A FORECASTING ENSEMBLE MODEL BASED ON TRANSFER LEARNING FOR ENERGY SUPPLIER FROM PROFILE GEOMETRIES OF WIND TURBINES

Jasson F. Gurgel[1], Felipe C. Sousa[1], Pedro L. B. Martins[1], Carla F. de Andrade[1] and Francisco Olimpio M. C.[2]

[1] University Federal of Ceará, Fortaleza (Brazil)

[2] University of International Integration of The Afro-Brazilian Lusophone World, Redenção (Brazil)

Abstract

Small-scale wind turbines (SSWTs) are promising solutions for decentralized electrification in urban and remote settings, but their performance depends strongly on blade aerodynamics and on reliable power prediction under limited data availability. This study combines Blade Element Momentum (BEM) analysis with supervised machine learning to predict turbine voltage from geometric and operational parameters, while supporting blade design exploration through chord and twist distributions. A compact dataset from wind-tunnel experiments was enriched through data augmentation and complementary laboratory measurements, resulting in 24 samples. Missing values were handled using KNN imputation. Several regression models were benchmarked under repeated cross-validation with hyperparameter optimization, and assessed using MAE, RMSE, and R^2. Among the evaluated methods, KNN achieved the best generalization on the test set, with RMSE = 3.35 and R^2 = 0.69, outperforming tree-based ensembles in the low-data regime. These results indicate that locally adaptive, nonparametric models can better capture the feature–response relationship when observations are scarce. Overall, the proposed lightweight and reproducible pipeline demonstrates that BEM-informed features combined with carefully tuned machine learning can provide accurate voltage forecasts for SSWTs.

Keywords: Renewable Energy, Wind Energy, Forecasting, Machine Learning, Ensemble Models.

1. Introduction

The search for renewable energy sources has intensified in recent decades in response to environmental challenges exacerbated by the increase in energy demand driven by global urbanization and industrial digitization. Among clean and sustainable alternatives, wind energy stands out for its high availability and low environmental impact (Burton et al., 2021). In Brazil, recent national energy reports indicate the growing strategic role of renewable sources in expanding supply security and diversifying decentralized generation pathways (De Janeiro, 2025). In this scenario, Small-Scale Wind Turbines (SSWTs) have gained prominence for offering decentralized energy generation solutions and for being effective in urban and remote areas with limited installation space (Mertens, 2006; Kumar et al., 2018).

The geometry of its blades fundamentally determines the aerodynamic performance of a wind turbine. The choice of aerodynamic profile, together with the chord and twist distributions along the blade length, significantly affects the power coefficient (CP), torque, and starting characteristics (Hansen, 2015).

The Blade Element Momentum (BEM) theory remains the most widely used model for aerodynamic analysis of rotors, offering a suitable balance between physical accuracy and computational efficiency (Glauert, 1935). Mathematically, the aerodynamic power is expressed as:

10.18086/swc.2025.06.22 Available at http://proceedings.ises.org

$$P = \frac{1}{2}\rho A v^3 C_p(\lambda, \beta) \quad \text{(eq. 1)}$$

$$C_p(\lambda, \beta) = \frac{P}{\frac{1}{2}\rho A v^3} \quad \text{(eq. 2)}$$

Where ρ is the air density, A the rotor swept area, v the wind velocity, and finally, λ and β are the tip speed ratio and blade pitch angle, respectively.

Recent advances in machine learning (ML) have enabled modeling of previously unexplored problems in wind energy, primarily using wind turbine performance data, complementing BEM-based analytical approaches. Recent studies have shown that machine learning can support forecasting tasks in wind-related energy systems under complex operating conditions, reinforcing its potential as a complementary tool to physics-based models (Javaid et al., 2022). Traditional ML models, such as Support Vector Machines and K Nearest Neighbors, advanced models such as neural networks, and ensembles such as Random Forest, XGBoost, and Gradient Boosting have achieved robust predictions of energy production and voltage generation, even with limited or noisy aerodynamic data (Rezaei, 2020; Nai-Zhi, 2022). Such techniques enable faster optimization cycles and facilitate exploration of parametric design.

This study aims to develop a voltage prediction model for small-scale wind turbines based on geometric parameters and to generate optimized blade profiles by integrating *BEM theory* with *machine learning*. By coupling physics-based and data-driven approaches, the proposed framework aims to improve prediction accuracy and enable efficient aerodynamic design workflows for decentralized renewable systems.

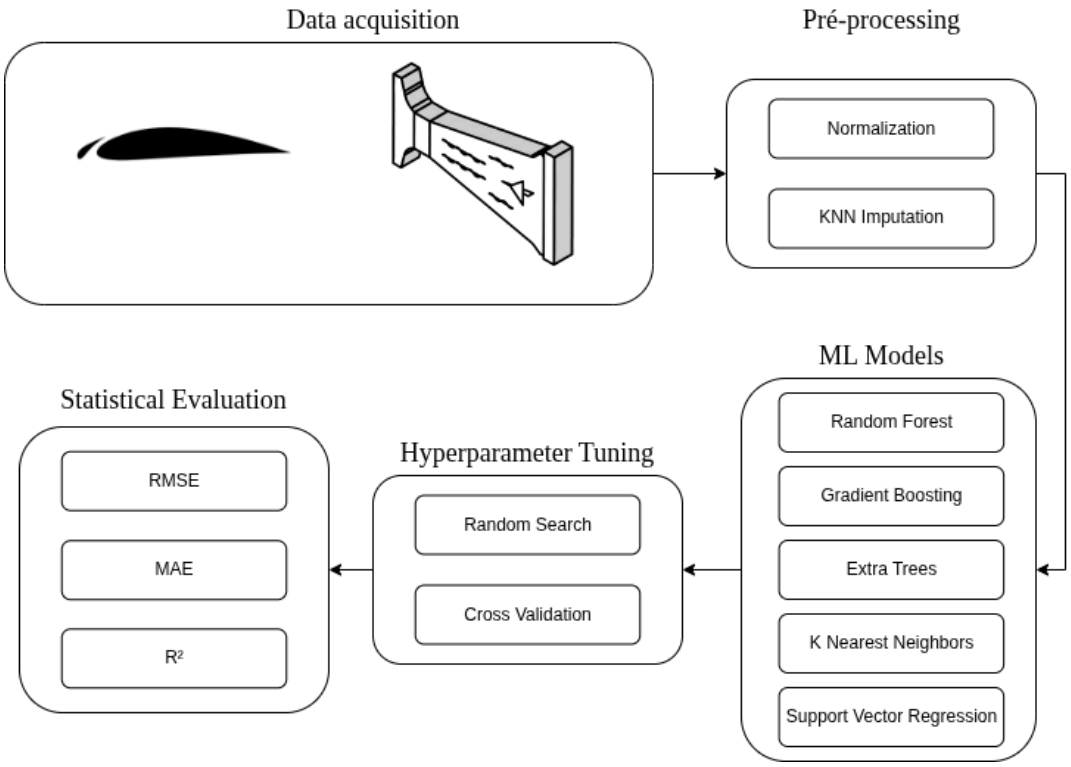
The paper is structured as follows: the introduction section contextualizes the problem, justification, and objectives. The methodology section details the proposed method, algorithms, and metrics. The results section presents the findings from the proposed method. Finally, the conclusion summarizes the findings and suggests directions for future research.

2. Methodology

The methodological framework adopted in this study aims to evaluate and compare the performance of five regression models such as Random Forest (RF), Gradient Boosting Regressor (GBR), Extra Trees (XT), K-Nearest Neighbors (KNN), and Support Vector Regression (SVR) for accurate, fast, and efficient prediction of electrical voltage from a wind blade generated by the BEM method.

The approach combines data preprocessing, hyperparameter optimization through random search with cross-validation, and statistical performance comparison using standard error metrics. Figure 1 illustrates the general workflow, which ensures a reproducible and unbiased comparison between algorithms.

Figure 1 - Diagram of the workflow from data collection to electrical-voltage prediction using wind-turbine blades.



2.1. Dataset Description

The data used in this study were acquired through experiments conducted in a closed-circuit wind tunnel, with external dimensions of 4.8 m x 2.4 m x 15.9 m (width, height, and length), test chamber dimensions of 1000 mm x 1000 mm x 3000 mm (width, height, and length), and air speeds ranging from 0 to 40 m/s. During data acquisition, the initial velocity (m/s), post-initial velocity (m/s), velocity reduction (%), velocity @ 5 m/s post-initial (RPM), TSR @ 5 m/s post-initial, design TSR, design TSR deviation, and stress @ 5 m/s were measured. However, only a small amount of experimental data could be collected, so a data augmentation dataset was used. Initially, tests were performed using only 12 samples collected from different airfoil models: AH93w215 G610, AH93w215 G915, AH93W215 G1360, GOE 770 G1360, GOE 770 G915, ONERA HOR20 G1360, ONERA HOR20 G915, NREL S809 G610, and NREL S809 G915.

Our dataset comprises 24 samples that map 3 key characteristics: rotation (rpm), reference speed (m/s), and voltage (V). To illustrate our data and visualize its distribution, we created a graph in Figure 1 that shows our data and its behavior in detail.

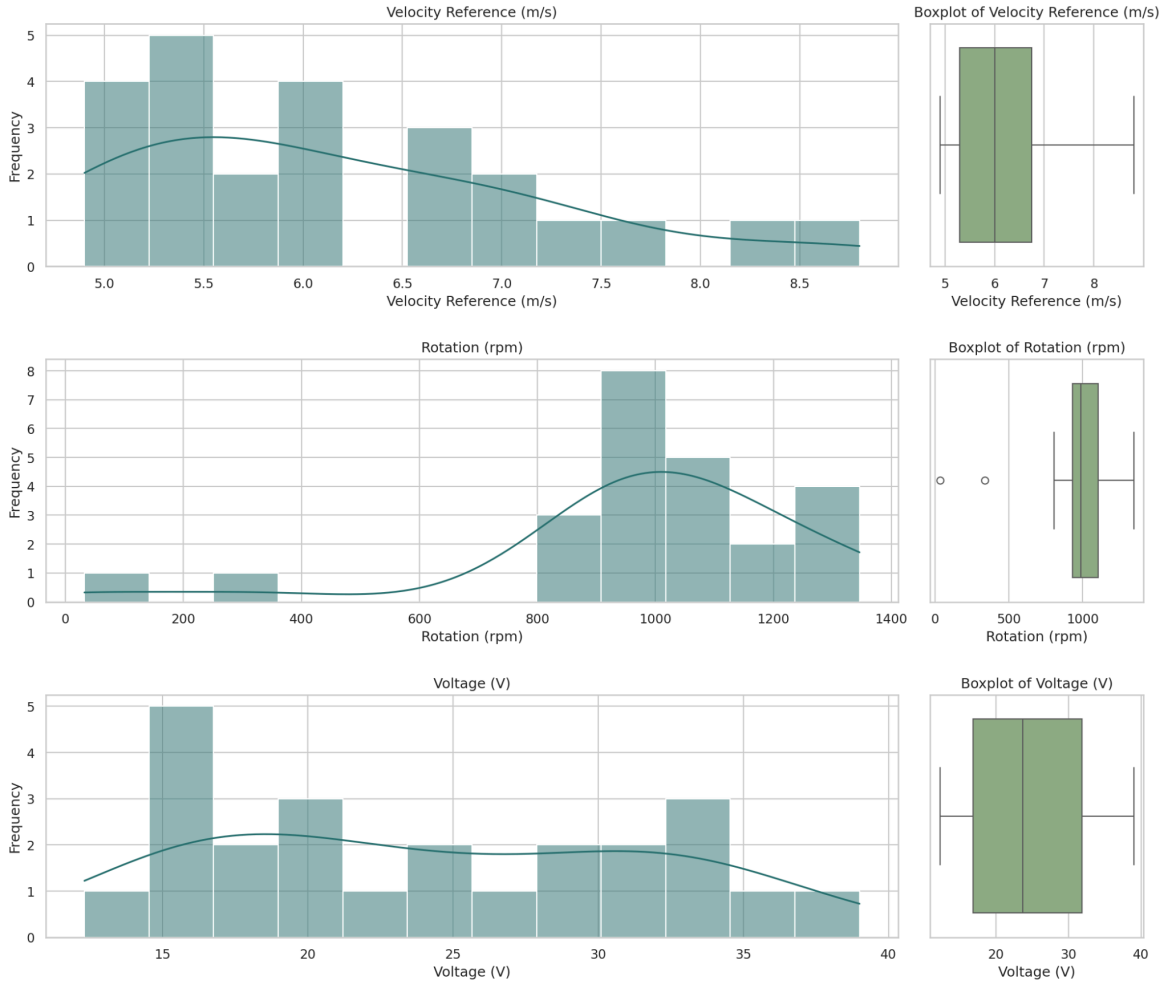


Figure 2: Empirical distributions of reference speed (m/s), rotation (rpm), and voltage (V) in the 24-sample dataset.

Figure 2 shows that the three variables are unevenly distributed and the sample is highly concentrated in a narrow range of rotation values, which is consistent with the small-sample setting. This observation supports the adoption of robust preprocessing and motivates the use of lightweight regression models.

2.1.1. Preprocessing

The pre-processing step in our database consisted of analyzing missing or null values and determining how to handle them. In the complete set, there were only four instances of invalid data in the columns “Rotation (rpm)” and “Voltage (V),” two in each. To better understand the data's behavior, an additional step was taken to examine their relationships. We performed an unconditional bivariate analysis, for example, without considering the data separated by the target variable, voltage (V). Therefore, we calculated the covariance matrix of the data and summarized the results in Figure 3.

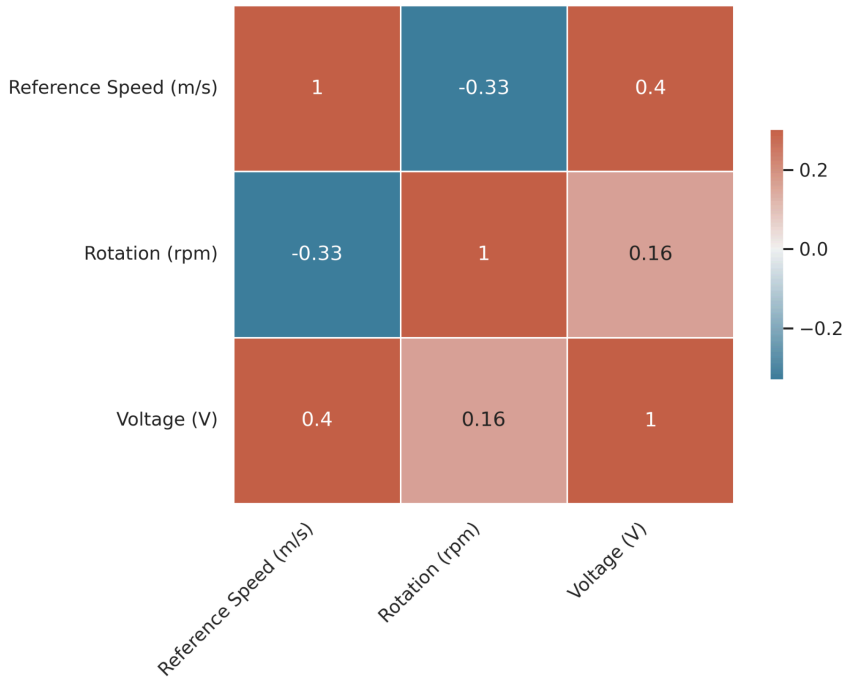


Figure 3: Pearson correlation matrix for reference speed, rotation, and voltage before imputation.

Figure 3 indicates a weak positive association between voltage and reference speed, a weaker association between voltage and rotation, and a moderate negative association between rotation and reference speed. Because the correlation structure is limited and the dataset is very small, removing incomplete observations would further reduce the effective sample size; therefore, KNN imputation was adopted.

2.2. Validation Metrics

To evaluate the predictive performance of the regression models in this study, three complementary metrics were used: Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Coefficient of Determination (R^2). These metrics quantify distinct aspects such as model accuracy and robustness, allowing for a comprehensive assessment of systematic and random deviations between observed and predicted values.

2.2.1. RMSE

The Square Root of Mean Squared Error (RMSE) measures the standard deviation of the residuals, representing how concentrated the data points are around the line of best fit. It penalizes larger errors more severely due to the quadratic term, making it particularly sensitive to outliers. The RMSE equation is defined as:

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}_i\right)^2} \qquad \text{(eq. 1)}$$

where y_i and $\hat{y}_i$ denote the observed and predicted values, respectively, and N is the total number of samples. The RMSE retains the same unit as the dependent variable, facilitating its physical interpretation.

2.2.2. MAE

The MAE quantifies the average magnitude of prediction errors without considering their direction. Unlike RMSE, it is less influenced by outliers since it uses the absolute deviation:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \qquad \text{(eq. 2)}$$

This metric provides a more robust indication of the average expected deviation per prediction, and is particularly useful when the error distribution is not Gaussian.

2.2.3. R2

The coefficient of determination R^2 measures the proportion of variance in the dependent variable that is explained by the model. It is computed as:

$$R^2 = 1 - \frac{\sum_{i=1}^{N}\left(y_i - \hat{y}_i\right)^2}{\sum_{i=1}^{N}\left(y_i - \bar{y}_i\right)^2} \quad \text{(eq. 3)}$$

Where $\bar{y}$ is the mean of the observed values. Values of R^2 close to 1 indicate strong explanatory power, whereas negative values suggest that the model performs worse than a simple mean-based prediction.

2.3. Data Augmentation

The use of data augmentation has become a well-established strategy to alleviate the limitations of small datasets by increasing sample diversity and improving model generalization, particularly in data-scarce or imbalanced scenarios (Wang et al., 2024; Moreno-Barea et al., 2020). In settings where collecting additional experimental observations is costly or impractical, controlled perturbation and replication strategies have been shown to improve predictive performance, especially when the original sample size is limited (Moreno-Barea et al., 2020). More broadly, recent surveys highlight that augmentation can be an effective mechanism for enhancing robustness when the available data are insufficient to support highly flexible models (Wang et al., 2024). Therefore, given the limited number of samples available in this study, two complementary augmentation strategies were adopted to expand the dataset and improve model generalization.

The first strategy consisted of generating *noisy replicas* of the original samples. For each instance in the initial dataset, synthetic samples were created by adding Gaussian noise with zero mean and a small standard deviation proportional to 5% of each feature's range. This procedure aimed to simulate natural measurement variability while preserving the overall statistical distribution of the data. As a result, the dataset size was doubled, from 12 to 24 samples.

The second strategy involved acquiring 12 additional samples through new laboratory experiments under the same experimental protocol as the original collection. These new measurements were integrated into the augmented dataset, enabling comparative analyses between the original dataset (12 samples) and the expanded version (24 samples).

Both augmentation procedures were evaluated by inspecting the statistical consistency of the features (mean, variance, and correlation structure) to ensure that the generated data remained within realistic physical limits.

2.4. Machine Learning Models

This section presents the set of supervised learning algorithms employed to model and predict the target variable from the given dataset.

The scenario for this work starts from a supervised learning problem. Therefore, given a dataset that can be represented by a matrix D_{ij}, where i are the samples of the dataset, j are the feature vectors, and y is the label or annotation of the dataset, the predictor variable, which in general occupies the j-th column of the dataset for concerns of a predictor vector, such as the one used in this work. The goal is to learn a function $f: \Re^d \rightarrow Y$ that generalizes well to unseen data, minimizing a loss function $L(f(x), y)$ with respect to the underlying data distribution (Hastie et al., 2009).

The models were selected to represent different paradigms of learning: margin-based (Support Vector Regression — SVR), instance-based (K-Nearest Neighbors — KNN), and ensemble-based (Random Forest, Extra Trees, Gradient Boosting), to ensure methodological diversity and robust results.

2.4.1. Support Vector Machines

Support Vector Regression (SVR) is the counterpart of the Support Vector Machine (SVM) for regression tasks, since the algorithm was originally designed for binary classification tasks(Cortes & Vapnik, 1995). The main objective of the algorithm is to find a function $f(x)$ that approximates a data set $D = \{(x_i, y_i)\}_{i=1}^{N}$, where $x_i \epsilon \Re^d$ and $y_i \epsilon \Re$ allows a maximum deviation ε (epsilon) from the actual values y_i.

Unlike methods that minimize quadratic error (such as Linear Regression), SVR uses an ε -insensitive loss function. This function does not penalize errors that fall within a tolerance ε around the function $f(x)$. Only points that fall outside this tolerance contribute to the cost. SVR can be defined as:

$$min_{w,b,\xi,\xi^*} \frac{1}{2}||w||^2 + C\sum_{i=1}^{N}(\xi_i + \xi_i^*) \qquad \text{(eq. 4)}$$

ε is what defines tolerance. Minor errors that ε are ignored. C is the parameter that controls the trade-off between the smoothness of the function (model complexity $\frac{1}{2}||w||^2$) and the amount of error (points outside the tube $\sum(\xi_i + \xi_i^*)$ that is tolerated.

2.4.2. K-Nearest Neighbors

K-Nearest Neighbors (KNN) is a nonparametric, instance-based algorithm in which the prediction function is estimated locally rather than globally (Cover & Hart, 1967). Instead of learning an explicit mapping function from the training data, KNN stores the entire dataset and makes predictions based on the proximity of new samples to existing ones in the feature space. For a new query point x_q, the algorithm identifies the subset $D_q \subset D$ that contains the k pairs (x_i, y_i) corresponding to the k nearest neighbors of x_q according to a chosen distance metric (typically the Euclidean distance). The predicted value $\hat{y}_q$ is then computed as the average of the target values of those neighbors:

$$\hat{y}_q = \frac{1}{k}\sum_{(x_i,y_i)\in D_q} y_i \qquad \text{(eq. 5)}$$

2.4.3. Random Forest

Random Forest (RF) is an ensemble-based machine learning method proposed by Breiman (2001) that combines the bagging (Bootstrap Aggregating) technique with random attribute selection.

In each iteration m, a training subset D_m is generated from the original data set D by random sampling with replacement (bootstrap). From this subset, a decision tree $h_m(x)$ is constructed, whereby at each node only a random subset of k variables among the d available ($k < d$) is considered to determine the best split point. The final model is obtained by aggregating the M trees generated:

$$\widehat{f}(x) = \frac{1}{M}\sum_{m=1}^{M} h_m(x) \qquad \text{(eq. 6)}$$

for regression, or by majority vote in the case of classification.

This combination reduces the variance of the estimator without significantly increasing bias, resulting in a model that is robust against noise and overfitting. The diversity among trees, promoted by the randomization of samples and variables, is the main factor that ensures the stability and generalization of the method. Random Forest has been widely used in regression tasks, despite having its variant for classification tasks, presenting competitive performance in comparison with more complex models, while maintaining partial interpretability through measures such as feature importance (Hastie et al., 2009; Liaw & Wiener, 2002).

2.4.4. Extra Trees

Extra Trees is a joint method proposed by Geurts et al. (2006), similar to Random Forest, but which introduces an additional level of randomness to further reduce variance. The goal is to further reduce variance by also introducing randomness in the selection of cut-off points. The main differences between Extra Trees (ET) and Random Forest are the cut-off point selection and sampling. The Cut-off Point Selection is that instead of calculating the optimal cut-off point (via Gini or MSE) for a characteristic, ET selects a completely random cut-off point within the range of that characteristic. The sampling step, traditionally, is ET grows each tree using the entire original training set D (without bootstrapping), although bagging can be enabled. The algorithm selects the best split (among k features) by comparing the gain (variance reduction) obtained by these pairs (feature, random cutoff point) (Geurts et al., 2006).

The formulation of the final prediction (aggregation) of the ensemble $F(x)$ is identical to that of Random Forest, both for regression (mean) and classification (majority vote). The difference lies solely in the method of constructing the trees $h_m(x)$.

2.4.5. Gradient Boosting

Gradient Boosting (GB) is an ensemble method that uses the boosting technique to build models in an additive and sequential manner (Friedman, 2001). The algorithm trains new models (typically decision

trees) to iteratively correct the errors (residuals) of previous models, optimizing a loss function L through gradient descent.

Since the GB model is based on sequential additive ensemble methods, its main objective is to minimize an arbitrary loss function by combining several weak estimators (which are generally shallow decision trees) into a strong model (Friedman, 2001). Formally, GB seeks to find a function $F^*(x)$ that minimizes the expected error:

$$F^*(x) = arg\ min_F \Xi_{y,x}[L(y, F(x)] \qquad \text{(eq. 7)}$$

where $L(y, F(x))$ it is a differentiable loss function (such as the mean square error $L(y, F(x)) = \frac{1}{2}(y - F(x))^2$), for example). The model, in turn, is constructed iteratively every M step, according to the additive form:

$$F_M(x) = F_0(x) + \sum_{m=1}^{M} \gamma_m h_m(x) \qquad \text{(eq. 8)}$$

where $F_0(x)$ is the first model, which is typically the average of the y values, $h_m(x)$ is the baseline model, such as a regression tree, and γ_m is the learning coefficient associated with each $h_m(x)$ model. At each iteration M, the algorithm adjusts the new estimator $h_m(x)$ to the negative residuals of the loss function gradient relative to the previous model's predictions:

$$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right] \qquad \text{(eq. 9)}$$

the estimator $h_m(x)$ is then trained to predict these residues, and the model is updated according to:

$$F_M(x) = F_{m-1}(x) + v\gamma_m h_m(x) \qquad \text{(eq. 10)}$$

where $0 < v < 1$ is the learning rate that controls the update step, avoiding overfitting.

2.5. Hyperparameter Optimization

The hyperparameter optimization process was carried out to ensure a fair, reproducible, and unbiased comparison among the models evaluated.

For each algorithm, a Random Search strategy was applied within predefined parameter ranges, established from literature guidelines and preliminary experiments. A 5-fold cross-validation scheme with 3 repetitions was employed, using the Root Mean Square Error (RMSE) as the objective loss function to minimize. This approach enables efficient exploration of the hyperparameter space, mitigating overfitting risks and ensuring statistical robustness in selecting optimal configurations (Bergstra and Bengio, 2012; Probst et al., 2019).

2.6. Setup Environment

The experiments were performed on the Ubuntu 24.04.4 LTS 64 operating system and an Intel Core i5-9300H CPU with 8 cores operating at 2.40 GHz and 16 GB of available memory. The implementation was performed using Python 3.12 with uv, a Python package and project manager for reproducing experiments in a robust and fast manner. The core scientific libraries used were Scikit-learn (version 1.7.0) for creating machine learning models and preprocessing steps in the database, Pandas (version 2.3.1) to load all datasets in .csv format, NumPy (version 2.3.1) to perform specific algebra or mathematics steps, and Matplotlib (3.10.3) with Seaborn (0.13.2) to plot the graphs and figures.

3. Results

This section of the paper is responsible for showing all the results obtained from the comparative analysis of machine learning models for predicting electrical voltage from wind turbine blade models using BEM theory.

3.1. Statistical Analysis

As mentioned in the section describing the database, techniques were also applied to prepare the data preliminarily, in particular, the use of KNN Imputer to replace missing values, so that robust and concise results could be obtained when applied to the regression model to predict electrical voltage.

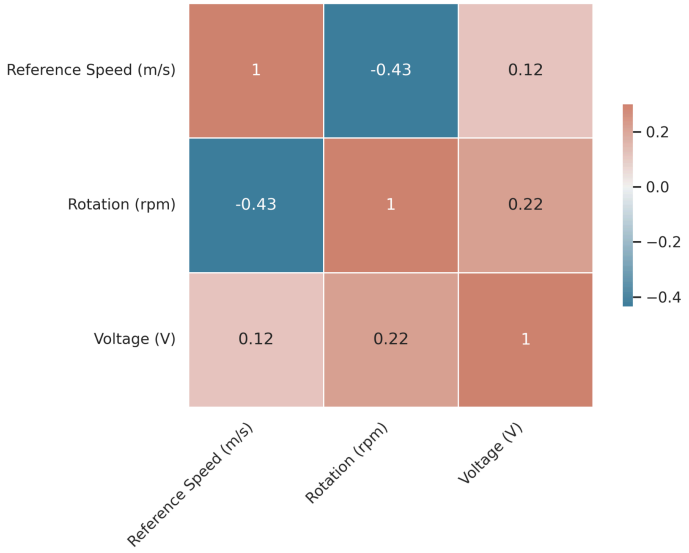


Figure 4: Pearson correlation matrix after KNN imputation.

As shown in Figure 3, the choice to use KNN Imputer proved effective, given that the linear correlation between the Rotation (rpm) and Voltage (V) characteristics was maintained, thus not compromising the dataset and making it effectively suitable for use as training data for the models.

3.2. Hyperparameter Tuning and Model Configuration

This section reports the optimization procedure and the optimal settings obtained for each model, followed by their performance evaluation. To ensure a fair and unbiased comparison among models, a systematic hyperparameter optimization was performed using Random Search with 5-fold cross-validation, repeated 3 times. The Root Mean Squared Error (RMSE) was adopted as the loss function to minimize, and the search ranges for each model were defined according to prior literature and preliminary experiments, as shown in Table 1. The optimal configurations obtained are summarized in Table 1.

Table 1 – Best hyperparameters obtained after optimization via Random Search (5 folds with 3 repetitions using RMSE metric).

Models	Best Hyperparameters
SVM	model__C: 395.7428423834928, model__epsilon: 0.1389069254026063, model__gamma: 0.9894367254602554
Extra Trees	max_depth: 14, max_features: 0.6742756945800147, min_samples_leaf: 1, min_samples_split: 5, n_estimators: 963
KNN	model__leaf_size: 22, model__n_neighbors: 2, model__p: 1, model__weights: distance
Random Forest	bootstrap: False, max_depth: 16, max_features: 0.2751524086067534, min_samples_leaf: 2, min_samples_split: 6, n_estimators: 978
Gradient Boosting	learning_rate: 0.27161723090553724, max_depth: 9, min_samples_leaf: 4, n_estimators: 830, n_iter_no_change: 49, subsample: 0.6827480531317052, validation_fraction: 0.10498839212016019

The Gradient Boosting model achieved its best performance with a learning rate of 0.27 and 830 estimators, indicating a strong sequential learning process combined with early stopping regularization. Similarly, the Random Forest reached optimal generalization with 978 estimators and a maximum depth of 16, balancing variance reduction and computational cost. For the SVR (known as SVM for regression tasks), the optimal values ($C = 395.7, \gamma = 0.99$) suggest a highly nonlinear mapping, capturing fine-grained

variations in our dataset. On the other hand, the KNN model performed better with only two neighbors and distance-based weighting, reflecting a strong locality in the prediction behavior.

After selecting these configurations, all models were retrained on the full training data and evaluated on the unseen test set (Section 3.3).

3.3. Predictive Performance and Model Analysis

We performed a comparative analysis of machine learning models in order to select the best model among those compared based on the evaluation metrics mentioned in Section 2.3. Table 2 summarizes the results obtained. The downward arrows (↓) indicate that lower values correspond to better performance for error-based metrics, while upward arrows (↑) denote higher performance for R^2.

Table 2 – Comparative analysis of model metrics in the test set.

Models	MAE (↓)	RMSE (↓)	R2 (↑)
SVM	2.568159	3.790073	0.607356
Extra Trees	3.597915	4.778258	0.375916
KNN	**1.821651**	**3.352089**	**0.692861**
Random Forest	3.233330	4.383432	0.474791
Gradient Boosting	2.300572	4.472849	0.453145

Among the tested models, KNN achieved the lowest MAE (1.82) and RMSE (3.35), and the highest determination coefficient ($R^2 = 0.69$), indicating the best generalization capacity on the test set.

The superior performance of KNN suggests that the relationship between features and target values is locally smooth, allowing instance-based models to perform well when appropriate distance metrics and scaling are applied. This is consistent with the results of hyperparameter tuning, in which the optimal KNN configuration used only two neighbors and distance-based weighting, enhancing the model's sensitivity to nearby samples. In contrast, tree-based ensemble models (Random Forest, Extra Trees, and Gradient Boosting) exhibited higher errors (RMSE between 4.3 and 4.7), possibly due to the relatively small dataset size and limited feature interactions, which reduce the benefits of ensemble averaging. The SVR model achieved moderate results ($R^2 = 0.61$), recommending that although its nonlinear RBF kernel was capable of capturing part of the variance, its high penalty parameter $C = 395$ and narrow tolerance $\varepsilon = 0.14$ may have led to overfitting in local regions of the input space. Overall, these results reveal that nonparametric local models (KNN) can outperform more complex ensemble or kernel-based approaches when the feature space is well-behaved and data availability is limited. However, ensemble methods demonstrated greater stability across cross-validation folds, as observed during training, indicating stronger robustness to new data distributions.

Future work will explore hybrid approaches combining instance-based and ensemble learning to achieve both local adaptability and global stability.

4. Conclusion

This study conducted a comprehensive comparative analysis of five machine learning regression models, K-Nearest Neighbors (KNN), Support Vector Regression (SVR), Random Forest (RF), Extra Trees (XT), and Gradient Boosting (GBR), optimized through Randomized Search and evaluated using MAE, RMSE, and R^2 evaluation statistical metrics.

The results demonstrated that the KNN model achieved the best overall predictive performance on the test set (RMSE = 3.35 and R^2 = 0.69), outperforming ensemble and kernel-based approaches. This suggests that the underlying relationship between the input variables and the target output is locally smooth, favoring nonparametric, instance-based methods when data availability is limited. Conversely, ensemble methods such as Random Forest, Extra Trees, and Gradient Boosting yielded higher error metrics, possibly due to the relatively small dataset and the limited number of feature interactions, which limited the benefits of model aggregation.

These findings emphasize the importance of model–data alignment, showing that simpler, localized algorithms can outperform more complex ensembles in specific contexts. Moreover, the hyperparameter optimization procedure proved essential in revealing these performance differences, confirming that model tuning plays a critical role in fair benchmarking.

Future work will focus on extending the analysis to larger, more heterogeneous datasets and integrating additional techniques, such as Bayesian Optimization, cross-model ensembles, and explainable AI methods, to interpret local and global feature contributions. Such advancements are expected to enhance predictive robustness and provide deeper insights into the underlying mechanisms governing the target variable.

5. Acknowledgments

FUNCAP — Ceará Foundation for Scientific and Technological Development (Research and Innovation Network on Renewable Energy, Rede VERDES, grant 07548003/2023), CNPq – National Research Council, Universal — Proc. No. 408431/2023-7 and Research Productivity Grant Proc. No. 307096/2021-1

6. References

Breiman, L., 2001. Random forests. *Machine Learning*, 45(1), pp. 5–32.

Burton, T.L., Jenkins, N., Bossanyi, E., Sharpe, D. and Graham, M., 2021. *Wind Energy Handbook*, 3rd ed. Wiley, Chichester (UK).

Cortes, C. and Vapnik, V., 1995. Support-vector networks. *Machine Learning*, 20(3), pp. 273–297.

De Janeiro, R.J., 2025. *Relatório Síntese / Ano Base 2020.* Empresa de Pesquisa Energética (EPE). Available at: https://www.epe.gov.br/sites-pt/publicacoes-dados-abertos/publicacoes/PublicacoesArquivos/publicacao-819/topico-715/BEB_Summary_Report_2024.pdf , accessed 30 Apr 2025.

Friedman, J.H., 2001. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, 29(5), pp. 1189–1232.

Glauert, H., 1935. Airplane propellers. In: Durand, W.F. (Ed.), *Aerodynamic Theory*, Vol. 4. Springer, Berlin, pp. 169–360. https://doi.org/10.1007/978-3-642-91487-4_3

Hansen, M.O.L., 2015. *Aerodynamics of Wind Turbines*, 3rd ed. Routledge, London and New York.

Javaid, A., Ahmed, S., Rashid, T., Farooq, M. and Shabbir, N., 2022. Forecasting hydrogen production from wind energy in a suburban environment using machine learning. *Energies*, 15(23), p. 8901. Available at: http://dx.doi.org/10.3390/en15238901.

Kumar, R., Singh, B. and Pillai, R., 2018. Wind energy in the built environment: potential applications and challenges. *Renewable and Sustainable Energy Reviews*, 96, pp. 697–713.

Liaw, A. and Wiener, M., 2002. Classification and regression by randomForest. *R News*, 2(3), pp. 18–22.

Moreno-Barea, F.J., Jarne, G. and Gómez-Vela, F., 2020. Improving classification accuracy using data augmentation on small data sets. *Expert Systems with Applications*, 161, 113698. Available at: https://www.sciencedirect.com/science/article/pii/S0957417420305200.

Nai-Zhi, L., 2022. Deep learning-based fault diagnosis for rotating machinery: a survey. *Neural Computing and Applications*, 34(17), pp. 14239–14259. Available at: https://link.springer.com/article/10.1007/s00521-021-06799-6.

Rezaei, S., 2020. Smart buildings and IoT-based energy efficiency: a review of technologies and applications. *Journal of Engineering, Design and Technology*, 19(8), pp. 1722–1743. Available at: https://www.emerald.com/insight/content/doi/10.1108/JEDT-06-2019-0154/full/html.

Wang, Z., Wang, P., Liu, K., Wang, P., Fu, Y., Lu, C-T., Aggarwal, C.C., Pei, J. and Zhou, Y., 2024. A comprehensive survey on data augmentation. *arXiv preprint arXiv:2405.09591.* Available at: https://arxiv.org/abs/2405.09591, accessed 27 Oct 2025.

***Bergstra, J. and Bengio, Y., 2012**. Random search for hyper-parameter optimization. Journal of Machine Learning Research, 13(10), pp. 281–305.*

***Probst, P., Boulesteix, A.-L. and Bischl, B., 2019**. Tunability: importance of hyperparameters of machine learning algorithms. Journal of Machine Learning Research, 20(53), pp. 1–32.*

***Mertens, S., 2006**. Wind Energy in the Built Environment. Multi-Science Publishing, Brentwood.*

***Cover, T.M. and Hart, P.E., 1967**. Nearest neighbor pattern classification. IEEE Transactions on Information Theory, 13(1), pp. 21–27.*

***Geurts, P., Ernst, D. and Wehenkel, L., 2006**. Extremely randomized trees. Machine Learning, 63(1), pp. 3–42.*

***Hastie, T., Tibshirani, R. and Friedman, J., 2009**. The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd ed. Springer, New York.*