Using Random forest and Gradient boosting trees to improve wave forecast at a specific location

Aurélien Callens, Denis Morichon, Stéphane Abadie, Matthias Delpey, Benoit Liquet

Using Random forest and Gradient boosting trees to improve wave forecast at a specific location

Aurélien Callens[a,*], Denis Morichon[b], Stéphane Abadie[b], Matthias Delpey[c], Benoit Liquet[a,d]

[a] *Université de Pau et des Pays de l'Adour, E2S UPPA, CNRS, LMAP, Pau, France*
[b] *Université de Pau et des Pays de l'Adour, E2S UPPA, SIAME, Anglet, France*
[c] *Centre Rivages Pro Tech SUEZ EAU FRANCE, Bidart, France*
[d] *Department of Mathematics and Statistics, Macquarie University, Sydney, Australia*

Abstract

The main objective is to present alternative algorithms to neural networks when improving sea state forecast by numerical models considering main spectral bulk parameters at a specific location, namely significant wave height, peak wave period and peak wave direction. The two alternatives are random forest and gradient boosting trees. To our knowledge, they have never been used for error prediction method. Therefore, their performances are compared with the performances of the usual choice in the literature: neural networks. We showed that the RMSE of the variables updated with gradient boosting trees and random forest are respectively 20 and 10% lower than the RMSE obtained with neural networks. A secondary objective is to show how to tune the hyperparameter values of machine learning algorithms with Bayesian Optimization. This step is essential when using machine learning algorithms and can improve the results significantly. Indeed, after a fine hyperparameter tuning with Bayesian optimization, gradient boosting trees yielded RMSE values in average 8% to 11% lower for the correction of significant wave height and peak wave period. Lastly, the potential benefits of such corrections in real life application are investigated by computing the extreme wave run-up ($R_{2\%}$) at the study site (Biarritz, France) using the data corrected by the different algorithms. Here again, the corrections made by random forest and gradient boosting trees provide better results than the corrections made by neural networks.

Keywords: Artificial neural networks, Data assimilation, Error prediction, Gradient boosting trees, Random forest, Wave forecasting.

*Corresponding author
Email address: aurelien.callens@univ-pau.fr (Aurélien Callens)

1. Introduction

Nowadays, numerical wave models are routinely used to forecast wind generated waves. Although they provide satisfactory predictions at a regional scale and during mean wave conditions, it has been shown that they are less accurate for forecasting at a specific location (Londhe et al., 2016) and have a tendency to underestimate wave height during energetic wave conditions. This underestimation has been observed in different state-of-the-art wave models: see the work of Arnoux et al. (2018) for Wind Wave Model II (WWMII) and WAVEWATCH–III (WW3) ; Rakha et al. (2007) for WAve Model (WAM) and Moeini et al. (2012) for the Simulating WAves Nearshore model (SWAN). The errors in wave predictions are mainly due to inaccuracies in the wind input that forces the model. The winds used as forcing are numerically simulated and are known to underestimate high wind speeds (Moeini et al., 2012). This results in the underestimation of wave parameters by numerical wave models. Simplifying assumptions, approximations employed in the modeling process, discretization of the domain and a potentially wrong parametrization of the model can also be sources of inaccuracies in wave model predictions (Babovic et al., 2001, 2005).

When observation data are available, data assimilation can be used to improve the predictions made by numerical models. There are 4 main categories of data assimilation procedures (Refsgaard, 1997; Babovic et al., 2001): updating the input parameters, updating the state variables, updating the model parameters and finally updating the output parameters. The last procedure is called ”Error prediction” method and is the most suitable approach to improve model predictions of different output variables at a specific location (Babovic et al., 2005). This procedure presents several advantages comparing to the other data assimilation procedures. First, it covers inaccuracies coming from all sources because it improves directly output variables. In addition, it can use a combination of external variables such as meteorological or wind data to increase the accuracy of the predictions. Lastly, it is easy to implement because it consists in only three steps and does not require multiple runs of numerical wave model. First, the deviations between the modeled values and measured values are computed. Then, ma-

chine learning algorithms are used to forecast these deviations. Finally the deviations predicted by the algorithms are incorporated to the predictions of the numerical model for the next time steps, resulting in a more accurate wave forecast.

This method has been successfully applied on hindcast data (Makarynskyy et al., 2005; Deshmukh et al., 2016) and has even been implemented in real time setting in the works of Babovic et al. (2001) and Londhe et al. (2016). To our knowledge, only artificial neural networks have been tested to forecast the errors in the data assimilation. However, according to the so-called "No Free Lunch" theorem, there is no single model that works best for all problems (Wolpert, 2002). It is therefore necessary to try multiple models and find the one that works best for our particular problem. The performance of artificial neural networks must be compared with other algorithms in the data assimilation task. Random forest and gradient boosting trees are strong candidates for this comparison. Indeed, these two methods are known for their performance and unlike neural networks, they also provide valuable information by computing the predictive power of each variable used as input. The predictive power or variable importance refers to how much a model relies on that variable to make accurate predictions. A variable with high predictive power means that its values have a significant impact on the prediction values. By contrast, a variable with low predictive power have a limited impact on the prediction values and it can be substracted from the model to make it simpler and faster.

To explore the performance of random forest and gradient boosting trees, we use as a test case the Basque coast (South west of France). Every winter, the basque coast faces numerous coastal flooding events. To prevent and mitigate the risk of flooding, wave forecast are used to compute the extreme run-up values either by using parametric models such as the formula of Stockdon et al. (2006) or process based models such as Xbeach (Vousdoukas et al., 2012; de Santiago et al., 2017). In both cases, the accuracy of this forecast is of utmost importance as the issuing of the early warning depends on it, especially during energetic wave climate where coastal flooding risk is the highest. In this study, we employ the error prediction method with the different machine learning algorithms and use local meteorological conditions and measured wave parameters from a local buoy to improve the wave forecast. Lastly, we investigate the potential

benefits of using such corrections in the computation of extreme run-up values.

This study aims to present two alternatives (random forest and gradient boosting trees) to neural networks by comparing their performances when improving regional numerical models. A secondary objective is to show how to tune the hyperparameter values of machine learning algorithms with Bayesian Optimization. In machine learning, a hyperparameter is a parameter whose value is specified by the user before the learning process begins, it will affect how well a model trains and therefore it will have a non negligible impact on the final results. Bayesian optimization is an efficient hyperparameter optimization algorithm and it is widely used to optimize the results of any given machine learning method.

Lastly, we investigate if the error prediction method makes a difference in a real application such as the computation of extreme run-up for the beach of Biarritz. Section 2 will introduce the study area, the data and all the statistical methods used. Results will be presented and discussed in Section 3. Finally, Section 4 will cover the conclusion.

2. Data and Methods

2.1. Study site and Data

The Basque coast is a 150 km long rocky coast facing the Bay of Biscay (Figure 1). Every winter, it is battered by numerous storm events. This results in frequent and sometimes intense coastal flooding which can severely damage seafront infrastructures. The city of Biarritz is particularly affected as the buildings and infrastructures are located right behind a sea wall that is located at the top of the beach. The damages associated with coastal flooding are costly for nearshore cities which try to prevent and mitigate the risks by developing early warning systems. Such systems rely on the knowledge of the sea state and its forecast.

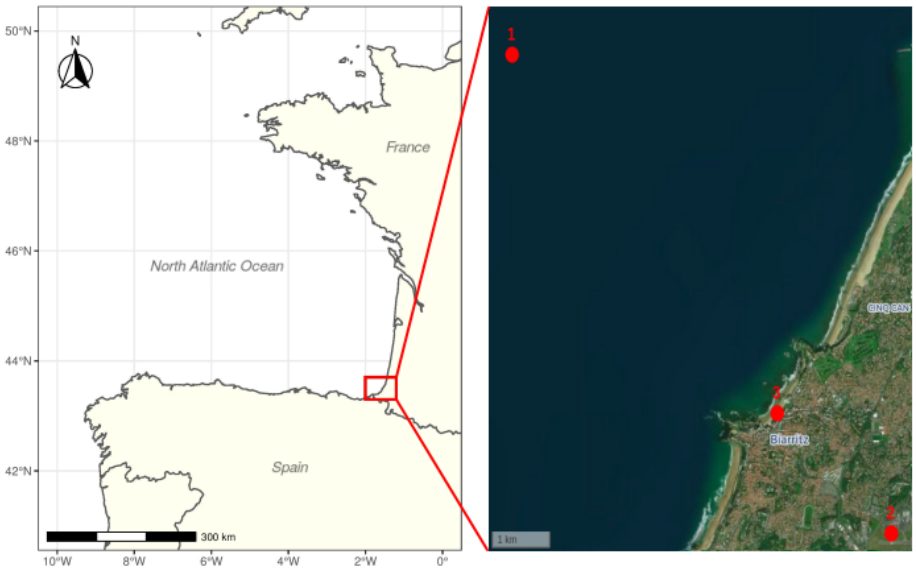


Figure 1: Map showing the location of the study site. The red dots show of the locations of the directional wave buoy (1), the meteorological station (2) and the beach called "Grande Plage de Biarritz" (3).

This work focuses on the forecast improvement of three wave integrated parameters which describe the sea state in this area: the significant wave height (H_S), the peak period (T_p) and the peak wave direction (θ_p). Direct measurements of these parameters are obtained from the National Center for Archiving Swell Measurements (L'her et al., 1999). They were made by a directional wave rider buoy (DWR MKIII) operated by the Centre for Studies and Expertise on Risks, Environment, Mobility, and Urban and Country Planning (CEREMA) and the University of Pau and Pays de l'Adour (UPPA). The buoy is located a few miles off the Basque Coast (Figure 1) at 50 meters water depth. Since its deployment in 2009, this buoy have been recording the parameters of interest every 30 minutes. The measuring range of this buoy is [-20m; 20m] for heave motion, [1.6s; 30s] for wave period and [0°; 360°] for wave direction. It has a resolution of 1 cm in heave motion and a directional resolution of 1.5°. To be consistent with the numerical wave data and meteorological data, a 1 hour time step was adopted for the buoy data.

The three parameters simulated at the buoy coordinates by the Meteo-France WAM

model were provided by the Copernicus Marine Environment Monitoring Service. This reanalysis ("ibi_reanalysis_wav_005_006") covers the period 2007-2019 with a hourly time-step. The MFWAM model is derived from the third generation wave model WAM (Group, 1988). It is forced by wind fields obtained from a regional numerical weather prediction model (AROME). A more complete description of the MFWAM model can be found in Lefèvre and Aouf (2012).

Meteorological data, including average wind speed above 10 meters, wind direction and atmospheric pressure were furnished by the French national meteorological service MeteoFrance. The data were collected hourly by the meteorological station of the Biarritz airport, located only a few kilometers from the study site (Figure 1). It covers the period ranging from 2013-01-01 to 2018-12-31. By assembling the wave buoy data, the wind wave parameters and the meteorological data we obtain a dataset of 41439 hourly observations ranging from 2013-01-01 to 2018-12-31.

In this work, we are improving the wave forecast by correcting the systematic errors of the wind wave model. Therefore, we are not considering any temporal effects while improving H_S, T_p and θ_p. The dataset was randomly divided into 2 parts: the training part containing 70% of the observations (n = 28797) and the testing part containing the remaining 30% (n = 12342).

2.2. Error prediction method

The error prediction method consists in three steps:

- Step 1: Deviations between model predictions and measured values are computed:

$$E_{model} = X_{measured} - X_{modeled},$$

  where E_{model} is the error of the model, $X_{measured}$ is the measured value of an output variable provided by the wave buoy and $X_{modeled}$ is the value of the same variable computed by the wave model.

- Step 2: E_{model} is predicted with an appropriate supervised machine learning algorithm.

- Step 3: The predicted error is added to the prediction of the wave model to obtain an updated numerical prediction:

$$X_{updated} = X_{modeled} + E_{predicted},$$

where $X_{updated}$ is the updated prediction of wave model and $E_{predicted}$ is the predicted error given by the supervised learning method.

This method is repeated separately for each output variable to improve (H_s, T_p, θ_p). The performance of this data assimilation method relies on two things: the quantity of data and the machine learning algorithm used. Since the machine learning algorithm are generally more suited to interpolate rather than extrapolate, the available data for learning process should cover as much as possible the range of all the probable events in the study area. Concerning the learning method, only neural networks have been used for the step (2) of the error prediction method to our knowledge(Makarynskyy et al., 2005; Moeini et al., 2012; Londhe et al., 2016). Because we want to compare the performance between different machine learning algorithms, we use random forest and gradient boosting trees. All the tested algorithms use the same input variables to improve the model accuracy: the three wave parameters (H_S, T_p, θ_p) given by the numerical model, the atmospheric pressure, the wind direction and speed.

2.3. Neural networks

Artificial neural networks have been extensively used in the domain of wave modelling (Deo et al., 2001; Makarynskyy et al., 2002; Makarynskyy, 2005; Mandal and Prabaharan, 2006) or wave parameters assimilation (Makarynskyy et al., 2005; Moeini et al., 2012; Londhe et al., 2016). It is why technical details will be avoided in this study and only the general concepts will be presented. The readers can find more details and information on the working of neural networks in Liang and Bose (1996) or Friedman et al. (2001).

The most common class of neural networks is the multilayer perceptron. The neurons in this network are organized in three layers: the input layer that receive the input variables, the output layer that performs the final predictions and between these two

layers there is the hidden layer. Neurons in the hidden layer transmit the signal to the output layer by transforming the weighted sum of the neurons present in the input layer with a non linear function called activation function. The weights between each neuron of the network are adjusted through the iterative process of backpropagation to minimize the error between the variable we want to predict and the variable predicted by the network (output layer).

As other machine learning methods, hyperparameters need to be specified before the training of neural networks. Some hyperparameters control the network architecture (number of neurons, layers, activation function used, etc...) while others control the training process (learning rate, bacth size, number of epochs, etc...). Hyperparameters must be tuned carefully in order to achieve optimal results with neural networks.

2.4. Tree based algorithms

Unlike neural networks, random forest and gradient boosting have never been used in the error prediction method. They are state-of-the-art ensemble learning techniques for classification and regression tasks. An ensemble learning technique commonly refers to a method that combines the predictions from multiple machine learning algorithms, called base learners, to produce more accurate predictions.

Random forest is an algorithm that builds many decision trees in parallel. These trees are the base learners for random forest and they have the following characteristics:

- Each tree is built using a different bootstrap sample of the data-set. This mechanism is called bagging.
- At each node, a given number (hereafter ”mtry”) of variables are randomly sampled as candidates at each split. The best split point is then selected within this random set of variables. This process is called feature sampling. The value ”mtry” is fixed before growing the forest.
- Unlike the classification and regression trees of Breiman et al. (1984), the trees in random forest are fully grown (no pruning step).

Bagging and feature sampling are the core principles of random forest. They are two randomizing mechanisms which ensure that the trees are independent and are less

correlated with each other. The final prediction of a random forest is obtained by averaging the results of all the independent trees in case of regression or using the majority rule in case of classification.

The most important hyperparameters in random forest are the number of trees and "mtry": the number of variables randomly sampled as candidates at each split when building the trees.

Gradient boosting is an algorithm that trains many weak learners sequentially to provide a more accurate estimate of the response variable. A weak learner is a machine learning model that perform slightly better than chance. In case of gradient boosting trees, the weak learners are shallow decision trees. Each new tree added to the ensemble model (combination of all the previous trees) minimizes the loss function associated with the ensemble model. The loss function depends on the type of the task performed and can be chosen by the user. For regression, the standard choice is the squared loss. By adding sequentially trees that minimize the loss function (i.e. follow the gradient of the overall loss function), the overall prediction error decreases. Technical details about gradient boosting trees can be found in (Friedman, 2001).

Many hyperparameters have to be tuned for gradient boosting trees, some of them control the gradient boosting process, such as the learning rate, the number of trees to be used whereas others regulate the construction process of the trees: minimal node size, sample of the dataset to be used, maximum depth.

2.5. Hyperparameter tuning

Hyperparameters influence significantly the training of the machine learning algorithms and therefore the quality of their predictions. The objective of hyperparameter tuning is to find the values of hyperparameters that yield the lowest error (RMSE in our case) for unseen data. Two types of methods exist to find the optimal values of hyperparameters: uninformed or informed.

In uninformed methods, many combinations of hyperparameter values are tested one after the other and the best combination is the one that yields the lowest error on unseen data. The values of hyperparameters are either sampled randomly (random search) or sampled along a grid (grid search). In both cases, each combination tested

are independent from another. With grid and random search, it is not guaranteed to find the optimal set of hyperparameters and it usually requires a lot of iterations (combinations tested).

In informed methods, the results obtained by the past combinations are used to choose the next combination to evaluate. Bayesian optimization algorithm is an informed method that aims to minimize an objective function, in our case the errors of the machine learning algorithms on unseen data. First, it builds a probability model (Gaussian process) of the objective function. Then it uses this surrogate model to select the most promising values of hyperparameters to evaluate. Once the promising combination of values have been evaluated, the probability model is updated and searched again for the most promising combination. This process is repeated several times. This method is employed in this article because it is very efficient for tuning hyperparameter values and it usually requires less iterations than uninformed methods (Bergstra et al., 2011). In-depth details of this method are given in the works of Snoek et al. (2012); Marchant and Ramos (2012) and Shahriari et al. (2015).

2.6. Training the algorithms

The machine learning algorithms described above are trained to predict the deviations of H_s, T_p or θ_p (one model for each variable), using 6 input variables: the three wave parameters (H_S, T_p, θ_p) given by the numerical model, the atmospheric pressure, the wind direction and speed.

The neural networks are built and trained with the R package **keras**. The input variables are centered and scaled to improve the result of neural networks and the weights are updated with the adam optimization algorithm (Kingma and Ba, 2014). Random forest and gradient boosting model are fitted in R using respectively the **ranger** package which provide fast implementation of Random Forests (suited for high dimensional data) and the **xgboost** package which is an efficient R implementation of the gradient boosting framework from Chen and Guestrin (2016). The input variables are not centered or scaled before the training of random forest and gradient forest because it does not influence the training of these algorithms.

The training is done twice: once with the default values of the hyperparameters in

the R packages and once with the optimal values found with the Bayesian optimization method.

The best hyperparameter values are found by the means of Bayesian optimization method coupled with a 5-fold cross validation in the training dataset. That is, the training data are split into five equal-sized partitions and a machine learning model is recursively built on four partitions (80% of the training data) with a given hyperparameter combination. A performance metric, in our case the root mean square error is assessed on the remaining partition (20% of the training data). The resulting five performance metrics are averaged to provide an estimated out-of-sample performance of the respective hyperparameter combination. The objective function to minimize for the Bayesian optimization method is the average out-of-sample performance value. The Bayesian optimization for our data is performed using the R package **RBayesianOptimization**. First, random combinations of hyperparameter values are evaluated to serve as search base for the informed method (5 in this study), then an acquisition function (upper confidence bound) is used to find the next combination values to evaluate (this step is repeated 25 times).

3. Results and Discussion

3.1. Model comparison

To assess the accuracy of the numerical model and the proposed corrections, several metrics are computed including the root mean square error (RMSE), the correlation coefficient, the bias and the scatter index (SI). The bias represents the average error between the observed and modeled data and allows one to detect under or over estimation of the value of one parameter. The scatter index is a measure of the error normalized by the observation values. It is a standard metric for wave model inter-comparison (Londhe et al., 2016). More details about the computation of these two metrics can be found in the work of Mentaschi et al. (2013). The metrics are computed twice: once with the whole test data and once with a subset of the test data where $H_s > 3m$ because the underestimation of H_s is known to become larger above this height (Arnoux et al., 2018).

Table 1: Statistical metrics for the three variables of interest before the hyperparameter tuning. "Ann" stands for artificial neural networks, "Rf" for random forest and "Gb" for gradient boosting tree.

	Hs				Tp				θ_p			
	Numerical model	Ann Corr.	Rf Corr.	Gb Corr.	Numerical model	Ann Corr.	Rf Corr.	Gb Corr.	Numerical model	Ann Corr.	Rf Corr.	Gb Corr.
					Computed with all data							
Biais	-0.201	0.005	-0.002	-0.004	0.712	0.002	-0.026	0.005	-0.249	0.698	0.976	0.765
RMSE	0.399	0.306	0.248	0.267	1.839	1.603	1.282	1.388	13.803	12.391	9.749	10.645
SI	0.166	0.148	0.120	0.129	0.153	0.145	0.116	0.125	0.045	0.041	0.032	0.035
Cor	0.954	0.962	0.975	0.972	0.78	0.804	0.880	0.857	0.330	0.366	0.455	0.386
					Computed with data where $H_s > 3m$							
Biais	-0.536	-0.156	-0.124	-0.126	0.683	-0.026	-0.090	-0.041	1.925	-0.561	0.052	0.046
RMSE	0.766	0.515	0.420	0.433	1.348	1.083	0.956	1.022	7.351	5.769	4.449	4.931
SI	0.133	0.120	0.098	0.101	0.089	0.083	0.073	0.078	0.024	0.019	0.015	0.016
Cor	0.818	0.857	0.902	0.898	0.832	0.850	0.886	0.869	0.589	0.604	0.790	0.726

Table 1 presents the metrics obtained with no assimilation (numerical model) and after a preliminary data assimilation with the three machine learning algorithms. The term "preliminary" refers to the lack of hyperparameter tuning. The learning has been done using the default hyperparameter values given in table 2.

For significant wave height, the numerical model shows a negative bias. This indicates that the MFWAM model has a tendency to underestimate H_s such as other wind wave models (Moeini et al., 2012; Arnoux et al., 2018). The negative bias increases as the value of H_s becomes larger ($H_s > 3m$), meaning that H_s is more likely to be underestimated during energetic events. For the peak period, the numerical model shows a positive bias. When $H_s > 3m$, the bias and the RMSE for this parameter are smaller. The predictions of T_p are therefore better during energetic conditions. For wave direction, a small bias is observed in average and is greater when the waves are larger. The large difference in RMSE computed with data where $H_s > 3m$ and with all data is explained by the distribution of the wave direction according to the wave height. When the significant wave height is below 2 meters, the wave direction at the buoy is more variable (Figure S1, supplementary material) and the spectral wave model has more difficulties to predict correctly the direction. This can be confirmed by looking at the θ_p errors of the numerical model: we see that they are larger and occur more often

when $H_s < 2m$ (Figure S2, supplementary material). A potential explanation of this phenomenon could be that below 2 meters, the sea state is more likely to be influenced by local wind conditions which are difficult to reproduce by the spectral wave model (Rascle and Ardhuin, 2013). When the significant wave height is above 2 meters, the wave directions are a lot less variable and the predictions of the spectral wave model are more accurate.

When we look at the metrics computed with all data, we see that the correction made by the three machine learning algorithms removes the bias and greatly reduces the RMSE and the scatter index for H_s and T_p. For θ_p, the mean bias is slightly larger after data assimilation for all algorithms. The correction of the machine learning algorithm could be less efficient for θ_p due to the high variability of the observed deviations they try to model (see the explanation in the paragraph above). However, lower value of RMSE and scatter index and larger correlation coefficients still indicate that the corrected data are closer to the observed values at the buoy.

For the metrics computed with data where $H_s > 3m$, the correction does not remove the bias for H_s and T_p but reduces it greatly. For the wave direction, the updated parameters are closer to the reality. Indeed, bias and RMSE obtained by the corrections are smaller than the numerical model and the correlation coefficients are larger for corrected data.

For this preliminary assimilation, random forest yields the best results for all the parameters. It reduces the RMSE values computed with all test data by 37.7%, 30% and 29% respectively for H_s, T_p and θ_p. Gradient boosting trees is close second and decreases the RMSE values by 33%, 24.5% and 22.8%. Finally, data assimilation with neural networks decreases the RMSE of H_s, T_p and θ_p by 23%, 12.8% and 10.2%.

As stated earlier, the performance of machine learning algorithms might depend on the choice of the hyperparameter values. The Bayesian optimization was therefore performed and optimal values were selected (Table 2). The selected hyperparameter values are quite different from the default values. Indeed, for neural networks, the best results were obtained with more epochs and more neurons in the hidden layer. For random forest, only the number of trees seems to have some effect on the results and models with a large number of trees performs better. Finally, for gradient boosting

Table 2: Default values, ranges and selected value of hyperparameters for the machine learning algorithms

Machine learning algorithms	Hyperparameters	Default value	Range searched	Selected value for Hs	Selected value for Tp	Selected value for Dir
Neural networks	No. of units in hidden layer	13 ($2 \times h + 1$)	{1-40}	26	20	40
	Activation function	sigmoid	{relu,sigmoid,tanh}	sigmoid	sigmoid	relu
	Learning rate	0.001	{0.0001-0.1}	0.021	0.016	0.005
	Epochs	30	{10,30,50,100,150}	50	100	150
	Batch size	32	{16,32,64,128}	32	64	64
Gradient Boosting	Number of trees	100	{100-2000}	560	1150	1990
trees	Learning rate	0.3	{0.0001-0.3}	0.072	0.028	0.069
	Max depth	6	{1-20}	14	20	20
	Minimal node size	1	{1-15}	7	1	1
	Subsample	1	{0.5-1}	0.57	0.82	0.79
	Col sample	1	{0.5-1}	0.99	0.85	0.9
Random forest	Number of trees	500	{100,200,500,800,1000}	1000	1000	1000
	Mtry	2 ($\sqrt{h}$)	{2-6}	2	2	2

Note: h corresponds to the number of input variables (6 in our case).

trees, models with a large number of trees and a small learning rate are preferred.

Metrics calculated with data corrected by the tuned machine learning algorithms are presented in Table 3. Overall, tuning the hyperparameter values has improved the results of all the algorithms. However, the degree of improvement differs depending on the algorithm. We observe the smallest improvements for random forest where the RMSE of every parameters seems to decrease by less than 1% in average. For neural networks, tuning hyperparameter values has a more significant effect by reducing the RMSE by 2 to 3% in average. The largest effect of tuning the hyperparameters are observed with gradient boosting trees. The RMSE is 8 to 11% lower for every parameter. The only exception is θ_p computed with all data where we have a small increase (2%) of RMSE. In general, the mean bias for H_s, T_p and θ_p remains the same before and after hyperparameter tuning expect for the bias of H_s computed when $H_s > 3m$ which is significantly lower after the tuning.

For this dataset, gradient boosting algorithm shows the best performances for all parameters. Assimilation with this algorithm decreases the RMSE values computed with all test data by 39.8% for H_s, 33% for T_p and 31% for θ_p. For H_s and θ_p, the reduction are even lower for the RMSE values computed with $H_s > 3m$: 47% for

Table 3: Statistical metrics for the three variables of interest after the hyperparameter tuning."Ann" stands for artificial neural networks, "Rf" for random forest and "Gb" for gradient boosting tree.

	Hs				Tp				θ_p			
	Numerical model	Ann Corr.	Rf Corr.	Gb Corr.	Numerical model	Ann Corr.	Rf Corr.	Gb Corr.	Numerical model	Ann Corr.	Rf Corr.	Gb Corr.
					Computed with all data							
Biais	-0.201	0.026	-0.002	-0.001	0.712	0.007	-0.022	0.003	-0.249	0.790	0.979	0.714
RMSE	0.399	0.300	0.246	0.240	1.839	1.553	1.269	1.231	13.803	12.07	9.646	9.501
SI	0.166	0.144	0.118	0.116	0.153	0.140	0.114	0.111	0.045	0.04	0.032	0.031
Cor	0.954	0.964	0.976	0.977	0.78	.817	0.882	0.889	0.330	0.36	0.461	0.421
					Computed with data where H_s > 3m							
Biais	-0.536	-0.117	-0.120	-0.099	0.683	0.032	-0.084	-0.051	1.925	-1.114	0.062	0.056
RMSE	0.766	0.495	0.417	0.404	1.348	1.064	0.950	0.943	7.351	5.820	4.412	4.365
SI	0.133	0.117	0.097	0.095	0.089	0.081	0.072	0.072	0.024	0.019	0.015	0.015
Cor	0.818	0.861	0.903	0.908	0.832	0.856	0.888	0.889	0.589	0.609	0.793	0.793

the significant wave height and 40% for wave direction. The performances of random forest for H_s, T_p and θ_p are slightly better than the results obtained before tuning the hyperparameters: respectively 38.3%, 30.9%, 30.1%. The performances are also better for neural networks after hyperparameter tuning: it decreases the RMSE values by 24.8% for H_s, 15.5% for T_p and 12.5% for θ_p. The differences in efficiency between neural networks and ensemble learning techniques could be explained by the architecture chosen for the neural networks. Indeed, this work shows the results for multilayer perceptrons with only one hidden layer which is the typical choice in the literature (Londhe et al., 2016; Moeini et al., 2012). By choosing an architecture with more hidden layers, the networks might be able to model more complex phenomena and bring a better improvement for the three wave parameters.

The distribution of the errors after the different corrections are presented in the figure 2. For all wave parameters, the distributions of the errors after a correction have narrowed and are now more centered in zero. The differences in performance between algorithms are confirmed with these violin plots. Indeed, when the correction is made with random forest or gradient boosting trees, the distributions of the errors are more narrow than the distributions of the errors obtained with neural networks. The difference in efficiency between random forest and gradient boosting trees is not dis-

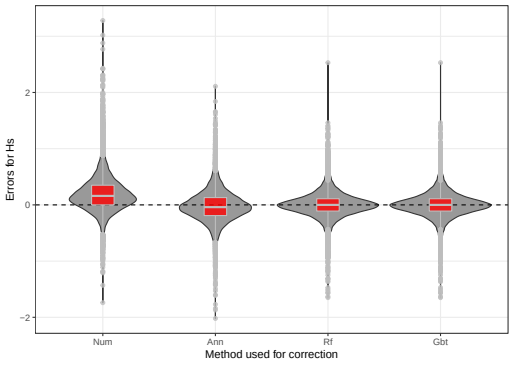
tinguishable graphically. It is expected as the metrics of the two algorithms only differ by a few percents. For H_s and T_p, the corrections have also removed the bias observed for numerical model. The large errors of θ_p for the numerical model (Figure **??**) are observed when $H_s < 2m$ and are not corrected by the machine learning algorithms. Figures showing the observed values versus the corrected values are available in the supplementary material for the three wave parameters.

3.2. Predictive power of the input variables

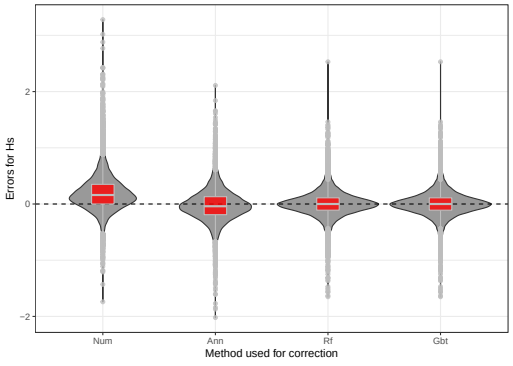
In addition to their performance, random forest and gradient boosting algorithms can provide a measure of importance for each variable used as input. This importance indicates the predictive power of the variable. It can be used to sort variable from most to least predictive, allowing one to have more insight on the problem and to perform feature selection when there are too many input variables. The figure 3 shows the importance measure of each variable computed by the random forest depending on the parameter to improve. For H_s and T_p, the most important variables are the value of H_s and T_p modeled by the wind wave model. It is different for the direction where the most important variables are the value of θ_p and H_s given by the model. The predictive power of local meteorological variables is quite low, suggesting that local and instantaneous meteorological variables does not bring valuable information in the assimilation process. The wind wave formation process is not instantaneous and occurs in large regional scale, therefore using meteorological variables from the past (several days before) and from different locations (located in the ocean) could lead to a better predictive power which means better updated wave predictions.

3.3. Example of application

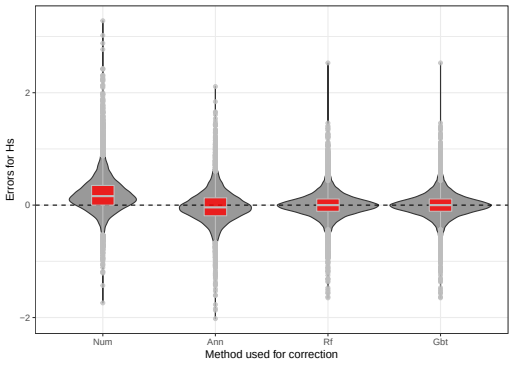
To investigate the potential effect of the different corrections in a real case scenario, the extreme wave run-up $R_{2\%}$ at the Grande Plage de Biarritz has been computed for the test period with the Stockdon formula (Stockdon et al., 2006) which uses H_s and T_p and the beach slope as parameters. The beach slope is fixed to 8% according to the work of Morichon et al. (2018). Using the extreme wave run-up calculated with the buoy data as reference, the metrics presented previously have been computed for the numerical



(a) H_s correction



(b) T_p correction



(c) θ_p correction

Figure 2: Distribution of the errors computed between values observed at the buoy and values corrected or not with the different machine learning algorithms. ”Num” stands for numerical model (no correction), ”Ann” for artificial neural networks, ”Rf” for random forest and ”Gb” for gradient boosting trees. The horizontal lines in the red boxplots represent from top to bottom: the third quartile, the median and the first quartile.

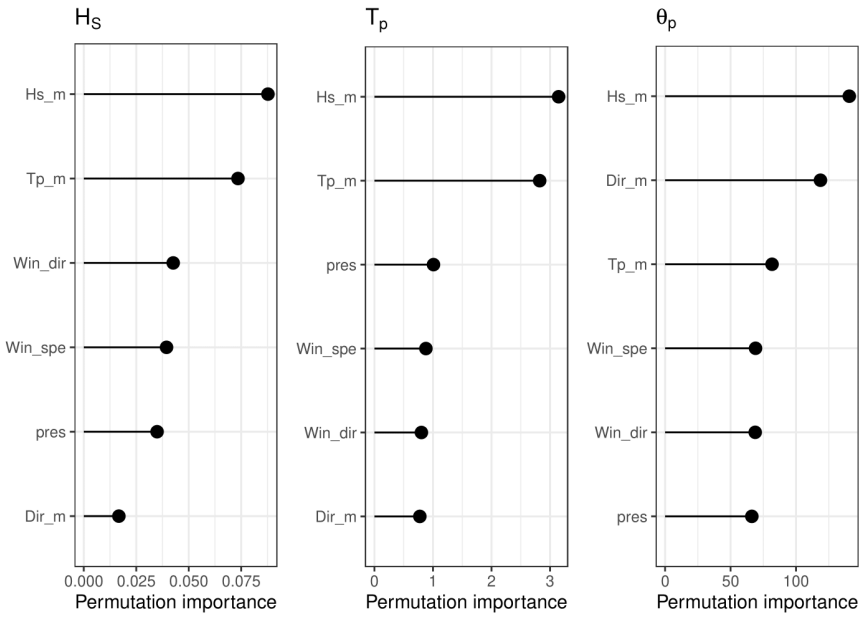


Figure 3: Variable importance for the correction of the three wave integrated parameters.

model and the different corrections (Table 4). From this table, it is evident that the data corrected with machine learning algorithms provide wave run up values that are closer to the "real" values with lower RMSE, Scatter index and greater correlation coefficient. Although the bias remains, the correction made by the gradient boosting tree algorithm decreases the RMSE of the extreme wave run-up by 22% (for all data and data where $H_s > 3m$). Random forest shows almost the same reduction of RMSE values: 21.5% for all data and 20.7% for data where $H_s > 3m$. The correction obtained by neural networks is less efficient: it reduces the RMSE computed with all data and data where $H_s > 3m$ by 6.2 and 9.9% respectively.

4. Conclusion

In this work, random forest and gradient boosting trees were employed for the first time in the error prediction method. These ensemble learning techniques based on decision trees performed better than neural networks for improving the wave forecast of the Basque Coast. The correction made by gradient boosting trees yielded the best

Table 4: Statistical metrics of the R2% calculated with Stockdon's formula. These results are obtained by taking the R2% computed with buoy data as reference. "Ann" stands for artificial neural networks, "Rf" for random forest and "Gb" for gradient boosting tree.

	Numerical model	Ann Corrected	Rf Corrected	Gb Corrected
		Computed with all data		
Biais	0.003	0.019	0.002	0.003
RMSE	0.223	0.209	0.175	0.172
SI	0.145	0.136	0.114	0.112
Cor	0.943	0.950	0.965	0.966
		Computed with data where Hs > 3m		
Biais	-0.042	-0.030	-0.054	-0.042
RMSE	0.313	0.282	0.248	0.242
SI	0.119	0.108	0.093	0.092
Cor	0.854	0.862	0.897	0.901

results for all the wave parameters: it reduced the RMSE values by nearly 40% for H_s, 33% for T_p and 31% for θ_p. The reduction of RMSE values for random forest was only a few percents lower than gradient boosting trees. The corrections made by neural networks were significant but yielded reductions in RMSE not as high as the two ensemble learning techniques: 24.8% for H_s, 15.5% for T_p and 12.5% for θ_p.

As expected, tuning the hyperparameters of the machine learning algorithms had a positive effect on the final results. However, the effect of the tuning differed depending on the algorithms. Indeed, random forest was less affected as it only reduced the RMSE values by 1% in average. The tuning had more effect on neural networks reducing the RMSE values by 2 to 3%. Gradient boosting tree algorithm was the most affected by hyperparameter tuning as the results were improved by 8 to 11% in average. One of the main advantage of random forest over gradient boosting trees is that it doesn't need this tuning step in order to yield great results. This is not negligible as hyperparameter tuning step can be time consuming and computationally demanding depending on the complexity of the search (number of hyperparameters).

Contrary to neural networks, Random forest and Gradient boosting trees provided valuable insights by giving the predictive power of each input variable. The predictive power of variable brings interpretability to the model and can give a better understanding of the variable we try to predict. For example, we know that the significant wave height modelled by the numerical wave model was the most important variable in the correction of the three parameters. In cases where there are a lot of input variables, knowing their associated predictive power helps developing more parsimonious models by keeping the pertinent variables and subtracting the less informative ones from the model.

The error prediction method has proven to be useful in improving wave forecast. This had an impact in a real life application by improving the accuracy of the extreme run-up computed at the Grande Plage de Biarritz. Here again the corrections brought by random forest and gradient boosting tree were better than the correction made by neural networks. The decrease in RMSE values was around 22% for the two ensemble techniques and 6.2% for the neural networks. Even though the differences in performance might not appear significant, it can make a difference when using these corrections in

an early warning system. It is especially true when dealing with storm events where H_s and T_p are large.

The differences between machine learning algorithms observed in this article are specific to Biarritz site. The results might differ for another study site. Therefore, we can only advise to test and compare several machine learning algorithms to find the optimal one associated with the site of interest.

Finally, the assimilation made in this study did not account for the temporal aspect in the errors of the numerical model, it only corrected systematic errors of the wave model. In the future, this work could be extended by adding input variables containing temporal aspect. This could be the values of a modeled parameter at previous time steps such as the work of Londhe et al. (2016). In this framework, neural networks could perform better as they are known to handle efficiently time series. Other input variables could be also used to improve the wave forecast such as the meteorological data from the past or at different locations. Because the success of the error prediction method depends on the quantity of data, it would be also interesting to perform a sensitivity analysis on the quantity of data used in the training process. This could give us some insights on the minimal quantity of data required to obtain a desirable assimilation procedure.

Acknowledgments

Funding was provided by the Energy Environment Solutions (E2S-UPPA) consortium and the BIGCEES project from E2S-UPPA (”Big model and Big data in Computational Ecology and Environmental Sciences”). The authors would like to thank the French national meteorological service ”MeteoFrance” and Copernicus Marine Environment Monitoring Service for providing data.

Reproducibility

Meteorological data used in this article are private and can not be provided by the authors. However, the R code to perform the analysis and an example of data

assimilation on wave forecast data (used in operational) are provided in this Github repository.

References

Arnoux, F., Abadie, S., Bertin, X., Kojadinovic, I., 2018. A database to study storm impact statistics along the Basque Coast. Journal of Coastal Research 85, 806–810.

Babovic, V., Cañizares, R., Jensen, H.R., Klinting, A., 2001. Neural networks as routine for error updating of numerical models. Journal of Hydraulic Engineering 127, 181–193.

Babovic, V., Sannasiraj, S.A., Chan, E.S., 2005. Error correction of a predictive ocean wave model using local model approximation. Journal of Marine Systems 53, 1–17.

Bergstra, J.S., Bardenet, R., Bengio, Y., Kégl, B., 2011. Algorithms for hyper-parameter optimization , 2546–2554.

Breiman, L., Friedman, J., Olshen, R.A., Stone, C.J., 1984. Classification and regression trees Chapman & Hall. New York .

Chen, T., Guestrin, C., 2016. Xgboost: A scalable tree boosting system, in: Proceedings of the 22nd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining, ACM. pp. 785–794.

de Santiago, I., Morichon, D., Abadie, S., Reniers, A.J., Liria, P., 2017. A comparative study of models to predict storm impact on beaches. Natural Hazards 87, 843–865.

Deo, M.C., Jha, A., Chaphekar, A.S., Ravikant, K., 2001. Neural networks for wave forecasting. Ocean engineering 28, 889–898.

Deshmukh, A.N., Deo, M.C., Bhaskaran, P.K., Nair, T.B., Sandhya, K.G., 2016. Neural-network-based data assimilation to improve numerical ocean wave forecast. IEEE Journal of Oceanic Engineering 41, 944–953.

Friedman, J., Hastie, T., Tibshirani, R., 2001. The Elements of Statistical Learning. volume 1. Springer series in statistics New York.

Friedman, J.H., 2001. Greedy function approximation: A gradient boosting machine. Annals of statistics , 1189–1232.

Group, T.W., 1988. The WAM model—A third generation ocean wave prediction model. Journal of Physical Oceanography 18, 1775–1810.

Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 .

Lefèvre, J.M., Aouf, L., 2012. Latest developments in wave data assimilation, in: ECMWF Workshop on Ocean Waves, pp. 25–27.

L'her, J., Goasguen, G., Rogard, M., 1999. CANDHIS database of in situ sea states measurements on the French coastal zone, in: The Ninth International Offshore and Polar Engineering Conference, International Society of Offshore and Polar Engineers.

Liang, P., Bose, N.K., 1996. Neural network fundamentals with graphs, algorithms and applications. Mac Graw-Hill .

Londhe, S.N., Shah, S., Dixit, P.R., Nair, T.M.B., Sirisha, P., Jain, R., 2016. A Coupled Numerical and Artificial Neural Network Model for Improving Location Specific Wave Forecast. Applied Ocean Research 59, 483–491. doi:10.1016/j.apor.2016.07.004.

Makarynskyy, O., 2005. Neural pattern recognition and prediction for wind wave data assimilation. Pac Oceanogr 3, 76–85.

Makarynskyy, O., Pires-Silva, A.A., Makarynska, D., Ventura-Soares, C., 2002. Artificial neural networks in the forecasting of wave parameters, in: 7th International Workshop on Wave Hindcasting and Forecasting. Banff, Alberta, Canada, pp. 514–522.

Makarynskyy, O., Pires-Silva, A.A., Makarynska, D., Ventura-Soares, C., 2005. Artificial neural networks in wave predictions at the west coast of Portugal. Computers & Geosciences 31, 415–424.

Mandal, S., Prabaharan, N., 2006. Ocean wave forecasting using recurrent neural networks. Ocean engineering 33, 1401–1410.

Marchant, R., Ramos, F., 2012. Bayesian optimisation for intelligent environmental monitoring, in: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE. pp. 2242–2249.

Mentaschi, L., Besio, G., Cassola, F., Mazzino, A., 2013. Problems in RMSE-based wave model validations. Ocean Modelling 72, 53–58.

Moeini, M.H., Etemad-Shahidi, A., Chegini, V., Rahmani, I., 2012. Wave data assimilation using a hybrid approach in the Persian Gulf. Ocean Dynamics 62, 785–797.

Morichon, D., de Santiago, I., Delpey, M., Somdecoste, T., Callens, A., Liquet, B., Liria, P., Arnould, P., 2018. Assessment of flooding hazards at an engineered beach during extreme events: Biarritz, SW France. Journal of Coastal Research 85, 801–805.

Rakha, K.A., Al-Salem, K., Neelamani, S., 2007. Hydrodynamic atlas for Kuwaiti territorial waters. Kuwait Journal of Science and Engineering 34, 143.

Rascle, N., Ardhuin, F., 2013. A global wave parameter database for geophysical applications. part 2: Model validation with improved source term parameterization. Ocean Modelling 70, 174–188.

Refsgaard, J.C., 1997. Validation and intercomparison of different updating procedures for real-time forecasting. Hydrology Research 28, 65–84.

Shahriari, B., Swersky, K., Wang, Z., Adams, R.P., De Freitas, N., 2015. Taking the human out of the loop: A review of Bayesian optimization. Proceedings of the IEEE 104, 148–175.

Snoek, J., Larochelle, H., Adams, R.P., 2012. Practical bayesian optimization of machine learning algorithms, in: Advances in Neural Information Processing Systems, pp. 2951–2959.

563 Stockdon, H.F., Holman, R.A., Howd, P.A., Sallenger Jr, A.H., 2006. Empirical pa-
564 rameterization of setup, swash, and runup. Coastal engineering 53, 573–588.

565 Vousdoukas, M.I., Ferreira, Ó., Almeida, L.P., Pacheco, A., 2012. Toward reliable
566 storm-hazard forecasts: XBeach calibration and its potential application in an oper-
567 ational early-warning system. Ocean Dynamics 62, 1001–1015.

568 Wolpert, D.H., 2002. The supervised learning no-free-lunch theorems, in: Soft Com-
569 puting and Industry. Springer, pp. 25–42.