

Evaluation of XGBoost vs. other Machine Learning models for wind parameters identification

B. García-Puente¹, A. Rodríguez-Hurtado¹, M. Santos² and J.E. Sierra-García³

¹ Faculty of Computer Sciences, Complutense University
C/ Profesor García Santesmases 9, 28040-Madrid (Spain)
e-mail: beleng11@ucm.es, antrod03@ucm.es

² Institute of Knowledge Technology, Complutense University
C/ Profesor García Santesmases 9, 28040-Madrid (Spain)
e-mail: msantos@ucm.es

³ Electromechanical Engineering Department, University of Burgos
Avn. Cantabria sn, 09006-Burgos (Spain)
e-mail: jesierra@ubu.es

Abstract. Wind energy is one of the most promising renewable energies. But wind is a quite unstable resource due to its continuous variation and random nature. This uncertainty affects the production cost. Therefore, accurate forecasting of wind and energy is very interesting for energy markets. In this work, we test a recent and powerful intelligent technique, extreme gradient boosting (XGBoost), for wind prediction. The forecasting models of some wind features with XGBoost are compared with Support Vector Regression (SVR), Gaussian Process Regression (GPR) and Neural Networks (NN) models. Specifically, the three features predicted are the active power generated by the turbine, the wind speed, and the wind direction. The results conclude that these techniques are useful for wind and energy forecasting, with XGBoost being the most outstanding one, especially for short-term predictions.

Key words. Wind energy, forecasting, Machine learning, XGBoost.

1) Introduction

Within the European framework, some objectives have been set to promote the use of renewable energies. The European Union (EU) has pledged that a 32% of energy production will come from renewable sources by 2030. In 2018, the EU produced 160 GW of onshore and 19 GW of offshore wind energy. This made-up 14% of the electricity demand. Nowadays, it continues to be the second form of energy generation capacity [1].

Wind power forecasting plays an active role in reducing operating costs and enhancing the competitiveness of wind power. Even more, it is a must for the integration of the wind power with existing electricity grids of different scales.

There are two ways to predict wind power, namely: directly, based on historical power data, or to forecast

wind speed first, and use the wind power curve of a specific wind turbine to obtain the output wind power [2]. We focused on the first approach.

Nowadays, different Artificial Intelligence (AI) and Machine Learning (ML) techniques have been applied to deal with wind turbines [3], [4] and specifically for forecasting wind features [5]. For instance, in [6], authors reach a 99% accuracy applying techniques such as Extreme Gradient Boosting (XGBoost), decision trees or Random Forest (RF) when forecasting long-term wind speed values.

In this work, we have tested four machine learning algorithms to model different wind parameters in the short-term. We have used XGBoost and compared the predictions with the models obtained with Support Vector Regression (SVR), Gaussian Process Regression (GPR) and Neural Networks (NN) models. The three features predicted are the active power generated by the turbine, the wind speed, and the wind direction.

Different error metrics have been used to compare those forecasting models and, although all of them are efficient, XGBoost has shown the best performance. Besides, it has been proved the importance of the processing of the data.

The structure of the paper is as follows: In section 2, the chosen dataset and the pre-processing methods applied in the project are described. In section 3, we discuss the classification of the forecasting methods, and describe XGBoost and the algorithms to compare. In section 4, we present the evaluation metrics applied in the training and testing processes of the algorithms. In section 5, the results of XGBoost are shown and compared with the other algorithms. In section 6, we summarize the work and draw some conclusions and future works.

2) Materials: Dataset and data pre-processing

The dataset used has been obtained from the SCADA system (Supervisory Control and Data Acquisition) of a wind turbine which is generating power in Turkey [7]. The data have been measured every 10-minute intervals from 01/01/2018 to 12/12/2018.

Figure 1 shows an example of some of the data collected, units, and main features.

	Date/Time	LV ActivePower (kW)	Wind Speed (m/s)	Wind Direction (°)
0	01 01 2018 00:00	380.047791	5.311336	259.994904
1	01 01 2018 00:10	453.769196	5.672167	268.641113
2	01 01 2018 00:20	306.376587	5.216037	272.564789
3	01 01 2018 00:30	419.645905	5.659674	271.258087
4	01 01 2018 00:40	380.650696	5.577941	265.674286
...
50525	31 12 2018 23:10	2963.980957	11.404030	80.502724
50526	31 12 2018 23:20	1684.353027	7.332648	84.062599
50527	31 12 2018 23:30	2201.106934	8.435358	84.742500
50528	31 12 2018 23:40	2515.694092	9.421366	84.297913
50529	31 12 2018 23:50	2820.466064	9.979332	82.274620

50530 rows × 4 columns

Fig. 1. Samples of the dataset with values of the wind features

The wind parameters selected for the study are LV Active Power (kW), the power generated by the turbine; Wind Speed (m/s), the wind speed at the hub height of the turbine; and Wind Direction (°), the wind direction at the hub of the turbine.

A. Pre-processing

The following pre-processing steps have been applied to the data.

- 1) *Downsampling*. To reduce the computational load and to prevent overfitting, the dataset, the signals have been decimated by D ; that is, keep one sample every D samples. In our case, $D = 2$, thus the time interval has been extended using measures every 20 minutes.
- 2) *Split the dataset into training and test*. The dataset is split into two different subsets. One is used to train the algorithm, with the 60% of the samples. The other 40% is used to validate and evaluate the fitted model.
- 3) *Normalization*. To better discriminate and to minimize the presence of outliers in the subsets and speed up the algorithm training, the subsets have been normalized applying the following formula:

$$Z_i = \frac{(X_i - \mu)}{\sigma} \quad (1)$$

Where Z_i is the normalized value of X_i , μ is the mean of the dataset and σ is the standard deviation.

- 4) *Formatting of the subsets*. Finally, the inputs and outputs of the algorithms must follow some guidelines [8]. They are defined as:

$$\begin{aligned} \text{input} &= \{x_t, x_{t-1}, x_{t-2}, \dots, x_{t-M}\}; \\ \text{output} &= x_{t+\Delta} \end{aligned}$$

Where t is the reference timestamp, from which the prediction will be made, x is the value of the time series in t , M is the number of past values to use as input, and Δ is time at when we want to obtain the predicted value.

It is important to carefully select M and Δ , which once set, will be constants during the training. To obtain the best combination of those parameters for each algorithm, different values have been tested and compared:

- M : 72, 216 and 504 (1 day, 3 days and 1 week, respectively).
- Δ : 3, 18, 36 and 72 (1 hour, 6 hours, 12 hours and 1 day, respectively).

3) Methods

There are different ways to classify forecasting methods, according to time intervals, applied models, accuracy, etc. According to [9], we can classify the forecasting models here presented as follows:

- *By time scale*: short, medium, and long term. Our models have been tested considering different time intervals.
- *By the prediction model*:
 - Statistical, machine learning methods: Gradient Tree Boosting and Multi-layer Perceptron (MLP) Regression.
 - Combined methods: SVR and GPR.
- *By the accuracy of output data*: deterministic/point prediction.
- *By the prediction physical quantity*: The models have been trained with three parameters: wind speed, wind direction, and power.
- *By the input data*: historical data. The data used in the project was measured by a SCADA system.

In this project, we are going to focus on the behaviour of the following forecasting algorithms, which are briefly described:

A. XGBoost

XGBoost is an optimized distributed gradient boosting algorithm designed to be highly efficient, flexible, and portable, available as an open-source package. The most important factor behind the success of XGBoost is its scalability in all scenarios. It is faster than other popular solutions on a single machine and scales to billions of examples in distributed or memory-limited settings. [10].

This work uses the XGBoost gradient tree boosting algorithm, which is an evolution of the tree ensemble model:

Given a dataset D with n examples and m features, a tree ensemble model uses K additive functions to predict the output:

$$D = \{(x_i, y_i)\} (\|D\| = n, x_i \in \mathbb{R}^m, y_i \in \mathbb{R}), \quad (2)$$

$$\hat{y}_i = \phi(x_i) = \sum_{k=1}^K f_k(x_i), f_k \in \mathcal{F}, \quad (3)$$

$$\mathcal{F} = \{f(x) = w_{q(x)}\} (q: \mathbb{R}^m \rightarrow T, w \in \mathbb{R}^{|T|}) \quad (4)$$

Where \mathcal{F} is the space of regression trees; q is the structure of each tree that maps an example to the corresponding leaf index; T is the set of leaf indexes, thus $|T|$ is the number of leaves in the tree; and f_k is an independent tree structure q ; w is the leaf weights. Unlike decision trees, each regression tree contains a continuous score on each leaf. We use w_i to represent the score on the i -th leaf.

For a given example, we will use the decision rules in the trees (given by q) to classify it into the leaves and calculate the final prediction by summing up the score in the corresponding leaves (given by m).

To learn the set of functions used in the model, we minimize the following regularized objective:

$$\mathcal{L}(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k), \quad (5)$$

$$\Omega(f) = \gamma T + \frac{1}{2} \lambda \|w\|^2 \quad (6)$$

Where \mathcal{L} is the differentiable convex loss function that measures the difference between the prediction \hat{y}_i and the target y_i ; Ω penalizes the complexity of the model; γ is the first regularization term; and λ denotes the second regularization term, which smooths the final learnt weights to avoid over-fitting.

Intuitively, the regularized objective will tend to select a model employing simple and predictive functions.

However, the tree ensemble model cannot be optimized using traditional optimization methods in Euclidean space, so it must be trained in an additive manner [10]:

$$\mathcal{L}^{(t)} = \sum l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) \quad (7)$$

Where \mathcal{L} is the regularized learning objective; l is a differentiable convex loss function that measures the difference between the prediction and the target; $\hat{y}_i^{(t)}$ is the prediction of the i -th instance of the t -th iteration; Ω is the function that penalizes the complexity of the model; and f_t is the function added to minimize the objective.

B. Gaussian Process Regression

Gaussian process regression is a non-parametric (not limited by a functional form) Bayesian approach towards regression problems [11, 12]. In GPR, we select a prior distribution over a function f and condition this

distribution on the observations, using the posterior distribution to make predictions. Within this prior GP, prior knowledge about the space of functions can be incorporated through the selection of the mean and covariance functions. The prior Gaussian process is defined by its mean and covariance functions as:

$$f(x) \sim GP(m(x), k(x, x')) \quad (8)$$

Where $m(x)$ is the mean function and $k(x, x')$ is the covariance function, also known as the kernel function.

Then, a posterior distribution is generated. Given X (the input variables), the expected value of the output variables y can be predicted. Previous observations and predictions follow a multivariate normal distribution, as:

$$\begin{bmatrix} y_t \\ f_* \end{bmatrix} \sim N \left(0, \begin{bmatrix} K(X_t, X_t) + \sigma_\epsilon^2 I & K(X_t, X_*) \\ K(X_*, X_t) & K(X_*, X_*) \end{bmatrix} \right) \quad (9)$$

Where $K(X_t, X_t)$ is the covariance matrix between all observed points so far, $K(X_*, X_*)$ is the covariance matrix between the newly introduced points, $K(X_*, X_t)$ is the covariance matrix between the new input points and the already observed points, $K(X_t, X_*)$ is the covariance matrix between the observed points and the new input points, I is identity matrix, and σ_ϵ^2 is assumed noise level of observations.

In this work we used a Radial Basis Function RBF (Squared Exponential) kernel. The Gaussian RBF is:

$$K(X_1, X_2) = \exp \left(- \frac{\|X_1 - X_2\|^2}{2\sigma^2} \right) \quad (10)$$

Where σ is the overall variance and $\|X_1 - X_2\|$ is the Euclidean Distance between two points. Also, we applied the range [1, 20] of integers on intervals of three for alpha, which establishes a variance of the additional Gaussian measurement of noise on training observations.

C. Support Vector Regression

When a Support Vector Machine (SVM) is used in regression, it is called Support Vector Regression or SVR. SVR, unlike SVM, only has one kind of sample points, and the optimal hyperplane it seeks is to minimize the total deviation between sample points and that hyperplane [13].

SVR utilizes a subset of the provided dataset to construct a function estimator, as follows:

$$f(x) = \langle w, \Phi(x) \rangle + b \quad (11)$$

Where w is a weighted feature vector, b is the intercept, $\Phi(\cdot)$ represents the mapping and x is the input vector.

To allow SVR to handle nonlinear data, a kernel function that transforms the original input data to a higher-dimensional space, referred to as a kernel space, is proposed [14]. SVR finds a hyperplane that maximizes the distance between two training data subsets and

minimizes the error between the forecasted value and the actual value [15].

After several simulations to find the best configuration, we used the RBF kernel. For the C parameter, a hyperparameter to control error, we used 1, 100 and 1000. For epsilon, which defines a margin of tolerance where no penalty is given to errors, we used a range of [0.1, 0.9], intervals of 0.2. Finally, we considered two values for gamma, which defines how far the influence of a single training example reaches, where low implies it reaches ‘far’ and high implies ‘close’. Those values are defined as:

$$\gamma_1 = \frac{1}{n \text{ features}} \quad \gamma_2 = \frac{1}{n \text{ features} \cdot \sigma^2} \quad (7)$$

Where “n features” refers to the number of features and σ is the overall variance.

D. Multi-layer Perceptron Regressor

MLP is a supervised neural network algorithm that learns a nonlinear function and maps inputs to outputs by training on a dataset [16]. The MLP consists of three or more layers (an input layer, an output layer, and one or more hidden layers). Each node in one layer connects with a certain weight to every node in the following layer.

The input layer consists of a set of neurons, with X representing the inputs. The output layer receives information from the last hidden layer and transforms it into output values. In each hidden layer, each neuron accumulates the values from the previous layer as a weighted linear summation with a bias, followed by a nonlinear activation function.

For instance, the output at the j-th node of the first hidden layer is given by:

$$out = g\left(\sum_i w_{ji}x_i + b_j\right) \quad (12)$$

Where g is the nonlinear activation function, w_{ji} is the weight of the i-th input in the j-th neuron of the first hidden layer and b_j the bias of the j-th neuron of the first hidden layer.

4) Evaluation of the forecasting methods

To determine which algorithm is the best for forecasting each wind feature, training and testing were carried out combining each wind feature, different values of M and Δ , and the algorithms. The metrics to evaluate and compare the algorithms are the following [17]:

A. *RMSE (Root Mean Square Error)* as a measure of how spread out the residuals:

$$RMSE = \left[\sum \frac{(\hat{x}_i - x_i)^2}{N}\right]^{1/2} \quad (13)$$

B. *MAE (Mean Absolute Error)* is the mean amount of error in the measurements:

$$MAE = \frac{1}{n} \sum |x_i - \hat{x}| \quad (14)$$

C. *Coefficient of determination R^2* is the ratio of total variation of data points explained by the regression line and total variation of data points from the mean:

$$R^2 = \frac{SSR}{SST} = \frac{\sum(\hat{x}_i - \bar{x})^2}{\sum(x_i - \bar{x})^2} \quad (15)$$

5) Results and discussion

Detailed results are shown of the best technique for each wind feature, although all of them are compared in the last subsection. The following tables show the best results obtained for each evaluation method and each combination of M and Delta parameters.

A. Active Power

In Figure 2, the forecasting of active power using the XGBoost technique is shown. The blue line are the real data and the red one the prediction. As it is possible to observe, the algorithm fits well the predictions, although the higher and lower values are harder to predict correctly.

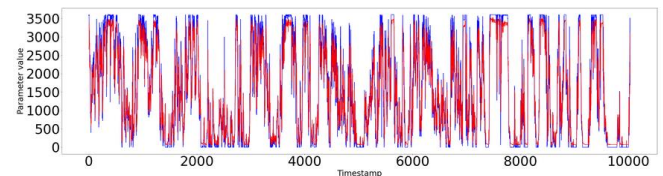


Fig. 2. Active Power prediction for M = 72, $\Delta = 3$

The numerical results for all the combinations are shown in Table 1. The boldfaced values are the best results.

Table 1. Active Power results for each combination of M and Δ with XGBoost

M	Δ	RMSE	MAE	R2
72	3	482.4018	335.0285	0.864667
	18	941.2131	750.04142	0.479454
	36	1138.9793	959.85287	0.240119
	72	1284.1493	1096.6035	0.082377
216	3	502.0565	340.33766	0.864416
	18	953.8013	750.01071	0.481711
	36	1144.7827	972.82888	0.242018
	72	1288.3964	1105.88849	0.07601
504	3	532.6081	336.25796	0.863716
	18	994.7580	759.937	0.470599
	36	1182.2283	958.34544	0.229319
	72	1258.1828	1095.18485	0.07357

B. Wind Speed

In Figure 3, the forecasting of wind speed using XGBoost is shown. The blue line are the real data and the red one the prediction. As it is possible to observe, predicted values fit well, being lower and higher values harder to obtain correctly.

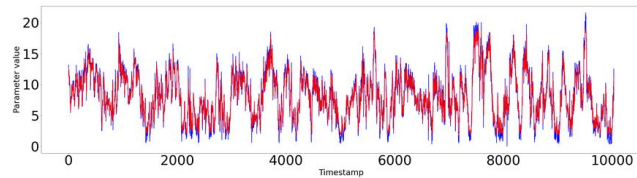


Fig. 3. Wind Speed prediction for $M = 72$, $\Delta = 3$

The numerical results for all the combinations are shown in Table 2. The boldfaced values are the best results.

Table 2. Wind Speed results for each combination of M and Δ with XGBoost

M	Δ	RMSE	MAE	R2
72	3	1.436387	1.07365	0.865559
	18	2.80246	2.19508	0.488321
	36	3.386817	2.71107	0.2537
	72	3.740795	2.99105	0.09255
216	3	1.443943	1.07902	0.8656
	18	2.812568	2.20391	0.4904
	36	3.411496	2.71501	0.25128
	72	3.768315	3.01633	0.08846
504	3	1.44859	1.08109	0.86428
	18	2.83819	2.24	0.47841
	36	3.43103	2.7308	0.23731
	72	3.7695	2.9956	0.0817

C. Wind Direction

In Figure 4, the forecasting of wind direction using the XGBoost technique is shown. The blue line are the real data and the red one the prediction. As it is possible to observe, the predicted values fit worse than in the other features.

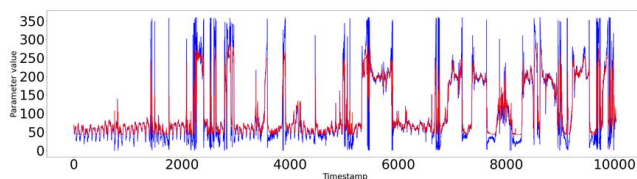


Fig. 4. Wind Direction prediction for $M = 72$, $\Delta = 3$

As it is possible to observe, the predicted values fit worse than in the other features.

The numerical results for all the combinations are shown in Table 3. The boldfaced values are the best results.

Table 3. Wind Direction results for each combination of M and Δ with XGBoost

M	Δ	RMSE	MAE	R2
72	3	44.3778	20.57955	0.726623
	18	63.43407	38.83586	0.441982
	36	70.35669	48.42757	0.31444
	72	77.1177	59.58934	0.17778
216	3	44.64334	20.654462	0.725628
	18	63.540016	38.610825	0.444848
	36	70.2878	47.94277	0.321609
	72	77.12268	56.95441	0.184393
504	3	45.22584	21.10568	0.723168
	18	63.9277	38.929135	0.44744
	36	70.9534	48.4449	0.320285
	72	77.5956	58.828	0.188304

The following tables show the best results of each model for the three wind parameters.

Table 4. Comparison between the algorithms results for Active Power

Algorithm	M	Δ	Best RMSE	MAE	R2
XGBoost	72	3	482.4018	335.0285	0.8646
GPR	72	3	487.8276	338.8071	0.8616
SVR	72	3	497.8148	330.3092	0.8558
MLP	72	3	529.3002	371.6042	0.837

Table 5. Comparison between the algorithms results for Wind Speed

Algorithm	M	Δ	Best RMSE	MAE	R2
XGBoost	72	3	1.4363	1.0736	0.8655
GPR	72	3	1.4402	1.0705	0.8648
SVR	72	3	1.4907	1.1035	0.8551
MLP	72	3	1.6242	1.228	0.8281

Table 6. Comparison between the algorithms results for Wind Direction

Algorithm	M	Δ	Best RMSE	MAE	R2
XGBoost	72	3	44.3778	20.5795	0.7266
GPR	72	3	45.8228	21.0369	0.7085
SVR	72	3	46.9885	19.9221	0.6935
MLP	72	3	46.9521	22.1195	0.6939

The best results are always the combination of the smallest values of M and Δ . For each M , the smaller the parameter delta is, the better the predictions result. The same happens

if we compare the results between different values of M . Also, each model shows a similar behaviour: as soon as we increase any of those parameters, the results worsen. We can conclude that all of them work better for immediate-short-term or short-term predictions rather than long-term.

Regarding the predictions, they all follow the same evolution regardless of the model. As the prediction time increases, the results worsen. Also, the predicted errors stop reaching higher values but accumulate a larger one instead. These models are good for short-term forecasting, but when it comes to using them, we must be aware that the further we want to predict, the less accurate those predictions will be. Regarding the error obtained based on Δ value, all models worsen in a similar way. Despite of that, it is worth mentioning that XGBoost has the best R^2 results in almost any case, although with a slight difference.

Besides, we tested the models removing 10000 samples from the dataset, and the results obtained were worse than the ones we show here. With these tests, it is possible to conclude that the results shown above can be improved by extending the training data.

Looking at the tables above, we can see that XGBoost has the best results for every error metric, except for the MAE, being the GPR and SVR algorithms which obtain the best results for wind power and wind direction, and wind speed respectively. This happens because the predictions obtained from these algorithms do not reach extreme values as XGBoost. This means that XGBoost can accumulate a larger error, which is translated in a higher MAE.

6) Conclusions and future works

In this project, different machine learning algorithms have been applied to predict the behaviour of the wind and, consequently, the produced power. Nonetheless, predicting the future with a 100% of accuracy is a difficult task. We can only make more or less precise estimations of future values based on the fidelity of measured data in the past. The wind features studied here are the active power generated by the turbine, the wind speed, and the wind direction.

The forecasting algorithms used are gradient boosting (XGBoost), Support Vector Regression (SVR), Gaussian Process Regression (GPR) and neural networks (NN) models, being the best results for the chosen features obtained with XGBoost.

That is, XGBoost is a good tool for forecasting. Compared to other ML models, it is accurate and trustworthy, and it gave better results than the other algorithms tested, with acceptable errors.

Future works include combine some of these machine learning techniques, in an assemble system, to improve the results, and to forecast other variables.

Acknowledgments

This work was partially supported by the Spanish Ministry of Science, Innovation and Universities under MCI/AEI/FEDER Project no. PID2021-123543OB-C21.

References

- [1] European Commission, "Guidance document on wind energy projects and EU legislation on nature protection", (2020).
- [2] Wang, Y., Zou, R., Liu, F., Zhang, L. and Liu, Q., "A review of wind speed and wind power forecasting with deep neural networks", *Applied Energy* (2021). Vol. 304, pp. 117766.
- [3] Sierra-García, J. E. and Santos, M. "Switched learning adaptive neuro-control strategy", *Neurocomputing* (2021). Vol. 452, pp. 450-464.
- [4] Sierra-García, J. E., and Santos, M. "Neural networks and reinforcement learning in wind turbine control". *Revista Iberoamericana de Automática e Informática Industrial* (2021). Vol. 18, no 4, pp. 327-335.
- [5] Sacie, M., Santos, M., López, R., and Pandit, R., "Use of state-of-art machine learning technologies for forecasting offshore wind speed, wave and misalignment to improve wind turbine performance", *Journal of Marine Science and Engineering* (2022). Vol. 10, no 7, pp. 938.
- [6] Ahmadi, A., Nabipour, M., Mohammadi-Ivatloo, B., Amani, A. M., Rho, S. and Piran, M. J., "Long-term wind power forecasting using tree-based learning algorithms", *IEEE Access* (2020). Vol. 8, pp. 151511-151522.
- [7] <https://www.kaggle.com/datasets/berkerisen/wind-turbine-scada-dataset>
- [8] Filik, Ü. B. and Filik, T., "Wind speed prediction using artificial neural networks based on multiple local measurements in Eskisehir", *Energy Procedia* (2017). Vol. 107, pp. 264-269.
- [9] Wang, Y., Zou, R., Liu, F., Zhang, L. and Liu, Q., "A review of wind speed and wind power forecasting with deep neural networks", *Applied Energy* (2021). Vol. 304, pp. 117766.
- [10] Chen, T. and Guestrin, C., "Xgboost: A scalable tree boosting system", In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (2016). Pp. 785-794.
- [11] Schulz, E., Speekenbrink, M. and Krause, A., "A tutorial on Gaussian process regression: Modelling, exploring, and exploiting functions", *Journal of Mathematical Psychology* (2018). Vol. 85, pp. 1-16.
- [12] Guevara, C., and Santos, M. "Intelligent models for movement detection and physical evolution of patients with hip surgery". *Logic Journal of the IGPL* (2021). Vol. 29, no 6, pp. 874-888.
- [13] Quan, Q., Hao, Z., Xifeng, H. and Jingchun, L., "Research on water temperature prediction based on improved support vector regression", *Neural Computing and Applications* (2020). Pp. 1-10.
- [14] Zhang, F. and O'Donnell, L. J., "Support vector regression", In *Machine Learning*, Academic Press (2020). Pp. 123-140.
- [15] Zhang, Z., Ding, S. and Sun, Y., "A support vector regression model hybridized with chaotic krill herd algorithm and empirical mode decomposition for regression task", *Neurocomputing* (2020). Vol. 410, pp. 185-201.
- [16] Feng, X., Ma, G., Su, S. F., Huang, C., Boswell, M. K. and Xue, P. "A multi-layer perceptron approach for accelerated wave forecasting in Lake Michigan", *Ocean Engineering* (2020). Vol. 211, pp. 107526.
- [17] Grandini, M., Bagli, E. and Visani, G., "Metrics for multi-class classification: an overview", *arXiv preprint arXiv:2008.05756* (2020).