Hindawi
Advances in Meteorology
Volume 2019, Article ID 3181037, 14 pages
https://doi.org/10.1155/2019/3181037

Research Article

Statistical Postprocessing of Different Variables for Airports in Spain Using Machine Learning

David Quintero Plaza and José Antonio García-Moya Zapata

AEMET, Agencia Estatal de Meteorología (Spanish National Weather Service), C/ Historiador Fernando de Armas 12, Las Palmas de Gran Canaria, Spain

Correspondence should be addressed to David Quintero Plaza; dquinterop@aemet.es

Received 22 January 2019; Revised 15 March 2019; Accepted 10 April 2019; Published 30 April 2019

Academic Editor: Helena A. Flocas

The results of a deterministic calibration for the nonhydrostatic convection-permitting LAM-EPS AEMET-γSREPS are shown. LAM-EPS AEMET-γSREPS is a multiboundary condition, multimodel ensemble forecast system developed for Spain. Machine learning tools are used to calibrate the members of the ensemble. Machine learning (hereafter ML) has been considerably successful in many problems, and recent research suggests that meteorology and climatology are not an exception. These machine learning tools range from classical statistical methods to contemporary successful and powerful methods such as kernels and neural networks. The calibration has been done for airports located in many regions of Spain, representing different climatic conditions. The variables to be calibrated are the 2-meter temperature, the 10-meter wind speed, and the precipitation in 24 hours. Classical statistical methods perform very well with the temperature and the wind speed; the precipitation is a subtler case: it seems there is not a general rule, and for each point, a decision has to be taken of what method (if any) improves the direct output of the model, but even recognizing this, a slight improvement can be shown with ML methods for the precipitation.

1. Introduction

The necessity for calibration of meteorological models is known for many years. Whether by the use of classical statistical methods or by more modern and advanced techniques [1], the art and science of forecasting takes advantage of the momentum that calibration has to offer. There are methods that help to improve the forecasts, such as MOS (Model Output Statistics) [2] and Perfect Prog [3], with the main difference being that the model is considered in MOS but not in Perfect Prog. Later on, specific calibration for ensembles was developed, such as the BMA (Bayesian Model Averaging) [4], obtaining good results.

In this work, the nonhydrostatic convection-permitting LAM-EPS AEMET-γSREPS has been used. LAM-EPS AEMET-γSREPS is a short-range mesoscale forecast system developed for Spain. This is an ambitious and original short-range ensemble which mixes different boundary conditions and NWP models (so, it is a multiboundary, multimodel ensemble). It is constituted by 20 members and runs at 2.5 kilometres resolution, and it is convection permitting. It uses two branches of the European model Harmonie (ALARO and AROME), the WRF-ARW from NOAA-NCAR, and the NMMB from NOAA-NCEP. The boundary conditions come from 5 Global NWP models (centre/NWP model): ECMWF/IFS, NOAA-NCEP/GFS, Canadian CMC/GEM, Japanese JMA/GSM, and Météo-France/ARPÈGE. Multimodel is the approach to take into account NWP model's errors and uncertainties mainly in the mesoscale, and the initial and boundary conditions uncertainties are dealt through multiboundaries approach which are more related to synoptic uncertainties. The multiboundaries and multimodel design of AEMET-γSREPS is the same as its predecessor AEMET-SREPS [5] because of the same reason: better performance in terms of more consistent EPS, with better skill than using other EPS approaches as multiphysics, stochastic parameterizations, multiparameters, and boundary conditions from a global EPS [6].

For the calibration, it was decided to use a deterministic approach with different machine learning (ML) methods; that is, it was decided to calibrate each of the 20 members as if they were a deterministic model, and 5 airports that represented different climatic conditions of Spain were chosen; these airports were Madrid-Adolfo Suárez-Barajas, Barcelona-El Prat, Vigo-Peinador, Palma de Mallorca-Son San Juan, and Málaga-Costa del Sol. Madrid has an airport in the middle of the Iberian Peninsula, with a continental and dry climate; two airports were close to the coast (Barcelona and Palma de Mallorca): one of them (Palma de Mallorca) is in an island with Mediterranean climate. The other two airports are in the wet Atlantic facade of Spain (Vigo) and in the hot land of Andalucía in the South of Spain (Málaga).

It was decided to calibrate 3 variables that have a clear impact for the sensitive weather in the surface: temperature, wind speed, and precipitation in 24 hours. The sophistication of the calibration was roughly in the increasing order from temperature to precipitation, due to the inherent difficulties associated with such variables. As mentioned, machine learning (ML) tools were used, a range of powerful statistical methods that are growing in popularity due to their success. A brief overview of ML methods is shown in the next section.

Some efforts have been done in the past using ML as a calibration tool. There are new and promising results using ML for instance for the nowcasting of precipitation [7] along with older achievements like in [8–11]. This work is original because the calibration has been done massively in an ensemble with 20 members and because of the physical considerations added to the approach to complement the ML techniques.

2. Materials and Methods

Machine learning methods are a wide range of statistical tools that allow extraction of meaning from data. Indeed, statistics and machine learning can be synonyms. Both are concerned with learning from data. Simplifying perhaps too much, one could say that statistics puts an effort in formal inference for low-dimensional problems while machine learning deals with high-dimensional problems [12]. The key point seems to be that, with the increase of computational power, many problems previously considered intractable can be at least partially solved. Some terms that denote the same concept differ depending on whether the user comes from an ML or a statistics background (for instance, estimation in statistics and learning in ML). We will use in this article the ML terminology.

Machine learning can be divided in 3 big paradigms: reinforcement, supervised, and unsupervised learning. Reinforcement learning is applied when a system learns, while it evolves interacting with an environment. The learning is supervised or unsupervised depending on if there is a function or a set of labels that guide the learning. In this work, the supervised paradigm is used, with data (observations) that should be similar to the model. Inside the supervised paradigm, there are classification or regression problems depending on discrete or continuous variables, respectively. As in this work variables are continuous, regression is the technique used.

Among the many techniques present in the literature of ML, these methods were chosen: ridge regression, lasso, elastic net, Bayesian ridge, random forest regression, gradient boosting, XGBoost, AdaBoost, polynomial regression, singular vector regression (SVR), and feedforward neural networks (FNNs) that are briefly described in the next sections.

2.1. Ridge Regression, Lasso, Elastic Net, and Bayesian Ridge. These methods are sophisticated versions of the classical linear regression solved by Carl Gauss 200 years ago. They minimize a squared error function as in linear regression but with the peculiarity of adding an extra term to prevent *overfitting*. Overfitting is a word that will appear a lot in this work. It means that a model has finished extracting the main features from a dataset, and it is just memorizing that dataset. In such case, all the ability to get a good performance is lost when we apply the model to another, even similar, dataset. A way to prevent overfitting is to limit the number of free parameters in a model (see, for instance, John von Neumann ironic comments on this [13]).

In the case of ridge, the penalty term (also known as a *regularization* term) added to the square error is named *L2*. The error function for ridge is as follows:

$$\mathrm{MSE}_R = \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} X_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2, \tag{1}$$

with y_i the observations and $\sum_{j=1}^{p} X_{ij}\beta_j$ the linear model that performs the fit (X_{ij} are the elements of the predictor matrix, and β_j are the coefficients), λ is the penalty coefficient, n is the number of points in the dataset, and p is the number of predictors.

Lasso is very similar to ridge, but it uses a $L1$ penalty term, i.e., the absolute value instead of a square term, that is,

$$\mathrm{MSE}_L = \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} X_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} \left|\beta_j\right|. \tag{2}$$

One of the main features of lasso is that it reduces the number of predictors used in the regression to only those that provide more information for the function to be closest to the observations.

Elastic net is a combination of ridge and lasso at the same time. Bayesian ridge assumes a Bayesian probability thinking, assigning a Gaussian probability distribution for the parameters of the model and then estimating them during the regression; this results in an approach very similar to ridge [14].

2.2. Random Forests and Boosting Techniques. Random forest consists in the creation of an ensemble of decision trees and in taking the mean of the values that those trees estimate. A decision tree is a model similar to a flowchart consisting of branches and nodes. Nodes are functions of the data, such as mean squared error (MSE) or information-related metrics

(Akaike information criterion, for instance). Branches are the different outcomes of those operations performed in the nodes. At the end of the tree, we have the leaves, that is, the final outcome of the different operations. When working with random forests, it is crucial to set the adequate deepness of the trees, that is, the number of levels in the flowchart.

Boosting combines the random forest approach with the minimization of an error function, like MSE. They use the gradient descent technique for such minimization. Knowing that the gradient gives the direction of maximum growth of a function, we can move in the opposite direction in order to search for the minimum of such function:

$$w_{i+1} = w_i - \alpha \nabla g(w), \tag{3}$$

where the parameter w is what we search, iteratively, $g(w)$ is the error function (for instance, the MSE), and α is the learning rate, a constant that can be fine tuned. Gradient descent admits lots of variants to improve its performance, such as the momentum method, to diminish the oscillations of the movement through the parameter space, or the option to take the learning rate as a variable to optimize instead of a constant. Examples of these techniques are the AdaBoost, gradient boosting, and XGBoost (the last two differ mostly in the implementation details).

From a broader perspective, random forest can be seen as algorithms that reduce the variance of a model and the boosting algorithms can be seen as a reduction of the bias of models. Both techniques lead to a reduction of the MSE since we know that

$$\text{MSE} = \text{bias}^2 + \text{variance}. \tag{4}$$

2.3. Singular Vector Regression and Neural Networks. These are important on fashion techniques that have showed considerable power when dealing with big and complex datasets (especially the neural network). The details of their implementation and their working are quite convoluted, and here, only a very brief summary of their features is shown. Interested readers can consult [14, 15].

SVR (singular vector regression) is a technique based on going to higher dimensional spaces in order to convert a nonlinear problem into a linear one. Working in higher dimensional spaces has a cost, the so-called *curse of dimensionality*, but the clever use of some functions (kernels) simplify and reduce most of the computations.

A feedforward neural network (FNN) is an artificial imitation of a human brain, with hierarchical layers of neurons that receive a set of inputs and compute nonlinear functions or *activations*. These neurons depend on parameters that can be learned using a technique inspired in the gradient descent called *backpropagation*. Besides the parameters "inside" the neurons, there are many hyperparameters in neural networks that also need to be tuned, like the number of layers, the exact type of functions the neurons compute, or the algorithm that performs the backpropagation. And of course, like in any other ML method, it is necessary to fight against overfitting. All this shows that, although they have considerable power, neural networks can be very tricky to train.

3. Results and Discussion

3.1. Calibration for 2-Meter Temperature. The training dataset was from November 14, 2016, to January 22, 2018, roughly one year and two months. The observation of the 2-meter temperature came from the METAR reports of the 5 airports. The 4 closest points to the coordinates of the observation stations were the points chosen, covering a grid area of $2.5 \times 2.5\,\text{km}^2$. The reason of choosing these 4 points is because not always the closest point provides the better information and also because more information can be gained by adding other points. In Figure 1, the example of how much information is gained is shown with a method that performs especially well, the ridge regression. The example of Madrid airport is shown. The observation point is at latitude 40.485 degrees and longitude −3.570 degrees. The 4 points are at roughly 0.712 km, 1.992 km, 2.137 km, and 2.833 km from the observation point. These distances have been calculated using the Vincenty distance that comes from considering the Earth as an ellipsoid with the projection WGS-84, as provided for instance in the *geopy* library for Python. As it is possible to see in Figure 1, it is not the closest point to the observation point the one with the best R^2 coefficient. In this case, it is the second point, the one with the best R^2. It is possible to see in Figure 2 that, with 4 points, the best R^2 coefficient is obtained. Of course, more points could still be added to the regression, and in fact, this was done for the case of the precipitation, due to the spatial uncertainty of this variable; however, for the temperature, adding more points would mean losing the high resolution of our model. It was thought that, with 4 closest neighbours, there was a nice trade-off between resolution and extra information.

In some airports, such as Madrid, the 4 closest points are all land points, so no special measure need to be taken. But in cases like Barcelona or Palma, some of the 4 closest neighbours could be (and were) points over the sea. It is known that the diurnal cycles of the temperature over the ocean and over the land are different. A legitimate approach could be to include the 4 points in the regression, without considering if they are land or sea; the elimination of systematic errors (biases), like the differences in the temperature between land and sea, is something that ML algorithms are especially good at. However, it was decided that an extra "help" could be provided to the algorithms filtering those points over the sea. In the ML literature, this is called *feature engineering*, and in some cases, it is essential for a good result. So, the approach was to use a land-sea mask for the member of the LAM-EPS AEMET-γSREPS ensemble to perform the filtering. A function was implemented in the Python code with the help of the *ecCodes* library from the ECMWF to choose the 4 closest points. For every point, it was checked if it was from the sea or from the land. Points from the sea were discarded. An extra, special routine for the (quite exceptional) case of the 4 closest points coming from the sea was added; in such case, the routine would be searching until a land point appeared and that would be the chosen point. In the cases where some points were discarded due to the fact that they were sea points, it was decided not to

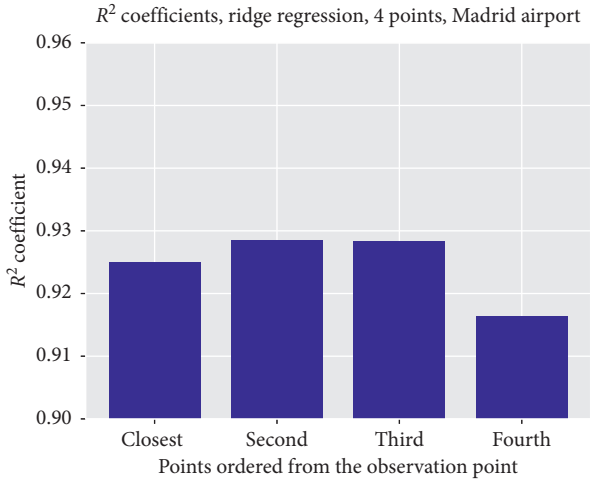

Figure 1: Quality of the ridge regression for different number of points (temperature).

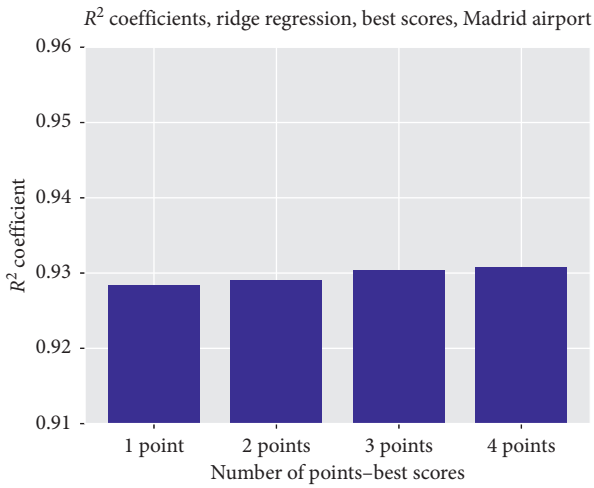

Figure 2: Quality of the ridge regression for the 4 closest points (temperature).

continue the search until completing the 4 points, in order to not be too far from the observation (and therefore losing the power of a high resolution model). A compromise solution was to select only those points that were land points, even if there was only one.

Data from the model from $H+06$ hours until $H+36$ hours were used, each 3 hours (observations were each thirty minutes as usual with METAR, but the model steps were each 3 hours). We performed a quality control over the model and over the observations deleting gross outliers (the threshold was put in values bigger than ±80 degrees Celsius); for the model, this was the only control. This threshold seems strange, but the reasons for it are part of the ML philosophy: to search a trade-off between the ability to discard very bad forecasts from the model that would spoil all the learning while, at the same time, being able to penalize the model if it showed wrong values (without spoiling the entire learning). Experience has shown that observations need a quality control too. For the observations, it was performed an extra, very draconian quality control: for each observation at hour H, the average of the temperature at H − 3 and $H+3$ was taken, and the value at H was kept if the difference with respect to the previously calculated average was below 5 degrees in the absolute value. Perhaps valid values were deleted with this procedure, but this method is robust to different types of changes in the temperature, even to some abrupt changes, and at least one can reasonably be sure that outliers were not present in the observations.

After these quality controls (and other basic checks such as deleting repeated values for the same hour), the 1 column for the observations and the 4 columns for the forecasts were confronted and prepared for the calibration or training with the different ML methods. This joining and preparation of the dataset was done using the very useful *Pandas* library from the Python environment.

The results of the training for the 5 airports chosen with one of the members of the LAM-EPS AEMET-γSREPS ensemble, member 019, are shown. There is no special reason to choose this member although the results exemplify what happens with a lot of the other members. The intention is to show the results from temperature, wind, and precipitation for members with completely different NWP models and boundary conditions, so different members representing different models and boundary conditions for each of the three variables have been chosen. Member 019 is the WRF-ARW model [16] with the boundary conditions from the Japanese Global Model, GSM (http://www.jma.go.jp). In the graphs, the MSE error is in the vertical bar and the different ML methods used in the horizontal bar. The horizontal lines are the MSE of the model without postprocessing: the green line is the MSE of the closest point to the observation of the 4 closest points, and the red line is the MSE of the point with the minimum MSE of the 4 original points. In many occasions, these lines coincide because the points coincide. For each ML method, the average performance and its standard deviation have been calculated. This calculation has been done using *cross validation*. In cross validation (CV), the dataset is divided in N parts and $N-1$ parts are used for the learning or inference of the parameters of the model and an evaluation is performed with the remaining part. This procedure is repeated until each one of the N parts has been the part of evaluation. Then, the average and the standard deviation are calculated, and an honest evaluation of the model is obtained. A CV with $N=10$ parts was used except for the SVR, where for computational optimization, only 5 parts were used. In two of the models, the FNN and the SVR, a technique called *nested cross validation* was used; this is basically 2 CVs, the deepest one to find the best hyperparameters that define the model and the other one for the performance of the model with the selected hyperparameters.

The graphs for the 5 selected airports for the member 019 are shown. The MSE is in the vertical axis and in the horizontal the different ML methods. The average of the performance for each method is shown, and its standard deviation is calculated with cross validation. For each MSE, it represents the MSE plus the standard deviation as the top of the bar and the MSE minus the standard deviation as the

lowest part of the bar, to give an idea of the range of variability. As previously stated, the red line is the model output without postprocessing for the minimum MSE of the 4 points. The green line is the MSE of the closest point to the observation, so it is really the point (or the line) to use to compare between the model and the ML methods. The calibrations of the 5 airports are shown in Figures 3–7.

As an extra, a scatter plot between the point with the minimum MSE and the observations is shown (Figure 8). In an ideal model, all the points would be in a diagonal line at 45 degrees. A fit to a straight line and to a second-order polynomial is performed, to provide intuition.

It is necessary to comment that the same weights and biases have been used in the algorithms for all hours from $H+06$ to $H+36$. A more rigorous approach would have been to train an algorithm for each time, taking into account the fact that the model degrades with the passing of time. However, this would mean a severe stratification of the dataset (a reduction of more than a tenth of our original dataset), and these datasets are still small since LAM-EPS AEMET-γSREPS has been running one year and a half in the moment of writing this, and less time when the calculations were performed. Besides, from $H+06$ to $H+36$, the degradation of the model is still small, and it is not strange to see in the daily practice that, for instance, a forecast for $H+12$ is more accurate than a forecast for $H+09$. In the end, it all turned to be a practical decision: was the training good with this procedure or not? As the graphs show, the answer is that the training was good, so this approach was used. Perhaps in the future, with bigger datasets, as the LAM-EPS AEMET-γSREPS accumulates more data, such stratification could be done.

As it can be seen, the classical statistical and linear methods perform very well. The ridge method seems to stand out. Comparing with the green line (the closest point to the observation) it is possible to see there are improvements (in some cases high, in some cases not so high). For the cases of two airports (Vigo and Málaga), there is no clear improvement, but the ridge method in such case is similar to the model performance, so there is no spoiling either.

3.2. Calibration for the Wind speed. As with the temperature, the dataset was from November 14, 2016, to January 22, 2018. The main ideas implemented for the case of the temperature are also applied for the wind speed at 10 meters. However, some caveats need to be considered. First of all, as the LAM-EPS AEMET-γSREPS works in the Lambert conformal conic projection, the wind, a vector, needs to be adequately rotated in order to be compared with the wind from the observations, which it is in Cartesian coordinates. Some people have suggested that the differences between the Lambert conic conformal and the Cartesian coordinates, for a limited area model like the LAM-EPS AEMET-γSREPS, centred in the Iberian Peninsula, are very small and could be overlooked. That is true, i.e., the angles between the Cartesian grid and the Lambert grid were very small (the biggest angle was below 3 degrees). Besides, the ML methods are especially good subtracting systematic errors, as previously commented. However, the spirit of this work was to do as much feature engineering as possible (in other words, to reason as physically as possible), so the wind vectors were rotated from Lambert to Cartesian.

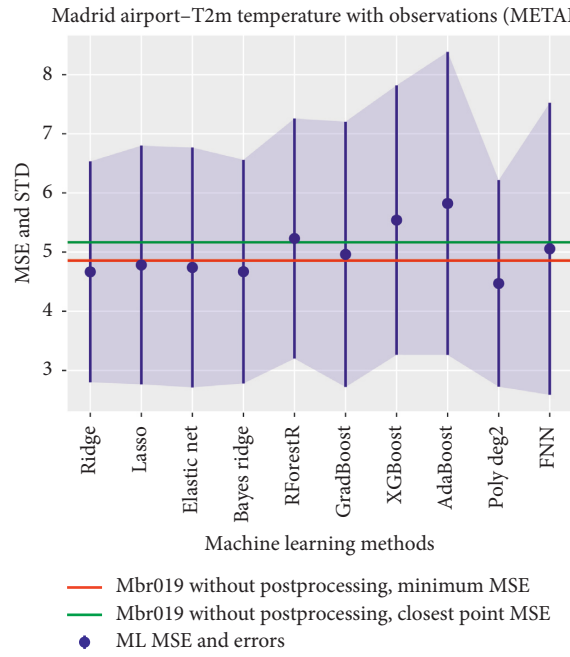

Figure 3: T2m calibration plot for Madrid airport. Horizontal axis, from left to right: ridge, lasso, elastic net, Bayesian ridge, random forest regression, gradient boosting regression, XGBoost regression, AdaBoost regression, polynomial of order 2, and feed-forward neural network.

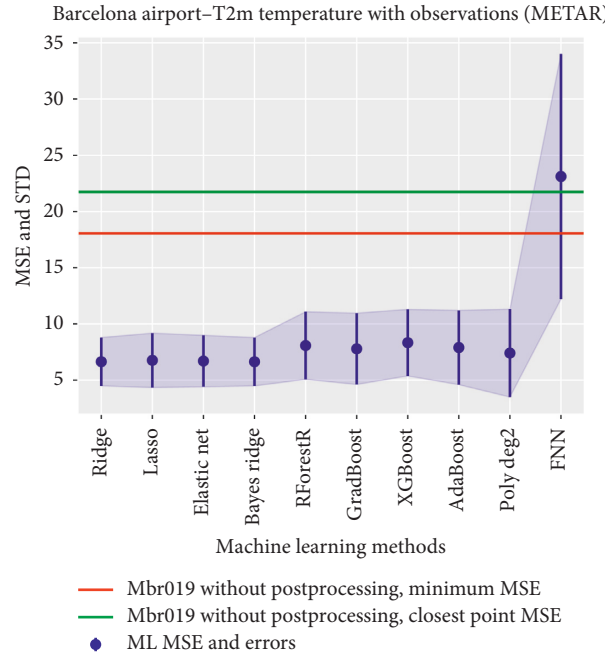

Figure 4: T2m calibration plot for Barcelona airport. Horizontal axis as in Figure 3.

The wind components u and v were extracted, the rotation carried on, and a *dataframe* with the *pandas* library

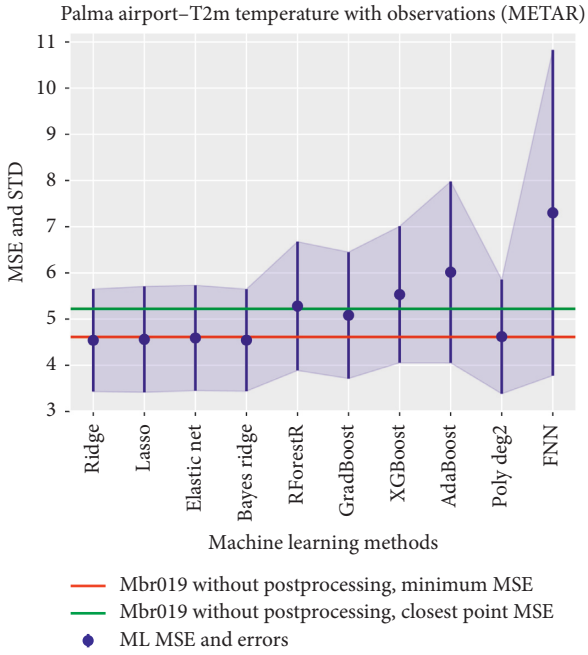

Figure 5: T2m calibration plot for Palma airport. Horizontal axis as in Figure 3.

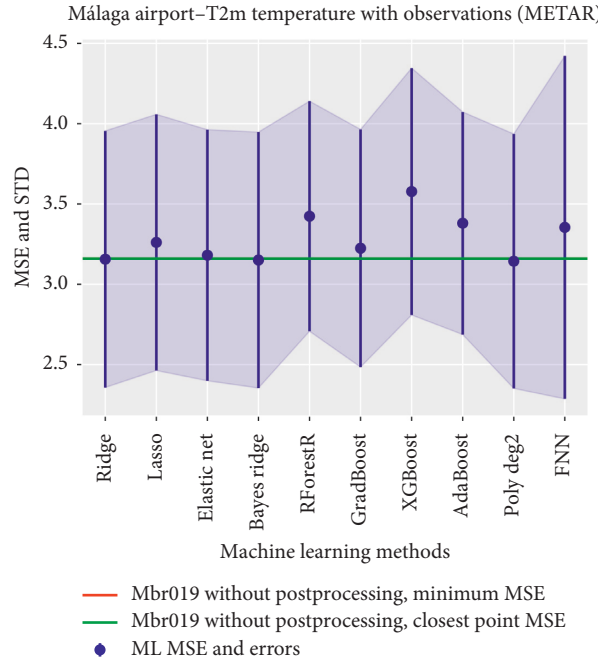

Figure 7: T2m calibration plot for Málaga airport. Horizontal axis as in Figure 3.

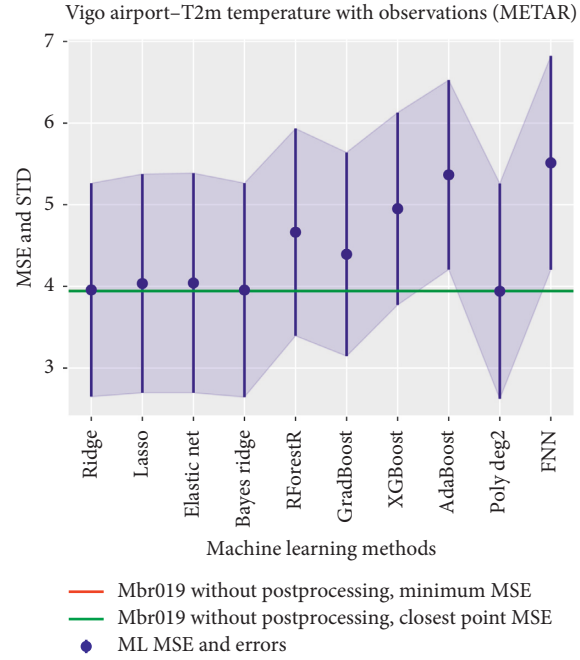

Figure 6: T2m calibration plot for Vigo airport. Horizontal axis as in Figure 3.

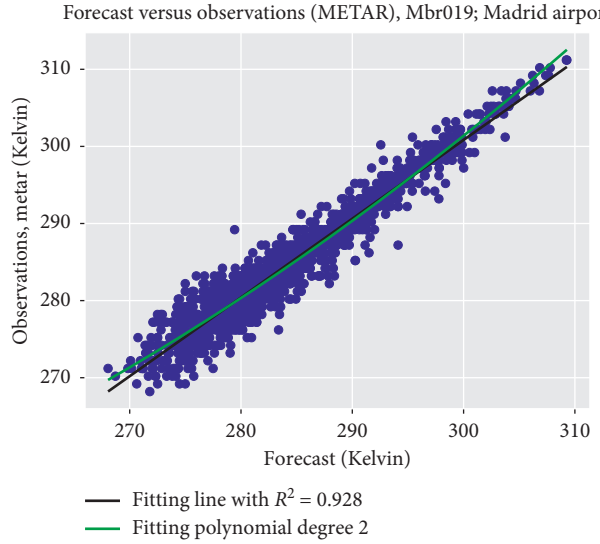

Figure 8: T2m scatter plot for Madrid airport, γSREPS member 019: LBCs JMA/GSM with the WRF-ARW NWP model.

from Python was created, as in the case of the temperature. The time range was from $H+06$ to $H+36$ each 3 hours. An stratification in the dataset (training an algorithm for each time step) was not done for the same reasons that, for the temperature, an stratified dataset would be a reduction of 1/11 of the size of the dataset, the limit of 36 hours was a relatively short limit for the degradation of the model, and, on a practical level, the approach chosen showed good results.

An important thing to comment is that the training was done for the wind speed only, that is, the scalar value, not for the wind vector with its magnitude and direction. The reason of this was purely a matter of choice. It was checked that when training for the wind as a vector, part of the learning went to learn the direction and part of the learning went to learn the magnitude. It was decided that a wind vector whose direction differs from the METAR in some degrees was not very relevant, but that a difference in a couple of knots (or m/

s) is more substantial and more helpful to the forecasters. So the modulus of the wind vector was calculated from the u and v components and the training performed with the magnitude of the wind.

It is also relevant to remark that the only quality control performed over the observations and the model was a basic quality control to delete the presence of gross outliers, in a similar spirit to the case of the temperature, that is, balancing out the necessity of avoiding gross outliers that would ruin the learning while at the same time penalizing the model for bad values. The threshold was put in 100 m/s. Unlike what happens with the temperature, it is quite difficult to perform a quality control over the wind that does not discard numerous valid measures. Also, unlike with the temperature, it is unrealistic to expect a regular pattern in the evolution of the wind that allows us to make valid comparisons between the values some hours before and later.

The 4 closest grid points to the observation point were searched, and a multivariable regression was performed. Not all the points were land points. For the case of the wind, measured at 10 meters, there is also a difference between land and sea, but it was thought that this difference was not as important as in the case of the temperature (with its strong diurnal and nocturnal cycle for land points) and that other factors were more important (type of terrain, for instance). As in the case of the temperature, it is possible to see in the example of Figure 9 that it is not the closest point the one who provides the best information (as measured by the R^2 coefficient). In this case, it is the farthest point. In the example of Figure 10, it is shown that using 4 neighbours adds extra quality to the regression, and this was the chosen procedure. The results are shown for the member 001, that is, the HARMONIE-AROME NWP model [17] with the boundary conditions of the ECMWF/IFS (http://www.ecmwf.int).

It is also shown the scatter plot (Figure 11) of the model with minimum MSE versus the observation. The fit is not as perfect as in the case of the temperature, as expected. Very low-scale phenomena like small changes in the terrain, buildings, or other obstacles can modify the wind measured considerably. As these are systematic errors, it is expected for the ML methods to deal very well with them.

For each MSE, the MSE plus the standard deviation is represented as the top of the bar and the MSE minus the standard deviation is represented as the lowest part of the bar, to give an idea of the range of variability. The red line is the model output without postprocessing for the minimum MSE of the 4 points. The green line is the MSE of the closest point to the observation, so it is really the point (or the line) to use to do the comparison between the model and the ML methods. As it is shown in the calibration graphs (Figures 12–16) there is an improvement of the forecasted wind speed with many ML methods. Like in the case of the temperature, ridge seems to offer a great improvement while being at the same time computationally acceptable for an operational environment.

3.3. Calibration for the Precipitation. As for the cases of the wind and the temperature, the same dataset was used, from

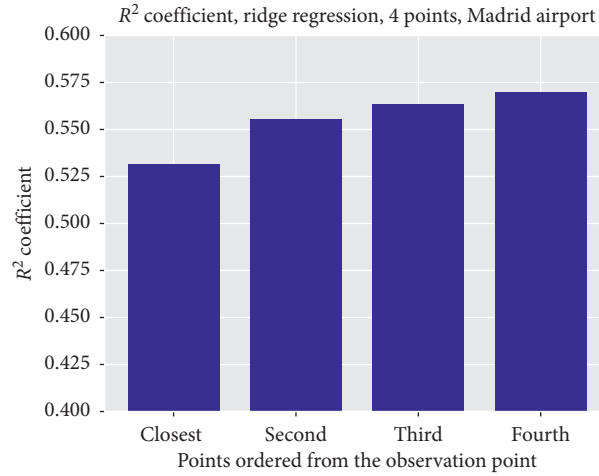

Figure 9: Quality of the ridge regression for the 4 closest points (wind speed).

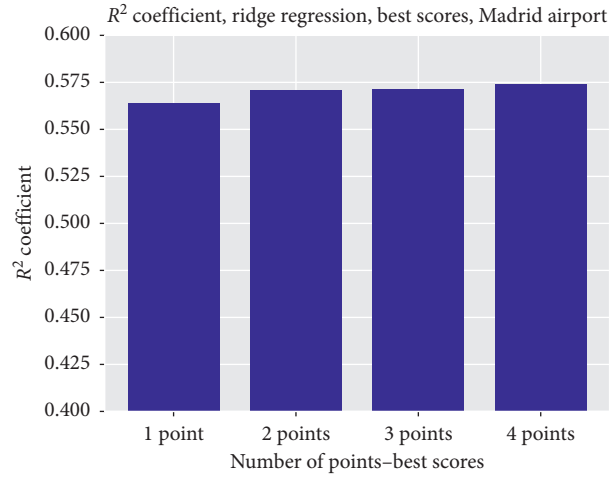

Figure 10: Quality of the ridge regression for different number of points (wind speed).

November 14, 2016, to January 22, 2018. Calibrating the precipitation is a very subtle issue. It is thoroughly known that precipitation does not follow a Gaussian distribution. It is also known that when calibrating, the precipitation is necessary to take into account that, besides the numerical quantities, the structure of the precipitation is also important. That is why the approach followed was different. Unlike with the cases of the wind and the temperature, the points used were the 12 closest neighbour points of the model, not the 4 closest ones. It was thought that, with this number of points, the high spatial uncertainty that affects the precipitation was taken into account. With this number of points, the features of a precipitation structure are collected and that at the same time, there is not a renounce to the high resolution properties of the LAM-EPS AEMET-γSREPS. Of course, other choices were possible, but, as before in this work, the approach chosen was the one that balanced out computational efficiency with physical insights. In the end, the structure under consideration was an irregular octagon

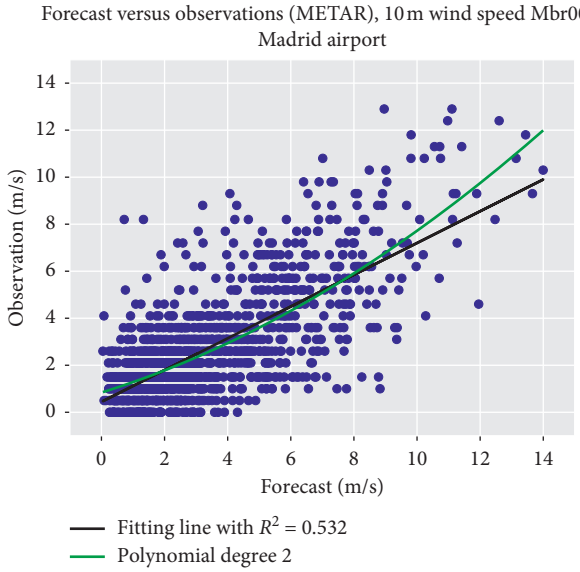

Figure 11: 10 meters total wind scatter plot for Madrid airport, LAM-EPS AEMET-γSREPS member 001: HARMONIE-AROME with LBC from ECMWF/IFS.

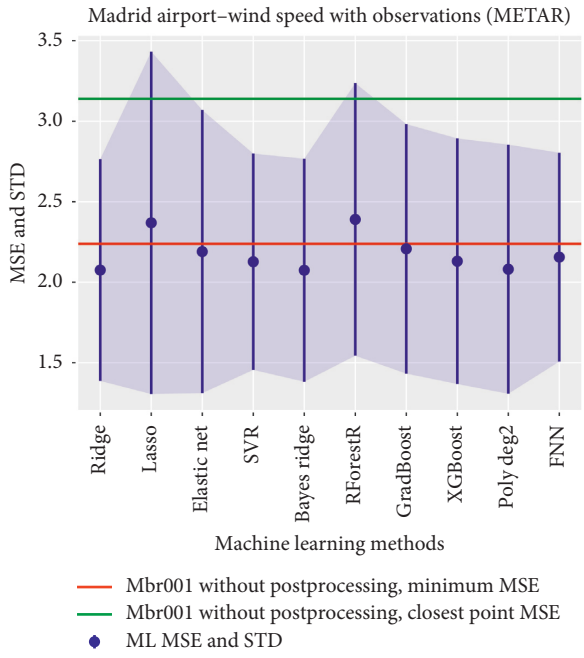

Figure 12: 10 meters total wind calibration plot for Madrid airport. Horizontal axis, from left to right: ridge, lasso, elastic net, singular vector regression, Bayesian ridge, random forest regression, gradient boosting regression, XGBoost regression, polynomial of order 2, and feedforward neural network.

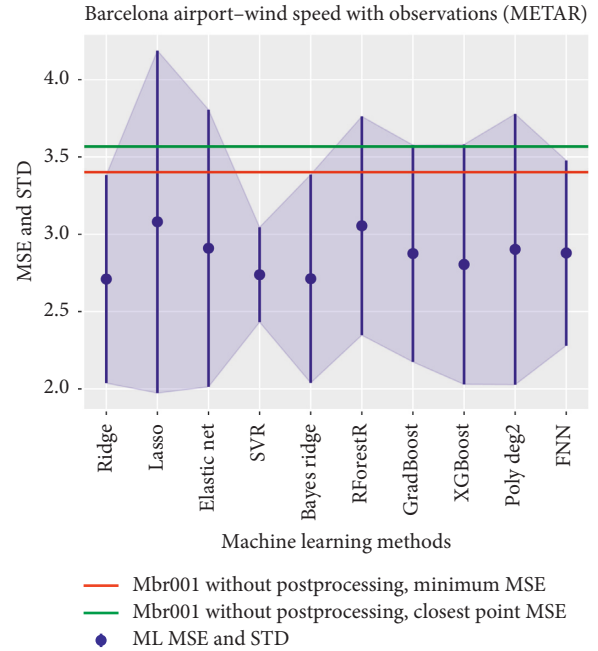

Figure 13: 10 meters total wind calibration plot for Barcelona airport. Horizontal axis as in Figure 12.

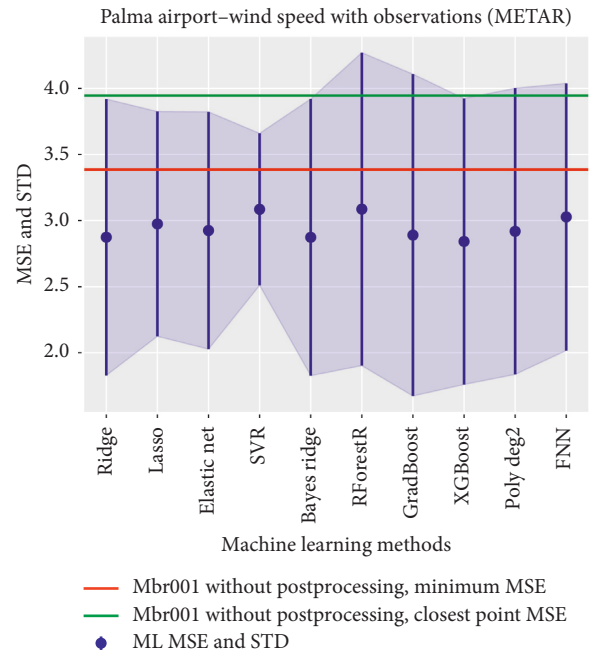

Figure 14: 10 meters total wind calibration plot for Palma airport. Horizontal axis as in Figure 12.

(8 points) plus the 4 points inside the octagon (Figure 17). The precipitation was calibrated for 24 hours, measured from 06 to 06 UTC in the model but from 07 to 07 UTC in the network of observations. Of course, it would be ideal to have model from 07 to 07, but sadly this is not the case. In a near future, the LAM-EPS AEMET-γSREPS will have outputs each hour, but, right now, with outputs every 3 hours, the 1 hour lag between the model and the observation is an unavoidable pitfall that there is no option but to assume.

The temperature at 2 meters and the u and v components of the wind at 10 meters were also added to improve the calibration. For these variables, it was decided to use the point closer to the observation, without distinguishing if the point was land or sea; this was done because, on the one

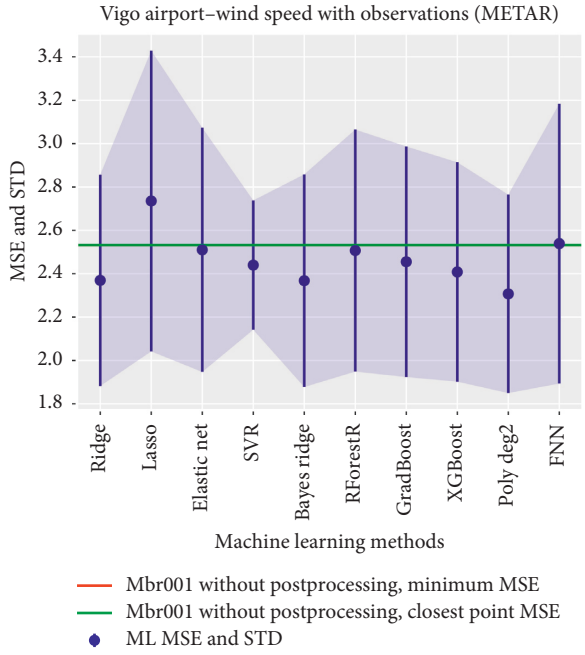

FIGURE 15: 10 meters total wind calibration plot for Vigo airport. Horizontal axis as in Figure 12.

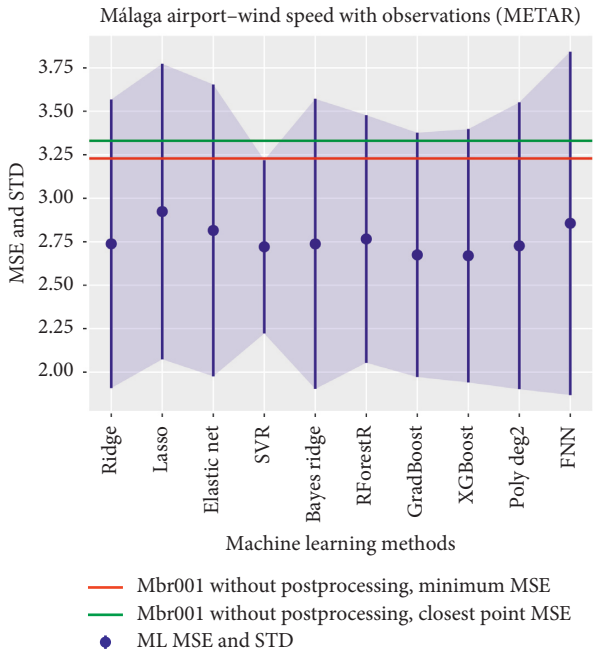

FIGURE 16: 10 meters total wind calibration plot for Málaga airport. Horizontal axis as in Figure 12.

hand, there was a high confidence that the ML methods can deal with systematic errors very well and, on the other hand, because these variables were just a help to improve the regression, they were extra information, not the desired outcome. A check was done for the 3 possible combinations: 12 points of precipitation plus the closest point of the temperature, 12 points of precipitation plus the closest point of the u and v components of the wind, and 12 points of

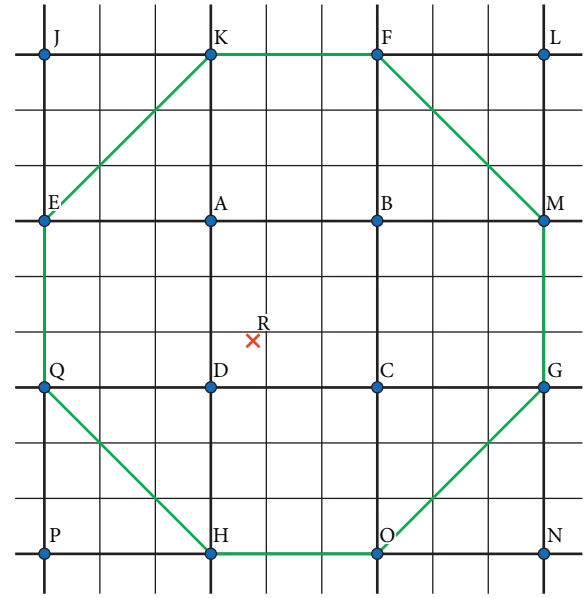

FIGURE 17: Irregular octagon representing the 12 points considered to calibrate the precipitation. The blue points are the grid points from the model, and the red point is the observation point.

precipitation plus both the closest points of temperature and u and v components of the wind field. In the end, it was decided to provide more variables for a better algorithm, and the last option was chosen.

As in the case of the wind field, the quality control was to discard gross outliers (if any) in both the model and the observations. In the case of the observations, quality controls are done before incorporating any data to the Spanish climatological database. For the model and as in the case of the wind speed, it is possible to discard only gross outliers that are clear indication that something was wrong when computing or storing the data; these are outliers due to mechanical or operational issues, not related to the model design and performance. Except for gross outliers, bad values from the model were included and would be a penalization in the training. The level was put in 2000 millimetres in 24 hours for the precipitation and in 100 m/s in wind speed and ±80 degrees in temperature, as before. As a safety check, rows which had negative values were deleted: this can happen when transforming data (for instance, in the Spanish climatological database, values are stored as tenths of millimetres and for this work, they were converted to millimetres); this phenomenon is called *underflow* in the computer science literature.

When dealing with this type of regression, the possibility of standardizing the dataset was considered. *Standardization* is a procedure where for each independent x variable in the regression equation (for each predictor), the mean is calculated, $\overline{x}$, and its standard deviation, σ, and then the operation $(x_i - \overline{x}_i)/\sigma_i$ is carried out, where i runs through all the samples in the dataset. A good point about standardization is that all the variables have the same relative weight when doing the regression; this is very nice since in this work there are variables that are in the scale 200–300

(temperature), variables distributed around 0 (u and v components of the wind field), and positive variables with a great spectrum of variation (precipitation). Another good point of the standardization is that some algorithms work in the range of normally distributed values or with uniform distributed values between 0 and 1. This is especially true in the case of the FNN (feedforward neural network). In the cases of the wind and the temperature, reasonably good results were achieved without the necessity of standardization, and some experiments with standardization changed the performance of the algorithms but not the substantial results, so it was decided not to use standardization for the wind and the temperature.

In the graphs, for each MSE, the MSE plus the standard deviation is represented as the top of the bar and the MSE minus the standard deviation is represented as the lowest part of the bar, to give an idea of the range of variability. The red line is the model output without postprocessing for the minimum MSE of the 12 points. The green line is the MSE of the closest point to the observation, so it is really the point (or the line) that should be used to do the comparison between the model and the ML methods. The results are shown for the precipitation without standardization (Figures 18–22) and with standardization (Figures 23–27) for member mbr010. Mbr010 is the Harmonie-ALARO NWP model with the boundary conditions from the ARPÈGE model [18] by Météo-France (http://www.meteofrance.fr).

Note that sometimes the blue bars that denote the standard deviation have negative values. Of course, this does not mean that the MSE is a negative magnitude. It is simply a reflection of the fact that the cross-validation technique has showed a wide variability of our MSE. The MSE varies a lot depending on what slice is the validation set and what slices are the training set. Blue bars are by definition symmetric around the average of the MSE, that is, the top of a bar is the average of the MSE plus the standard deviation and the lowest part of a bar is the MSE minus the standard deviation. So, bars in the negative values are really MSEs with a great positive value.

As it is possible to see, precipitation is a very subtle variable to calibrate. For the case of the precipitation, each point has its peculiarities in an even stronger way than with the wind speed or the temperature. What it is possible to say is that standardization helps (however, perhaps not always). For the precipitation, the most sophisticated methods such as singular vector regression and neural networks begin to show their strength although still reasonable results are achieved with ridge.

4. Conclusions

As it has been shown, ML methods are a great tool for the calibration of meteorological models. Classical linear regression, with the added help of regularization, works very well for the temperature and the wind speed. In the case of the precipitation, there is no preferred method and things seem to depend on the point and on the nature of the dataset, something that is not surprising, because it is known that there is not a universally valid ML method, valid for all the datasets [19]. For the precipitation, standardization of the dataset can be helpful, and with respect to the methods, neural networks offer a good alternative although other methods such as lasso, elastic net, or ridge have performances that could be close to those of neural networks but

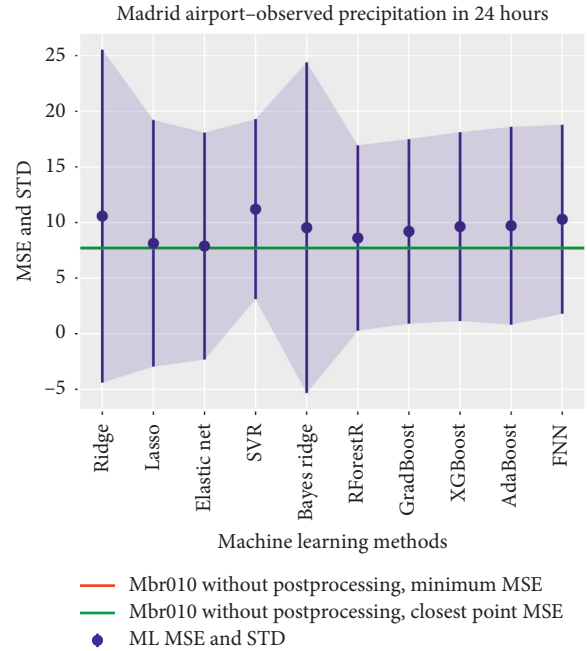

Figure 18: Precipitation in 24 hours, calibration plot for Madrid airport. Horizontal axis, from left to right: ridge, lasso, elastic net, singular vector regression, Bayesian ridge, random forest regression, gradient boosting regression, XGBoost regression, AdaBoost regression, and feedforward neural network.

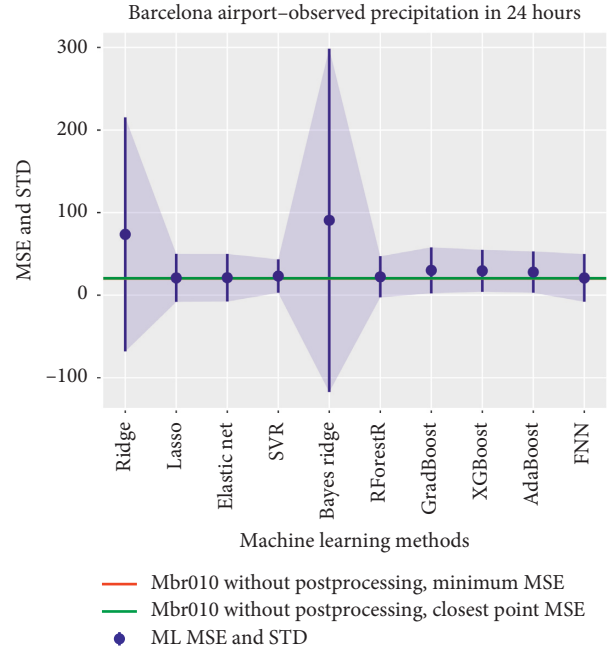

Figure 19: Precipitation in 24 hours, calibration plot for Barcelona airport. Horizontal axis as in Figure 18.

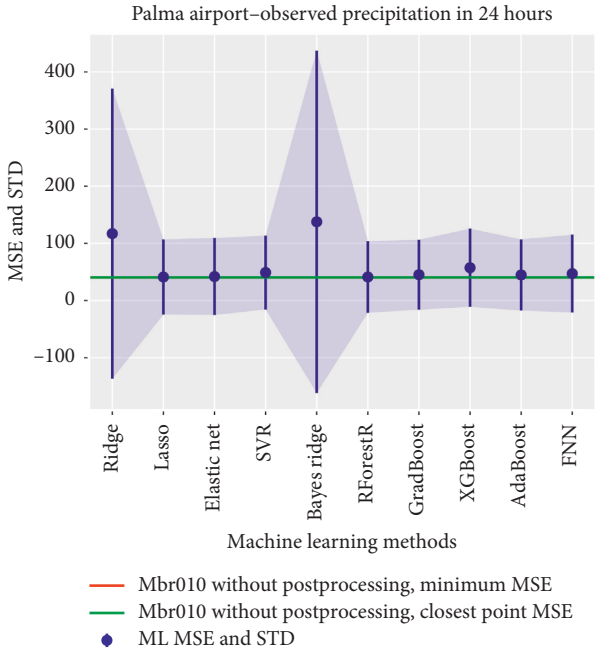

Figure 20: Precipitation in 24 hours, calibration plot for Palma airport. Horizontal axis as in Figure 18.

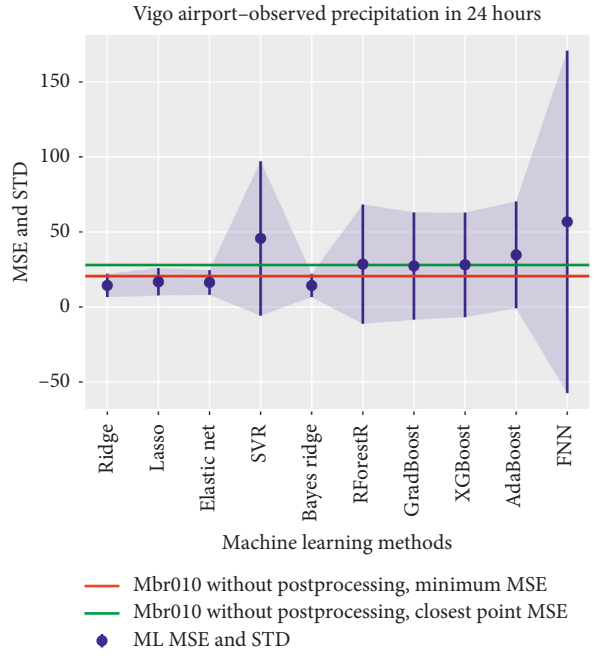

Figure 21: Precipitation in 24 hours, calibration plot for Vigo airport. Horizontal axis as in Figure 18.

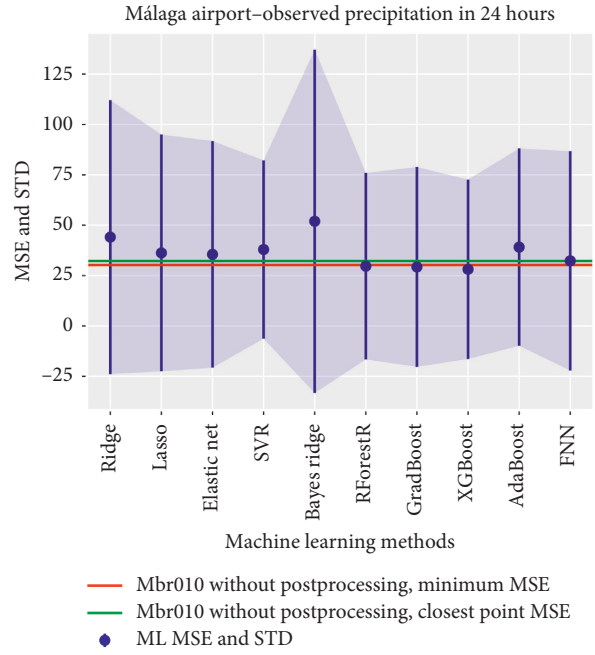

Figure 22: Precipitation in 24 hours, calibration plot for Málaga airport. Horizontal axis as in Figure 18.

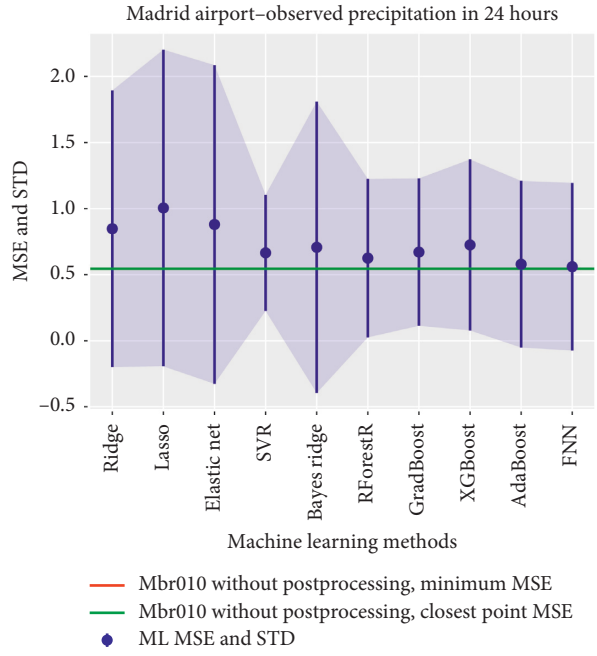

Figure 23: Precipitation in 24 hours, standardized calibration plot for Madrid airport. Horizontal axis as in Figure 18.

offering considerable easier training. In an operational environment, for each member of the ensemble and for each point, a training will have to be performed and, after cross validation, the best method will be chosen. The methods mentioned (ridge, FNN, etc.) do not have to be always valid, and each dataset has its own method. We can make guesses about what method will be the best based on physical and statistical considerations, but in the end, only once the calibration is applied can we decide.

It is legitimate to ask oneself what these ML methods are really doing (at least, what they are *probably* doing since there are still open questions about how ML methods work). It is important to differentiate between wind speed and temperature on the one side and precipitation on the other

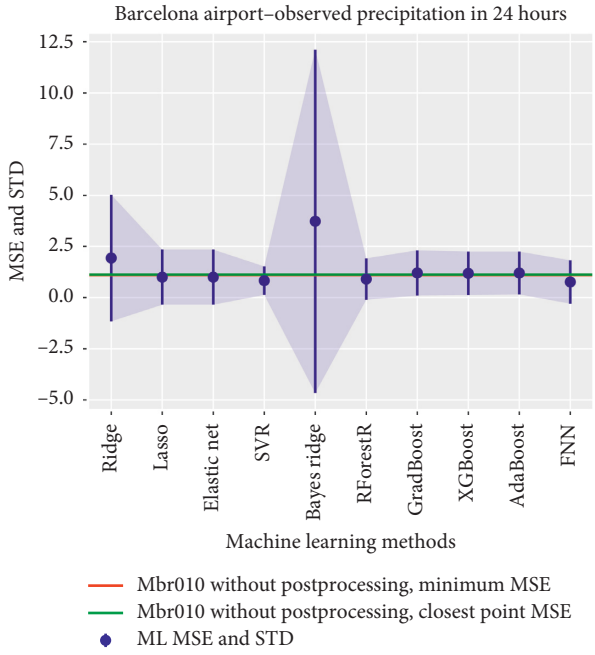

Figure 24: Precipitation in 24 hours, standardized calibration plot for Barcelona airport. Horizontal axis as in Figure 18.

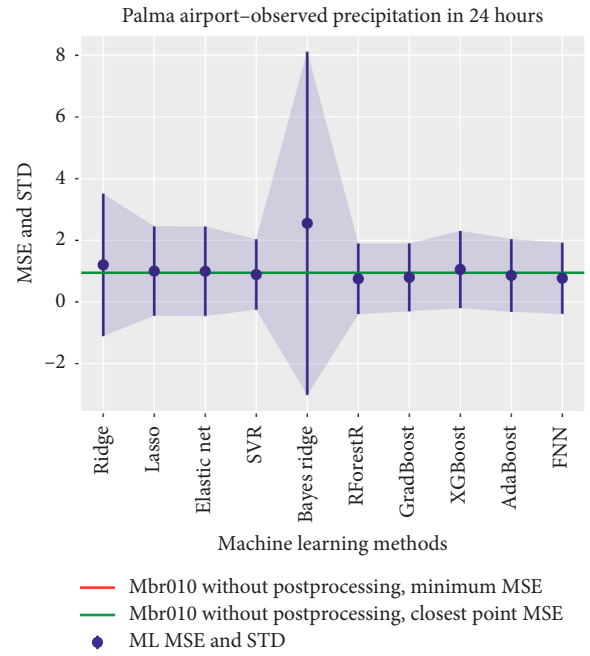

Figure 25: Precipitation in 24 hours, standardized calibration plot for Palma airport. Horizontal axis as in Figure 18.

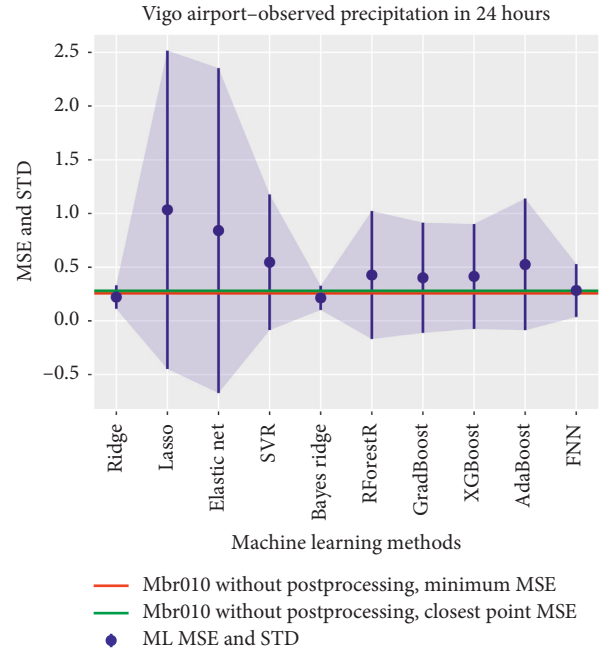

Figure 26: Precipitation in 24 hours, standardized calibration plot for Vigo airport. Horizontal axis as in Figure 18.

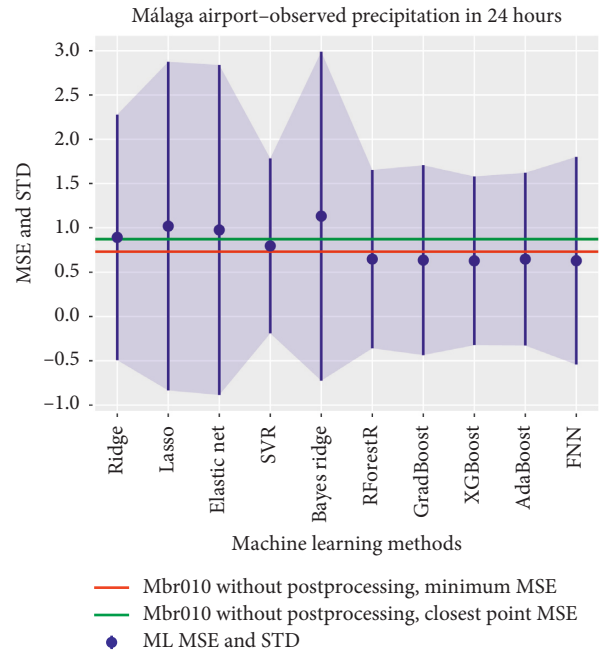

Figure 27: Precipitation in 24 hours, standardized calibration plot for Málaga airport. Horizontal axis as in Figure 18.

side. For the wind speed and the temperature, ML methods are probably correcting the biases (systematic errors) that the models have for typical patterns of synoptic situations (and probably even for mesoscale situations). For the case of the precipitation, the errors have many different origins, due to not only biases and systematic errors but also due to all the approximations taken into account to model the precipitation, like the cloud microphysics scheme used or the parametrization of the onset of convection; in this case, ML methods are managing greater complexity and uncertainty.

Why do some methods perform better than others? In most of the occasions, when doing ML, it is hardly known a priori which method will be the right one. It is proof and error what finally determines what method has the best performance. However, and from purely physical considerations, for the wind speed and the temperature, the

success of relatively simple methods like ridge, elastic net, lasso, or Bayesian ridge, which are basically extensions of a linear regression, is probably linked to the facts mentioned in the previous paragraph: the correction of mainly systematic errors due to relatively few and controlled sources of error for these variables. In the case of the precipitation, with all the uncertainties and complexities involved, more sophisticated methods like the FNN, that are capable of discerning more subtle signals in the data, begin to give better results. FNN and the rest of sophisticated methods are harder to train, with a tendency to overfit among other subtleties; these methods are not geared to relatively better determined problems like the forecast of the wind speed or the temperature.

It is important to remark that the calibration goes well when the ML methods deal with values that are in the range of the minimum or maximum values in the dataset, in other words, values that are in the range of what the algorithm has "seen." When a calibrated algorithm faces a value that is outside the trained range, anything can happen. Depending on their nature, some algorithms will perform a linear extrapolation and others could fit the value to some complex, high-order polynomial curve. To avoid this behaviour, it is possible to establish a flag or similar warning advice to deactivate the algorithm for such a value, letting the direct (uncalibrated) output of the model to be the definitive value. At least the extreme value is incorporated to the dataset and it will be part of a future training process.

With respect to the calibration with ML, there are many lines of research that can be explored in the future. It is possible to dive deeper in the realm of ML methods, searching for instance how *deep learning* (neural networks with many layers) performs: an interesting method could be the recurrent neural network, for example, perhaps deep learning could serve to improve the results of the precipitation. It is possible to think in extending these calibration methods from points to surfaces, following some kinds of classification in function of the type of terrain, weather, or climate. Or one could use algorithms that offer probabilistic outputs (FNN, for instance) to calibrate the ensemble directly instead of member by member. There is no doubt that this is an interesting topic to delve in.

Data Availability

Huge amounts of data have been used for this work, and parts of them could be released (although we cannot guarantee it) if needed by contacting the corresponding author via dquinterop@aemet.es.

Disclosure

A previous version of this article appeared in a Spanish book about different strategies regarding weather forecasting. It was a summary of what it has been shown here, and entire sections were omitted, like the analysis of the precipitation. The authors did not earn any amount of money with the publication of the book.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

The authors thank the Spanish weather service, AEMET, for its funding and support, both the headquarters office and the local office of the Canary Islands. They also thank the ECMWF, the Canadian CMC, the French Météo-France, the Japanese JMA, and the North American NOAA for their kindness providing the boundary conditions for the LAM-EPS-AEMET-γSREPS, and also they thank the North-American NOAA, NCEP, NCAR, NWS, and related communities of WRF-ARW and NMMB models and the Harmonie community. The authors also thank all the team of the LAM-EPS AEMET-γSREPS ensemble, for its kind support and help in many topics, especially to José Antonio García-Moya Zapata: he has been very kind, showing the path when David Quintero Plaza was a total beginner; more than a team leader he has been a mentor. Special thanks go to José Luis Casado Rubio, for his design of a very great software library. We also want to thank Álvaro Subías Díaz-Blanco and Alfons Callado Pallarès for their help and useful comments. We thank the community of the data science and Machine Learning in Python for developing really great tools.

References

[1] J. Pathak, Z. Lu, E. Girvan, B. R. Hunt, and E. Ott, "Using machine learning to replicate chaotic attractors and calculate Lyapunov exponents from data," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 27, no. 12, article 121102, 2017.

[2] H. R. Glahn and D. A. Lowry, "The use of model output statistics (MOS) in objective weather forecasting," *Journal of Applied Meteorology*, vol. 11, no. 8, pp. 1203–1211, 1972.

[3] W. H. Klein, B. M. Lewis, and I. Enger, "Objective prediction of five-day mean temperatures during winter," *Journal of Meteorology*, vol. 16, no. 6, pp. 672–682, 1959.

[4] J. A. Hoeting, D. Madigan, A. E. Raftery et al., "Bayesian model averaging," *Proceedings of the AAAI Workshop on Integrating Multiple Learned Models*, vol. 335, pp. 77–83, Madison, WI, USA, July 1998.

[5] J. A. García-Moya, A. Callado, C. Santos et al., "Multi-model ensemble for short-range predictability," in *Proceedings of 3rd International Verification Methods Workshop*, ECMWF, Reading, UK, January-February 2007.

[6] E. Kalnay and D. Hou, "Objective verification of the SAMEX'98 ensemble forecasts," *Monthly Weather Review*, vol. 129, no. 1, pp. 73–91, 2001.

[7] S. Xingjian, Z. Chen, H. Wang, and D.-Y. Yeung, "Convolutional LSTM network: a machine learning approach for precipitation nowcasting," *Advances in Neural Information Processing Systems*, pp. 802–810, 2015.

[8] J. Olsson, C. B. Uvo, K. Kawamura et al., "Neural networks for rainfall forecasting by atmospheric downscaling," *Journal of Hydrologic Engineering*, vol. 9, no. 1, pp. 1–12, 2004.

[9] A. Weichert and G. Bürger, "Linear versus nonlinear techniques in downscaling," *Climate Research*, vol. 10, no. 2, pp. 83–93, 1998.

[10] R. M. Trigo and J. P. Palutikof, "Precipitation scenarios over Iberia: a comparison between direct GCM output and different downscaling techniques," *Journal of Climate*, vol. 14, no. 23, pp. 4422–4446, 2001.
[11] B. Hewitson and R. Crane, "Climate downscaling: techniques and application," *Climate Research*, vol. 7, no. 2, pp. 85–95, 1996.
[12] L. Wassermann, "Statistics versus machine learning," in *Normal Deviate*, , 2012.
[13] F. Dyson, "A meeting with Enrico fermi," *Nature*, vol. 427, no. 6972, 2004.
[14] J. Friedman, T. Hastie, and R. Tibshirani, "The elements of statistical learning: data mining, inference, and prediction," in *Springer Series in Statistics*, , 2nd edition, 2017.
[15] C. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Oxford, UK, 1995.
[16] W. C. Skamarock, J. B. Klemp, J. Dudhia et al., *A Description of the Advanced Research WRF Version 3, NCAR Technical Note, Mesoscale and Microscale Meteorology Division*, National Center for Atmospheric Research, Boulder, CO, USA, 2008.
[17] L. Bengtsson, U. Andrae, Y. Batrak et al., "The HARMONIE-AROME model configuration in the ALADIN-HIRLAM NWP system," *Monthly Weather Review*, vol. 145, no. 5, pp. 1919–1935, 2017.
[18] P. Termonia, C. Fischer, F. Bouyssel et al., "The ALADIN System and its canonical model configurations AROME CY41T1 and ALARO CY40T1," *Geoscientific Model Development*, vol. 11, no. 1, pp. 257–281, 2018.
[19] D. Whitley and J. P. Watson, "Complexity theory and the no free lunch theorem," in *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*, pp. 317–339, 2005.