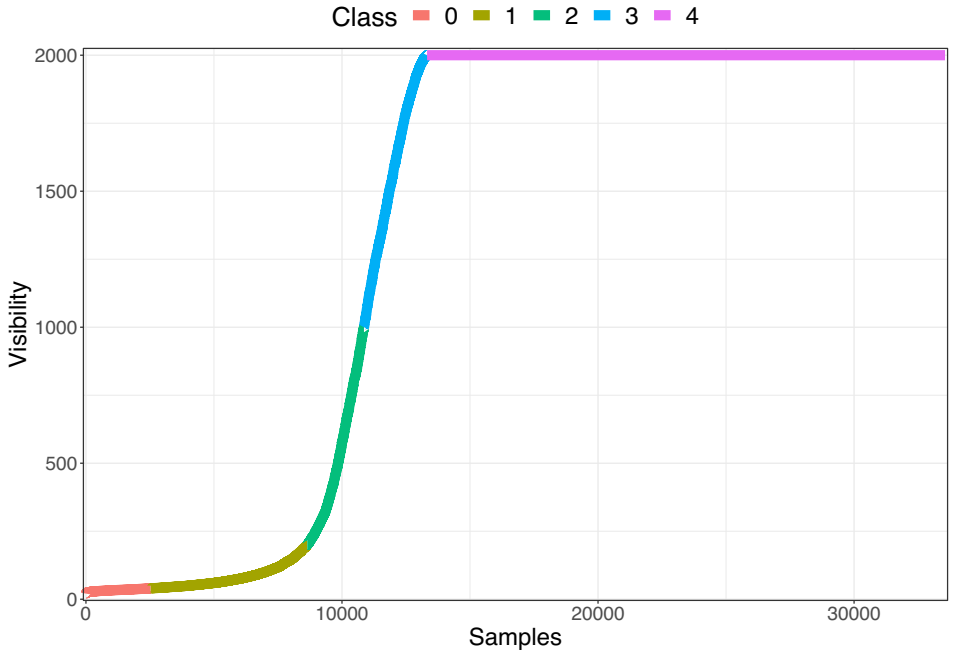


Fig. 2. Correspondence of the visibility measured with the sensor data acquisition in meters, to each of the five classes.

makes the problem particularly difficult. As it is explained later, this, among the other reasons, justifies the preference for non-linear models. To sum up, we are treating a regression problem also as a classification problem precisely to account for the frequency of the underlying phenomena and also to try to correct the inherent imbalanced nature of the data. We are also trying to use the knowledge provided by using classes to measure the degree in which the visibility is affected and take cautionary actions in foresight.

The reader might have noticed that the same threshold applied to the target variable could be applied to the output of a regression model, which could be used as a class prediction. We think that in training different methods with different methodologies, namely regression and classification, we can see which methods perform better as the training process is correcting for different errors in each case and one strategy may improve the final forecast.

Table 1 summarizes all the predictive variables (inputs) considered in this work. It is worthy of mention and clarification that, as mentioned earlier, we predict the value of visibility in the next half hour, so we are using $\boldsymbol{x}_t$ to predict y_{t+1}, meaning the we use the data we have now ($\boldsymbol{x}_t$) to predict the target in the next half hour (y_{t+1}). This required to shift the visibility value from the raw data, *vs*, with respect to the other input variables, $\boldsymbol{x}$ to get the target variable y. To improve the predictions we are using the visibility in instant t, vs_t also as an input variable to predict the visibility target (be it in the regression or classification problem).

Table 1
Predictive variables (inputs) used for the low-visibility event prediction at Mondoñedo, Galicia, Spain.

Variable	Abbreviation	Units
Accumulated precipitation	*qpRa*	mm/24 h
Air temperature	*at*	°C
Atmospheric pressure	*ap*	hPa
Dew temperature	*td*	°C
Floor temperature	*st*	°C
Global solar radiation	*gr*	W/m^2
Relative Humidity	*hr*	%
Salinity	*sa*	%
Visibility	*vs*	m
Wind direction	*wd*	Degrees
Wind speed	*ws*	km/h

The variable presented in Table 1 as *vs* is the input visibility from the current instant, (the one included in $\boldsymbol{x}_t$), although the distribution of both variables is the same except y is missing one value from the shift.

Table 2 summarizes the principal descriptive statistics of each predictive variable (see Table 1) for each of the considered classes.

In Fig. 3 a violin-plot of all predictive variables is shown. In this plot the distribution of each variable, disaggregated and colored differently for each visibility class, is shown. In other words, for each variable the samples with the same visibility class are taken into account jointly, and the distribution for the respective variable is calculated and plotted separately. Inside each of the "violins", a boxplot is also included so that it shows the median and Interquartile range (IQR). The dots outside the violin denote the outlier values. The shapes of the violin also show the modes of the distribution, so we can see that in some variables, e.g. *at* and *td*, the distribution for class 0 has two modes while all the other classes have unimodal distributions. We can also see that some variables have a very narrow interval of values regardless of the class considered, like *qpRa*, *sa*, *hr* (except for class 4) and *wd*, in which most of the distributions overlap. It can be seen that, in general, there are no clear differences in the distribution of any variable for each class, which means that in the data there are no clear indicators of what the visibility value will be in the next hour from the values of a single variable. Generally, the medians of each variable are pretty close and have a similar distribution regardless of the class. The variables that are more clearly distinct in their distribution for each class are *td*, *wd*, in which the medians do not overlap, but in any case there is still too much overlap in the rest of the distribution to confidently predict or to deduct any relation with the class based on a single variable.

Some of the reasoning applied in the last paragraph with violin-plots can also be seen in Fig. 4, which shows a correlation plot between all variables. In this case we can see that the input variables are in general not very correlated with each other, which provides important information when facing a prediction problem. First, as there are no cases of high correlation between variables, we can see that there are no redundant variables, and no variable should be deleted. Also, if we compare the distribution (Fig. 3) of variables with high correlation values, namely *at* with *st* and *td*, and *st* with *td* and *gr*, we can see that the overall shape of distributions is somewhat close and similar for each class, but still different enough so that each variable introduces different

Table 2
Summary of the most important distribution moments of each predictive variable per classes.

Class.	Metr.	qpRa	sa	vs	hr	at	st	td	gr	ws	wd	ap
	min	0.00	0.0	20.66	100.0	2.60	4.54	2.59	9.00	0.00	0.83	1001.0
	mean	0.09	0.0	36.38	100.0	13.82	16.30	13.81	42.08	4.98	239.18	1021.9
0	med.	0.00	0.0	34.16	100.0	14.40	16.37	14.39	10.00	5.03	323.33	1021.3
	max	6.90	0.0	987.00	100.0	19.76	30.57	19.75	594.00	9.96	359.33	1036.0
	std	0.39	0.0	30.81	0.0	2.74	3.53	2.74	59.95	1.51	133.00	3.8
	min	0.00	0.00	25.16	68.16	−4.73	−2.49	−4.74	8.83	0.00	0.66	984.0
	mean	0.74	0.31	170.14	99.97	10.28	12.88	10.26	47.64	4.99	190.26	1021.6
1	med	0.00	0.00	69.66	100.00	10.96	13.48	10.95	10.00	4.60	168.66	1022.0
	max	32.60	100.00	2000.00	100.00	22.13	41.66	22.11	731.66	19.01	358.83	1040.0
	std	2.52	3.97	307.49	0.53	4.40	5.92	4.40	72.09	2.65	125.97	6.6
	min	0.00	0.00	29.00	71.50	−4.44	−1.84	−4.51	9.00	0.00	0.50	981.1
	mean	2.29	0.56	808.50	99.79	8.72	11.05	8.68	59.53	6.17	185.05	1018.4
2	med.	0.25	0.00	495.83	100.00	8.43	9.68	8.41	10.00	5.33	166.00	1020.0
	max	46.06	100.00	2000.00	100.00	21.79	48.41	21.77	949.00	23.03	357.66	1037.0
	std	4.69	5.30	711.48	1.39	4.52	6.84	4.52	96.28	4.24	97.02	8.4
	min	0.00	0.00	38.00	39.33	−4.12	−1.73	−4.89	8.83	0.00	1.33	975.0
	mean	2.78	0.49	1355.65	99.33	8.64	11.20	8.52	78.67	6.22	185.15	1017.6
3	med.	0.30	0.00	1694.41	100.00	8.35	9.75	8.30	10.00	5.38	168.50	1019.0
	max	46.75	100.00	2000.00	100.00	25.99	50.50	22.78	915.50	22.01	358.00	1040.0
	std	5.29	5.08	710.96	2.77	4.76	7.61	4.75	125.25	4.20	95.39	9.3
	min	0.00	0.00	139.66	18.33	−4.32	−2.67	−11.98	8.83	0.00	0.66	975.0
	mean	2.32	0.73	1968.04	87.35	10.29	15.21	8.00	200.05	6.44	190.65	1018.7
4	med.	0.10	0.00	2000.00	92.33	9.76	12.25	7.56	56.33	5.46	174.16	1020.0
	max	49.51	100.00	2000.00	100.00	31.57	58.38	23.09	1222.50	27.73	358.83	1039.6
	std	5.31	7.59	173.02	14.88	5.23	11.04	4.87	257.89	4.38	90.39	9.2

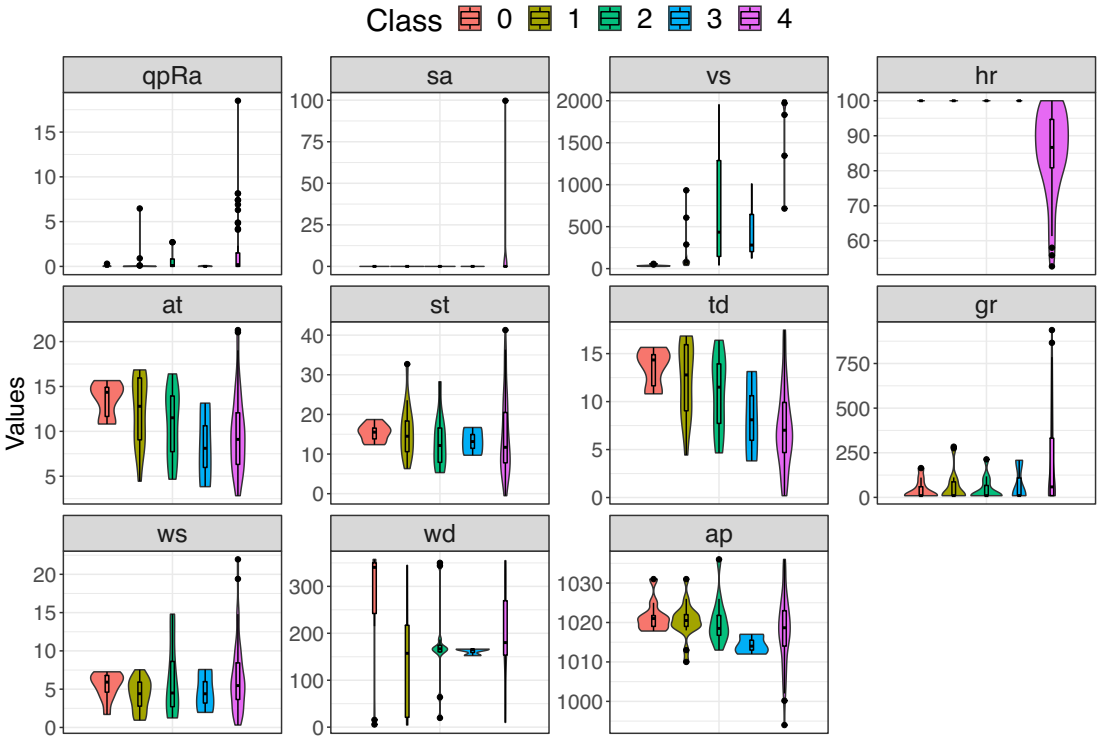


Fig. 3. Violin plot of all the predictive variables in the low-visibility prediction problem considered.

information into the problem. Second, we can see that there is no direct nor linear correlation between the chosen input variables and the current value of visibility (variable *vs*) which means that, most likely, linear prediction models will not suffice to obtain good enough predictions, and that non-linear prediction models should work better when working with this dataset.

As for correlations between pairs of variables, we can see that *vs* has a negative correlation with relative humidity (*hr*) (the higher *hr* is, the lower *vs* is in general, as expected, since in a fog event $hr = 100\%$), and positive with solar radiation (*gr*) (the higher is *gr*, the higher is the *vs*). Note, however, that individual correlation among variables is only an indicative parameter and must be taken with caution, since there are non-linear interactions among meteorological variables which trigger the fog events.

3. Supervised ML methods compared

In this work we carry out an exhaustive comparison among several state-of-the-art ML methods for classification and regression in a problem of low-visibility events prediction due to fog. We classify the

	qpRa	sa	vs	hr	at	st	td	gr	ws	wd	ap
qpRa	1.000	-0.011	0.153	0.100	-0.219	-0.213	-0.167	-0.075	0.187	0.088	-0.383
sa	-0.011	1.000	0.034	-0.033	-0.166	-0.126	-0.179	-0.039	-0.011	-0.018	-0.008
vs	0.153	0.034	1.000	-0.427	-0.064	0.078	-0.271	0.301	0.149	-0.053	-0.138
hr	0.100	-0.033	-0.427	1.000	-0.272	-0.331	0.255	-0.417	0.063	0.067	-0.105
at	-0.219	-0.166	-0.064	-0.272	1.000	0.820	0.858	0.385	-0.198	-0.021	0.215
st	-0.213	-0.126	0.078	-0.331	0.820	1.000	0.655	0.786	-0.243	0.014	0.198
td	-0.167	-0.179	-0.271	0.255	0.858	0.655	1.000	0.179	-0.161	0.014	0.158
gr	-0.075	-0.039	0.301	-0.417	0.385	0.786	0.179	1.000	-0.083	0.008	0.072
ws	0.187	-0.011	0.149	0.063	-0.198	-0.243	-0.161	-0.083	1.000	-0.051	-0.379
wd	0.088	-0.018	-0.053	0.067	-0.021	0.014	0.014	0.008	-0.051	1.000	-0.009
ap	-0.383	-0.008	-0.138	-0.105	0.215	0.198	0.158	0.072	-0.379	-0.009	1.000

Fig. 4. Correlation matrix of the predictive variables.

evaluated methods in four different categories: linear methods described in Section 3.3, ensemble methods, detailed in Section 3.1, Artificial Neural Network-based methods (ANN), described in Section 3.2 and regular ML methods, which have been presented in Section 3.4. Table 3 summarizes the evaluated ML methods for classification and regression problems using this categorization. Some of them are commonly used in both classification and regression problems, but others have been restricted to one of them (in this particular problem), as can be seen the third column of Table 3.

3.1. Ensemble methods

Ensemble methods train a set of several base ML models and assume that their collective predictions can surpass the accuracy of the individual ones, or even improve properties such as the robustness or generalizability. The ensemble methods follow a learning paradigm which is summarized with the expression "better together", and in many cases, they are able to overcome the predictions of other more complex ML methods, such as, the ANNs, which involve a huge number of parameters. The base learners which form the ensemble are called learners.

Table 3
Summary of the evaluated ML methods in both classification and regression tasks. We structured the considered ML methods in three different categories: ensemble methods (Section 3.1), ANN-based methods (Section 3.2) and regular ML methods (Section 3.4).

Category	Method	Type	Abbreviation
Ensemble	AdaBoost	Class / Reg	AB
	Gradient Boosting	Class / Reg	GB
	Random Forest	Class / Reg	RF
	Bagging	Class	Bagg
ANN-based	Multilayer Perceptron	Reg	MLP
	Extreme Learning Machine	Reg	ELM
Linear	Linear Regressor	Reg	LREG
	ElaticNet Regressor	Reg	EREG
	Generalized Linear Model	Class	GLM
	Support Vector Regression	Reg	SVR
	Support Vector Machine	Class	SVM
Other	Gaussian Process	Reg	GP
	Gaussian Naive Bayes	Class	GNB
	K Nearest Neighbours	Class	KNN
	Decision Tree	Class	DT

There exists a wide variety of ensemble algorithms, most of them are grouped into two categories: The *bagging* and *boosting* methods. One of the most extended ensemble methods used in ML for both classification and regression is Random Forest (RF) (Breiman, 2001). In this work, we evaluate several ensemble methods, specifically: Bagging (Bagg) of decision trees, Adaboost (AB), Gradient Boosting (GB) and Random Forest (RF).

3.1.1. Bagging

Bootstrap aggregating or simply *bagging* are the simpler ensemble technique to train multiple learners and provide an unified output. Bagging considers an ensemble composed by learners with equal architecture, that is, with same topology, number of input-output variables and parameters. The most common learners for creating the bagging ensemble method are decision trees, with same branches, parameters to train and input-output variables, as illustrated in Fig. 5.

Note that the ensemble model provides the output by applying a decision rule which combines the individual outputs of each model: averaging their outputs (regression) or by majority voting (classification) (Mohandes et al., 2018).

3.1.2. Random forest

Random Forest (RF) (Breiman, 2001) is arguably among the most renowned bagging-like techniques for classification and regression problems. The method specifically uses decision or regression trees as the learners, and then creates subsets from the bootstrap aggregating technique as the bagging method, but differs from the pure bagging technique in the topology of the trees changes among them. Trees of the ensemble (the forest) may have different length, topology or use different input variables which greatly increase the variability of the learners, but it is contrary to the bagging paradigm from a theoretical point of view.

3.1.3. Boosting

Boosting methods are a kind of ensemble algorithm which follows a special procedure for training their learners. They obtain excellent performance in both classification and regression problems (Ferreira and Figueiredo, 2012). All boosting methods establish the same structure for all the learners involved in the ensemble, that is, same architecture, number of parameters, or input-output variables. After creating

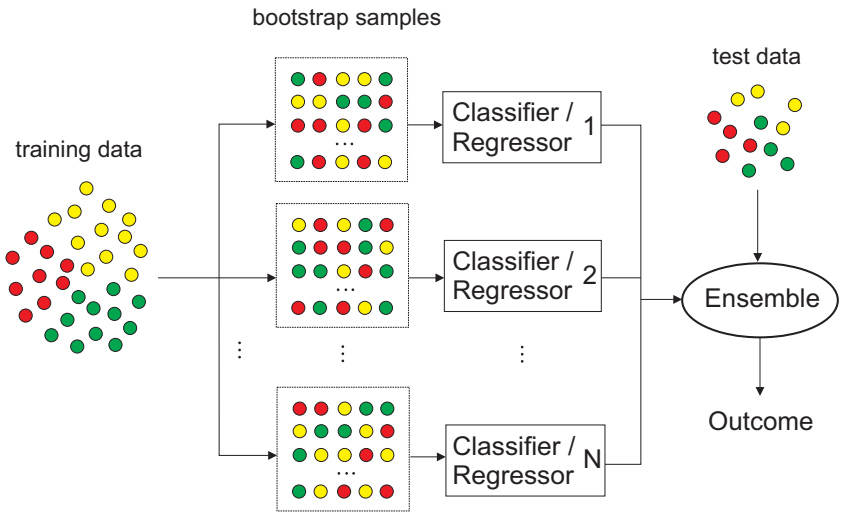


Fig. 5. Bagging technique for classification or regression problems.

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, see Fig. 6.

Adaptive Boosting (AdaBoost) is a kind of boosting method that proposes to train each of these machines iteratively, in such a way that each learner focuses on the data that was misclassified by its predecessor, to iteratively adapt its parameters and achieve better results (Ferreira and Figueiredo, 2012; González et al., 2020). Fig. 6 shows an outline of the Adaboost algorithm for multi-class classification.

Gradient Boost (Friedman, 2001) (GB) also combines a set of learners to create an stronger ensemble. Here, the residual of the a learner in the chain becomes the input for the next consecutive learner, and hence it is an additive model. The residuals are used in a step-by-step manner by the learners, in order to capture the maximum variance within the data.

3.2. ANN-based methods

We also briefly summarize here the description two ANN-based methods considered in this work: the Multi-Layer perceptron (MLP), and the Extreme Learning Machine (ELM).

3.2.1. Multi layer perceptron

A multi-layer perceptron (MLP) is a particular class of ANN which has been successfully applied to solve a large variety of non-linear classification and regression problems (Haykin and Network, 2004; Bishop et al., 1995). The multi-layer perceptron consists 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, see Fig. 7.

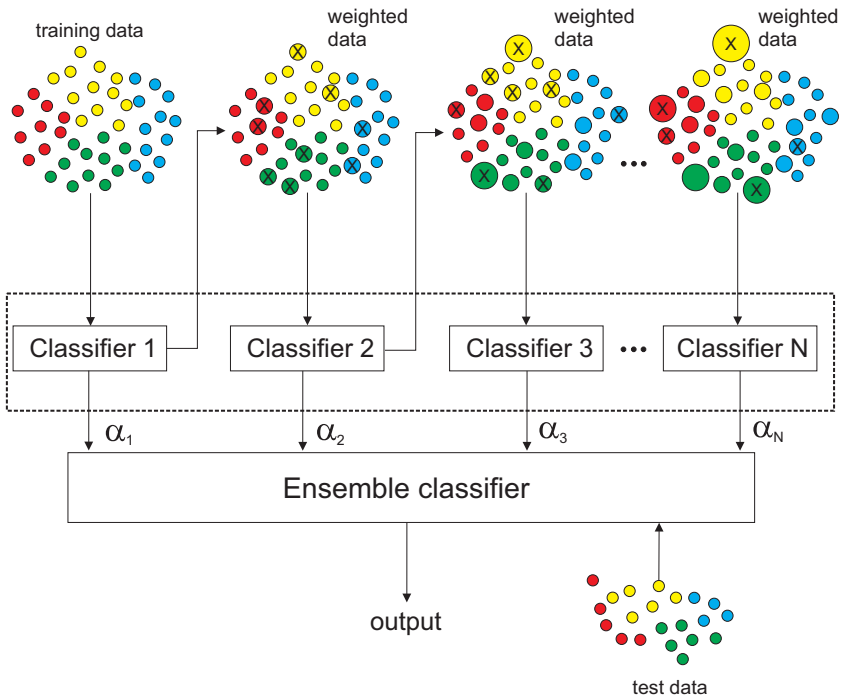


Fig. 6. Diagram of the AdaBoost algorithm exemplified for multi-class classification problems. Different size circles stand for samples with more associated weight (w) due to misclassification in the previous step (marked with X).

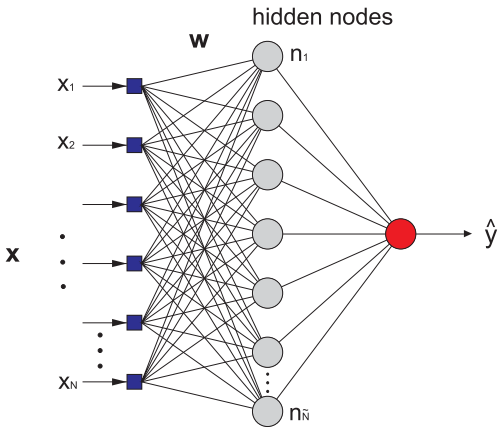


Fig. 7. Structure of a MLP, with one hidden layer.

It is the called feed-forward MLP. 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 labelled examples is known as the training process of the perceptron. Obtaining adequate values for the weights minimizes the error between the output given by the MLP and the corresponding expected output in the training set. The number of neurons in the hidden layer is also a parameter to be optimized (Haykin and Network, 2004; Bishop et al., 1995). The well-known Stochastic Gradient Descent (SGD) algorithm is often applied to train the MLP (Rumelhart et al., 1986). There are also alternative training algorithms for MLPs which have shown excellent performance in different problems such as the Levenberg-Marquardt algorithm (Hagan and Menhaj, 1994).

3.2.2. Extreme learning machines

An Extreme Learning Machine (ELM) (Huang et al., 2006) is a type of training method for MLPs (see Section 3.2.1), 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 (see (Huang et al., 2011) for details). The ELM obtains competitive results with respect to other classical training methods, while its training computation efficiency overcomes other classifiers or regression approaches such as SVM algorithms or MLPs (Huang et al., 2011).

3.3. Linear methods: Linear models, ElasticNet and the Generalized Linear Model

In this category we have included all linear models for both classification and regression problems, specifically we include Linear Regression (LREG), ElasticNet Regression (EREG) and Generalized Linear Model (GLM).

LREG (Freedman, 2009) is the simplest ML method for both regression and estimation tasks. The model assumes that there exists a linear relationship between the dependent variable y and the explanatory variables (the independent variables) $\mathbf{x} = (x_1, \cdots, x_n) \in \mathbb{R}^N$ which are expressed as follows:

$$y = \boldsymbol{\omega}^{\mathbf{T}}\bar{\mathbf{x}} + \varepsilon, \tag{2}$$

where $\boldsymbol{\omega} = (\omega_0, \ldots, \omega_n)$ are the coefficient vector of the linear regression and $\bar{\mathbf{x}} = (1, \mathbf{x})$ represents the extender predictive variable vector. The variable ε is a random variable with zero mean that measures how far the linear model is from the real function f.

Linear models are attractive since the relationship between coefficients and feature importance is direct, and it is more interpretable. The linear regressor computes its coefficients minimizing the mean squared error of the residuals:

$$R = \frac{1}{N}\sum_{i=1}^{N} r_i = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}_i\right)^2 = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \boldsymbol{\omega}^{\mathbf{T}}\bar{\mathbf{x}}_{\mathbf{i}}\right)^2. \tag{3}$$

Minimizing this error directly drives to a the following closed-form solution:

$$\boldsymbol{\omega} = \left(X^T X\right)^{-1} X^T Y, \tag{4}$$

where $X^T = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$ is a matrix of the observations (the predictive variables of the observations) and $Y = (y_i)$ are the corresponding response variable of a training set $\mathscr{D} = \left\{\left(\mathbf{x}_i, y_i\right) \middle| 1 \le i \le N\right\}$.

Linear regression has some important disadvantages. One of the most straightforward is that the model does not consider the possible correlation of the predictive variables $\{x_i\}$. Also, the mean squared error of the residuals is strongly sensitive to outliers. For solving both issues, instead of minimizing the mean squared of the residuals, the ElasticNet model (EREG) (Zou and Hastie, 2005) minimizes a function which involves both L1 and L2 regularization as follows:

$$\mathscr{L} = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \boldsymbol{\omega}^{\mathbf{T}}\bar{\mathbf{x}}_{\mathbf{i}}\right)^2 + \alpha \parallel \omega\parallel_2^2 + \beta \parallel \omega\parallel_1, \tag{5}$$

where α and β are hyperparameters that balance the amount in which both norms are involved in the function, and $\|\cdot\|_2$ and $\|\cdot\|_1$ refers to the norms:

$$\begin{aligned} \parallel \mathbf{x}\parallel_2 &= \sqrt{x_1^2 + \cdots + x_n^2}. \\ \parallel \mathbf{x}\parallel_1 &= |x_1| + \cdots + |x_n| \end{aligned} \tag{6}$$

The hyperparameters α and β must be fitted from a validation dataset. In this work, a grid search with a validation partition has been used in order to discover the best values for both hyperparameters. Observe that if α and β are zero, then an ElasticNet becomes a plain linear regressor, see Eq. (4). If the parameter $\beta = 0$, then the resultant regressor is a Ridge regressor (Hoerl et al., 1975), which removes correlated input variables. If the parameter $\alpha = 0$, then we have a LASSO regressor (Tibshirani, 1996), which is less sensitive to possible outliers. ElasticNet generalizes all these regressors in one by introducing these two regularization terms in the problem.

A GLM is a generalization of ordinary linear regression. This generalized linear model includes a link function which describes the distribution of the response variable. So if we had a binary classification problem, a Binomial distribution would be needed as the link function. The link functions are chosen from distributions belonging to the Exponential family of distributions. This allows to, as the name implies, use a linear model with the same parameterization as a linear regression to be used with all kinds of target values, generalizing the concept of linear regression further than a regression problem where the predicted variable $y \in \mathbb{R}$ and $\sim \mathscr{N}\left(\mu, \sigma^2\right)$.

They were proposed in the work (Nelder and Wedderburn, 1972) to unify some other statistical ML models, including Poisson, logistic or linear regression in one single model. The GLM has the following expression:

$$y = g^{-1}(\mu) = g^{-1}\left(\omega^T x\right) \tag{7}$$

The GLM includes three main elements:

1. The response variable y is assumed to be a random variable that follows one of the probability distribution of the exponential family, which include Poisson, gamma, binomial and normal distributions. This property generalizes the linear regression model that assumes that the response variable is a Gaussian variable.

2. The domain of the link function $g^{-1}(\cdot)$ remains to be a linear predictor $\mu = \omega^T x$, which eases the computation of the coefficients.
3. The link function $g^{-1}(\cdot)$ provides the relationship between the linear predictor $\mu = \omega^T x$ and the mean of the distribution function $E[y|x] = g^{-1}(\mu)$.

In this work, the GLM used for fog events classification tasks uses a multinomial link function. This enables the model to work in multiclass classification problems, namely, a problem where there are more than two classes.

3.4. Statistical-based and other regression and classification methods

In this category we have included all the evaluated ML methods for regression and classification tasks related to fog event prediction, which are not strictly ensemble, ANN-based or linear methods. We include statistical-based approaches such as Gaussian Process (GP) and the Gaussian Naïve Bayes (GNB), and other classical algorithms for classification and regression such as Support Vector Machine (SVM) and Support Vector Regression (SVR), K-nearest neighbours algorithm (KNN) and Decision Tree (DT).

3.4.1. Support vector machines

The Support Vector Machine (SVM) (Schölkopf et al., 2002; Schölkopf et al., 2000) is a learning algorithm for classification problems defined as follows: Notationally, given a labelled training data set $\{\mathbf{x}_i, y_i\}_{i=1}^n$, where $\mathbf{x}_i \in \mathbb{R}^N$ and $y_i \in \{-1, +1\}$, and given a nonlinear mapping $\phi(\cdot)$, the SVM method solves the following problem:

$$\min_{\mathbf{w},\xi_i,b}\left\{\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i\right\} \tag{8}$$

constrained to:

$$y_i(\langle \boldsymbol{\phi}(\mathbf{X}_i), \mathbf{w}\rangle + b) \geq 1 - \xi_i \forall i = 1, \ldots, n \tag{9}$$

$$\xi_i \geq 0 \forall i = 1, \ldots, n \tag{10}$$

where $\mathbf{w}$ and b define a linear classifier in the feature space, and ξ_i are positive slack variables enabling to deal with permitted errors (Fig. 8). 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. Details on the solution process for the SVM algorithm and its tuning and optimization can be found in (Schölkopf et al., 2002).

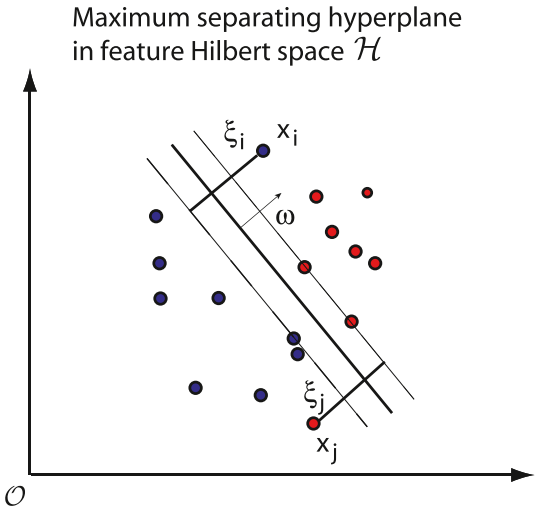


Fig. 8. Illustration of the SVM process: Linear decision hyperplanes in a nonlinearly transformed, feature space, where *slack* variables ξ_i are included to deal with errors.

3.4.2. Support vector regression

The Support Vector Regression (SVR) (Smola and Schölkopf, 2004) is a well-established algorithm for regression and function approximation problems. The SVR takes into account an error approximation to the data, as well as the capability to improve the prediction of the model when a new dataset is evaluated. Although there are several versions of the SVR algorithm, we use the classical model (ε-SVR) described in detail in (Smola and Schölkopf, 2004), which has been used for a large number of problems and applications in science and engineering (Salcedo-Sanz et al., 2014). Fig. 9 shows an example of the process of a SVR for a two-dimensional regression problem, with an ε-insensitive loss function. Details on the solution process for the SVR algorithm and its tuning and optimization can be found in (Smola and Schölkopf, 2004).

3.4.3. Gaussian process

Gaussian Processes (GP) (Rasmussen, 2003) are non-parametric kernel-based probabilistic ML models for both regression and classification problems. The Gaussian process models a continuous random process $f(x)$ where the variable x is dense. Similar to the linear regression, which estimates the output y from the input variables x trough a linear relation $y = \omega^T x + \varepsilon$ where $\varepsilon \sim \mathcal{N}(\mu, \sigma^2)$, GP models predict the response y by introducing a set of latent variables $\{f(x_i)\}_{i=1}^n$ from a Gaussian process, and explicit basis functions, $h(\cdot)$. Details on the solution process for the GP algorithm and its tuning and optimization can be found in (Rasmussen, 2003).

3.4.4. K-Nearest neighbours

K-nearest neighbour (KNN) (Shakhnarovich et al., 2008) is a non-parametric ML method which looks for a set of K observations of the training set which are the closest to the new test observation. The term "closest" to an observation x_i is measured with respect to a metric or distance $d(\cdot)$ which fulfils:

- $d(x_i, x_j) \geq 0 \quad \forall\, i, j$ and $d(x_i, x_j) = 0 \Leftarrow x_i = x_j$
- $d(x_i, x_j) = d(x_j, x_i)$
- $d(x_i, x_j) \leq d(x_i, x_k) + d(x_k, x_j)$

The most frequent distance used in the KNN is the Euclidean distance.

3.4.5. Decision trees

Decision Trees (DT) (Rokach and Maimon, 2005), although also a common tool used for decision making, are a kind of ML method which builds a tree that follows branching decision paths to reach a classification. A graphical representation of a decision tree can be seen in Fig. 10. In ML Decision Trees, the training data is used to calculate the thresholds that better split the data according to a partition criterion. This criterion searches for the best gain of information possible in the current node, meaning, what is the optimum split on which variable so that it allows for a better prediction of the target.

3.4.6. Gaussian Naive Bayes

Naive Bayes methods (Zhang, 2004) are supervised learning methods based on the Bayes' theorem. These methods assume the "naive" 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 trough x_n:

$$P(y|x_1, \ldots, x_n) = \frac{P(x_1, \ldots, x_n|y)P(y)}{P(x_1, \ldots, x_n)} \tag{11}$$

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

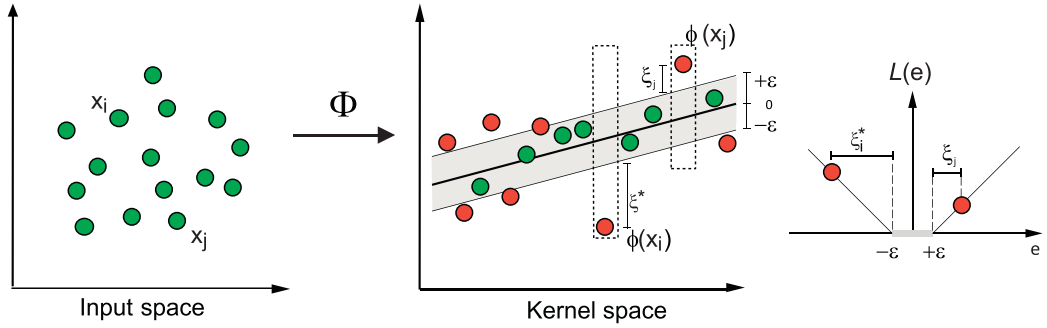


Fig. 9. Example of a Support-Vector-Regression process for a two-dimensional-regression problem, with an ε-insensitive loss function.

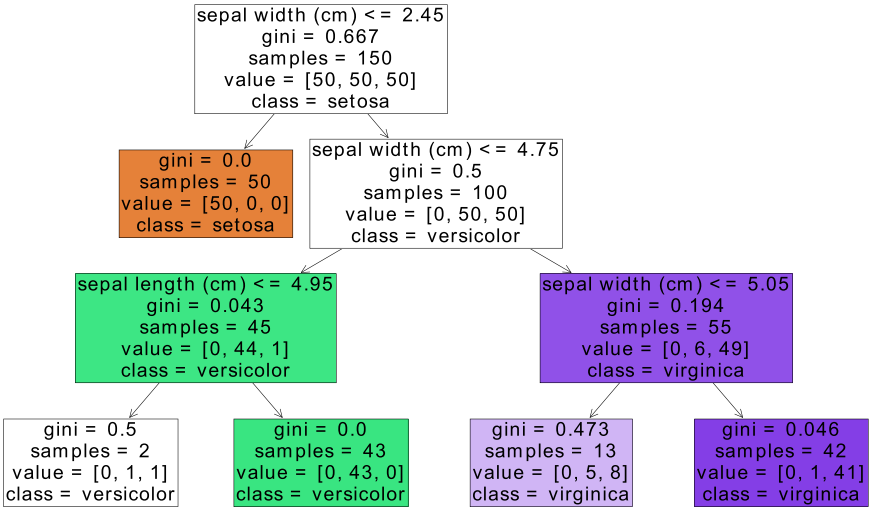


Fig. 10. Graphic representation of a decision tree trained on the Iris dataset with a maximum depth of three levels.

$$P(y|x_1,\ldots,x_n) = \frac{P(y)\Pi_{i=1}^{n} P(x_i|y)}{P(x_1,\ldots,x_n)}. \tag{12}$$

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

$$\hat{y} = \text{argmax}_y P(y|x_1,\ldots,x_n) = \text{argmax}_y P(y)\Pi_{i=1}^{n} P(x_i|y) \tag{13}$$

Gaussian Naive Bayes (GNB) simply assumes that each $x_i \mid y$ is a

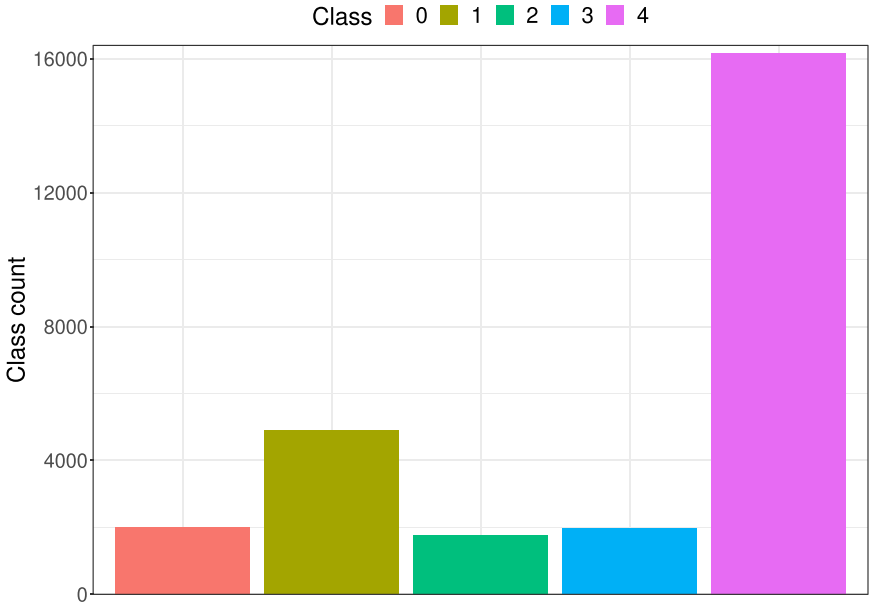


Fig. 11. Original dataset.

normal random variable:

$$P(x_i|y) = \frac{1}{\sqrt{2\pi}\sigma_y} e^{-\frac{\left(x_i-\mu_y\right)^2}{2\sigma_y^2}} \quad (14)$$

3.5. Balancing techniques

The fog event database considered in this work is highly imbalanced, meaning there are a huge amount of high visibility events compared to the amount of low visibility events, as seen Fig. 11. For class 4, the one with highest visibility, we have almost 16000 samples, for class 1 we have about 5000 samples and for each of the other classes we usually have about 2000. As seen previously in Section 2 with Fig. 2 this imbalance is in the underlying nature of the phenomena.

It is well known that this leads to many problems when dealing with classification tasks. There are a number of different ways of approaching imbalance in classification problems (López et al., 2013), and all of them have some associated problems. The three most common ways to treat imbalanced problems are:

1. *Cost-sensitive classification*: the error for the minority classes is weighted higher than the error for the majority class to make learning these classes a priority during training. With this method, finding the right weighting is difficult and is in the end an heuristic process.
2. *Algorithmic modification*: modifying the algorithm so that it takes into account different sets of the data. The most commonly used and well known are ensemble methods, where many weak learners, such as decision trees, are combined, and each is trained with different samplings of the data. With this method there is no explicit enforcing to consider the under-represented classes as such, but the weak learners are trained with different samplings that are likely to contain more samples of minority classes than of the majority class for some of the underlying weak models used.
3. *Resampling*: resampling the different classes to achieve a better balance. This can mean generating new samples, oversampling, or deleting existing samples, undersampling. Oversampling generates new non-existing samples close to existing ones. When dealing with multidimensionality, generated samples could be created in problematic regions where the decision boundary is not clear which may lead to introducing misinformation in the data, but in general new samples are generated in regions where they would be likely to exist. As for undersampling, when removing samples is not done carefully important and representative data may be removed.

In this paper, in order to treat the imbalance problem, we have combined two approaches: we have used mostly ensemble methods as classifiers and regressors (detailed in Section 3.1), but as the differences among classes are so vast we have considered that in this case it would be also be important to balance the number of samples we have per class, so we have used techniques based on sample generation and reduction. Specifically we use three balancing strategies on the original data:

1. *Oversampling* the minority classes
2. *Undersampling* the majority class
3. *Hybrid method*: undersampling the majority class and oversampling the minority classes

Table 4 summarizes the balancing techniques employed including their abbreviations as used from here on, and their original publications. These methods will be explained next.

The oversampling procedure consists of increasing the number of observations of all the under-represented classes by generating new data samples. When performing oversampling it is important that the statistics of each class remain the same. The algorithm must create new samples that remain in the same "class region" inside the problem's feature space and do not cross the decision boundaries into other classes as this would introduce misinformation that would taint the data. In this work, we use the most common oversampling technique, the SMOTE (SMT) algorithm (Chawla et al., 2002), which creates new samples taking into account the existing ones, diminishing the risk of creating samples in "wrong" areas.

Table 4
Summary of the balancing techniques used.

Over./Under.	Balancing Technique	Abbreviation	Work
Oversampling	SMOTE	SMT	(Chawla et al., 2002)
Undersampling	Condensed Nearest Neighbours	CoNN	(Hart, 1968)
	Neighbourhood Cleaning Rule	NCR	(Laurikkala, 2001)
	Tomek Links	TL	(Tomek et al., 1976)
	Random UnderSampler	RUS	–

On the contrary, undersampling methods decrease the number of observations of the majority classes by removing samples. As it happens with oversampling, undersampling has to reduce the number of samples while also maintaining the statistical properties of the classes, so it should remove, when possible, redundant information. In this work, we consider four different undersampling techniques:

1. *Condensed Nearest Neighbours (CoNN)*: This algorithm uses a 1-NN rule to iteratively decide if a sample of the majority class should be removed or not based on whether or not it is missclasified. This method is mostly focused on removing samples.
2. *Neighbourhood Cleaning Rule (NCR)*: The data is undersampled using the neighbourhood cleaning rule: all the minority samples are kept, and using instance based methods such as k-NN the samples from the majority class that are ambiguous are identified and removed. NCR places less emphasis on removing redundant examples than cleaning the data and keeping the neighbourhood of the samples "clean", so this results on the final number of removed samples not being as big as with CoNN.
3. *Tomek Links (TL)*: This method finds in the data the so called tomek's links, samples where the distance to samples of another class is less than to samples of its own class. The sample that belongs to the majority class is removed. This method performs a cleaning of the boundary between classes.
4. *Random UnderSampler (RUS)*: As the name implies, samples are deleted randomly without attending to their characteristics, positioning or values. It is mostly used as a baseline method to compare how the others perform. It's main benefit is that it is very fast compared to the previous methods which are very computationally expensive.

As mentioned earlier, in this work we combine both undersampling and oversampling techniques in order to evaluate what combination works best for the evaluated classification methods. The results of applying these resampling methods to the dataset are shown in Fig. 12: on the top row the undersampling methods are included, in the bottom, the oversampling with SMOTE on the previously undersampled datasets. So in Fig. 12e SMOTE is shown, and then in Fig. 12f, g and h the hybrid methods that include both oversampling and undersampling. This way, the columns have the same undersampling method and the differences can be easily compared. The colors and the axis are the same as in Fig. 11, so the classes are incremental from left to right.

As it can be seen in Fig. 12a, CoNN is an aggressive undersampling procedure, and it removes almost all of the instances of the majority class, which surprisingly leaves class 4 to be the minority class. This is the reason why on the lower row there is no oversampling applied to this

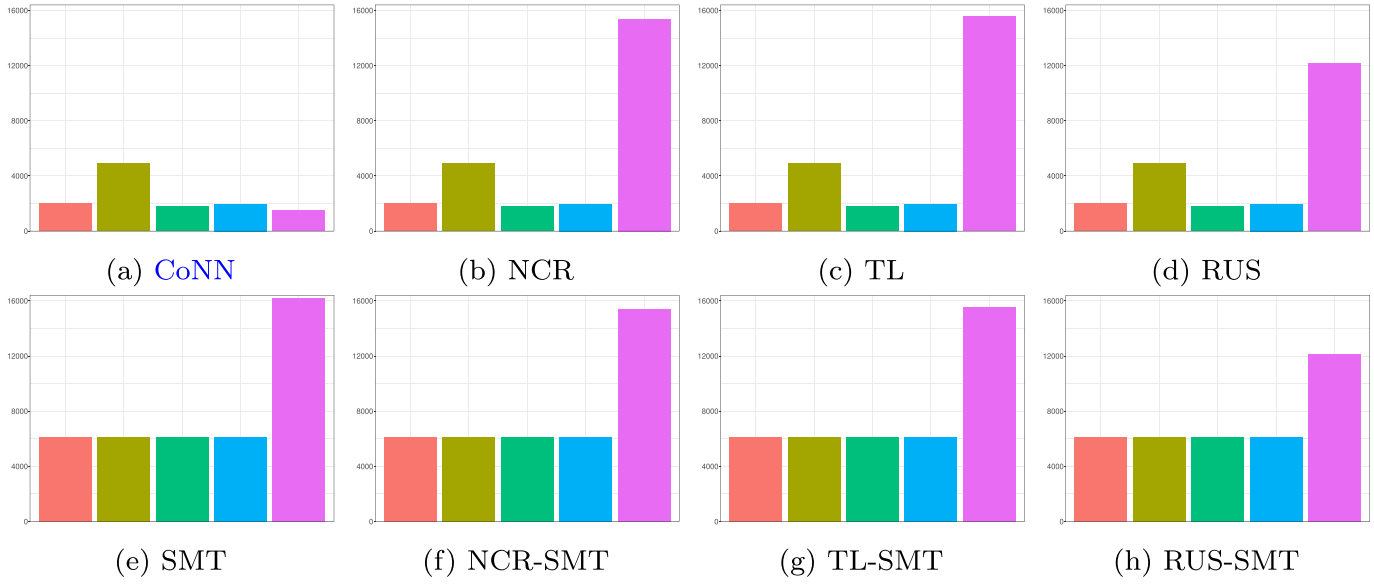


Fig. 12. Different evaluated balancing techniques over the original dataset.

method, just using CoNN has balanced the dataset. As seen in Fig. 12b and c, the amount of undersampling achieved by NCR and TL is not too different, although as there is a difference in the methods, the selected data is also different; the difference is not quantitative as much as qualitative. Lastly, RUS, in Fig. 12d, has removed a higher number of instances than the previous methods although we are not as certain of the quality of the resulting data.

As for oversampling methods, we have decided to equalize the number of instances for all minority classes, as seen in Fig. 12e. The remaining Fig. 12f, g and 12 h show the results of applying SMT to their respective undersampling methods which are shown in the upper row, namely Fig. 12b, c and d.

All this resulting datasets have been used in the classification problem and are discussed in Section 4.2.2.

3.6. Data normalization

Predictions obtained by the different ML classification and regression techniques considered are affected by the type of normalization applied over the predictive variables. In this work, we evaluate three different types of normalization, described below:

1. **Min-Max Normalization:** Min-Max normalization maps the whole range of each input variable into the interval $[0,1]$ using the following formula

$$\widetilde{x}_j = \frac{x_j - \min\limits_i x_i}{\max\limits_i x_i - \min\limits_i x_i}, \quad \forall j \tag{15}$$

2. **Standardization**: The standardization treats each input variable as a random variable. The standardization process consists of removing the mean of each variable and then normalize the result by its standardization as follows:

$$\widetilde{x}_j = \frac{x_j - \frac{1}{n}\sum_1^N x_i}{\left(\frac{1}{n}\sum_{l=1}^N \left(x_i - \frac{1}{n}\sum_{m=1}^N x_m\right)^2\right)^{\frac{1}{2}}}, \quad \forall j \tag{16}$$

Observe that the standardization technique does not necessary map the range of each variable to the interval $[-1,1]$. Only the most probable observations, i.e. those which are close to the mean, are mapped to this interval.

3. **Robust Standardization:** The robust standardization works with the median and the deviations from the median to compress

$$\widetilde{x}_j = \frac{x_j - \text{med}_i x_i}{\text{med}_j \left|x_j - \text{med}_i x_i\right|}, \quad \forall j \tag{17}$$

It is less sensitive to outliers than the standard deviation. As the standardization technique, it does not necessary map the range of each variable to the $[-1,1]$ interval.

In this work, we also use the PCA of the input data matrix X. PCA directly standardizes the data before obtaining the principal components, so we cannot use other types of normalization in this case.

4. Experiments and results

In this section we carry out an exhaustive comparison between the different ML algorithms considered (Section 3) in the low-visibility event prediction problem described in Section 2. We first describe the metrics used to evaluate the regression and classification methods compared, see Section 4.1. Then the results obtained are reported: we first discuss the results of the ML regression techniques in Section 4.2.1 and then the ML classification methods are presented and evaluated in Section 4.2.2.

4.1. Evaluation metrics

For measuring the performance of the regression methods we consider three common metrics: Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) and coefficient of determination (R^2), which are briefly described here:

$$\begin{aligned} RMSE &= \sqrt{\frac{1}{N}\sum_{i=1}^{n}(y_i - \widehat{y}_i)^2} \\ MAE &= \frac{1}{N}\sum_{i=1}^{n}|y_i - \widehat{y}_i| \\ R^2 &= 1 - \frac{\sum_i (y_i - \widehat{y}_i)^2}{\sum_i (y_i - \overline{y})^2} \end{aligned}, \tag{18}$$

where y is the ground truth, $\bar{y}$ is the average value of y, and $\hat{y}$ represents the prediction of the regressor. The subscript i is used to refer to a single sample $y_i = y[i]$.

The formulation of the coefficient of determination, R^2, makes an assumption that the predictions come from a linear model (Colin Cameron and Windmeijer, 1997), so it should be used only for linear regressors. When using this statistic with non-linear regressors, problematic behaviour may arise such as values greater than 1 or lower than 0 as this metric is defined as $R^2 \in [0, 1]$. Even if this is the case, we have left it for all regressors as another performance comparison metric.

Regarding the classification task, we use four different metrics: Accuracy (ACC), F1-score (F1), Recall (REC) and Quadratic Weighted Kappa (QWK), which have the following expressions:

$$\begin{aligned} ACC &= \frac{TP + TN}{P + N} \\ F1 &= \frac{2TP}{2TP + FP + FN} \\ Recall &= \frac{TP}{TP + FN} \\ QWK &= 1 - \frac{\sum_{i,j} w_{i,j} O_{i,j}}{\sum_{i,j} w_{i,j} E_{i,j}} \end{aligned} \tag{19}$$

where TP and TN are the true positives and the true negatives achieved by the method, respectively; and P and N are the total amount of positive and negative test samples.

Note that most of the metrics above are defined for binary classification, however, they can be used for multi-class classification by adopting a "One vs Rest" or "One vs All" scheme to transform a multi-class problem into multiple binary problems. With this scheme we iteratively take one class, which we consider the positive class, and consider the rest of the classes as the negative class. We then average all the binary problems and have an equivalent to multi-class.

With regards to the F1-score, we use a weighted average to account for the imbalance present in the classes. This average is computed with the support of each class and serves to equally represent all classes in the final score even if some have a less samples. So classes with less samples are weighted with a higher value than classes with more samples. With this weighting we can be assured that we are not getting a very high score just because the classifier is predicting all the samples from the majority class and omitting the minority classes, which would give us a trivial classifier that could not predict the event of interest. For this reasons this metric has been chosen as the main comparison metric for classification in Section 4.2.2. We also use a weighted recall by support when comparing classification results, following the same reasoning. In this case, using recall directly gives us a numeric representation of the number of positive (or relevant) samples classified out of all the positive samples; we see how many samples of class i were correctly predicted as belonging to class i.

Regarding QWK, O is the $N \times N$ confusion matrix, E is the expected matrix and w is the weight matrix. Each coefficient of the confusion matrix O_{ij} represents the number of predictions of the class i when the true class was j. The expectation matrix is computed as:

$$E = \mathbf{h}_i \mathbf{h}_p^T, \tag{20}$$

where $\mathbf{h}_p$ and $\mathbf{h}_t$ are the histograms of the prediction and the ground truth, respectively, and the super-index T refers to the transpose. Finally, the weight matrix with quadratic weights is computed as follows:

$$w_{i,j} = \frac{(i-j)^2}{(N-1)^2}, \tag{21}$$

where N is the number of classes.

The closer to 1 these metrics are, the better the performance of the analyzed methods. ACC, REC and F1 score are all nominal metrics, that is, they do not consider an ordering of the classes. On the other hand, QWK considers the ordering of the classes by penalizing errors with the quadratic of the distance between classes.

4.2. Results

The results obtained are presented and discussed in this section, which is structured into two subsections: we first discuss the results of the regression methods in Section 4.2.1, and then the results of the ML classification techniques in Section 4.2.2.

4.2.1. ML regression results

Tables 5, 6, 7 and 8 detail the performance of the ensemble, the ANN-based, linear and statistical-based (named as others in Section 3.4) regression methods respectively, including all variations of the input variable normalization/standarizations (see Section 3.6), and if we use PCA or not to reduce the number of predictive variables. There could be large differences in performance of the ML regressors depending on the normalization considered, including (or not) PCA, and, of course, among the different techniques.

It is well known that some methods require previous normalization of the input data, and also that some of the methods are more sensitive to the changes in the normalization method than others. As it can be seen in Table 5 and in Table 6, changing the data normalization can greatly alter the results obtained for different algorithms; it can produce an excellent or a bad performance, as is the case when using Min-Max normalization in RF and in MLP_{big}. RF seems to be less affected by normalization of input data than MLP-based approaches. SVR and Gaussian Processes, Table 8, seem to have a poor performance in this problem, worse than the other regression techniques evaluated in this work.

Table 5 specifically details the performance of the evaluated ensemble methods: RF, GB and ADB_{MLP}. For RF and GB, we have used the hyperparameters that appear on the original study, which is cited in the description of every problem (implemented in sklearn (Pedregosa et al., 2011)), but for computing time and resources we have used only 10 MLP estimators for ADB_{MLP}. Out of all the algorithms considered, RF obtains the best RMSE, MAE and R^2 for all the evaluated normalizations without considering the one using PCA. The best RF result in terms of RMSE, MAE and R^2 is obtained with the Min-Max normalization with values of 340.55, 175.36 and 0.8288 respectively. We do not perceive significant differences among the evaluated input normalizations in RF, but it seems that PCA negatively affects its performance; after applying PCA, RF obtains the worst performance among all the evaluated ensemble methods. The reason behind this is that RF benefits from the different regression trees structure. Since PCA reduces the number of

Table 5
Evaluated ensemble methods for regression. We classify the methods according to the kind of scaling we apply to the inputs and if they use the original data or PCA decomposition. We sort the methods in ascending order of their RMSE.

Model	Scaling	PCA	RMSE	MAE	R^2
RF	MinMax	No	**340.55**	**175.36**	**0.8288**
RF	Standard	No	341.59	176.28	0.8278
RF	Robust	No	341.59	175.53	0.8278
GB	Standard	Yes	344.71	187.32	0.8246
GB	Robust	No	344.71	187.33	0.8246
GB	Robust	No	344.73	187.33	0.8246
GB	Standard	No	344.78	187.52	0.8246
ADB_{MLP}	Standard	Yes	383.09	265.91	0.7834
ADB_{MLP}	Robust	No	400.27	286.93	0.7636
ADB_{MLP}	Standard	No	403.54	298.73	0.7597
ADB_{MLP}	MinMax	No	419.81	325.78	0.7399
RF	Standard	Yes	737.64	537.06	0.1972

Table 6
Evaluated ANN-based methods for regression. We classify the methods according to the kind of scaling we apply to the inputs and if they use the original data or PCA decomposition. We sort the methods in ascending order of their RMSE.

Model	Scaling	PCA	RMSE	MAE	R^2
MLP_{big}	Standard	No	**327.55**	**164.30**	**0.8416**
MLP	Standard	No	340.64	179.11	0.8288
MLP_{big}	Robust	No	347.96	183.36	0.8213
MLP	Robust	No	348.68	187.09	0.8206
MLP	MinMax	No	350.51	188.52	0.8187
MLP_{simp}	Standard	Yes	359.32	210.89	0.8095
ELM_{GS}	MinMax	No	360.68	215.22	0.8080
MLP_{simp}	Standard	No	371.83	211.98	0.7960
ELM_{GS}	Standard	No	375.79	217.05	0.7916
ELM_{GS}	Robust	No	380.69	221.51	0.7861
MLP_{simp}	Robust	No	381.23	227.44	0.7855
MLP_{big}	Standard	Yes	389.35	242.63	0.7763
MLP_{simp}	MinMax	No	395.73	246.03	0.7689
MLP	Standard	Yes	409.30	260.97	0.7528
MLP_{big}	MinMax	No	824.82	759.17	−0.0037

Table 7
Evaluated linear methods. We classify the methods according to the kind of scaling we apply to the inputs and if they use the original data or PCA decomposition. We sort the methods in ascending order of their RMSE.

Model	Scaling	PCA	RMSE	MAE	R^2
LREG	Standard	Yes	**389.51**	**240.95**	**0.7761**
LREG	MinMax	No	389.51	240.95	0.7761
LREG	Standard	No	389.51	240.95	0.7761
LREG	Robust	No	389.51	240.95	0.7761
EREG	Robust	No	389.52	240.94	0.7761
EREG	MinMax	No	389.52	240.94	0.7761
EREG	Standard	No	389.61	240.93	0.7760
EREG	Standard	Yes	405.61	247.10	0.7572

Table 8
Evaluated statistical-based regression methods. We classify the methods according to the type of scaling we apply to the inputs and if they use the original data or PCA decomposition. We sort the methods in ascending order of their RMSE.

Model	Scaling	PCA	RMSE	MAE	R^2
SVR	MinMax	No	**407.97**	**212.27**	**0.7544**
SVR_{simp}	MinMax	No	407.98	212.26	0.7544
SVR_{simp}	Standard	Yes	426.58	188.05	0.7315
SVR	Standard	Yes	434.47	189.11	0.7214
SVR	Standard	No	448.78	331.49	0.7028
SVR_{simp}	Standard	No	477.95	322.77	0.6629
GP	MinMax	No	484.77	258.40	0.6532
SVR	Robust	No	690.34	560.44	0.2968
SVR_{simp}	Robust	No	879.17	559.62	−0.1404
GP	Standard	No	974.58	626.15	−0.4013
GP	Robust	No	1015.74	677.53	−0.5221
GP	Standard	Yes	1624.30	1400.51	−2.8925

variables in the input data, and in turn the tree length of RF, the variability of the ensemble is also reduced, and its performance worsens in consequence. GB obtains similar results to RF for all evaluated normalizations with or without PCA, with a RMSE, MAE and R^2 of around 344.7, 187.3 and 0.8246, respectively. The GB algorithm computes the residuals of one tree to feed the following tree. There is no variability on the topology of the trees in the ensemble, and therefore it remains robust even if we apply PCA. ADB_{MLP} obtains by far worse results than the previous ensembles for all the normalizations, with RMSE, MAE and R^2 values of 383.09, 265.91 and 0.7834 respectively (the best case uses standard scaling with PCA). We observe an improvement using PCA since it may avoid a possible overfitting in the MLP.

Table 6 focuses on the performance of the evaluated ANN-based methods: MLP and ELM. We evaluate three different MLPs according to the number of neurons and the number of hidden layers. We refer to the biggest MLP as MLP_{big}, to the medium as MLP, and to the one with less parameters as MLP_{simp}. Both MLP_{big} and MLP have been optimized varying the number of hidden layers from 1 to 3 and the number of neurons per layers from 50 to 150, and as the number of layers of the network increased the number of neurons per layer decreased (remaining MLP_{big} more dense than MLP). Besides the architecture, we also use both ADAM and SGD in order to determine what is the best training algorithm, and also change the activation function, using tanh and logistic. To better train the network we have also used an adaptive learning rate, early stopping and various initial learning rate values (1×10^{-4},7.5×10^{-4}, 5×10^{-3}). MLP_{simp} represents the sklearn library implementation (Pedregosa et al., 2011) but changing the training parameters: training, we used an adaptive learning rate and early stopping. The number of neurons in the ELM hidden layer has been set by using a grid search. We indicate this with the subscript *GS*. Note that the MLP obtains excellent results in this problem. The best out of this ANN-based category is also the best-over-all for all the evaluated regression methods. As it can be seen, the MLP is very sensitive to the different normalizations applied. The best results have been obtained with the MLP_{big} and the standard scaling, with values of RMSE, MAE and R^2 of 327.55, 164.30 and 0.8416 respectively. The worst result has been obtained with the Min-Max normalization. The MLP is the most accurate method in this study. Regarding the size of the MLP, we observe that a MLP with a high number of neurons in the hidden layers obtains better results than the one with less neurons because we have enough data with enough complexity to require some architectural complexity. We also observe that using PCA does not improve the results in the MLP case. In the case of the ELM, it obtains the best results with the Min-Max normalization, with RMSE, MAE and R^2 of 360.68, 215.22 and 0.8080 respectively.

Table 7 shows the performance of the linear regression methods considered: LREG and EREG. The parameter α of the elasticnet regularization has been tuned using a grid search. We only show the regressor with the coefficient that obtained the best results in the grid search. The LREG obtains the best results in this category with RMSE, MAE and R^2 of 389.51, 240.95 and 0.7761, respectively. It is worthy of attention that the same results are obtained independently of the normalizations or use of PCA for many of the linear models, which can sometimes happen in linear models when an optimum is achieved. EREG, which includes weight regularization, obtains close results to the LREG for all types of normalization. It does not seem that the regularization has a very big effect for this the regressors for this dataset. However, including PCA in the EREG worsens the results. The best value found for the α parameter of the elasticnet regularization is $\alpha = 1$, for which the regularization is equal to the Lasso regularization. Note that any of these linear regressors obtain better results than the statistical-based evaluated ML methods, which is somehow surprising, mainly for the poor performance of statistical-based algorithms in this problem.

Table 8 shows the performance of two statistical-based methods for regression: SVR and GP. SVR_{simp} uses the default hyperparameters, and the SVR's have been selected from a grid search. We have used two kernels, RBF and sigmoid, For the GP we have used: multiple kernel functions: dot product, dot product + white noise, RBF, RBF + white noise, and have used a range of values for the α parameter. The SVR has obtained the best results in this category, but it is far from those obtained by the other evaluated ML methods. We observe a big sensitivity to the different normalizations in the case of the SVR. The best result for the SVR is obtained with the Min-Max normalization with RMSE, MAE and R^2 of 407.97, 12.27 and 0.7544 respectively. Other normalizations decrease the performance of this technique. GP obtains worse results than SVR. Only the MaxMin normalization produces adequate results combined with the GP. The rest of the evaluated normalizations produce poor results, with GP obtaining very high values of RMSE and MAE, 1624.30 and 1400.51 respectively (with standarization and PCA). It is also surprising that GP is the method which obtains the worst results in

this study. This could be because of the approach taken for time series in this work; maybe its performance would have been better if the time series had been considered sequentially. It is also in this particular case when it becomes apparent that the R^2 is not well suited for non-linear regressors, we see negative values appear, and some are very big, like the last GP which has a $R^2 = -2.8925$.

Table 9 details the performance of the best methods evaluated for each category, corresponding to RF, MLP_{big}, LREG and SVR, respectively, see Tables 5, 6, 7 and 8. Note that the best version of the RF uses the Min-Max normalization, and both MLP and LReg the standardization of the input variables. We can observe that the MLP_{big} achieves the most accurate results for visibility prediction, with the lowest RMSE. RF obtains an acceptable result, and the LREG is far away from the other two ML approaches, as expected.

Fig. 13 shows the prediction of three of these approaches in a chosen fragment of about 250 samples of the test set. We can observe that all the models can predict the vertical transitions in visibility and most follow the continuous values pretty well. The MLP_{big} is the one that better approximates the top and bottom areas of the visibility variable followed by the RF, whereas the LREG is unable to reach the full range of values of the target variable. We can observe this in the places where the visibility value is constant in a few consecutive samples, the LREG predictions introduce a lot of noise and never quite reach the bottom or top target value. This may be because the linear model is not complex enough to model the target variable from the predictors used, so this model would have a high bias and a high variance, its linear assumption is wrong and small changes in the data create big changes in the prediction. As the linear models have been regularized we can conclude that a linear model is not suited for this task. In the non-linear models shown we can see that there is not much difference between the MLP_{big} and the RF, and at a glance it may seem that the MLP_{big} has a higher error in the predictions compared to the RF, but when carefully observing the predictions we can see that RF has trouble following sudden changes in the ground truth values, we can see very big spikes, which mean that it can't get the amount of visibility correct after a sudden change while the MLP_{big} can better assess what the value after a sudden change will be. It can also be seen when carefully observing that RF has a tendency to hold values and can not subtly change its predictions at the extremes. As for the MLP_{big} we can observe that the error seems a bit high and that model has a little bit of trouble when the target values are close to 0, introducing some noise in the predictions, but in every other section of the target variable it performs extremely well, modelling sudden changes and continuous high visibility values much better than the any of the other models used.

4.2.2. Classification results

In this section we analyze the classification results and provide an exhaustive discussion on the performance of the ordinal and the nominal classification methodologies in this problem. The purpose of this is to deduce which of the methodologies are better for the classification task with this dataset. In order to carry out this study, we consider a GLM model, which is simple, easy to train and has nominal and ordinal versions readily available. This enables us to carry out a quick and fair comparison. To explore the performance of this technique in each classification task, the GLM model has been trained varying two parameters: first, we evaluate all combinations of the balancing techniques

Table 9
Comparison of the best methods of the ensemble, ANN-based and Linear and statistical-based regression methods. MLP_{big} with the standard input normalization obtains the best results in terms of RMSE, MAE and R^2.

Model	Scaling	PCA	RMSE	MAE	R^2
MLP_{big}	Standard	No	**327.56**	**164.30**	**0.8417**
RF	Min-Max	No	340.55	175.35	0.8288
LREG	Standard	Yes	389.51	240.95	0.7761
SVR	Min-Max	No	407.97	212.27	0.7544

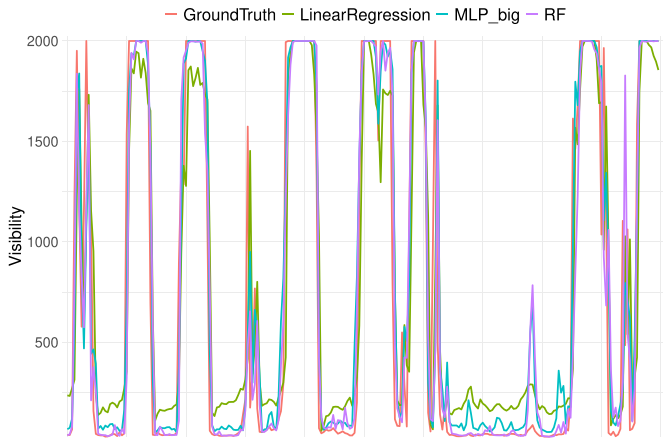


Fig. 13. Predictions obtained by the best ensemble (RF), ANN-based (MLP_{big}) and Linear regression method (LReg), in the prediction of low-visibility events with regression approaches.

described in Section 3.5; then, we use a grid search to select the best hyper-parameters for the regularization term. We test α values ranging from 0 (which is equal to ridge regularization (L_2 norm)) to 1 (which is Lasso regularization (L_1 norm)):

$$\mathscr{L} = \| \mathbf{y} - f(\boldsymbol{\omega}; \mathbf{x}) \|_2 + \alpha \| \boldsymbol{\omega} \|_1 + (1-\alpha) \| \boldsymbol{\omega} \|_2 \tag{22}$$

This results in 11 α values and 9 datasets which provides a total of 99 combinations in the grid search process for each (i.e., the nominal and the ordinal) task. The complete results of all 99 combinations are shown in Figs. 14 and 15, but for the sake of brevity in Tables 10 and 11 only the best results (the model with the best performing α) for each dataset and for the nominal and ordinal problem are shown; we present 18 combinations: for each of the datasets the best nominal and ordinal GLM.

Table 10 shows the accuracy (ACC), quadratic weighed kappa (QWK) and weighted F1-score ($F1_w$) metrics for the best hyper-parameters found on the grid search, in both nominal and ordinal classification problems. The trained GLMs have been sorted in descending according to the weighted F1-score ($F1_w$) value achieved in the test partition. The best performance for all rebalancing techniques and both ordinal and nominal version has been consistently obtained when $\alpha = 1$, namely, when using Lasso regularization. This informs us that in this particular problem it is better to prune some input variables for simple models that may suffer from redundant information (Tibshirani, 1996).

We observe that nominal GLMs consistently obtain better results than ordinal GLMs for all metrics shown, including an ordinal metric like QWK, and a metric fit for imbalanced data like weighted F1-score ($F1_w$). This difference in results also happens for all balancing techniques combinations; even when using the original dataset without rebalancing, the nominal version outperforms the ordinal, with the single exception of the CoNN undersampling technique which scores lower values than two ordinal predictions, and this is possibly because of the aggressive nature of CoNN's undersampling, which turned class 4 into the minority class. So, in this particular case, we can clearly see that there is no benefit or improvement when treating the problem as an ordinal classification task.

The best results in terms of ACC are obtained by the nominal GLM using RUS without any oversampling method, with a value of 0.8275. However, given the nature of the data and the underlying problem the other two metrics are more trustworthy because we know that ACC is only taking into account the amount of correct predictions and in this case, as there is no oversampling, there are less samples of minority classes which means that the model is not making a better prediction across classes, it is just mostly predicting class 4 with a lower share of

Fig. 14. F1 score of the evaluated GLMs in the nominal version. We show all combinations of alpha value and the balancing techniques studied.

samples of the minority classes. The best results in terms of QWK and $F1_w$ were obtained by the nominal GLM using both NCR undersampling and SMT oversampling method with a value of 0.9310 and 0.7998, respectively. The worst nominal GLM results have been obtained for the CoNN undersampling technique. This method drastically reduces the number of samples in the majority class leaving it as the class with least samples, as seen previously in Fig. 12a. For the nominal task the ACC, QWK and $F1_w$ scores range between $0.75 - 0.827$, $0.909 - 0.931$ and $0.76 - 0.799$, respectively.

We now show an exhaustive comparison of both GLMs with all the combinations tried during the grid search, namely all datasets and all alpha values, showing the values of weighted F1 score ($F1_w$) in descending order. In Fig. 14 we show the nominal GLM F1 score and in Fig. 15 we show the ordinal ones. In both figures the colors represent the dataset used, and the x-axis value shows the alpha values.

It is noticeable in Figs. 14 and 15 that the best values of the $F1_w$ are obtained for each dataset when the regularization parameter $\alpha = 1.0$, so, in all of the GLM tables with the results presented, we have omitted the unity value of alpha. In both nominal and ordinal GLMs, the NCR with SMT is the balancing technique method that obtains the best $F1_w$ value.

It is noteworthy from Fig. 14 that the CoNN method is the worst undersampling technique for the particular nominal task, having all appearances at the very end of the sorted values, but on the contrary, for the ordinal task, as seen in Fig. 15 it is among the best performing. This perhaps means that when treating the problem as ordinal situation, we need to more clearly delimit the classes, which would then require us performing a higher undersampling on the majority class to equalize the number of samples of all classes.

Table 11 shows the recall per class parameter for the best GLMs obtained from the grid search process and this has been sorted out by average weighted Recall. As we are not only interested in having very high scores in these metrics but also having a balanced prediction of when the fog appears and the actual level of visibility, we use the Recall weighted by support, which accounts for class imbalance. We use this metric because the main problem when working with this dataset is correctly recognizing the samples from minority classes, and this metric represents the amount of samples correctly classified for each class, so we are measuring for each class how many samples out of all the samples belonging to this class are predicted correctly. This way we can ensure that we are not being deceived by a high metric value that hides within it a poor classification performance in minority of the classes.

Overall, we observe that, as previously seen in Table 10, the nominal GLM generally obtains better results than the ordinal GLMs for all classes. The first peculiar feature is that that there is no overall best combination that achieves the highest values across all classes although by choosing this metric for comparison, we can be certain that we have a sorting by performance across all classes. For the best combination achieved, nominal GLM with NCR and SMT data, the highest overall values are achieved for class 0 and 2, but this comes at the cost of decreasing the performance in other classes like 1, 3 and 4, where other other combinations make better predictions. For this best model we can see that the performance is excellent in classes 0 and 4, so we are correctly discerning very high visibility and very low visibility, but for the classes in between there is more difficulty. In general, we can also see that for classes 4 and 1 some other combinations obtain better results, like Ordinal GLMs, that achieve values of 0.97 and 0.8 respectively more consistently, but their performance for other classes is not as good, specially with classes 2, 3 where the values are very low, and for class

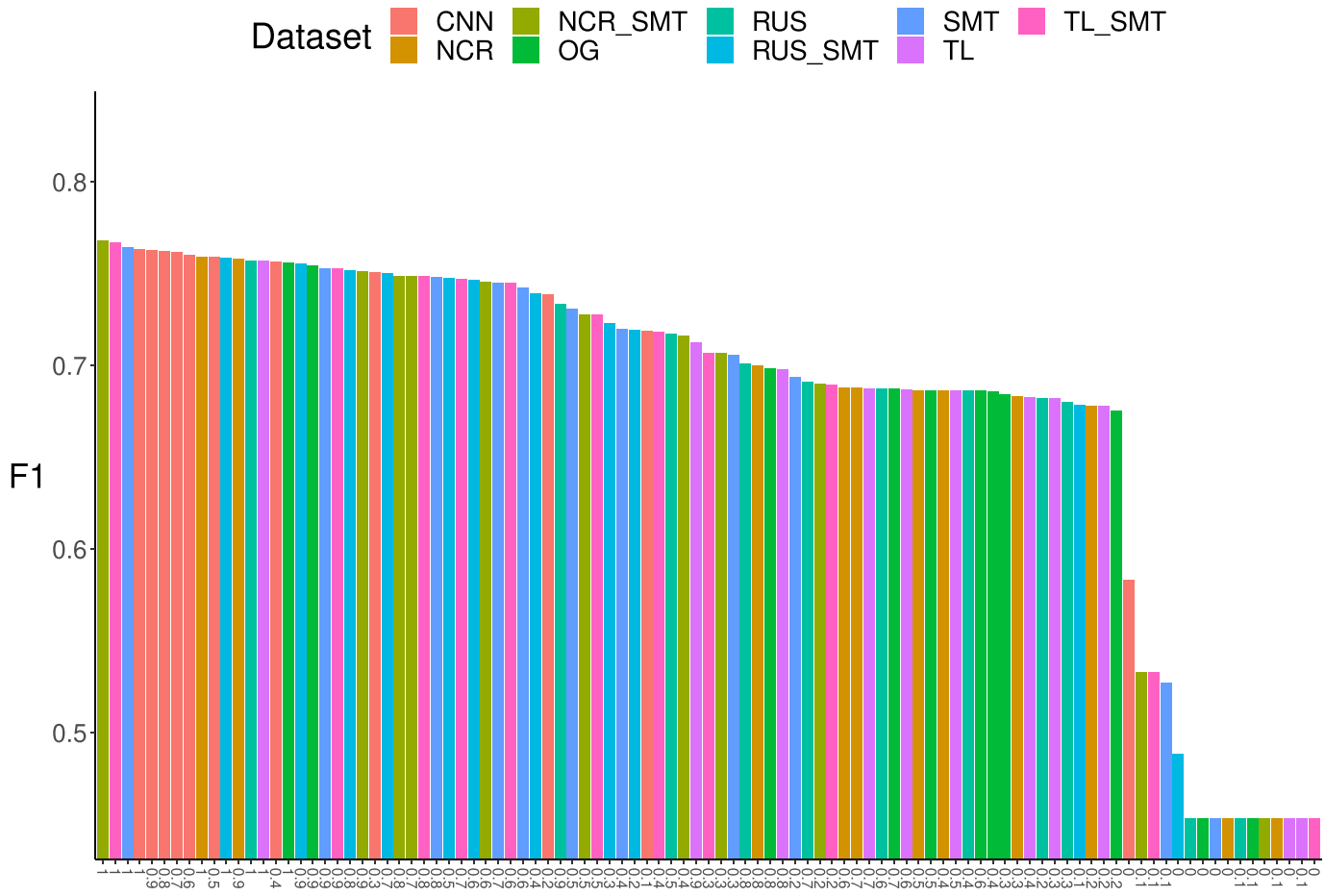


Fig. 15. F1 score of the evaluated GLMs in the ordinal version. We show all combinations of alpha value and the balancing techniques studied.

Table 10
Accuracy, quadratic weighed kappa and F1 score of the best grid search of the GLM. The best performance of the GLM method for all balancing techiniques and both ordinal and nominal versioan has been obtained fixing the hyperparamenter α to 1. Nominal GLM generally obtains better results than the ordinal GLM.

UnderS.	OverS.	Ord./Nom.	ACC	QWK	F1_w
NCR	SMT	Nom.	0.8064	**0.9310**	**0.7998**
RUS	SMT	Nom.	0.8036	0.9297	0.7995
TL	SMT	Nom.	0.8056	0.9296	0.7970
NCR	–	Nom.	0.8225	0.9206	0.7950
TL	–	Nom.	0.8265	0.9193	0.7944
–	SMT	Nom.	0.8059	0.9287	0.7939
RUS	–	Nom.	**0.8275**	0.9184	0.7933
–	–	Nom.	0.8247	0.9144	0.7861
NCR	SMT	Ord.	0.7779	0.9177	0.7681
TL	SMT	Ord.	0.7778	0.9172	0.7668
CNN	–	Nom.	0.7505	0.9090	0.7648
–	SMT	Ord.	0.7767	0.9158	0.7642
CNN	–	Ord.	0.7706	0.9086	0.7634
NCR	–	Ord.	0.7922	0.9064	0.7594
RUS	SMT	Ord.	0.7732	0.9150	0.7588
RUS	–	Ord.	0.7933	0.9056	0.7573
TL	–	Ord.	0.7934	0.9050	0.7569
–	–	Ord.	0.7951	0.9041	0.7562

0 when comparing with nominal GLMs.

Focusing on the nominal GLMs, the best Recall metrics have been obtained for the class 0 with 0.9544 and the class 4 with 0.9844 using the NCR with SMT and OG, respectively. These are quite good Recall values for detecting the positives cases. Both classes correspond to the highest and the lower visibility classes, respectively. Nominal GLMs get worse Recall results for the class 1 with 0.8586 in case of RUS undersampling, and definitely even worse Recall metrics for classes 2 and 3, with maximum values of 0.3408 and 0.2530 with NCR and SMT balancing technique. As we can see, nominal GLMs do not obtain good results predicting positives for classes 2 and 3 which have similar characteristics. Regarding the ordinal GLMs, the behaviour is similar but they obtain worse results. Maximum Recalls for classes 0 and 4 are 0.7900 and 0.9814 for the same balancing technique that in the nominal case NCR with SMT and OG, respectively. The Recall metric for the class 1 is better than in case of ordinal with 0.8790 using CoNN, but for classes 2 and 3 the results are poor. The best recalls for both classes are 0.3160 and 0.2718 obtained using no balancing technique, and CoNN oversampling, respectively.

Overall the best linear classification model we have achieved is the one that was nominal uses Lasso regularization and was trained with NCR performing undersampling and then oversampling with SMOTE. With this configuration we can achieve an good average relation between detecting relevant samples and discerning different classes. As for the ordinal task we do not think that it is fruitful to carry on with that approach.

We now select the best GLM methods in both ordinal and nominal versions, which are the GLM with NCR and SMT in versions (referred as GLM_n and GLM_o for the nominal and ordinal versions repectively) and carry out a comparison including more complex classifiers in the experimentation. Specifically, we include: GB, RF, Bagg, KNN, DT, AB and GNM, together with the GLM_n and GLM_o, discussed previously in Section 3.1. We use these ensemble methods because we think that their underlying structure and methodology may be able to sort some of the

Table 11
Recall per classes of the the best grid search of the GLM. The best performance of the GLM method for all balancing techiniques and both ordinal and nominal versioan has been obtained fixing the hyperparamenter α to 1. Nominal GLM generally obtains better results than the ordinal GLM.

UnderS.	OverS.	Ord./Nom.	Recall 0	Recall 1	Recall 2	Recall 3	Recall 4
NCR	SMT	Nom.	**0.9544**	0.6519	**0.3408**	0.2530	0.9530
RUS	SMT	Nom.	**0.9544**	0.6642	0.3340	**0.2718**	0.9431
TL	SMT	Nom.	**0.9544**	0.6503	**0.3408**	0.2231	0.9560
NCR	–	Nom.	0.8495	0.8521	0.1106	0.1663	0.9681
TL	–	Nom.	0.8594	0.8545	0.1128	0.1217	0.9780
–	SMT	Nom.	**0.9544**	0.6511	0.3386	0.1784	0.9619
RUS	–	Nom.	0.8693	0.8586	0.1128	0.0993	0.9799
OG	–	Nom.	0.8435	0.8513	0.1106	0.0709	**0.9844**
NCR	SMT	Ord.	0.7900	0.5784	0.3137	0.1764	0.9609
TL	SMT	Ord.	0.7881	0.5751	0.3182	0.1643	0.9629
CNN	–	Nom.	0.8752	0.8627	0.1151	0.4543	0.8067
–	–	Ord.	0.7900	0.5678	0.3160	0.1501	0.9651
CNN	–	Ord.	0.4811	0.8627	0.1422	**0.2718**	0.9085
NCR	–	Ord.	0.3821	0.8790	0.1106	0.1277	0.9728
RUS	SMT	Ord.	0.8455	0.5580	0.2370	0.1602	0.9626
RUS	–	Ord.	0.4000	0.8766	0.1106	0.0933	0.9772
TL	–	Ord.	0.3861	0.8807	0.1128	0.0933	0.9777
OG	–	Ord.	0.3861	**0.8823**	0.1083	0.0851	0.9814

pitfalls present in the data that add great difficulty for other kinds of models. Just as we did with GLMs, for all of these classifiers, we have previously evaluated all combinations of balancing techniques described in Section 3.5. We have chosen the best combination for presenting the results. The complete comparison with all the datasets is shown in Fig. 17 but as previously we have decided to show only the best dataset

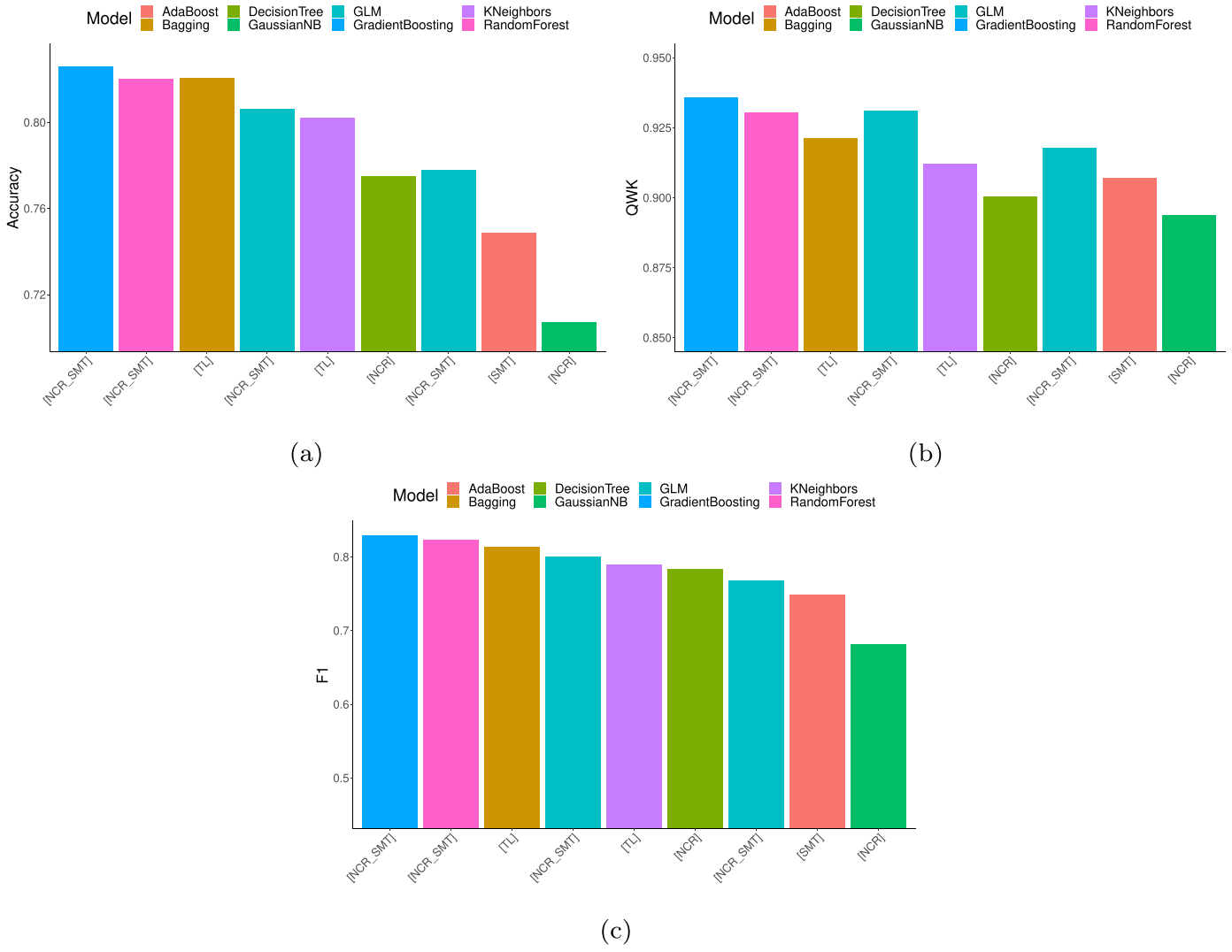


Fig. 16. (a) Accuracy, (b) Quadratic Weighted Kappa and (c) F1 score of the best classifiers methods analyzed. We choose the best balancing technique for each method in order to report the accuracy, a total of 9 different classifiers.

for each model for the sake of brevity. As for the model's hyperparameters, we have choosen the ones appearing in the original works where they first appeared, and as they are included in the sklearn library (Pedregosa et al., 2011).

Fig. 16 and Table 12 show the ACC, the QWK and the $F1_w$ for the best ML classification methods for this problem including GB, RF, Bagg, GLM_n, KNN, DT, GLM_o, AB and GNB.

GB obtains the best result in terms of ACC, QWK and $F1_w$ scores with 0.8258, 0.9359 and 0.8285 respectively. RF and Bagg closely follow the performance of the GB on these metrics. The best results are obtained by the ensemble ML methods, except for AB that obtains the second worst result. GB and RF obtain the best performance with an NCR undersampling followed by a SMOTE oversampling. Bagging's best configuration is achieved when only training with a TL undersampling. There is a gap between the previous ML methods and the results by the next set of classification methods, formed by GLM_n and KNN, although GLM_n is the second best in QWK. In both cases, the ACC, QWK and $F1_w$ have been reduced at least 2%. It is curious that the combination of NCR and SMT undersampling-oversampling technique reaches the best result (the same was obtained by RG and RF). DT and GLM_o obtain similar results in terms of ACC, QWK and $F1_w$ but perform worse than the previous one. Finally, AB and GNB obtain the worst results. AB is specially sensitive to the way in which the weighting of the samples is propagated in the sequence of the learners. And GNB has pretty strong constraints and assumptions such as the normality of the input variables, or the independence which are not true in this problem.

Fig. 17 shows the F1 score provided by the evaluated classifiers for the studied balancing techniques. We can see that in almost all the classifiers, the different balancing techniques do not lead to a substantial improvement on the $F1_w$ metric. The colored bars, denoting the model, appear contiguous, and we can see that almost all of them are grouped together for all cases. We can deduct from this that, to some extent, the model used is more important in the quality of the final predictions than the dataset used, although the dataset clearly improve to a considerable degree the predictions made by the model. This is to say that carefully choosing the models is crucial to get a good performance, and that having good data to train on improves the resulting predictions. We can also see that the CoNN undersampling technique gets the worst results in terms of F1 score for almost all classifiers, decreasing significantly the predictions obtained when trained on the original imbalanced data. We conclude that this technique, as remarked previously in Section 3.5, is not apt for this problem.

Finally, Table 13 shows the recall per class for the best set of classification methods. For all methods, the recall in classes 0 and 4 is quite good. Even GNB, GLM_o and GLM_n obtain the best recall metrics for these classes. However, for class 2, their recall is significantly reduced compared to the other methods which surpass them. It is worthy of mention that GNB has the best value of recall for class 0 and the second best for class 4, which means that for a binarized problem it would most likely have a very good performance, but it is unable to discern different levels of visibility. As we can see, the values of recall strongly drop for all methods on classes 2 and 3. Distinguish these classes with the input variables given is a very difficult task. In the best case, with GB we are able to correctly detect about half of instances of a visibility of class 2 and a third of occurrences of class 3. For class 3, AB, RF and GB obtain the best recall metric, around 0.31, clearly a very low value. But on the other hand we can very confidently predict with GB and RF instances of very low visibility, classes 0 and 1 recognizing about 88% and 75% of relevant cases, and periods of very high visibility, class 4 with a recall of 94% of relevant cases.

Fig. 18a and b show the aggregate recall and $F1_w$ score by class for all classifiers. These cumulative Figures allow us to graphically see each individual Recall and F1 score per class and the relations between them. On the x-axis the model and the dataset is shown. The ordering is according to the weighted F1 on both cases, so that is the reason for the non-monotonic decreasing order. This way we can see that the order changes between the aggregate and the weighted average used.

In this figures we can see that on both metrics the values for class 4 are very high and are almost equal across all classifiers, with a value close to 1. For class 0 the values also tend to be quite high although we can see some inverse relation of the high values in classes 0 and 1 and low values on classes 2 and 3. For classes 2 and 3, we see a considerable reduction of the Recall and F1 score for all metrics in accordance with the ordering of the classifiers.

Table 12
Accuracy, quadratic weighed kappa and F1 score of the best classification methods. Best GLMs in both ordinal and nominal version have been obtained fixing the hyperparamenter α to 1.

Model	UnderS.	OverS.	ACC	QWK	F1_w
GB	NCR	SMT	**0.8258**	**0.9359**	**0.8285**
RF	NCR	SMT	0.8202	0.9304	0.8235
Bagg	TL	–	0.8205	0.9212	0.8134
GLM_n	NCR	SMT	0.8064	0.9310	0.7998
KNN	TL	–	0.8022	0.9120	0.7899
DT	NCR	–	0.7749	0.9003	0.7837
GLM_o	NCR	SMT	0.7779	0.9177	0.7681
AB	–	SMT	0.7486	0.9071	0.7490
GNB	NCR	–	0.7071	0.8937	0.6813

4.3. Final discussion and remarks

The prediction of low-visibility events associated to fog, either defined as a regression or a classification problem, is generally more difficult than some of the other prediction problems associated with meteorological events such as wind speed, solar radiation and even the precipitation events. This is attributable to the extreme local characteristics of the fog events. In spite of this challenge, any advancement towards accurate techniques to improve the explicit variance of low-visibility event prediction is crucial for the characterization and accurate prediction of these events, and the respective application of such predictive models in fog modelling. According to the results obtained, important conclusions can be drawn from these experimental work carried out with a number of competing ML algorithms. In respect to the regression problem, we have obtained the following conclusions:

- Considering the four ML model categories that were explored, i.e., ensemble, ANN-based, linear and statistical ML methods (and the others) we conclude that the ANN-based methods are the most appropriate methods to solve a fog-events regression problem, particularly the MLP method that was able to obtain the best results in all of the regression metrics reported. The RF and GB methods were seen to obtain close the results to those of the ANN-based methods but were still relatively inferior. In general, these techniques appeared to have a better performance in the classification task formulation than in the regression problem. The rest of the ML methods are not suitable for tackling this low-visibility event prediction problem as a regression task.
- This study reveals that ML methods are influenced differently from the standarization and the normalization of the input variables depending upon the considered ML method. Some of them are strongly sensitive, such as the MLP, which reported the best results with a simple standarization but a worse result with the MinMax normalizationm equation. However, the other algorithms, such as the ensemble-based approaches seem to be robust against different standarizations and normalization for the regression taskas assigned.
- The linear regressors were seen to generate a poor result, and these were even worse with the regularization (EREG) method. Our results clearly point out to a highly non-linear structure of the fog-event datasets in the regression problem.

In respect to the classification problem associated to low-visibility

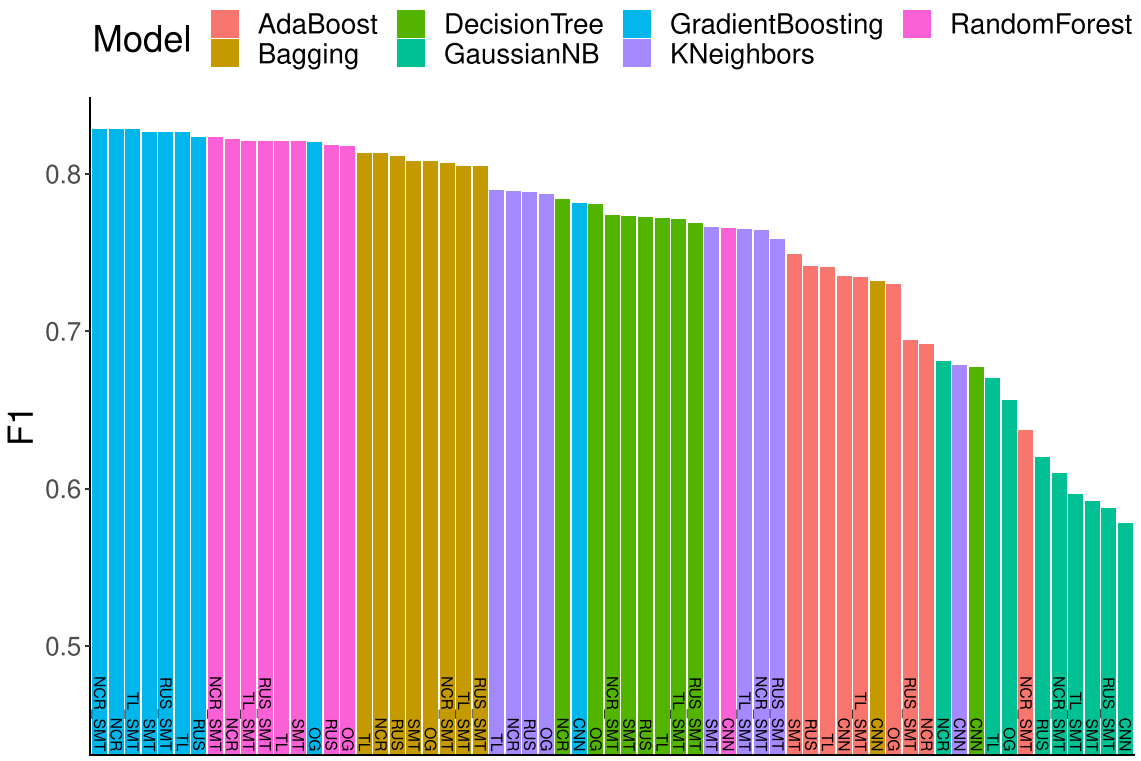


Fig. 17. F1 score of the evaluated classification methods. We show all possible combination of classifiers and balancing techniques study. GB obtains the best F1 score above 0.8 with almost all balancing techniques employed (blue bars). GNB obtains the worst results with F1 score below 0.6. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.) (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

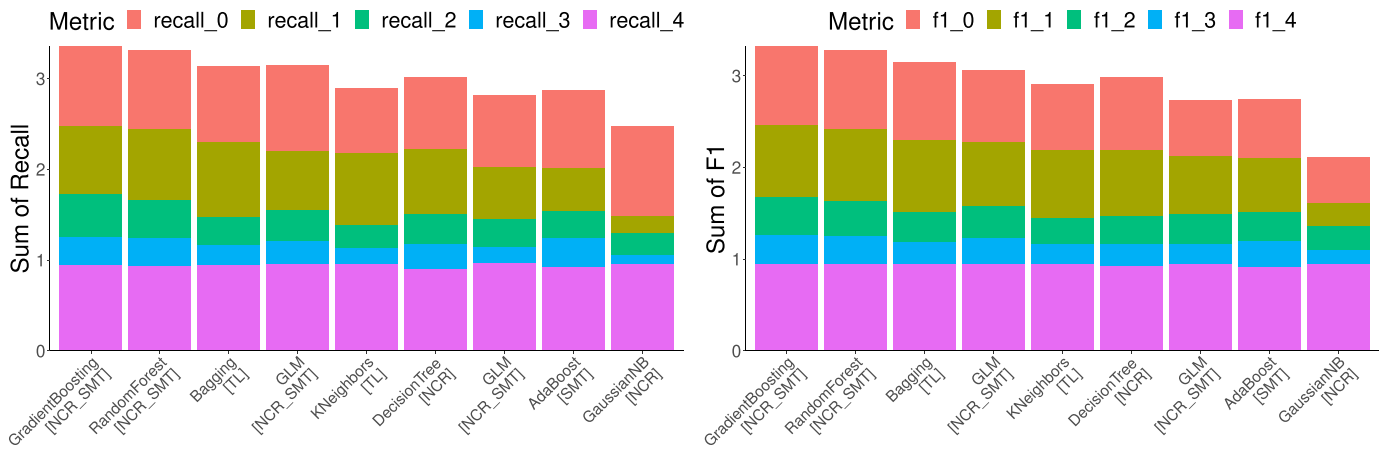
Table 13
Recall per class for the best classification methods. Best GLMs in both ordinal and nominal version have been obtained fixing the hyperparamenter α to 1.

Model	UnderS.	OverS.	Recall 0	Recall 1	Recall 2	Recall 3	Recall 4
GB	NCR	SMT	0.8772	0.7540	**0.4762**	0.3062	0.9426
RF	NCR	SMT	0.8673	0.7834	0.4176	0.3103	0.9317
Bagg	TL	–	0.8435	**0.8202**	0.3115	0.2150	0.9473
GLM_n	NCR	SMT	0.9544	0.6519	0.3408	0.2535	0.9530
KNN	TL	–	0.7128	0.8022	0.2550	0.1784	0.9493
DT	NCR	–	0.7940	0.7205	0.3273	0.2758	0.8989
GLM_o	NCR	SMT	0.7900	0.5784	0.3137	0.1764	**0.9609**
AB	–	SMT	0.8633	0.4722	0.2957	**0.3265**	0.9189
GNB	NCR	–	**0.9960**	0.1805	0.2415	0.1034	0.9550

events prediction, the following conclusions can be drawn:

- For this case we have analyzed four ML categories for this particular classification (ensemble, ANN-based, linear and statistical ML methods) and in this case we conclude that the ensemble methods are the most suitable for fog-event classification problem, particularly the GB and RF, which obtains the best results. It is easy to see that in both ensemble methods, the learners are tree structures, which behave good for fog-events classification. ADB with MLPs as learners produces a poorer behaviour in this problem.
- Since the fog-event classification problem is highly unbalanced, several balancing techniques have been evaluated in order to determine what is the best combination. As a result, we conclude that balancing techniques do not condition the ML performance of the evaluated ML methods. CoNN is an undersampling method to avoid, since it produces a worsen of the result for all the methods considered. NCR with a SMT seems to be a good combination of undersampling and oversampling for the ML methods, but slightly improves the performance, except for CoNN.
- We have carried out an analysis of nominal classifiers versus ordinal classifiers in this problem, finding out that ordinal version of the classifiers did not obtain good results for low-visibility classification. This indicates that taken into account the natural ordering in the classes does not contribute to improve the results obtained, and in this particular problem, ordinal classifiers are not a good option.
- We have chosen 5 different classes according to the 5 different low-visibility categories considered in this application. We have shown that all ML methods struggle classifying classes 2 and 3. We found out that input variables for both classes are very similar, leading to misclassification of these samples. Improving the recall and F1 score metrics for these particular classes is still a challenge after the present work, but it opens a room of opportunity. We suggest to introduce new variables which allow a better separation of both classes.

Finally, a note on the use of ML approaches versus NWM in fog events and low-visibility events prediction. As we have shown in this paper, ML approaches are able to obtain excellent results in terms of specific aspects of the event, for example, visibility prediction (as in our work). On the other hand, NWM usually analyze fog events in more



(a) Recall per classes and the overall.

(b) F1 score per classes and the overall.

Fig. 18. Recall and F1 score per classes and total of the best classifiers methods.

details and from different aspects related to physical properties of the event, but in many occasions the accuracy of NWM in visibility prediction is not so good as ML methods, as discussed in the introduction of the paper. Note that, in short prediction time-horizons ML approaches are able to exploit their ability to extract information from data, whereas NWM usually fail or do not reproduce the event with enough accuracy to provide a reliable prediction of the visibility event in the short-time. Another important aspect of ML approaches which must be taken into account is that the proposed ML algorithms may be run in a general purpose PC or even laptop, whereas NWM, even meso-scale models, are difficult to be tuned and very difficult to be run in a PC, since they usually require data assimilation and they are computationally demanding.

5. Conclusions

Fog events prediction is a significant task in aviation, road transport and several of the daily life activities in both urban and rural areas. Forecasting fog events is difficult due to its intermittent nature depending on the vertical mixing of atmospheric air with surface moisture which can change in very short timescales such as seconds. However, its proper prediction is extremely important as proper prediction of fog events can enable people to be better prepared to avoid delays for work and in many cases avoiding traffic jams, prevention of accidents at airports etc. In this paper we have carried out a comprehensive comparison of different ML classification and regression methods in a problem of hourly low-visibility events prediction due to fog. We have considered a problem of hill-fog prediction through its related low-visibility events, using visibility and atmospheric predictive variables from a motor-road in Lugo, Spain. ML classification and regression techniques have been compared in this particular problem, considering different types of input data normalization, feature reduction through a PCA algorithm and, in classification problems, including different balancing techniques (oversampling and undersampling approaches). In the analysis carried out with ML classifiers, we have shown that nominal classification techniques obtain better results for this problem than ordinal classification. The unbalanced nature of the classification problem is an important issue to obtain competitive results in this prediction problem with classification techniques, though the Gradient Boosting with oversampling has obtained the best results among all ML classifiers analyzed. On the other hand, regression techniques have the advantage of not needing any re-balancing techniques, obtaining accurate prediction results with absolute errors around 350 m in the best results obtained with a Multi-Layer Perceptron.

CRediT authorship contribution statement

C. Castillo-Botón: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data curation, Writing – original draft, Visualization. **D. Casillas-Pérez:** Conceptualization, Methodology, Software, Validation, Investigation, Resources, Writing – original draft, Writing – review & editing, Visualization. **C. Casanova-Mateo:** Conceptualization, Resources, Supervision. **S. Ghimire:** Data curation, Writing – review & editing. **E. Cerro-Prada:** Resources, Data curation. **P.A. Gutierrez:** Investigation, Project administration, Funding acquisition. **R.C. Deo:** Data curation, Writing – review & editing, Funding acquisition. **S. Salcedo-Sanz:** Methodology, Validation, Investigation, Resources, Supervision, Project administration, Funding acquisition.

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.

Acknowledgement

This research has been partially supported by Spanish Ministry of Science and Innovation (MICINN), through Project Number PID2020-115454GB-C21.

Appendix A. Acronyms

Table A.1 shows a list of alphabetically ordered acronyms that appear in this paper.

Table A.1
List of acronyms.

Acronym	Full name
AB	Adaboost
ACC	Accuracy
ANN	Artificial Neural Network
Bagg	Bagging
CoNN	Condensed Nearest Neighbours
DT	Decision Tree
ELM	Extreme Learning Machines
EREG	ElasticNet Regression
F1	F1-score
GB	Gradient Boosting
GLM	Generalized Linear Model
GNB	Gaussian Naïve Bayes
GP	Gaussian Process
IQR	Interquartile range
KNN	K-nearest neighbours
LREG	Linear Regression
MAE	Mean Absolute Error
ML	Machine Learning
MLP	Multi-Layer perceptron
NCR	Neighbourhood Cleaning Rule
NWP	Numerical Weather Prediction
PCA	Principal Component Analysis
QWK	Quadratic Weighted Kappa
R^2	Coefficient of Determination
REC	Recall
RF	Random Forest
RMSE	Root Mean Squared Error
RUS	Random UnderSampler
SGD	Stochastic Gradient Descent
SMT	SMOTE
SVM	Support Vector Machine
SVR	Support Vector Regressions
TL	Tomek Links
TN	True Negatives
TP	True Positives

References

Abdel-Aty, M., Radwan, E., Oloufa, A., Rodgers, M., 2015. et al. A Comprehensive Investigation of Visibility Problems on Highways: Developing Real Time Monitoring and Prediction System for Reduced Visibility and Understanding Traffic and Human Factors Implications.

Anber, U., Gentine, P., Wang, S., Sobel, A.H., 2015. Fog and rain in the Amazon. Proc. Natl. Acad. Sci. 112 (37), 11473–11477.

Baldocchi, D., Waller, E., 2014. Winter fog is decreasing in the fruit growing region of the central valley of California. Geophys. Res. Lett. 41 (9), 3251–3256.

Bartok, J., Bott, A., Gera, M., 2012. Fog prediction for road traffic safety in a coastal desert region. Bound.-Layer Meteorol. 145 (3), 485–506.

Bartoková, I., Bott, A., Bartok, J., Gera, M., 2015. Fog prediction for road traffic safety in a coastal desert region: Improvement of nowcasting skills by the machine-learning approach. Bound.-Layer Meteorol. 157 (3), 501–516.

Belo-Pereira, J.A.S.M., 2016. A persistent wintertime fog episode at Lisbon airport (Portugal): performance of ECMWF and AROME models. Meteorol. Appl. 23 (3), 353–370.

Bendix, J., 2002. A satellite-based climatology of fog and low-level stratus in Germany and adjacent areas. Atmos. Res. 64 (1), 3–18.

Bergot, T., Terradellas, E., Cuxart, J., Mira, A., Liechti, O., Mueller, M., Nielsen, N.W., 2007. Intercomparison of single-column numerical models for the prediction of radiation fog. J. Appl. Meteorol. Climatol. 46 (4), 504–521.

Bishop, C.M., et al., 1995. Neural Networks for Pattern Recognition. Oxford University Press.

Boneh, T., Weymouth, G., Newham, P., Potts, R., Bally, J., Nicholson, A., Korb, K., 2015. Fog forecasting for Melbourne airport using a Bayesian decision network. Weather Forecast. 30 (5), 1218–1233.

Breiman, L., 2001. Random forests. Mach. Learn. 45 (1), 5–32.

Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P., 2002. SMOTE: synthetic minority over-sampling technique. J. Artif. Intell. Res. 16, 321–357.

Colabone, R.O., Ferrari, A., da Silva-Vecchia, F., Bruno-Tech, A., 2015. Application of artificial neural networks for fog forecast. J. Aerosp. Technol. Manag. 169, 1107–1119.

Colin Cameron, A., Windmeijer, F.A., 1997. An r-squared measure of goodness of fit for some common nonlinear regression models. J. Econ. 77 (2), 329–342.

Cornejo-Bueno, L., Casanova-Mateo, C., Sanz-Justo, J., Cerro-Prada, E., Salcedo-Sanz, S., 2017. Efficient prediction of low-visibility events at airports using machine-learning regression. Bound.-Layer Meteorol. 165, 349–370.

Cornejo-Bueno, S., Casillas-Pérez, D., Cornejo-Bueno, L., Chidean, M.I., Caamaño, A.J., Sanz-Justo, J., Casanova-Mateo, C., Salcedo-Sanz, S., 2020. Persistence analysis and prediction of low-visibility events at Valladolid airport, Spain. Symmetry 12 (6), 1045.

Cornejo-Bueno, S., Casillas-Pérez, D., Cornejo-Bueno, L., Chidean, M.I., Caamaño, A.J., Cerro-Prada, E., Casanova-Mateo, C., Salcedo-Sanz, S., 2021. Statistical analysis and machine learning prediction of fog-caused low-visibility events at A-8 motor-road in Spain. Atmosphere 12 (6), 679.

da Rocha, R.P., Gonçalves, F.L., Segalin, B., 2015. Fog events and local atmospheric features simulated by regional climate model for the metropolitan area of SÃ£o Paulo, Brazil. Atmos. Res. 151, 176–188.

Dey, S., 2018. On the theoretical aspects of improved fog detection and prediction in India. Atmos. Res. 202, 77–80.

Durán-Rosal, A., Fernández, J., Casanova-Mateo, C., Sanz-Justo, J., Salcedo-Sanz, S., Hervás-Martínez, C., 2018. Efficient fog prediction with multi-objective evolutionary neural networks. Appl. Soft Comput. 70, 347–358.

Fabbian, D., De-Dear, R., Lellyett, S., 2007. Application of artificial neural network forecasts to predict fog at Canberra international airport. Weather Forecast. 22 (2), 372–381.

Fernández-González, S., Bolgiani, P., Fernández-Villares, J., González, P., García-Gil, A., Suárez, J.C., Merino, A., 2019. Forecasting of poor visibility episodes in the vicinity of Tenerife Norte Airport. Atmos. Res. 223, 49–59.

Ferreira, A.J., Figueiredo, M.A., 2012. Boosting algorithms: a review of methods, theory, and applications. Ensemble Mach. Learn. 35–85.

Freedman, D.A., 2009. Statistical Models: Theory and Practice. Cambridge University Press.

Friedman, J.H., 2001. Greedy function approximation: a gradient boosting machine. Ann. Stat. 1189–1232.

González, S., García, S., Del Ser, J., Rokach, L., Herrera, F., 2020. A practical tutorial on bagging and boosting based ensembles for machine learning: Algorithms, software tools, performance study, practical perspectives and opportunities. Inform. Fusion 64, 205–237.

Guerreiro, P.M., Soares, P.M., Cardoso, R.M., Ramos, A.M., 2020. An analysis of fog in the mainland portuguese international airports. Atmosphere 11 (11), 1239.

Guijo-Rubio, D., Gutiérrez, P., Casanova-Mateo, C., Sanz-Justo, J., Salcedo-Sanz, S., Hervás-Martínez, C., 2018. Prediction of low-visibility events due to fog using ordinal classification. Atmos. Res. 214, 64–73.
Hagan, M.T., Menhaj, M.B., 1994. Training feedforward networks with the Marquardt algorithm. IEEE Trans. Neural Netw. 5 (6), 989–993.
Hart, P., 1968. The condensed nearest neighbor rule (corresp.). IEEE Trans. Inform. Theory 14 (3), 515–516.
Haykin, S., Network, N., 2004. A comprehensive foundation. Neural Netw. 2 (2004), 41.
Hoerl, A.E., Kannard, R.W., Baldwin, K.F., 1975. Ridge regression: some simulations. Commun. Stat. Theory Methods 4 (2), 105–123.
Huang, G.-B., Zhu, Q.-Y., Siew, C.-K., 2006. Extreme learning machine: theory and applications. Neurocomputing 70 (1–3), 489–501.
Huang, G.-B., Zhou, H., Ding, X., Zhang, R., 2011. Extreme learning machine for regression and multiclass classification. IEEE Trans. Syst. Man Cybern. Part B (Cybern.) 42 (2), 513–529.
Klemm, O., Schemenauer, R.S., Lummerich, A., Cereceda, P., Marzol, V., Corell, D., van Heerden, J., Reinhard, D., Gherezghiher, T., Olivier, J., Osses, P., Sarsour, J., Frost, E., Estrela, M.J., Valiente, J.A., Fessehaye, G.M., 2012. Fog as a fresh-water resource: overview and perspectives. AMBIO 41, 221–234.
Koziara, M., Robert, J., Thompson, W., 1983. Estimating marine fog probability using a model output statistics scheme. Mon. Weather Rev. 111 (12), 2333–2340.
Laurikkala, J., 2001. Improving identification of difficult small classes by balancing class distribution. In: Conference on Artificial Intelligence in Medicine in Europe. Springer, pp. 63–66.
López, V., Fernández, A., García, S., Palade, V., Herrera, F., 2013. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. Inf. Sci. 250, 113–141.
Miao, Y., Potts, R., Huang, X., Elliott, G., Rivett, R., 2012. A fuzzy logic fog forecasting model for Perth airport. Pure Appl. Geophys. 169, 1107–1119.
Miao, K.-C., Han, T.-T., Yao, Y.-Q., Lu, H., Chen, P., Wang, B., Zhang, J., 2020. Application of LSTM for short term fog forecasting based on meteorological elements. Neurocomputing 408, 285–291.
Mohandes, M., Deriche, M., Aliyu, S.O., 2018. Classifiers combination techniques: a comprehensive review. IEEE Access 6, 19626–19639.
Montecinos, S., Carvajal, D., Cereceda, P., Concha, M., 2018. Collection efficiency of fog events. Atmos. Res. 209, 163–169.
Nelder, J.A., Wedderburn, R.W., 1972. Generalized linear models. J. Royal Stat. Soc. 135 (3), 370–384.
Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: machine learning in Python. J. Mach. Learn. Res. 12, 2825–2830.
Peng, Y., Abdel-Aty, M., Lee, J., Zou, Y., 2018. Analysis of the impact of fog-related reduced visibility on traffic parameters. J. Transp. Eng. Part A 144 (2), 04017077.
Räsänen, M., Chung, M., Katurji, M., Pellikka, P., 2018. Similarity in fog and rainfall intermittency. Geophys. Res. Lett. 45, 10691–10699.
Rasmussen, C.E., 2003. Gaussian processes in machine learning. In: Summer School on Machine Learning. Springer, pp. 63–71.
Rokach, L., Maimon, O., 2005. Decision trees. In: Data Mining and Knowledge Discovery Handbook. Springer, pp. 165–192.
Román-Cascón, C., Yagüe, C., Sastre, M., Maqueda, G., Salamanca, F., Viana, S., 2012. Observations and WRF simulations of fog events at the spanish northern plateau. Adv. Sci. Res. 8 (1), 11–18.
Román-Cascón, C., Steeneveld, G., Yagüe, C., Sastre, M., Arrillaga, J., Maqueda, G., 2016. Forecasting radiation fog at climatologically contrasting sites: evaluation of statistical methods and WRF. Q. J. R. Meteorol. Soc. 142, 1048–1063.
Román-Cascón, C., Yagüe, C., Steeneveld, G.-J., Morales, G., Arrillaga, J.A., Sastre, M., Maqueda, G., 2019. Radiation and cloud-base lowering fog events: Observational analysis and evaluation of wrf and harmonie. Atmos. Res. 229, 190–207.
Rumelhart, D.E., Hinton, G.E., Williams, R.J., 1986. Learning representations by back-propagating errors. Nature 323 (6088), 533–536.
Salcedo-Sanz, S., Rojo-Álvarez, J.L., Martínez-Ramón, M., Camps-Valls, G., 2014. Support vector machines in engineering: an overview. Wiley Interdisc. Rev. 4 (3), 234–267.
Salcedo-Sanz, S., Piles, M., Cuadra, L., Casanova-Mateo, C., Caamaño, A., Cerro-Prada, E., Camps-Valls, G., 2021. Long-term persistence, invariant time scales and on-off intermittency of fog events. Atmos. Res. 252, 105456.
Schölkopf, B., Smola, A.J., Williamson, R.C., Bartlett, P.L., 2000. New support vector algorithms. Neural Comput. 12 (5), 1207–1245.
Schölkopf, B., Smola, A.J., Bach, F., et al., 2002. Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT press.
Shakhnarovich, G., Darrell, T., Indyk, P., 2008. Nearest-neighbor methods in learning and vision. IEEE Trans. Neural Netw. 19 (2), 377.
Shrestha, S., Moore, G.A., Peel, M.C., 2018. Trends in winter fog events in the Terai region of Nepal. Agric. Forest Meteorol. 259, 118–130.
Smola, A.J., Schölkopf, B., 2004. A tutorial on support vector regression. Stat. Comput. 14 (3), 199–222.
Steeneveld, G., Ronda, R., Holtslag, A., 2015. The challenge of forecasting the onset and development of radiation fog using mesoscale atmospheric models. Bound.-Layer Meteorol. 154, 265–289.
Stolaki, S., Haeffelin, M., Lac, C., Dupont, J.C., Elias, T., Masson, V., 2015. Influence of aerosols on the life cycle of a radiation fog event. A numerical and observational study. Atmos. Res. 151, 146–161.
Tapiador, F.J., Sánchez, J.-L., García-Ortega, E., 2019. Empirical values and assumptions in the microphysics of numerical models. Atmos. Res. 215, 214–238.
Tibshirani, R., 1996. Regression shrinkage and selection via the lasso. J. R. Stat. Soc. Ser. B Methodol. 58 (1), 267–288.
Tomek, I., et al., 1976. Two Modifications of CNN.
van der Velde, I.R., Steeneveld, G.J., Wichers Schreur, B.G.J., Holtslag, A.A.M., 2010. Modeling and forecasting the onset and duration of severe radiation fog under frost conditions. Mon. Weather Rev. 138 (11), 4237–4253.
Wu, Y., Abdel-Aty, M., Lee, J., 2018. Crash risk analysis during fog conditions using real-time traffic data. Accid. Anal. Prev. 114, 4–11.
Zhang, H., 2004. The optimality of naive bayes. AA 1 (2), 3.
Zhou, B., Du, J., Gultepe, I., Dimego, G., 2011. Forecast of low visibility and fog from NCEP: current status and efforts. Pure Appl. Geophys. 169, 895–909.
Zou, H., Hastie, T., 2005. Regularization and variable selection via the elastic net. J. R. Stat. Soc. Ser. B Methodol. 67 (2), 301–320.