mathematics

MDPI

Article

Machine Learning Prediction of Wind Turbine Blade Icing Considering Class Imbalance

Ruslan V. Fedorov *, Yuliya E. Kuvayskova, Anastasiya I. Izyurova, Vladimir N. Klyachkin and Andrei V. Chukalin

Laboratory of Interdisciplinary Problems in Energy Production, Ulyanovsk State Technical University, 32 Severny Venetz Street, 432027 Ulyanovsk, Russia; u.kuvaiskova@ulstu.ru (Y.E.K.)
* Correspondence: r.fedorov@ulstu.ru

Abstract

Wind turbine blade icing is one of the most significant operational problems for wind farms in cold-climate regions, leading to substantial losses in electricity generation, increased mechanical loads, and hazardous situations. Accurate icing prediction is critically important for optimizing wind farm operation and applying preventive measures in a timely manner. This study aims to apply and comparatively analyze sampling methods for predicting wind turbine blade icing under conditions of class imbalance. Six approaches are examined: class weighting, SMOTE, ADASYN, Tomek links, SMOTE-Tomek, and the proposed TaA method. Unlike a simple sequential combination of Tomek links and ADASYN, the TaA method incorporates three novel components: an adaptive Tomek links cleaning threshold that accounts for the local density of the minority class; weighted synthetic sample generation (based on ADASYN) that considers the distance to the nearest majority-class instance; and an iterative cleaning and generation procedure. Feature selection based on the Gini criterion and hyperparameter optimization are employed. Experiments are conducted using real meteorological data from the Ulyanovsk wind farm and its surrounding area. The study addresses both the detection of current icing and forecasting with a horizon of up to nine hours. Three machine learning algorithms—Random Forest, Support Vector Machines (SVM), and Gradient Boosting—are evaluated, alongside a Long Short-Term Memory (LSTM) neural network. Temporal data splitting is used to ensure objective evaluation. Performance is assessed using a set of metrics robust to class imbalance: Precision, Recall, F1-score, Specificity, Balanced Accuracy, MCC, AUC-ROC, and PR-AUC. The results demonstrate that hybrid sampling methods outperform class weighting as well as the individual SMOTE, ADASYN, and Tomek links methods. For the Random Forest model, the best performance is achieved using the TaA method: compared to the baseline model without balancing, the F1-score increases 2.7-fold, and the PR-AUC improves by 42%. The TaA method statistically significantly outperforms SMOTE-Tomek in terms of F1-score for Random Forest and gradient boosting, whereas SMOTE-Tomek proves preferable for the LSTM network. The practical significance of this work lies in the potential to use the obtained results to improve the quality of icing forecasting at wind farms in cold regions.

Keywords: wind farm; wind turbine blade icing; class imbalance; machine learning; hybrid sampling; Tomek links; ADASYN; forecasting

MSC: 68T05; 62H30; 80M50

Check for updates

Academic Editor: Renato Bruni

Received: 12 August 2026
Revised: 30 October 2026
Accepted: 30 October 2026
Published: 3 November 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Wind energy is one of the fastest-growing renewable energy sources in the world. According to the Global Wind Energy Council (GWEC), a record 117 GW of new capacity was installed in 2024, and total global wind power capacity reached 1136 GW. Growth accelerated further in 2025: 165 GW of new capacity was commissioned, which was 40% more than in the previous year, and total global capacity exceeded 1299 GW [1]. According to the GWEC forecast, average annual additions through 2030 will be approximately 194 GW, and total wind power capacity will exceed 2 TW as early as 2029 [1]. In the Russian Federation, according to the Association for the Development of Renewable Energy (ARVE), the installed capacity of wind power plants reached 2.57 GW as of 1 February 2025, while the total renewable-energy generation capacity, excluding large hydropower plants, was 6.59 GW [2].

However, a substantial part of this potential, approximately 20% of all installed wind turbines, is located in cold-climate regions, where low temperatures and atmospheric icing create significant operational problems. The Ulyanovsk wind farm, located on the bank of the Volga River, is a representative example: the region has a temperate continental climate with moderately cold winters, frequent temperature transitions through zero, and considerable precipitation during the winter period. The wind farm consists of 28 wind turbines installed along the Volga River shoreline, which creates specific microclimatic conditions that promote the formation of supercooled precipitation and fog during the cold season. Wind turbine blade icing occurs when supercooled water droplets collide with the blade surface and subsequently freeze, leading to ice accumulation that radically changes the aerodynamic profile of the blades [3,4]. The consequences are serious and multifaceted: reduced electricity-generation efficiency due to changes in aerodynamic characteristics; increased mechanical loads and rotor imbalance due to uneven ice distribution; potential structural damage and premature component failure; safety hazards caused by ice shedding from a great height; and, in severe cases, complete turbine shutdown.

According to industry reports, annual power losses caused by icing can reach 17% for wind farms in cold regions [5].

The ability to accurately predict icing events before they occur would allow wind farm operators to take proactive measures, including adjusting operating strategies, starting de-icing procedures, and planning maintenance, thereby reducing power losses and preventing damage. This has led to substantial research efforts aimed at developing reliable icing-prediction methods.

Existing approaches to the detection and prediction of wind turbine blade icing can be classified into two main categories: direct sensing methods and indirect modelling methods.

Direct sensing methods rely on physical sensors installed on the turbine or its components. These methods include ice-detection sensors [6], acoustic emission [7], vibration analysis [8], and visual monitoring systems [9]. Although such approaches can provide direct and accurate measurements, they require additional capital investment in equipment, involve ongoing operating costs, and may be impractical for existing wind farms without modernization [10].

Indirect modelling methods determine icing conditions from available operational and meteorological data without requiring additional sensors. These methods use Supervisory Control and Data Acquisition (SCADA) systems, which are already installed on modern wind turbines and provide extensive time series of operating parameters such as generated power, wind speed, rotor speed, temperature, and other variables [11]. The appeal of data-driven approaches lies in their use of existing infrastructure, which makes them cost-effective and ready for deployment.

Several methodological approaches have been investigated within the indirect modelling paradigm. Physical models based on computational fluid dynamics (CFD) and blade element momentum (BEM) theory have been used to simulate ice-accretion processes [12]. However, these approaches are computationally intensive and require detailed geometric and environmental input data. Research into the impact of icing on the aerodynamic characteristics of wind turbines [13] warrants particular attention. This study evaluated the effects of icing on the wake behind a vertical-axis wind turbine, revealing additional power loss mechanisms associated with changes in flow structure. These results underscore the need to comprehensively account for both direct losses due to ice accretion and indirect effects linked to alterations in the aerodynamic wake.

ML methods have become a particularly promising alternative because they can identify complex nonlinear relationships between input features and icing events without explicit physical modelling. Study [14] considers ML approaches to wind turbine icing prediction, with a focus on methods using SCADA data. Similarly, article [15] provides a review of prediction methods based on shallow and deep learning and discusses their principles, advantages, and limitations. The application of ML methods to the analysis of air-mass movement in the wind farm area [16] makes it possible to refine the meteorological conditions affecting icing processes and provides a basis for constructing predictive models.

Specific applications of ML include neural-network-based approaches. The authors of [17] used a multilayer perceptron (MLP), trained on meteorological parameters, for rapid prediction of ice accretion, and the MLP model achieved coefficient-of-determination values above 0.95 for predicting both rime ice and glaze ice. Deep learning architectures have also been widely investigated. In [18], an ensemble Parallel CNN–TCN–GRU (PCTG) network was developed that achieved an average prediction accuracy of approximately 97% for prediction horizons of up to two days and maintained an accuracy of at least 95% for horizons of up to 22 days. Study [19] presents an approach using a temporal convolutional network (TCN) for icing prediction based on SCADA data. The authors of [20] developed a wind turbine blade icing diagnosis model based on a hybrid ConvLSTM–GRU architecture. The proposed approach effectively extracts hidden features from SCADA data for accurate diagnosis of the icing state.

Among traditional ML algorithms, the random forest model demonstrates high effectiveness [21]. The effectiveness of this method for diagnosing the state of technical systems is confirmed in [22], which demonstrates the possibility of multiclass classification for recognizing different states of complex objects. In [23], random forest was compared with several other algorithms, including a gated recurrent unit (GRU), a GA–BP neural network, and an extreme learning machine (ELM), and showed the highest prediction accuracy, exceeding 99%. The application of ensemble approaches to technical diagnostic problems is also considered in [24], which demonstrates their robustness to noisy data and their ability to identify nonlinear relationships. SVM was used both independently and in combination with particle swarm optimization for tuning kernel parameters [25], as well as in a hybrid architecture with Bi-LSTM for short-term icing prediction [26]. Gradient boosting methods, particularly LightGBM, were compared with other ensemble approaches for blade icing detection [27].

Despite the impressive success of ML methods, substantial class imbalance inherent in the operational data of wind power installations remains one of the key problems. Blade icing is a rare phenomenon compared with normal operating conditions. In real datasets, the proportion of such anomalies generally does not exceed 5–10%. In the considered Ulyanovsk wind farm dataset, this value is approximately 8% of all observations.

Such a pronounced class imbalance becomes a serious obstacle for standard ML algorithms. Models constructed from imbalanced samples are naturally biased toward the

prevailing class, namely the absence of icing. This allows them to achieve high overall accuracy through error-free recognition of frequent situations, but it almost eliminates their ability to detect rare but critical icing events. The paradox is that these particular cases, which require an immediate response, are the least recognizable for the model. According to previous studies, standard models trained on imbalanced data can incorrectly classify up to 70% of such emergency states [28].

The problem of class imbalance in wind turbine blade icing detection is widely recognized in the current literature, and a number of strategies have been proposed to overcome it. They can conventionally be divided into several groups.

The most common approach is based on oversampling methods for the minority class. For example, in [29], the SMOTE algorithm was applied to balance monitoring data, which made it possible to achieve precision and recall values of 96.48% and 97.86%, respectively; a SMOTE–CNN modification for direct icing detection was also considered in that study. In [30], a Random–SMOTE modification was proposed in which synthetic samples are generated with consideration of the standard deviation of the features, which makes it possible to preserve the data structure and improve classification quality. In [20], the ADASYN method was used to generate synthetic samples. Further development of these ideas is presented in [27], where the γ-MiniDPC–SMOTE algorithm is proposed, which first divides icing examples into clusters before oversampling, and in [31], where MBK–SMOTE is combined with random forest to predict icing events.

Hybrid and generative models form a separate group. The effectiveness of combining undersampling with ensemble methods is demonstrated in [32], where a hybrid approach based on a multilayer perceptron, undersampling, and bagging substantially improved classification quality on real imbalanced data. In [33], a diagnostic system based on a Conditional Tabular Generative Adversarial Network (CTGAN) and LightGBM was developed, providing high performance even with a limited amount of training data. For high-dimensional imbalanced samples, Ref. [34] proposed a multiclass data-augmentation and fault-diagnosis model based on ISOMAP–CGAN. In [25], the authors used undersampling and oversampling techniques at the data-preprocessing stage to balance datasets containing icing and non-icing cases. For extreme imbalance, with class ratios of up to 1:100, Ref. [35] developed the Enhanced Tree Ensemble (ETE) method, which showed high effectiveness on strongly imbalanced datasets and confirmed the appropriateness of specialized ensemble approaches for rare-event prediction. Further development of hybrid approaches is presented in [36], where the Balanced Hoeffding Tree Forest (BHTF) structure is proposed, combining oversampling and undersampling for multi-purpose fault diagnosis. This approach demonstrated high effectiveness with limited training samples, confirming the prospects of combining different sampling strategies.

In parallel, approaches that take data heterogeneity into account are being developed. Thus, Ref. [37] present prototype federated-learning models specially adapted for class-imbalanced data in distributed environments. Alternative strategies for classifying imbalanced data are presented in [38], where an extended Broad TSK Fuzzy Classifier (B-TSK-FC) with special adaptation of rule weights for the minority class is proposed. This approach showed high effectiveness when working with complex data structures characteristic of technical diagnostic problems.

Study [39] provides a systematic review of data-driven methods for wind turbine blade icing detection, confirming the relevance and diversity of the considered problem.

Although current studies cover a wide range of wind turbine blade icing prediction problems, from algorithm selection and feature construction to data preprocessing, several substantial gaps remain unresolved. In particular, no systematic comparison of the effectiveness of different hybrid sampling strategies specifically for this problem has yet

been carried out. In addition, the combination of Tomek-link-based undersampling with ADASYN oversampling, which is the TaA method proposed by the authors, has not previously been tested on wind turbine icing data. It is important to note that the TaA method is not a simple sequential combination of Tomek links and ADASYN; it incorporates three original components that ensure adaptability to the local data structure. Finally, the literature lacks comprehensive methodologies that combine feature selection, hybrid sampling, and hyperparameter optimization within a single approach.

This study aims to apply and comparatively analyze sampling methods for forecasting wind turbine blade icing under conditions of class imbalance. The study's primary contribution is as follows: a modification of the TaA method is proposed, featuring an adaptive threshold for Tomek links cleaning and weighted synthetic sample generation via ADASYN; this approach preserves class boundary information in regions with high minority-class density. A systematic comparison of six class-balancing strategies—class weighting, SMOTE, ADASYN, Tomek links, SMOTE–Tomek, and TaA—was conducted using a unified experimental protocol for the task of wind turbine icing detection. This unified protocol incorporates temporal data splitting, feature selection performed solely on the training set, and sampling applied exclusively to the training data, thereby ensuring a valid comparison among the strategies. To assess the robustness of differences between hybrid methods, a statistical analysis was performed using the Wilcoxon signed-rank test. Additionally, classical machine learning algorithms were compared with an LSTM network using the same meteorological dataset, enabling a comparison of classical and neural network approaches in the context of class imbalance and limited sample size. The practical significance of this work lies in the potential to use the findings to improve icing forecasting accuracy at wind farms located in cold regions.

The article is structured as follows. Section 2 presents the mathematical problem formulation and describes the methods employed, ranging from preprocessing and feature selection to class-balancing strategies and hyperparameter optimization. Section 3 details the source data, the experimental setup, and the results obtained. The conclusion summarizes the study's key findings and outlines promising directions for future research.

2. Materials and ingMethods

2.1. Problem Formulation

Let the meteorological state of the wind farm area be characterized by the set of observed parameters $X = (X_1, X_2, \ldots, X_p)$ recorded at n weather stations surrounding the wind farm and at the reference station located at the wind farm itself.

The data are represented as a time series in which each time instant t is associated with a vector of meteorological parameters X_t.

The objective of this study is to forecast icing of wind turbine blades over a prediction horizon of h steps. Based on the meteorological parameters X_t available at time t, the presence of blade icing after h steps is determined as

$$Y_{t+h} = f(X_t), \qquad Y_{t+h} \in \{0, 1\}, \qquad h \geq 0, \tag{1}$$

where $Y_{t+h} = 1$ indicates the presence of blade icing conditions after h steps, whereas $Y_{t+h} = 0$ indicates their absence.

For $h = 0$, the problem corresponds to detecting the current icing state from meteorological data available at the same time instant. For $h > 0$, the problem corresponds to forecasting blade icing over a prediction horizon of h steps.

In this study, four prediction horizons are considered: $h = 0$ (detection), $h = 1$, $h = 2$, and $h = 3$ steps. The horizon $h = 0$ is used to evaluate model performance and to compare sampling methods, since it allows the ability of different approaches to distinguish between

icing and non-icing conditions to be assessed. The horizons $h = 1$, $h = 2$, and $h = 3$ are used to evaluate the possibility of forecasting icing in advance, which is particularly important for the timely implementation of preventive measures by wind farm operators.

To construct datasets for forecasting with a horizon of h, the target variable is shifted backward by h steps. Thus, for each time instant t, a pair (X_t, Y_{t+h}) is formed, while the last h observations are excluded from the sample because the corresponding values of Y_{t+h} are unavailable. A separate model is trained for each prediction horizon h, since the target variable Y_{t+h} differs for different values of h.

The data are represented by:

- An observation matrix $X \in \mathbb{R}^{N \times p}$, where N is the number of time points and p is the number of input features;
- A target-variable vector $Y \in \{0,1\}^N$, where $y_i = 1$ if icing occurred at time i.

The class-imbalance problem arises when the proportion of positive instances is substantially smaller than the proportion of negative instances:

$$r = \frac{N_+}{N}; N = N_+ + N_- \tag{2}$$

where N_+ is the number of icing observations and N_- is the number of non-icing observations.

For the Ulyanovsk wind farm data, $r \approx 0.08$, which means that icing observations account for only approximately 8% of all data.

The objective is to construct a predictive function $f : \mathbb{R}^p \to \{0,1\}$ that minimizes the prediction error for the minority class while maintaining acceptable performance for the majority class.

2.2. Feature Selection Based on Gini Importance

Feature selection is a crucial step in building machine learning models, as it enables the reduction in feature space dimensionality, mitigates the risk of overfitting, and enhances model interpretability.

In this study, Gini importance—implemented via the Random Forest algorithm [21, 40]—is used to assess feature importance. This method was chosen because Gini importance is a built-in feature of Random Forest and requires no additional computations. Furthermore, the method ensures robustness against class imbalance through the use of bootstrap sampling and ensemble averaging.

For each decision tree T_b in the ensemble, the Gini impurity at node t is defined as:

$$G(t) = 1 - \sum_{k=0}^{1} p_k^2. \tag{3}$$

where p_k is the proportion of observations belonging to class k at node t.

The decrease in Gini impurity achieved by splitting node t using feature X_j is:

$$\Delta G(t, X_j) = G(t) - \frac{N_L}{N} G(t_L) - \frac{N_R}{N} G(t_R). \tag{4}$$

where N_L and N_R are the numbers of observations in the left and right child nodes, respectively, and t_L and t_R are the child nodes.

The importance of feature X_j is calculated as the mean decrease in Gini impurity over all nodes in which X_j is used for splitting, averaged over all trees in the forest:

$$I(X_j) = \frac{1}{B} \sum_{b=1}^{B} \sum_{t \in \tau_b} \Delta G(t, X_j)\, I(X_j \text{ splits } t). \tag{5}$$

where B is the number of trees and τ_b is the set of nodes in tree b.

Features with the largest importance values are selected for model construction because they provide the greatest discriminatory ability for distinguishing icing and non-icing states.

Despite the well-known limitations of Gini importance related to its bias toward continuous features and categorical variables with many levels [41], the present study considers a binary target variable and a limited set of meteorological parameters, which reduces the influence of these biases.

To assess the robustness of the obtained results, Gini importance is compared with permutation importance, which is less susceptible to the aforementioned biases [21,42]. Permutation importance evaluates the contribution of each feature by randomly permuting its values and measuring the resulting decrease in model performance. For each feature X_j, the permutation procedure is repeated K times, after which the average decrease in the target performance metric is calculated:

$$PI(X_j) = \frac{1}{K}\sum_{k=1}^{K}\left(S - S_{j,k}\right), \tag{6}$$

where S denotes the original model performance, and $S_{j,k}$ is the model performance after the k-th permutation of feature X_j.

2.3. ML Models

For icing prediction in this study, three ML algorithms were selected: random forest, support vector machine, and gradient boosting. This choice is based on their proven effectiveness in wind turbine condition diagnostics and their confirmed suitability for constructing reliable models with limited training samples [43,44]. Additionally, an LSTM network is considered to compare classical and neural network approaches under conditions of class imbalance.

Random forest is an ensemble learning method that constructs multiple decision trees during training and outputs the class that is the mode of the classes predicted by the individual trees. For a given input vector x, the prediction is:

$$\hat{y}(x) = \text{mode}\{T_b(x)\}_{b=1}^{B}. \tag{7}$$

where $T_b(x)$ is the prediction of the bth tree.

Classification in random forest is performed by aggregating the decisions of the individual trees. Each tree T_i in an ensemble of B trees calculates a vector of posterior probabilities that an object characterized by the feature vector x belongs to each class. The value $p_{ik} = P(y = k \mid x, T_i)$ is an estimate of the probability that the object belongs to class k, obtained from the ith tree.

The probability that the object belongs to class k is determined by averaging the probabilities over all trees:

$$P_k(x) = \frac{1}{B}\sum_{i=1}^{B} p_{ik}. \tag{8}$$

The final class prediction is determined according to the maximum-probability rule:

$$\hat{y}(x) = \arg\max_{k\in\{0,1\}} P_k(x). \tag{9}$$

The first important hyperparameter of the random forest model is the number of trees B in the ensemble, which determines the complexity of the composition. As B increases,

model accuracy asymptotically approaches a limiting value, while the computational complexity of the algorithm grows linearly with the training-sample size n, which must be considered when selecting the optimal value.

The second significant hyperparameter is the maximum tree depth $d_{\max}$, which controls the complexity of the base classifiers. Deeper trees provide a better fit to the training data but simultaneously increase the risk of overfitting, which is expressed as a reduction in the model's generalization ability on the test sample. Limiting the number of leaves makes it possible to control the balance between accuracy and the generalization ability of the model.

The third parameter is the number of candidate features m considered at each node split, which substantially affects the degree of decorrelation among trees in the ensemble. The recommended value for classification problems provides a balance between tree diversity and their individual accuracy [45].

The support vector machine finds the optimal separating hyperplane that maximizes the margin between the classes. For the nonlinear case, SVM uses the kernel trick to map the input features into a higher-dimensional space.

The optimization problem for SVM with class weights is

$$\min_{w,b,\xi}\left\{\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N} w_i\xi_i\right\}. \tag{10}$$

subject to the constraints:

$$y_i\left(w^T\phi(x_i) + b\right) \geq 1 - \xi_i, \qquad \xi_i \geq 0. \tag{11}$$

where w is the weight vector, b is the bias, ξ_i are slack variables, C is the regularization parameter, w_i are class weights, and $\phi(\cdot)$ is the mapping function.

For the nonlinear case, the radial basis function (RBF) is used as the kernel:

$$K(x_i, x_j) = \exp\left(-\gamma\|x_i - x_j\|^2\right). \tag{12}$$

After training M classifiers, the probability of belonging to class k is calculated from the distances to the separating hyperplanes:

$$P_k(x) = \frac{1}{1 + \exp\left[-\left(w_k^T\phi(x) + b_k\right)\right]}. \tag{13}$$

The final decision is made according to the maximum-probability rule.

The central hyperparameter in the SVM model is the regularization coefficient C, which controls the trade-off between accuracy on the training sample and the model's generalization ability. This parameter determines the "hardness" of the separating hyperplane and makes it possible to control the acceptable degree of overfitting. The second parameter is the coefficient γ of the radial basis function.

Gradient boosting constructs an ensemble of weak learners, usually decision trees, sequentially, with each subsequent model correcting the errors of its predecessors.

For binary classification with a logarithmic loss function, the model is constructed iteratively. At step m, a new tree $h_m(x)$ is added to approximate the negative gradient of the loss function:

$$F_m(x) = F_{m-1}(x) + uh_m(x). \tag{14}$$

where u is the learning rate, also called the shrinkage parameter.

The loss function for binary classification takes the form of the cross-entropy criterion:

$$L\big(y, F(x)\big) = -y \log p(x) - (1-y) \log\big(1 - p(x)\big). \tag{15}$$

where $p(x) = \dfrac{1}{1+\exp[-F(x)]}$ is the probability of belonging to the positive class.

The hyperparameters that must be optimized to improve the algorithm include the learning rate u, the number of trees, tree depth d, and regularization-related parameters such as the minimum number of samples required to split a node and the minimum number of samples in a leaf node. Regularization helps balance model complexity and generalization ability. High values of the regularization parameters can lead to underfitting, whereas low values increase the risk of overfitting.

The optimal value of the maximum tree depth $d_{\max}$ is determined by the relation [46]:

$$d_{\max} \approx \log_2(M) + c. \tag{16}$$

where c is a constant that depends on data complexity.

It should be noted that increasing tree depth allows the model to identify more complex relationships in the data but simultaneously increases the risk of overfitting. Conversely, insufficient depth can produce an oversimplified model that is unable to adequately represent important patterns.

The learning rate u is a key parameter that determines the contribution of each tree to the final model. Smaller values of u require more trees to achieve optimal performance but help improve the model's generalization ability. At the same time, excessively high learning rates can cause premature overfitting.

The number of trees in the ensemble directly affects prediction accuracy. Increasing this parameter can potentially improve model quality but also increases the risk of overfitting and computational costs. Conversely, an insufficient number of trees can lead to underfitting.

An LSTM network [18,47] is used to compare classical machine learning algorithms with neural network-based approaches. LSTM is a type of recurrent neural network capable of effectively modeling long-term temporal dependencies. A key feature of LSTM is its gating mechanism, which regulates the flow of information through the state cell, allowing significant information to be retained over long time intervals while avoiding the vanishing gradient problem.

The LSTM architecture employed consists of a single recurrent layer and a fully connected output layer with a sigmoid activation function. The input sequence length (window) is 4 observations; given a 3 h data collection interval, this corresponds to a 12 h temporal context. Sequences are constructed strictly within each dataset (training, validation, or test); sequences crossing dataset boundaries are excluded to prevent temporal data leakage. When constructing the target variable for a prediction horizon h, the last h observations of each dataset are excluded, since the corresponding values of Y_{t+h} are unavailable. Training is performed using the Adam optimizer and a binary cross-entropy loss function. Dropout and early stopping based on validation error are applied to prevent overfitting.

The LSTM hyperparameters include the number of neurons in the recurrent layer, the dropout probability, and the learning rate. The number of neurons determines the model's capacity: too few neurons lead to underfitting, while an excessive number leads to overfitting. The dropout probability controls the level of regularization, and the learning rate determines the weight update step size: high values cause instability, while low values slow down convergence.

2.4. ingHyperparameter Optimization

The quality and generalization ability of ML models directly depend on the choice of hyperparameters; therefore, their optimizat,on is a critically important stage. In this study, two approaches are used for this purpose: random search and Bayesian optimization, each of which has its own area of effective application.

Random search performs stochastic sampling of hyperparameter combinations from a specified space. For a hyperparameter space θ, K random combinations are evaluated, after which the combination that maximizes the cross-validation score is selected. This approach is particularly effective for multidimensional spaces and, as shown in [48], outperforms exhaustive grid search with a substantially smaller number of evaluations.

Bayesian optimization constructs a probabilistic surrogate model of the objective function and uses it to select the most promising hyperparameters for evaluation at the next step. The surrogate model is usually a Gaussian process:

$$f(\theta) \sim GP\big(\mu(\theta), K(\theta, \theta')\big). \tag{17}$$

where $\mu(\theta)$ is the mean function and $K(\theta, \theta')$ is the covariance, or kernel, function.

The Expected Improvement (EI) acquisition function guides the search:

$$EI(\theta) = \mathbb{E}\big[\max\big(f(\theta) - f(\theta^+), 0\big)\big]. \tag{18}$$

where θ^+ is the best hyperparameter combination found up to the current step.

For a Gaussian process, this expression has an analytical form:

$$EI(\theta) = \big(\mu(\theta) - f(\theta^+) - \xi\big)\Phi(Z) + \sigma(\theta)\phi(Z). \tag{19}$$

where $Z = \dfrac{\mu(\theta) - f(\theta^+) - \xi}{\sigma(\theta)}$, Φ and ϕ are the cumulative distribution function and probability density function of the standard normal distribution, respectively, and ξ controls the balance between exploration and exploitation.

Thus, the EI function makes it possible to assess how advantageous it is to explore a new point θ on the basis of the current function estimates and their uncertainty. After the objective function is evaluated at the new point, the data are updated and the process is repeated until the specified stopping criterion is reached.

The advantage of the method is its ability to find the global optimum with a relatively small number of objective-function evaluations, which is especially important for computationally expensive ML models. However, the effectiveness of Bayesian optimization decreases in high-dimensional parameter spaces, where its convergence rate can be lower than that of random search.

2.5. Sampling Methods

Balancing the training sample is a critical step when working with imbalanced data. In this study, six class-balancing strategies are considered: class weighting, SMOTE, ADASYN, Tomek links, SMOTE–Tomek, and the proposed TaA method.

Class weighting (*class weights*) assigns a weight to each class inversely proportional to its frequency. The class weights are calculated as

$$w_k = \frac{N}{M \cdot N_k}, \tag{20}$$

where N is the total number of observations, M is the number of classes, and N_k is the number of observations belonging to class k.

This approach does not modify the original data and only adjusts the loss function, which makes it computationally efficient.

SMOTE (*Synthetic Minority Over-sampling Technique*) generates synthetic minority-class samples by linear interpolation between an observation x_i and a randomly selected neighboring observation from the same class:

$$x_{\text{new}} = x_i + \lambda \cdot (x_{i,\text{NN}} - x_i), \tag{21}$$

where $x_{i,\text{NN}}$ is one of the k nearest neighbors of x_i belonging to the minority class, and $\lambda \in [0, 1]$ is a random number.

ADASYN (*Adaptive Synthetic Sampling*) is a modification of SMOTE that adaptively distributes the generation of synthetic samples. For each minority-class observation x_i, an imbalance coefficient is calculated as

$$r(x_i) = \frac{\Delta_i}{k}, \tag{22}$$

where Δ_i is the number of majority-class observations among the k nearest neighbors.

The number of synthetic samples generated for x_i is determined as

$$g_i = \hat{G} \cdot \frac{r_i}{\sum_{j=1}^{N_{\text{min}}} r_j}, \tag{23}$$

where $\hat{G}$ is the total number of synthetic samples required to achieve complete balancing of the dataset:

$$\hat{G} = N_{\text{maj}} - N_{\text{min}}, \tag{24}$$

and N_{maj} and N_{min} are the numbers of majority- and minority-class observations, respectively.

Tomek links remove pairs of observations belonging to different classes, (x, x'), that are located in close proximity to each other. A Tomek link is formed by a pair of observations (x, x') belonging to different classes for which there is no other observation x'' located closer to x than x'. Pairs identified in this way indicate regions of class overlap. The majority-class observations participating in Tomek links are removed.

SMOTE–Tomek is a hybrid strategy combining synthetic oversampling of the minority class using SMOTE with subsequent dataset cleaning using Tomek links [49]. At the first stage, synthetic samples are generated, whereas at the second stage, majority-class boundary observations are removed. SMOTE–Tomek not only balances the classes by generating synthetic samples but also improves class separability by removing ambiguous and noisy observations, thereby contributing to improved generalization performance of subsequent machine-learning models.

The authors of the present study propose a hybrid sampling method, TaA (*Tomek links with Adaptive Threshold and ADASYN*), designed for balancing highly imbalanced datasets. The method combines boundary cleaning based on Tomek links with adaptive synthetic oversampling using ADASYN, but differs from a simple sequential combination of these methods by three key components that provide a higher quality of class balancing.

The first component is an adaptive cleaning threshold. At the first stage, the Tomek links algorithm is applied to identify pairs of observations located in close proximity to each other. Majority-class samples participating in such pairs are removed from the training set. Unlike the classical approach, not all Tomek links located near the minority class are removed. For each minority-class sample x_i, the imbalance coefficient $r(x_i)$ is calculated

according to Equation (22). The value of r_i lies in the range $[0, 1]$: the higher its value, the more majority-class neighbors surround the given observation and the more difficult it is to classify.

The adaptive threshold τ is defined as the median value of r_i over all minority-class samples:

$$\tau = \text{median}\{r_i,\ i = 1, \ldots, N_{\text{min}}\}. \tag{25}$$

Tomek links are removed only when the corresponding minority-class sample satisfies the condition $r_i > \tau$. This makes it possible to preserve information about the class boundary in regions where the minority class is already sufficiently represented while focusing the cleaning procedure on the most problematic regions.

The advantage of this component is the reduced risk of losing informative majority-class samples and the preservation of a representative class boundary.

The second component is weighted generation of synthetic samples. At the second stage, the ADASYN algorithm is applied, which generates synthetic samples predominantly in regions with a low density of the minority class. The number of synthetic samples generated for x_i is proportional not only to the imbalance coefficient r_i, but also to the normalized distance from x_i to the nearest majority-class sample d_i:

$$g_i = \hat{G} \cdot \frac{r_i d_i}{\sum_{j=1}^{N_{\text{min}}} r_j d_j}, \tag{26}$$

where $\hat{G}$ is the total number of synthetic samples required to achieve complete balancing.

The distance normalization is required to map d_i to the range $[0, 1]$, consistent with the range of the imbalance coefficient $r_i \in [0, 1]$; otherwise, the scale of the distances could dominate r_i and distort the distribution of the generated samples.

The introduction of the multiplier d_i makes it possible to generate more samples in regions where the class boundary is located farther away, and fewer samples in regions where the classes already overlap. The advantage of this component is the targeted "expansion" of the class boundary rather than merely filling the feature space.

The third component is an iterative procedure. After the generation of synthetic samples, the Tomek links evaluation is repeated and, if necessary, additional cleaning is performed. The procedure is repeated for up to three iterations or until the class ratio stabilizes:

$$\frac{N_{\text{maj}}^{(k)} - N_{\text{min}}^{(k)}}{N_{\text{min}}^{(k)}} < \varepsilon, \tag{27}$$

where ε is the stabilization threshold, for example, $\varepsilon = 0.01$.

The advantage of this component is its ability to adapt to changes in the class distribution after each iteration and to provide more complete balancing.

Thus, the TaA method differs from a simple sequential combination of Tomek links and ADASYN by three original components: an adaptive cleaning threshold, weighted generation of synthetic samples, and an iterative procedure. These components make it possible to account for local characteristics of the class distribution, reduce the risk of losing informative observations, and improve the quality of balancing in the case of strong class imbalance.

In previous studies on hydraulic system diagnostics [50], it was shown that the TaA method outperforms both the classical SMOTE–Tomek approach and the individual SMOTE, ADASYN, and Tomek links methods according to the key classification metrics.

2.6. Model Performance Evaluation Metrics

For binary classification with imbalanced data, accuracy is not an informative metric because a model can achieve a high value by correctly recognizing the majority class while completely ignoring the minority class. In this study, a set of metrics is used for a comprehensive assessment of prediction quality and for a detailed characterization of the model's ability to detect icing events [28,51].

Precision characterizes the proportion of objects assigned by the model to the positive class that actually belong to that class:

$$\text{Precision} = \frac{TP}{TP + FP}. \tag{28}$$

where TP (True Positive) is the number of correctly classified positive-class objects; FP (False Positive) is the number of negative-class objects incorrectly assigned to the positive class; and FN (False Negative) is the number of positive-class objects incorrectly assigned to the negative class.

Recall reflects the proportion of true positive-class objects that were correctly identified by the model:

$$\text{Recall} = \frac{TP}{TP + FN}. \tag{29}$$

The F1-score is the harmonic mean of precision and recall and makes it possible to evaluate the balance between these two metrics:

$$F1\text{-score} = 2\frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{30}$$

Specificity indicates the proportion of correctly classified negative samples:

$$\text{Specificity} = \frac{TN}{TN + FP}, \tag{31}$$

where TN (*True Negative*) denotes the number of correctly classified negative samples.

In rare-event detection tasks, Specificity is important for assessing how often the model incorrectly predicts icing when it is actually absent. An excessively low Specificity indicates a large number of false alarms, which reduces the practical value of the model.

Balanced Accuracy, unlike conventional accuracy, does not depend on the class ratio and is calculated as the arithmetic mean of Recall and Specificity:

$$BA = \frac{\text{Recall} + \text{Specificity}}{2}. \tag{32}$$

MCC (*Matthews Correlation Coefficient*) is the Matthews correlation coefficient, which takes into account all elements of the confusion matrix:

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}. \tag{33}$$

The MCC value ranges from $[-1, 1]$, where 1 corresponds to perfect prediction, 0 to random guessing, and -1 to complete disagreement between predictions and true labels.

MCC and Balanced Accuracy account for both classes, which is particularly important under class imbalance. These metrics identify models that correctly classify the minority class without a substantial deterioration in the classification of the majority class.

AUC-ROC (*Area Under the Receiver Operating Characteristic Curve*) is the area under the ROC curve, which measures the ability of the model to distinguish between classes across all possible classification thresholds. AUC-ROC values lie in the interval $[0.5, 1.0]$:

the closer the value is to 1, the higher the discriminative ability of the model, whereas a value of 0.5 corresponds to random guessing.

PR-AUC (*Area Under the Precision–Recall Curve*) is the area under the Precision–Recall curve and is particularly informative for strongly imbalanced datasets. Unlike AUC-ROC, PR-AUC is more sensitive to changes in the minority class, which makes it preferable for rare-event detection tasks.

Thus, under class imbalance, it is advisable to use a set of complementary metrics: Recall to assess the ability to detect rare events, F1-score to balance Precision and Recall, PR-AUC for an integral evaluation that accounts for imbalance, MCC and Balanced Accuracy to account for both classes, and Specificity to control false alarms.

To assess the statistical significance of differences between sampling methods, the Wilcoxon signed-rank test for paired samples is used. The test statistic is calculated as

$$W = \sum_{i=1}^{n} R_i \cdot I(d_i > 0), \tag{34}$$

where d_i is the difference between the metric values obtained by the compared methods in the i-th run, R_i is the rank of the absolute value of the difference, and n is the number of runs.

The null hypothesis H_0, stating that the median difference is equal to zero, is rejected when $p < 0.05$.

2.7. Flowchart

Figure 1 presents a flowchart of the proposed methodology, which consists of six consecutive stages: data preprocessing and formation of the input feature vector; splitting the data into training, validation, and test sets; feature selection based on the Gini criterion performed using the training set; class balancing using one of six methods (class weighting, SMOTE, ADASYN, Tomek links, SMOTE–Tomek, or TaA) applied only to the training set; training machine-learning models with hyperparameter optimization on the validation set; and final performance evaluation on the test set with calculation of the selected metrics and statistical assessment of the significance of the observed differences.

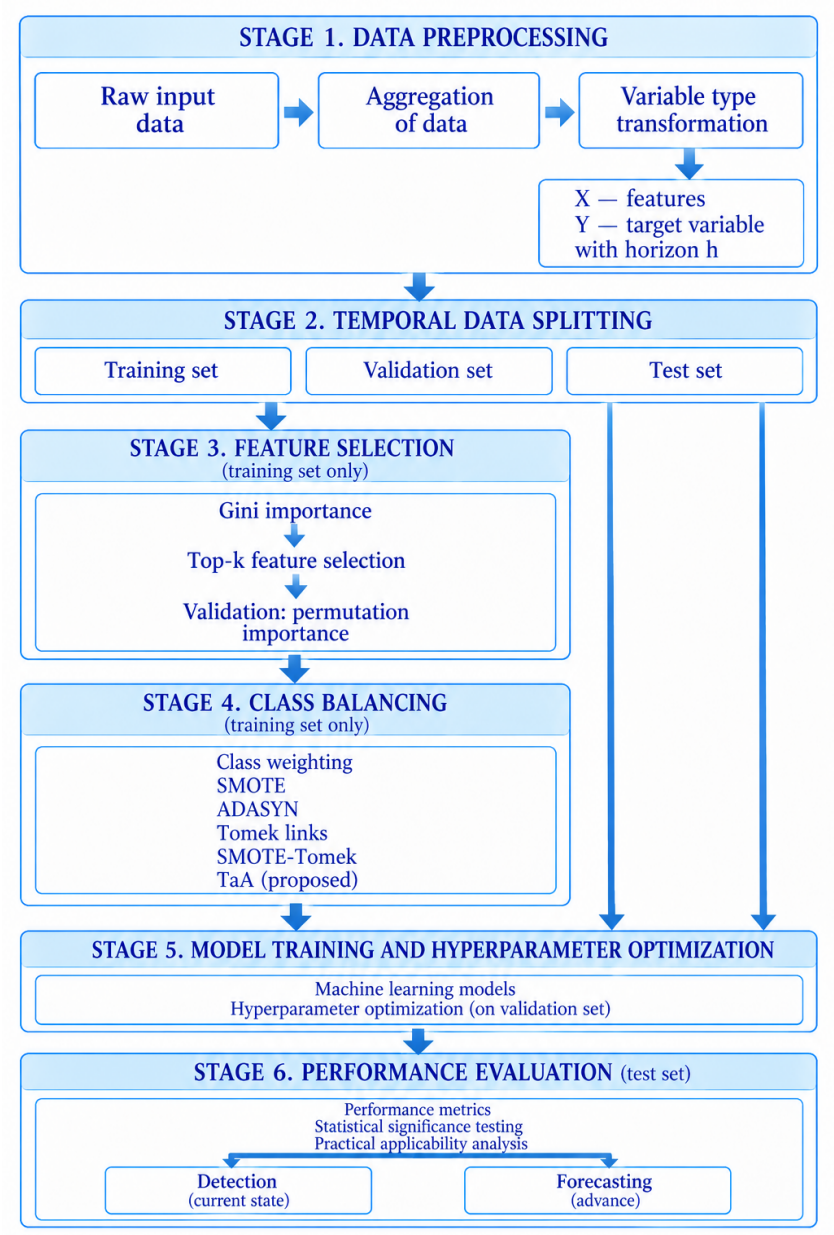


Figure 1. Flowchart of the proposed methodology.

3. Research Results

3.1. Data Description

The study was performed using meteorological data from the Ulyanovsk wind farm and its surroundings. The region has a temperate continental climate with cold winters and warm summers, with prevailing south-westerly and southerly winds.

The stations are distributed around the wind farm, which makes it possible to record meteorological parameters from different directions and provides spatial coverage for forecasting approaching weather fronts.

Each of the six surrounding stations records eight meteorological parameters characterizing the state of the atmosphere. These include air temperature (T, °C), atmospheric pressure (P_0, mmHg), relative humidity (U, %), wind direction (DD), wind speed (Ff, m/s), weather phenomenon code ($W1$: 0—no precipitation, 1—liquid precipitation, 2—solid precipitation, 3—other phenomena), dew-point temperature (T_d, °C), and precipitation amount (RRR, mm/h). In addition, for each station, a binary icing indicator is recorded (1—present,

0—absent), reflecting the occurrence of icing at the given station. The reference station in Ulyanovsk records a similar set of meteorological parameters (T, P_0, U, DD, Ff, $W1$, T_d, RRR) and the target variable Y (icing at the Ulyanovsk wind farm: 1—present, 0—absent).

The data were recorded every 3 h during the cold period (November–March) from 2020 to 2025. The total sample size is 7248 observations.

It should be noted that weather stations record meteorological parameters at a height of approximately 2 m, whereas wind turbine blades operate at heights of 80–120 m. This creates a systematic error associated with the vertical gradients of temperature, humidity, and wind speed. This effect is partly compensated for by using data from six stations surrounding the wind farm and icing indicators at neighboring stations, which indirectly reflect the vertical atmospheric profile. To further improve the accuracy, it is necessary to integrate SCADA data from the turbines themselves and measurements at hub height.

Among the listed parameters, several types can be distinguished. Continuous variables include temperature (T), pressure (P_0), humidity (U), wind speed (Ff), dew-point temperature (T_d), and precipitation amount (RRR). Wind direction (DD) is a cyclic variable. The weather phenomenon code ($W1$) is a categorical variable. Binary variables are the icing indicators at the stations and the target variable (Y).

The variables were transformed as follows. Continuous variables (T, P_0, U, Ff, T_d, RRR) were subjected to Z-score normalization. The cyclic variable DD was transformed into two components (sin and cos). The categorical variable $W1$ was encoded using one-hot encoding (4 categories). Binary icing indicators remained unchanged.

After generating synthetic samples using SMOTE, ADASYN, and the hybrid SMOTE-Tomek and TaA methods, additional post-processing is performed to ensure that the generated values are admissible. For categorical features represented by one-hot encoding (weather phenomenon code W1), the component with the maximum value is selected from the four components and set to 1, while the remaining three are set to 0. This guarantees that exactly one category is active and that the sum of the components equals one. In the case of equal values, a deterministic rule is applied: the component with the smallest index is selected. For binary features (the icing indicator at a meteorological station), a threshold transformation is applied: values greater than or equal to 0.5 are set to 1, while values below 0.5 are set to 0. For the cyclic wind direction components (sin and cos), renormalization is performed after interpolation by dividing the components by their Euclidean norm, ensuring that the identity $\sin^2(DD) + \cos^2(DD) = 1$ is satisfied. If the Euclidean norm of the vector is close to zero (less than 10^{-6}), the vector is replaced by the unit vector $(\sin(DD_{\text{ref}}), \cos(DD_{\text{ref}}))$, where DD_{ref} is the wind direction of the nearest real observation in the training dataset, determined using the Euclidean metric in the feature space.

Continuous meteorological variables are constrained to physically admissible ranges determined from the training dataset (minimum and maximum values with a small margin), preventing the generation of physically implausible synthetic values.

Validation of the synthetic data was performed. All synthetic samples had valid one-hot encoded, binary, and cyclic feature values after post-processing. For SMOTE and SMOTE-Tomek, the distributions of synthetic samples across W1 categories and the binary icing feature were consistent with those of the minority class in the original data (chi-square test, $p > 0.05$). For ADASYN and TaA, visual analysis and the values of the coefficient $r(x)$ indicated a shift toward regions with a low density of minority-class observations, consistent with the adaptive nature of these methods. Comparison of the marginal distributions of continuous features using the Kolmogorov–Smirnov test showed no statistically significant differences between the original and synthetic samples ($p > 0.05$). The physical admissibility of synthetic values was ensured by the post-processing constraints described above.

It should be noted that the described post-processing procedure is heuristic and does not guarantee exact preservation of the joint feature distribution. Nevertheless, the validation demonstrated that the admissibility of feature values and the marginal distributions meet the requirements of the considered problem.

After transformation, the input feature vector consists of 90 features. Each of the six surrounding stations is represented by thirteen features: six continuous meteorological parameters (temperature, pressure, humidity, wind speed, dew-point temperature, and precipitation), two cyclic wind-direction components (sin and cos), four binary weather-code features obtained by one-hot encoding, and one binary icing indicator for the station. Thus, the contribution of the surrounding stations amounts to 78 features. The reference station in Ulyanovsk adds twelve features: six continuous meteorological parameters, two cyclic wind-direction components, and four binary weather-code features.

The target variable "icing" was constructed using data from the wind turbine monitoring system, which includes icing sensors installed directly on the blades. These sensors operate continuously and transmit data to the wind turbine telemetry system every 10 s. When icing is detected, the monitoring system records the event and automatically initiates activation of the blade heating system. Thus, the monitoring system provides continuous instrumental detection of icing in real time and serves as the primary source of information on icing occurrence.

Additionally, when the monitoring system is triggered, the wind farm operation and maintenance personnel visually inspect the blades and prepare an inspection report documenting the presence of icing. Visual inspection reports contain a binary conclusion regarding the presence or absence of blade icing and are dated by calendar day. Furthermore, reports are prepared during each scheduled inspection, which is conducted at least once a day during the cold season. Visual inspection reports serve as a supplementary source of confirmation: they confirm the occurrence of icing on the corresponding calendar day but do not establish the exact time within that day and are not directly converted into three-hour labels.

The monitoring system data and visual inspection reports were aligned to a common three-hour time interval corresponding to the temporal resolution of the meteorological data. As a result, synchronized time series were obtained.

A positive label was assigned to a three-hour interval if the monitoring system recorded at least one icing event during that interval. The event timestamp was obtained from the monitoring system with a temporal resolution of 10 s. When a corresponding visual inspection report was available, it provided additional confirmation of the label. A visual inspection report does not override instrumental detection but only confirms it: instrumental detection is the primary source, while visual inspection provides supplementary confirmation.

An icing episode was defined as the longest continuous sequence of three-hour intervals with positive labels. Any interval with a negative label terminated the current episode. Intervals for which telemetry data were unavailable or the sensors were malfunctioning were excluded from the analysis: they were assigned neither positive nor negative labels and did not terminate an icing episode.

A negative label was assigned to three-hour intervals in which the monitoring system did not detect icing, provided that complete telemetry data were available throughout the interval. Visual inspection reports were not used to assign negative labels because their episodic nature does not allow the absence of icing to be confirmed for every three-hour interval. The absence of visual inspection reports alone was also not considered sufficient grounds for assigning a negative label. If neither the monitoring system nor visual inspection detected icing during a particular day, this indicated only the absence of

confirmed positive events; the corresponding intervals were assigned negative labels only when complete telemetry data were available.

The target variable and input features were obtained from different data sources: instrumental measurements (icing sensors installed on the wind turbine blades) and meteorological observations (ground-based stations, rp5.ru). Meteorological parameters were not used to construct the target labels and served exclusively as input features for the models.

The dataset has a serious class imbalance: of the 7248 observations, only 577, or 8%, correspond to icing, whereas the remaining 6671, or 92%, correspond to no icing. The imbalance ratio of approximately 1:12 creates a substantial problem for standard ML algorithms, which tend to prefer the majority class and fail to detect rare but critical icing events.

3.2. Software Implementation

For the experiments, a software package was developed that provides construction and comparative analysis of ML models, including random forest, support vector machine, gradient boosting, and the *k*-nearest-neighbour method, with automated hyperparameter selection using three optimization strategies: exhaustive grid search, random search, and Bayesian optimization. A key feature of the software implementation is the built-in set of methods for processing imbalanced data, including SMOTE, ADASYN, Tomek links, and the hybrid SMOTE–Tomek and TaA approaches. In the present study, the algorithms employed include Random Forest, Support Vector Machines, Gradient Boosting, and an LSTM network; while the k-Nearest Neighbors method and exhaustive grid search were implemented within the system, they were not used in these experiments.

The software was developed in Python 3.15 using the PyCharm Professional 2026.2 environment. The package has a modular architecture and uses tkinter for the graphical interface; Matplotlib 3.11, Seaborn 0.13, and Plotly 7.1 for visualization; Scikit-learn, XGBoost, Imbalanced-learn, and Bayesian-optimization for implementing ML algorithms; and Pandas, NumPy, and SciPy for data processing.

Functionally, the application covers the complete cycle of constructing predictive models. The initial data are loaded in CSV format through the graphical interface with the target variable specified. The built-in function for analysing factor significance based on the Gini criterion makes it possible to visualize the influence of each parameter and to select the most informative features in a justified manner. On the model-construction tab, the user selects the algorithm, the hyperparameter-optimization method, and the data-balancing method; after training is completed, the system generates a detailed report on model quality for the training and test samples, with the option of saving it to a text file. The application also provides the possibility of making predictions for new data, evaluating accuracy when reference values are available, and exporting the results in XLSX format.

A separate “Model Analysis” module provides automated comparison of the constructed models: initial filtering is performed using the AUC–ROC metric, models with values close to the random level are rejected, and the remaining models are then ranked by F1-score. The optimal model is presented in a separate report with detailed visualization of the quality indicators, and all results can be exported for subsequent analysis.

The developed software package is a tool for solving predictive-analytics problems in technical diagnostics and provides comprehensive implementation of current ML methods with automated hyperparameter optimization and processing of data imbalance. The adaptability of the application to different types of technical objects and the possibility of integration into existing monitoring systems make it a universal solution for a wide range of applied problems.

The developed software suite enables the comprehensive implementation of modern machine learning methods, featuring automated hyperparameter optimization and data imbalance handling.

3.3. Numerical Experiments

This section presents the results of computational experiments evaluating sampling methods for forecasting wind turbine blade icing. To assess the generalization ability of the models, a temporal data split was used. Data from 2020–2023 were used for training (4832 observations), data from 2024 (1208 observations) were used for validation and hyperparameter tuning, and data from 2025 (1208 observations) were used for the final performance evaluation. This scheme makes it possible to assess how well the models generalize to unseen future data and prevents information leakage from future observations into the past.

The 2025 test dataset is not involved in any stage of model development, including feature selection, hyperparameter optimization, sampling, threshold selection, and calibration.

Within the training dataset, three-fold chronological cross-validation with an expanding window is applied: the folds are formed sequentially, with earlier periods used for training and later periods for validation. All preprocessing steps, including feature selection, sampling, standardization, and post-processing of synthetic samples, are performed on the training portion of each fold; the resulting parameters are applied to the validation portion. The validation and test datasets remain imbalanced.

After cross-validation, the final model is trained on the entire training dataset (2020–2023). Feature selection based on the Gini criterion is performed on the training portion of each fold, and the selected features are applied to the validation portion. Class balancing is applied only to the training portion of each fold. Standardization is performed on the training portion of each fold: the Z-score normalization parameters are applied to the validation portion of the fold and, for final evaluation, to the validation and test datasets. The classification threshold is optimized on the validation portion of each fold; for final evaluation, the threshold is fixed at the value obtained from the 2024 validation dataset. Probability calibration is also performed using the 2024 validation dataset and fixed before testing. The 2025 test dataset is used exclusively for the final evaluation of model performance.

When constructing sequences for the LSTM network, windows are formed within each dataset, and sequences crossing dataset boundaries are excluded. During cross-validation, sequences are constructed separately for the training and validation portions of each fold. When constructing the target variable for a prediction horizon h, the last h observations of each dataset are excluded. The number of training sequences in each fold is reduced accordingly, which is taken into account when selecting the network architecture and interpreting the performance metrics.

At the first stage, the importance of meteorological features was evaluated in order to reduce dimensionality, decrease the risk of overfitting, and improve model interpretability. Feature selection was performed using the 2020–2023 training set to prevent information leakage from the validation and test sets. Feature importance was evaluated using two independent methods: Gini importance, embedded in the Random Forest algorithm, and permutation importance.

To determine the optimal number of selected features, a sensitivity analysis was performed. A Random Forest model was trained using different numbers of the most important features selected according to Gini importance: $k = 5, 8, 10, 12, 15, 18, 20, 22, 25, 28, 30, 32, 34, 35, 36, 37, 38, 39, 40, 42, 45, 48, 50, 55, 60, 65, 70, 75, 80$, and 90. To compensate for class im-

balance, the model was trained using the built-in class-balancing option `class_weight = balanced`, which automatically increases the contribution of the minority class during tree construction. For each value of k, F1-score and PR-AUC were evaluated on the validation set. To reduce the influence of random initialization, the results for each model were averaged over three independent random seeds. The validation dataset was used to determine the optimal number of selected features k and to tune the hyperparameters, but was not used for model training. The test set was not used at this stage. The results are presented in Figure 2.

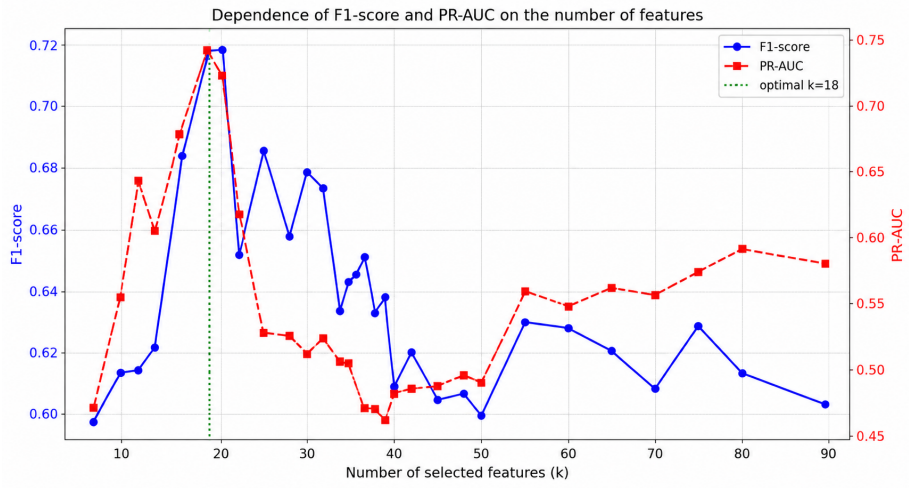


Figure 2. Dependence of F1-score and PR-AUC on the number of selected features k.

The analysis shows that model performance increases rapidly as k increases from 5 to 15, reaches a local maximum in the range $k = 18$–20, and subsequently decreases as additional features are included. This behavior is associated with the inclusion of weakly informative and correlated predictors. Four independent criteria were used to select the final value of k: maximum PR-AUC, maximum F1-score, maximum mean value of the normalized metrics, and the maximin rule, defined as the maximum of the minimum normalized metric values. Three criteria—maximum PR-AUC, maximum mean normalized metric, and the maximin rule—resulted in $k = 18$, whereas the maximum F1-score criterion resulted in $k = 20$. The final value was therefore selected as $k = 18$, representing a compromise between F1-score and PR-AUC. The difference between $k = 18$ and $k = 20$ is negligible in terms of PR-AUC, whereas $k = 18$ provides a lower-dimensional model with comparable predictive performance.

Figure 3 presents the Gini importance diagram for the selected features.

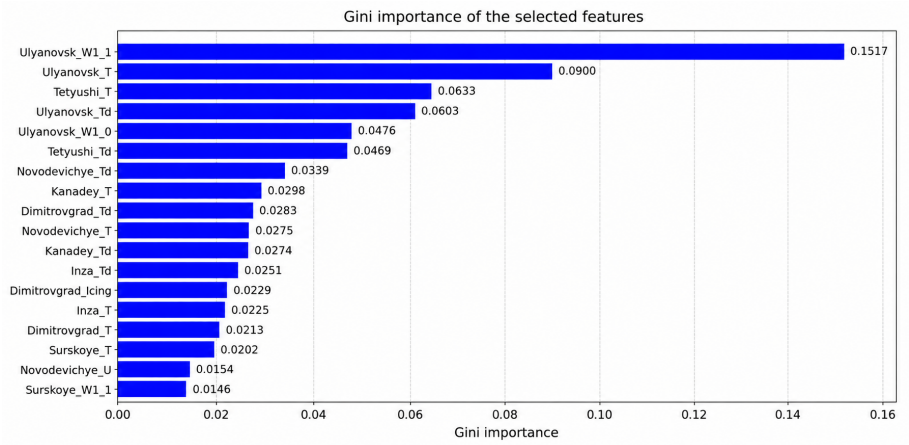


Figure 3. Feature-importance diagram based on the Gini criterion.

The Gini-importance analysis showed that icing of wind turbine blades at the Ulyanovsk wind farm is primarily affected by local meteorological conditions recorded at the reference station, as well as by air temperature and dew-point temperature at neighboring weather stations. The most important predictor is the binary weather-phenomenon feature corresponding to liquid precipitation at the reference station, with an importance value of 0.152. This result is physically justified because the presence of supercooled precipitation is one of the key conditions for ice formation on turbine blades. The second most important feature is air temperature at the reference station, with an importance of 0.090, reflecting the transition through 0 °C as a major trigger of icing. The third most important predictor is air temperature at the Tetyushi station, with an importance of 0.063.

The next most important features include dew-point temperature at the reference and neighboring weather stations, indicators of the absence of precipitation, and air temperature at the Tetyushi, Kanadey, and Dimitrovgrad stations. Their contribution is noticeably lower than that of the leading predictors but remains relevant and consistent with the physical mechanisms of icing. Icing indicators at neighboring stations, air temperature in Inza and Dimitrovgrad, as well as humidity- and wind-related characteristics, exhibit substantially lower importance. Wind characteristics, including the cyclic components of wind direction and wind speed, are almost absent from the final feature set, indicating their secondary role in the formation of icing conditions.

Gini importance was used as the primary feature-selection criterion because of its computational efficiency and its direct integration into the Random Forest model, which makes it possible to train the model and estimate feature importance simultaneously. Since Gini importance can be biased toward continuous and multi-category variables and its reliability may be limited in the presence of class imbalance [41], permutation importance was additionally used as an independent robustness check. Permutation importance is less susceptible to this type of bias. Agreement between the two independent methods provides additional evidence for the robustness of the selected leading predictors. The comparison was performed for the most important features selected according to Gini importance. Permutation importance was calculated using a separate 20% subset of the training data that was not used to train the model. Thus, no information from the validation or test sets was involved.

Figure 4 compares the normalized Gini and permutation importance values for the 18 features selected according to the Gini criterion. The features are shown along the horizontal axis in descending order of Gini importance, while the vertical axis represents normalized importance. The blue bars correspond to Gini importance and the red bars to permutation importance. Each importance measure was normalized by its own maximum value, allowing their relative feature contributions to be compared.

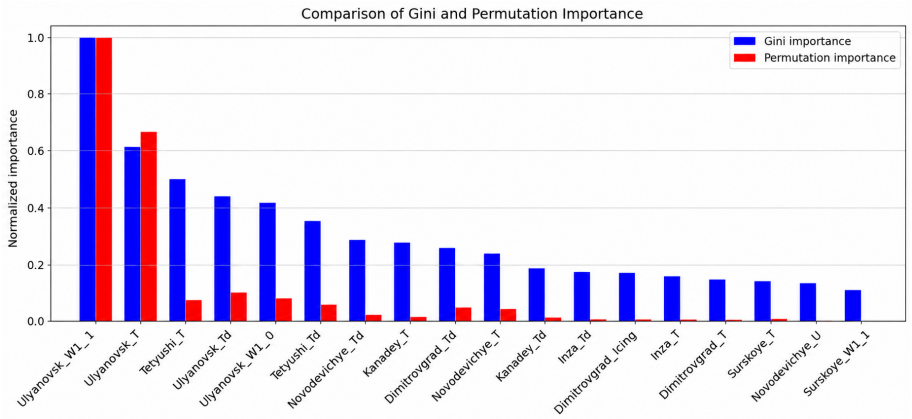


Figure 4. Comparison of Gini importance and permutation importance for the selected features.

Comparison of Gini and permutation importance showed that 8 of the 18 features selected as the most important according to the Gini criterion were also among the most important features according to permutation importance. However, only two of them—the liquid-precipitation indicator at the reference station (`Ulyanovsk_W1_1`) and air temperature at the reference station (`Ulyanovsk_T`)—have importance values of comparable magnitude and occupy leading positions according to both methods.

For the other six overlapping features, permutation importance is approximately one order of magnitude lower than Gini importance, reflecting their strong correlation with the leading predictors. For the features outside the intersection of the two rankings, permutation importance is also close to zero, whereas Gini importance is distributed relatively uniformly among them. Thus, selection of the 18 most important features according to the Gini criterion is robust for the leading predictors but requires further verification for features of moderate importance. The final set of 18 features was used in all subsequent modeling stages.

To select the hyperparameter-optimization method, two approaches were compared: Random Search and Bayesian Optimization. The comparison was performed for the current icing detection problem ($h = 0$). Both strategies followed the same experimental protocol: all four models were trained using class weights, PR-AUC was used as the optimization metric, the classification threshold was selected by maximizing the F1-score, model performance was evaluated on the validation set, and the test set was not used at this stage.

Random Search evaluated 50 randomly selected hyperparameter combinations, whereas Bayesian Optimization used a Gaussian process with the Expected Improvement acquisition function and 30 iterations. Three-fold chronological cross-validation was applied to the training dataset: the folds were constructed in chronological order, ensuring that later observations were not included in a training fold when earlier observations were used for validation. For the LSTM model, 15 Random Search iterations and 12 Bayesian Optimization iterations were used. The lower number of iterations was due to the substantially higher computational cost of recurrent neural-network training compared with classical machine-learning algorithms. All preprocessing procedures, including standardization, were performed exclusively on the training data and within the corresponding cross-validation folds.

Table 1 presents the best PR-AUC and F1-score values obtained on the validation set using each optimization strategy, together with the corresponding computational time.

Table 1. Comparison of Random Search and Bayesian Optimization for the four models.

Optimization Method	Model	PR-AUC	F1-Score	Time (s)
Random Search	Random Forest	0.7949	0.5868	121.0
	SVM	0.5232	0.5000	277.4
	Gradient Boosting	0.6399	0.4220	43.2
	LSTM	0.8248	0.7700	670.3
Bayesian Optimization	Random Forest	0.8029	0.6456	98.8
	SVM	0.5316	0.4469	136.6
	Gradient Boosting	0.7254	0.5371	61.9
	LSTM	0.8088	0.7070	408.3

Bayesian Optimization outperformed Random Search in terms of PR-AUC for three of the four models. The largest improvement was obtained for Gradient Boosting, reaching 13.4%. This can be explained by the relatively complex hyperparameter landscape of this model, which is poorly covered by a random sample of only 50 combinations. For

Random Forest and SVM, the improvement was considerably smaller, approximately 1.0% and 1.6%, respectively, because the corresponding search spaces are relatively limited and both optimization strategies identify similar near-optimal configurations. For LSTM, in contrast, Random Search produced both a higher PR-AUC and a noticeably higher F1-score.

In terms of computational cost, Bayesian Optimization was faster for three of the four models. The computation time was reduced by 50.7% for SVM, 39.1% for LSTM, and 18.4% for Random Forest. Gradient Boosting was the only model for which Bayesian Optimization required more computational time, with an increase of 43.3%. This corresponds to an additional 18 s and is compensated by the substantially higher PR-AUC obtained using this method.

These results demonstrate that the optimal optimization strategy depends on the particular model. In subsequent experiments, the hyperparameters obtained using Bayesian Optimization were used for Random Forest, SVM, and Gradient Boosting, whereas the LSTM model used the parameters obtained using Random Search. This approach allows the configuration providing the best validation-set performance according to the primary metric to be used for each model.

3.4. Baseline Performance Without Class Balancing

To quantitatively assess the class-imbalance problem, an experiment was first performed for the current icing detection problem ($h = 0$) using the original imbalanced data. All four models were trained without any imbalance-correction measures, i.e., without class weights or sampling. The classification threshold was selected on the validation set by maximizing the F1-score. Hyperparameters were optimized using Bayesian Optimization for Random Forest, SVM, and Gradient Boosting and Random Search for LSTM, with PR-AUC used as the optimization objective and without class weighting. Performance was evaluated on the test set, which was not involved in either model training or hyperparameter tuning.

Table 2 presents the performance metrics of the baseline models on the test set.

Table 2. Performance of baseline models without class balancing on the test set.

Metric	Random Forest	SVM	Gradient Boosting	LSTM
Accuracy	0.9288	0.9214	0.9354	0.9412
Precision	0.6667	0.5055	0.7143	0.6289
Recall	0.2083	0.4792	0.3125	0.6354
F1-score	0.3175	0.4920	0.4348	0.6321
Specificity	0.9910	0.9595	0.9892	0.9676
Balanced Accuracy	0.5997	0.7194	0.6509	0.8015
MCC	0.3465	0.4496	0.4455	0.6001
AUC-ROC	0.9641	0.9350	0.9566	0.9500
PR-AUC	0.6663	0.5444	0.6244	0.6875

All baseline models demonstrate high Accuracy values, ranging from 0.9214 to approximately 0.94. However, under strong class imbalance, where approximately 92% of observations belong to the “no icing” class, Accuracy does not adequately reflect the actual ability of the models to detect rare icing events. Recall, F1-score, MCC, Balanced Accuracy, and PR-AUC are considerably more informative in this case.

The classical models—Random Forest, SVM, and Gradient Boosting—detect only approximately 21–48% of actual icing events and therefore miss up to 80% of critical cases. Their F1-score does not exceed 0.49, while Balanced Accuracy remains below 0.72. At the same time, Specificity reaches 96–99%, which mainly reflects the strong tendency of the

models to predict the majority class. MCC values for the classical models range from 0.35 to 0.45, indicating only moderate agreement between predictions and true labels.

The LSTM model demonstrates noticeably better results. Its Recall is approximately 1.3–3.0 times higher than that of the classical models, while its F1-score and MCC are approximately 1.3–2.0 times higher. This can be explained by two factors. First, LSTM receives a sequence of four observations as input, making it possible to identify temporal patterns preceding icing, whereas the classical models use only the current observation. Second, the recurrent architecture with its gating mechanism can more effectively identify rare signals in a noisy time series. Nevertheless, even for LSTM, a Recall of approximately 0.63 means that more than one third of actual icing events remain undetected, which substantially limits practical applicability.

The difference between AUC-ROC (0.94–0.96) and PR-AUC (0.54–0.69) for all models is characteristic of strongly imbalanced classification problems. The models retain the ability to rank observations according to icing probability, but this ranking capability does not translate into sufficiently high minority-class detection performance at the selected classification threshold. This result additionally supports the use of PR-AUC as the primary metric for hyperparameter optimization.

Figure 5 presents a radar chart comparing the models according to seven performance metrics. The chart clearly illustrates the "flattened" profile of the classical models: their polygons extend toward high Precision and Specificity values but contract sharply along the Recall, F1-score, and MCC axes. The LSTM profile occupies a substantially larger area, although it is still far from ideal.

The results show that naive training without class-imbalance correction is insufficient for developing a practical wind-turbine icing prediction system. The classical models miss more than half of the critical icing events, whereas the LSTM model misses approximately one third. These results provide a quantitative baseline for evaluating the effectiveness of imbalance-correction methods in the subsequent experiments.

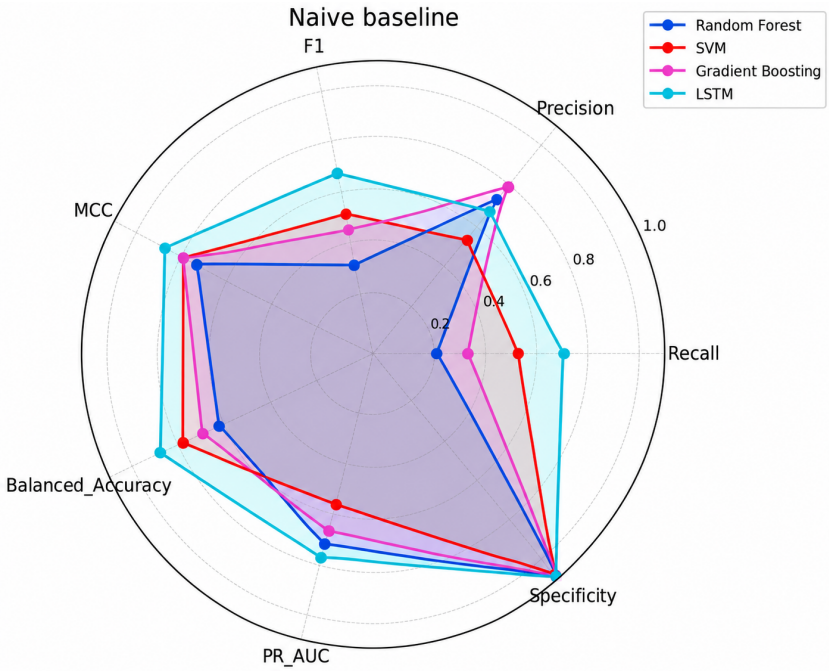


Figure 5. Radar-chart comparison of baseline models without class balancing.

3.5. Class Weighting

The next stage evaluated class weighting as a baseline imbalance-correction method that does not modify the original training sample but instead adjusts the loss function.

The experimental protocol was identical to that used for the previous baseline: current icing detection ($h = 0$), the same feature space, temporal data splitting, model-specific hyperparameter optimization using Bayesian Optimization or Random Search on the validation set, final evaluation on the test set, and classification-threshold selection based on the maximum F1-score.

The only difference was the introduction of class weights during training. For Random Forest and SVM, the built-in parameter `class_weight = balanced` was used. For Gradient Boosting, sample weights were calculated using `compute_sample_weight`, while for LSTM the corresponding weights were incorporated into the loss function. This experimental design ensures a consistent comparison with the naive baseline.

Table 3 presents the performance metrics of the four models trained using class weights.

Table 3. Performance of models trained with class weights on the test set.

Metric	Random Forest	SVM	Gradient Boosting	LSTM
Accuracy	0.9478	0.9230	0.9396	0.9487
Precision	0.7463	0.5143	0.6949	0.6667
Recall	0.5208	0.5625	0.4271	0.7083
F1-score	0.6135	0.5373	0.5290	0.6869
Specificity	0.9847	0.9541	0.9838	0.9694
Balanced Accuracy	0.7528	0.7583	0.7054	0.8389
MCC	0.5974	0.4960	0.5156	0.6593
AUC-ROC	0.9688	0.9345	0.9617	0.9645
PR-AUC	0.7120	0.5492	0.6412	0.7475

Analysis of Table 3 shows that class weighting substantially improves metrics sensitive to the minority class, although its effectiveness differs considerably among the models. The strongest effect is observed for Random Forest: Recall increases by approximately 2.5 times relative to the naive baseline and F1-score nearly doubles, while Precision remains high at approximately 0.75.

For SVM and Gradient Boosting, the improvement is less pronounced. Their F1-score increases by approximately 1.1–1.2 times and Recall by approximately 1.2–1.4 times. For the LSTM model, class weighting provides a moderate improvement relative to the naive baseline: F1-score increases by approximately 8% and Recall by approximately 11%, while Precision changes only slightly.

Among the class-weighted models, LSTM provides the highest F1-score, Recall, Balanced Accuracy, MCC, and PR-AUC. Random Forest provides the second-highest F1-score and PR-AUC while achieving higher Precision than LSTM. The gap between AUC-ROC and PR-AUC remains for all models, indicating that more advanced class-imbalance correction methods should also be investigated.

Comparison with the naive baseline demonstrates that class weighting improves rare-event detection but does not completely resolve the imbalance problem. Even the LSTM model still misses approximately 30% of icing cases, while Random Forest misses almost half of them. Class weighting can therefore be considered an effective and computationally inexpensive first-level imbalance-correction method, particularly for Random Forest, but the resulting performance remains insufficient for practical application, motivating the transition to sampling methods.

3.6. Comparison of Basic Sampling Methods

The next stage evaluated three basic training-set balancing methods—SMOTE, ADASYN, and Tomek links—and compared them with class weighting. Unlike class

weighting, these methods directly modify the training sample: SMOTE and ADASYN generate synthetic minority-class observations, whereas Tomek links remove majority-class observations located near the decision boundary.

The experimental protocol remained unchanged: current icing detection, 18 selected features, and temporal data splitting. For each model–method combination, hyperparameters were optimized independently according to PR-AUC on the validation set. The classification threshold maximizing F1-score was then selected using the same validation set. Balancing was applied exclusively to the training data and only within the corresponding cross-validation folds.

Figure 6 presents four radar charts, one for each model, comparing the profiles of the four imbalance-correction methods according to seven metrics. The results indicate that there is no universally superior method among the basic approaches: the most effective method depends both on the model and on the evaluation metric. According to F1-score, ADASYN provides the highest values for Random Forest and Gradient Boosting, whereas SMOTE provides the highest values for SVM and LSTM. The pattern differs for PR-AUC: Tomek links provides the highest values for Random Forest, SVM, and Gradient Boosting, whereas SMOTE provides the highest value for LSTM.

Figure 7 presents bar charts comparing F1-score and PR-AUC for the different model–method combinations.

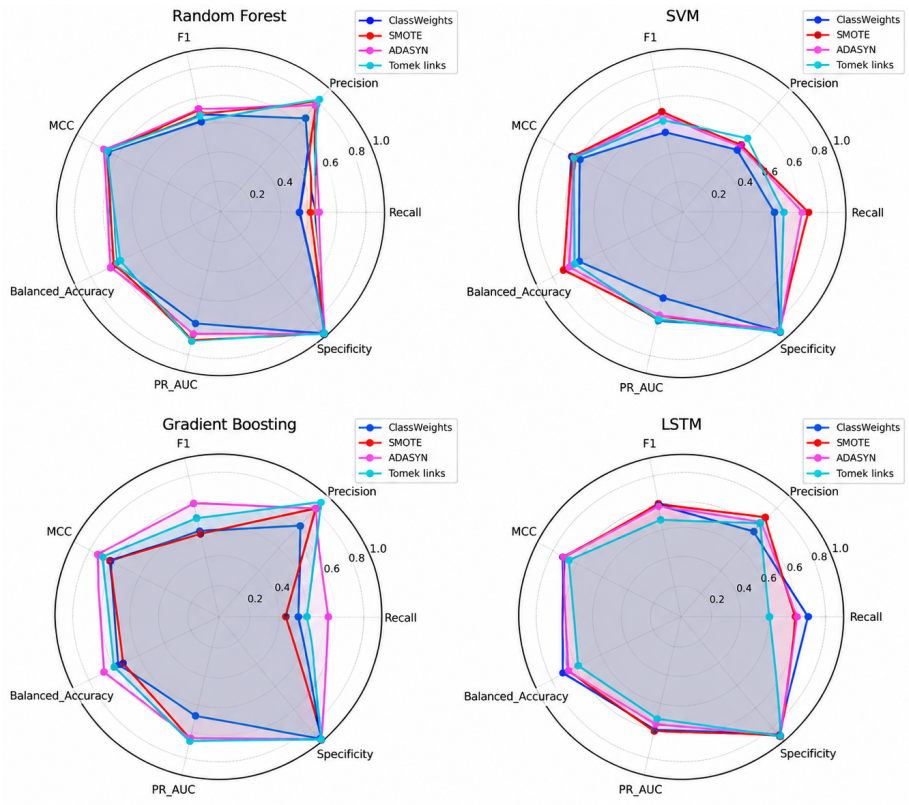


Figure 6. Radar-chart profiles of the four imbalance-correction methods for each model.

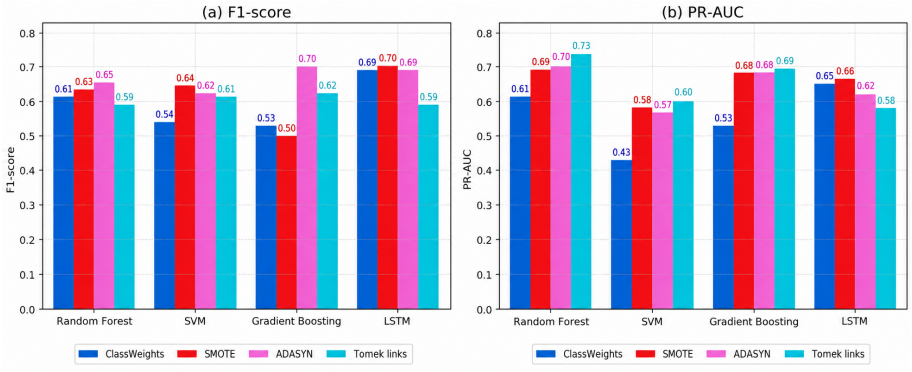


Figure 7. Comparison of sampling methods: (**a**) F1-score; (**b**) PR-AUC.

The F1-score and PR-AUC bar charts show that the optimal balancing method depends not only on the model but also on the metric considered. Relative to class weighting, the best sampling methods improve F1-score by approximately 2% for LSTM and up to 32% for Gradient Boosting. The corresponding PR-AUC improvement ranges from approximately 13% for LSTM to 29% for SVM.

The strongest effect is observed for Gradient Boosting. According to F1-score, ADASYN outperforms Tomek links by a factor of 1.14 and SMOTE by a factor of 1.38. In terms of PR-AUC, however, Tomek links provides the highest result and exceeds ADASYN by approximately 2%. For SVM, SMOTE provides the highest F1-score, with an improvement of approximately 19% relative to class weighting, whereas Tomek links provides the highest PR-AUC, with an improvement of approximately 29%.

For Random Forest, ADASYN provides the highest F1-score, approximately 6% higher than that obtained using class weighting, whereas Tomek links provides the highest PR-AUC and exceeds ADASYN by approximately 4%. For LSTM, both metrics consistently indicate SMOTE as the most effective of the basic sampling methods, although the improvement relative to class weighting is relatively small, approximately 7.6%.

The discrepancy between F1-score and PR-AUC is explained by the fact that F1-score is calculated at a single operating threshold selected on the validation set, whereas PR-AUC is threshold-independent and evaluates ranking quality over the entire range of possible thresholds. Therefore, the selection of a balancing method should take into account the priority of the evaluation metric: if performance at a specific operating threshold is of primary importance, F1-score should be emphasized; if robustness across the full threshold range is more important, PR-AUC is more informative.

Thus, the basic balancing methods provide a noticeable improvement over the naive baseline but remain insufficient for the problem considered. The best model–method combination provides an F1-score no higher than approximately 0.70, leaving around 30–40% of icing events undetected. This level of missed critical events limits practical applicability and motivates the use of hybrid sampling methods capable of modifying not only the class ratio but also the structure of the training sample.

3.7. Hybrid Sampling Methods

Hybrid balancing methods were subsequently evaluated: SMOTE–Tomek and the proposed TaA method. Balancing was applied exclusively to the training set, while performance was evaluated on the test set. Table 4 presents the model-performance metrics obtained using the two hybrid methods.

Table 4. Performance of models using hybrid class-balancing methods on the test set.

Metric	Random Forest	SVM	Gradient Boosting	LSTM
Balanced Accuracy	0.899/0.888	0.866/0.906	0.894/0.874	0.907/0.911
Precision	0.951/0.938	0.783/0.698	0.837/0.839	0.704/0.764
Recall	0.802/0.781	0.750/0.844	0.802/0.760	0.844/0.844
F1-score	0.870/0.852	0.766/0.764	0.819/0.798	0.768/0.802
PR-AUC	0.946/0.939	0.748/0.699	0.877/0.862	0.841/0.856

Values are reported as TaA/SMOTE–Tomek.

Comparison of TaA and SMOTE–Tomek shows that the difference between the methods depends on the particular model. For Random Forest, TaA produces higher values than SMOTE–Tomek for all five metrics, with the improvement reaching approximately 1–3% depending on the metric. For Gradient Boosting, TaA also provides higher values for most metrics. The improvement in Recall reaches approximately 6%, while Precision is marginally lower by less than 1%.

For SVM, the results are mixed. TaA provides improvements of up to approximately 12% in Precision and up to approximately 7% in PR-AUC, whereas SMOTE–Tomek provides higher Recall by up to approximately 12% and higher Balanced Accuracy by up to approximately 5%. For LSTM, in contrast, SMOTE–Tomek provides higher values for most metrics, with an advantage of up to approximately 9%, while Recall is identical for the two methods.

Thus, TaA provides stronger results for Random Forest and Gradient Boosting according to most of the considered metrics, with differences ranging from approximately 1% to 6%. For LSTM, SMOTE–Tomek provides higher values for most metrics, with differences reaching approximately 9%. For SVM, the choice depends on the metric considered: TaA provides higher Precision and PR-AUC, whereas SMOTE–Tomek provides higher Recall and Balanced Accuracy. These results demonstrate that the effectiveness of hybrid sampling methods depends on the model architecture and on the performance metric of interest.

To evaluate the statistical significance of the differences between SMOTE–Tomek and TaA, 30 paired runs were performed using the same temporal data split and identical random seeds for all stages unrelated to sampling. For each paired run, differences in F1-score and PR-AUC were calculated, and the null hypothesis of zero median difference was tested using the Wilcoxon signed-rank test at a significance level of 0.05.

For F1-score, TaA significantly exceeds SMOTE–Tomek for Random Forest, with mean values of 0.870 ± 0.018 and 0.852 ± 0.021, respectively ($p = 0.004$), and for Gradient Boosting, with 0.819 ± 0.019 and 0.798 ± 0.020, respectively ($p < 0.001$). For SVM, the difference is not statistically significant: 0.766 ± 0.017 versus 0.764 ± 0.018, with $p = 0.66$. For LSTM, the opposite pattern is observed, with SMOTE–Tomek significantly exceeding TaA ($p < 0.001$).

According to PR-AUC, TaA exceeds SMOTE–Tomek for Random Forest and SVM ($p < 0.05$), whereas the differences for Gradient Boosting and LSTM are not statistically significant ($p > 0.05$).

Therefore, the proposed TaA method demonstrates competitive performance: it provides stronger results than SMOTE–Tomek for Random Forest and Gradient Boosting according to the main reported comparisons, lower performance for LSTM, and comparable results for SVM depending on the metric considered.

3.8. Effect of TaA on Model Performance

To assess the generalization ability of the trained models and their suitability for practical application, predictions were generated for the test set, which was not involved

in either model training or hyperparameter optimization. Figure 8 presents confusion matrices for Random Forest and LSTM under two conditions: without balancing and with TaA balancing. Figures 8a and 8b correspond to Random Forest without balancing and with TaA, respectively, whereas Figure 8c,d correspond to LSTM without balancing and with TaA. This representation makes it possible to isolate the effect of balancing within each model.

Analysis of the confusion matrices shows that TaA balancing improves predictive performance for both models, although the magnitude of the effect differs substantially. For Random Forest without balancing, Recall is extremely low: of 96 actual icing events, the model correctly identifies only 20 and misses 76 critical cases. After applying TaA, the number of missed icing events decreases to 19, while the number of correctly detected events increases to 77. At the same time, the number of false positives decreases from 10 to 4. This corresponds to an approximately 3.9-fold increase in Recall and a 2.7-fold increase in F1-score.

For LSTM without balancing, 61 of the 96 actual icing events are detected. After applying TaA, this number increases to 81 of 96, while the number of false positives decreases from 36 to 34. Recall therefore increases by approximately 33%, Precision by approximately 12%, and F1-score by approximately 21%.

Thus, the strongest effect of TaA is observed for models that are initially strongly biased toward the majority class. For LSTM, which already exhibits a more balanced baseline performance profile, the additional improvement produced by hybrid sampling is more moderate.

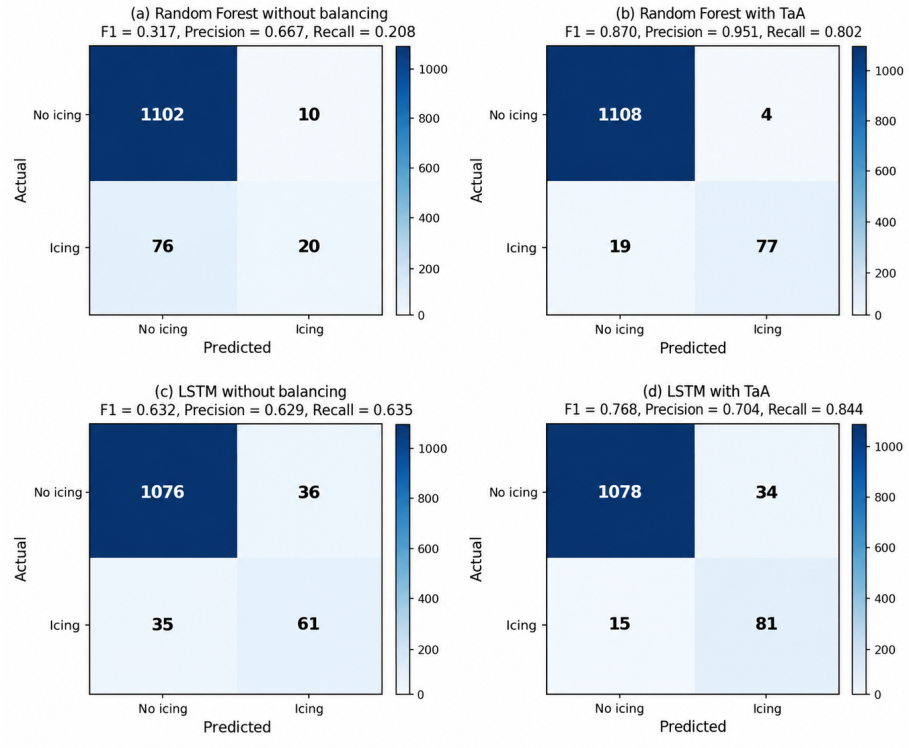


Figure 8. Confusion matrices on the test set: (**a**) Random Forest without balancing; (**b**) Random Forest with TaA; (**c**) LSTM without balancing; (**d**) LSTM with TaA.

Figure 9 presents bar-chart comparisons of Precision, Recall, F1-score, Balanced Accuracy, and PR-AUC for Random Forest and LSTM in two configurations: without balancing and with TaA.

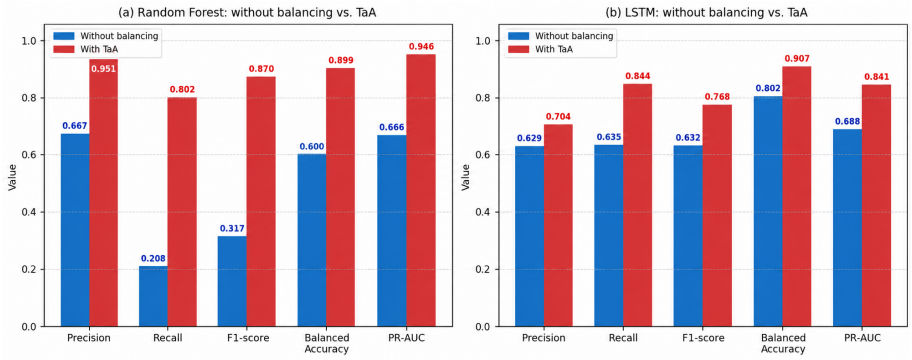


Figure 9. Comparison of performance metrics on the test set: (**a**) Random Forest without balancing and with TaA; (**b**) LSTM without balancing and with TaA.

For Random Forest, application of TaA improves all five metrics. Precision increases by approximately 42%, Recall increases by a factor of approximately 3.9, F1-score by a factor of approximately 2.7, Balanced Accuracy by approximately 50%, and PR-AUC by approximately 42%. The strongest improvement is observed for Recall and F1-score, reflecting the correction of the main limitation of the baseline model—its low sensitivity to rare icing events. Thus, for Random Forest, the hybrid TaA method produces a substantial improvement in predictive performance.

The effect of TaA on the LSTM model is less pronounced than that observed for Random Forest: F1-score increases by approximately 1.21 times, compared with approximately 2.7 times for Random Forest. This is explained by the more balanced baseline performance of LSTM resulting from its use of temporal context, whereas the baseline Random Forest is strongly biased toward the majority class. Nevertheless, LSTM with TaA exceeds the corresponding LSTM results without balancing by approximately 21–33% and those obtained with class weighting by approximately 13–20%.

The comparison demonstrates that Random Forest with TaA provides higher values for the reported set of metrics than the corresponding LSTM configuration in these experiments. At the same time, the positive effect of balancing is observed for both architectures, although its magnitude strongly depends on the initial model behavior. For Random Forest, which is initially highly biased toward the majority class, hybrid sampling produces a several-fold improvement in minority-event detection, whereas for LSTM the additional gain is less pronounced.

3.9. Icing Forecasting at Different Prediction Horizons

The final stage investigated icing forecasting at different prediction horizons. For each horizon h, the target variable was formed by shifting the original time series by h steps, and separate training, validation, and test sets were formed for each horizon. Since one time step corresponds to 3 h, the horizons $h = 1$, $h = 2$, and $h = 3$ correspond to forecasting 3, 6, and 9 h ahead, respectively. The final h observations were excluded from the corresponding dataset because the values of Y_{t+h} were unavailable. The horizon $h = 0$ corresponds to detection of the current icing state and was used in the preceding sections to compare the sampling methods.

For each horizon and each model, training was performed using the balancing method selected on the basis of the preceding experiments. Table 5 presents Precision, Recall, F1-score, and PR-AUC for the Random Forest model with TaA balancing at different forecasting horizons.

Table 5. Icing forecasting performance at different prediction horizons h.

Horizon h	Forecast Time	TP	FP	Precision	Recall	F1-Score	PR-AUC
0	0 h (detection)	77	4	0.951	0.802	0.870	0.946
1	3 h	77	20	0.794	0.811	0.802	0.844
2	6 h	74	26	0.740	0.787	0.763	0.879
3	9 h	73	35	0.676	0.768	0.719	0.764

Analysis of Table 5 shows that model performance generally decreases as the prediction horizon increases. The highest metric values are obtained for current icing detection ($h = 0$): the model correctly identifies 77 icing events with only 4 false positives. This is expected because, for $h = 0$, the model does not need to account for the future evolution of meteorological conditions.

When moving to a 3 h forecast ($h = 1$), the number of correctly identified icing events remains unchanged at 77, whereas the number of false alarms increases from 4 to 20. As a result, Precision decreases by 16.5% and F1-score by 7.8%. Recall increases slightly by approximately 1.1%, indicating somewhat greater sensitivity to icing events. PR-AUC decreases by approximately 10.8%.

Increasing the horizon to 6 h ($h = 2$) leads to a further decline in several metrics: the number of correctly identified icing events decreases from 77 to 74, while the number of false positives increases to 26. Performance metrics decrease by approximately 3–7% relative to the preceding horizon, although PR-AUC increases slightly, indicating somewhat better overall ranking performance for this particular horizon.

The largest reduction in performance is observed at the 9 h horizon ($h = 3$): the number of correctly detected icing events decreases to 73, whereas false positives increase to 35. Compared with the 6 h horizon, the performance metrics decrease by approximately 4–13%, with the largest reduction observed for PR-AUC. The deterioration in Recall and PR-AUC at this horizon is associated with the increasing uncertainty of meteorological conditions at longer forecast horizons and the limited amount of training data.

Overall, when moving from current icing detection to forecasting 9 h ahead, Precision decreases by approximately 29%, F1-score by approximately 17%, and PR-AUC by approximately 19%, whereas Recall remains relatively stable, decreasing by only about 4%. This means that the model retains the ability to identify most icing events, although the number of false alarms increases with the forecasting horizon. Consequently, longer-range predictions require more cautious interpretation in operational decision-making. The F1-score remains above 0.70 for all considered horizons, providing wind-farm operators with advance information that may be used when preparing preventive measures.

Modern deep-learning approaches, including PCTG [18], SMOTE-CNN [29], and ConvLSTM-GRU [20], have demonstrated high icing-prediction performance using SCADA data. However, direct comparison with the results of the present study is difficult because of differences in datasets, prediction horizons, and evaluation protocols. Deep-learning models may provide substantial predictive capability when sufficiently large training datasets are available; however, the dataset available in the present study is considerably more limited.

Under these limited-data conditions, class balancing provides a noticeable but less pronounced improvement for the LSTM model, with an approximately 12–17% increase in F1-score relative to class weighting. For the classical algorithms, however, imbalance correction has a substantially greater impact. Random Forest combined with TaA increases F1-score by a factor of approximately 2.7 and PR-AUC by approximately 42% relative to the corresponding baseline model. These results support the relevance of systematically

investigating sampling methods for wind-turbine icing forecasting under limited-data conditions.

4. Conclusions

This study presents a comparative analysis of six class-balancing strategies (class weighting, SMOTE, ADASYN, Tomek links, SMOTE-Tomek, and the authors' TaA method) for predicting wind turbine blade icing, using the Ulyanovsk wind farm as a case study. The research followed a unified experimental protocol involving temporal data splitting, feature selection based on the Gini criterion, hyperparameter optimization, and statistical significance testing (Wilcoxon test). The comparison encompassed four models: Random Forest, Support Vector Machine (SVM), Gradient Boosting, and an LSTM network.

Sensitivity analysis demonstrated model robustness regarding the number of features within the range of k = 15–20, with k = 18 selected as optimal. A comparison of hyperparameter optimization strategies revealed that Bayesian optimization outperformed random search for three of the four models; however, random search proved more effective for the LSTM model. The study demonstrates that the optimal balancing strategy depends on the model architecture; there is no universal method that outperforms all others across every algorithm. This result is of fundamental importance, indicating that the choice of sampling method should be determined not only by the nature of the class imbalance but also by the model architecture. Hybrid sampling methods outperformed both simple methods (SMOTE, ADASYN, Tomek links) and class weighting. The best overall performance across the evaluated metrics was achieved by the Random Forest model using the TaA method: compared to the baseline model without balancing, the F1-score increased 2.7-fold, PR-AUC rose by 42%, and recall increased 3.9-fold. Statistical analysis confirms that, for Random Forest and Gradient Boosting models, the TaA method significantly outperforms SMOTE-Tomek; this is attributed to the effectiveness of its three novel components: an adaptive Tomek link cleaning threshold, weighted synthetic sample generation (ADASYN), and an iterative cleaning and generation procedure.

The situation differs for recurrent architectures. Even in its baseline configuration, the LSTM network exhibits a more balanced profile than classical models, thanks to its use of temporal context. While applying TaA yields a notable performance boost for LSTMs—an increase of 21–33% over the baseline and 12–20% over class weighting—SMOTE-Tomek proves preferable to TaA, indicating that the choice of sampling method depends on the model architecture.

The practical significance of this work lies in the ability to forecast icing up to 9 h in advance while maintaining an F1-score above 0.70 across all forecast horizons. Prediction specificity ranges from 96.9% (at 9 h) to 98.2% (at 3 h), implying a limited number of false alarms and unnecessary preventive actions. A quantitative assessment of the economic impact—accounting for the costs associated with false alarms and missed events—requires a separate study and was not conducted here.

When training data is limited, classical algorithms employing hybrid balancing prove to be just as practical a solution as deep learning models, the latter of which require substantially larger datasets to realize their full potential.

Future research will involve integrating SCADA data from the turbines themselves with hub-height measurements to further improve forecast accuracy, as well as extending the methodology to other wind farms in various cold-climate regions to validate the generalizability of the results as well as shifting to hourly or half-hourly observations to more accurately capture rapid changes in icing conditions.

Author Contributions: Conceptualization, V.N.K.; methodology, Y.E.K. and V.N.K.; software, Y.E.K. and A.I.I.; validation, Y.E.K., A.I.I. and A.V.C.; formal analysis, Y.E.K. and V.N.K.; investigation, Y.E.K., A.I.I. and A.V.C.; resources, A.V.C.; data curation, Y.E.K., A.I.I. and A.V.C.; writing—original draft preparation, R.V.F. and A.I.I.; writing—review and editing, R.V.F., Y.E.K. and V.N.K.; visualization, Y.E.K. and A.I.I.; supervision, V.N.K.; project administration, V.N.K. and R.V.F. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ADASYN	Adaptive Synthetic Sampling
AUC–ROC	Area Under the Receiver Operating Characteristic Curve
BEM	Blade Element Momentum
CFD	Computational Fluid Dynamics
EI	Expected Improvement
FN	False Negative
FP	False Positive
ML	Machine Learning
RBF	Radial Basis Function
ROC	Receiver Operating Characteristic
SCADA	Supervisory Control and Data Acquisition
SMOTE	Synthetic Minority Over-sampling Technique
SVM	Support Vector Machine
TaA	Tomek links with Adaptive Threshold and ADASYN
TN	True Negative
TP	True Positive

References

1. Global Wind Energy Council (GWEC). Global Wind Report 2025. Available online: https://www.gwec.net/reports/globalwindreport/2025/ (accessed on 8 July 2026).
2. Association for the Development of Renewable Energy (ARVE). Renewable Energy Capacity in Russia. Available online: https://rreda.ru/en/ (accessed on 8 July 2026).
3. Vermeer, L.J.; Sørensen, J.N.; Crespo, A. Wind turbine wake aerodynamics. *Prog. Aerosp. Sci.* **2003**, *39*, 467–510.
4. Porté-Agel, F.; Bastankhah, M.; Shamsoddin, S. Wind-Turbine and Wind-Farm Flows: A Review. *Bound.-Layer. Meteorol.* **2020**, *174*, 1–59.
5. Parent, O.; Ilinca, A. Anti-icing and de-icing techniques for wind turbines: Critical review. *Cold Reg. Sci. Technol.* **2011**, *65*, 88–96.
6. Karlsson, T.; Begin-Drolet, A.; Godreau, C.; Rääf, J.; Fischerheit, J.; Kelly, M.; Sarlak, H.; Kim, T. Cold Climate Wind: Challenges, Technological Solutions and Policy. *Wiley Interdiscip. Rev. Energy Environ.* **2026**, *15*, e70025. Correction in *Wiley Interdiscip. Rev. Energy Environ.* **2026**, *15*, e70031.
7. Jiang, L.; Zhang, S.P.; Shen, G.Q.; Zhou, L. Acoustic emission-based wind turbine blade icing monitoring using deep learning technology. *Renew. Energy* **2025**, *247*, 122980.
8. Weijtjens, W.; Fernandes de Oliveira Junior, A.; Cloet, B.; Ceyhan Yilmaz, O.; Devriendt, C. Large scale test of vibration based icing detection for wind turbines. *J. Phys. Conf. Ser.* **2024**, *2647*, 192008.
9. Wei, F.; Guo, Z.; Han, Q.; Qi, W. Estimation of Wind Turbine Blade Icing Volume Based on Binocular Vision. *Appl. Sci.* **2025**, *15*, 114.
10. Chen, T.; Yang, C.; Xiao, Y.; Yan, C.; Lei, M.; Photong, C. Energy-Efficient Wind Turbine Operation via Multi-Class Blade Icing Recognition Using Mutual-Information Feature Selection and Ensemble Methods. *IEEE Access* **2025**, *13*, 156008–156020.

11. Wang, S.; Vidal, Y.; Pozo, F. Recent advances in wind turbine condition monitoring using SCADA data: A state-of-the-art review. *Reliab. Eng. Syst. Saf.* **2026**, *267*, 111838.
12. Contreras Montoya, L.T.; Lain, S.; Ilinca, A. A Review on the Estimation of Power Loss Due to Icing in Wind Turbines. *Energies* **2022**, *15*, 1083.
13. Sheidani, A.; Salavatidezfouli, S.; Stabile, G.; Barzegar Gerdroodbary, M.; Rozza, G. Assessment of icing effects on the wake shed behind a vertical axis wind turbine. *Phys. Fluids* **2023**, *35*, 169102.
14. Dai, J.; Zhang, Y.; Rotea, M.; Kehtarnavaz, N. A Review of Machine Learning Approaches for Prediction of Icing on Wind Turbines. In Proceedings of the 2024 IEEE 19th Conference on Industrial Electronics and Applications (ICIEA), Kristiansand, Norway, 5–8 August 2024; pp. 1–7.
15. Zhang, Y.; Ding, Y.; Zhang, Y.; Ge, F. A Review of Machine Learning-Based Icing Prediction Methods for Wind Turbine Blades. In *Mechanisms and Machine Science, Vol. 141, Proceedings of the TEPEN International Workshop on Fault Diagnostic and Prognostic (TEPEN 2024), Qingdao, China, 8–11 May 2024*; Wang, Z., Zhang, K., Feng, K., Xu, Y., Yang, W., Eds.; Springer: Cham, Switzerland, 2025.
16. Kovalnogov, V.N.; Fedorov, R.V.; Chukalin, A.V.; Klyachkin, V.N.; Tabakov, V.P.; Demidov, D.A. Applied machine learning to study the movement of air masses in the wind farm area. *Energies* **2024**, *17*, 3961.
17. Xie, N.; Cao, Q.; Zeng, Z.; Ma, K.; Zeng, S. Research on the Real-Time Prediction of Wind Turbine Blade Icing Process Based on the MLP Neural Network Model and Meteorological Parameters. *Processes* **2025**, *13*, 1910.
18. Dai, J.; Rotea, M.; Kehtarnavaz, N. An Ensemble Network for High-Accuracy and Long-Term Forecasting of Icing on Wind Turbines. *Sensors* **2024**, *24*, 8167.
19. Zhang, Y.; Kehtarnavaz, N.; Rotea, M.; Dasari, T. Prediction of Icing on Wind Turbines Based on SCADA Data via Temporal Convolutional Network. *Energies* **2024**, *17*, 2175.
20. Chen, W.; Zhou, H.; Cheng, L.; Xia, M. Wind Turbine Blade Icing Diagnosis Using Convolutional LSTM-GRU With Improved African Vultures Optimization. *IEEE Open J. Instrum. Meas.* **2022**, *1*, 3500809.
21. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32.
22. Klyachkin, V.N.; Karpunina, I.N. Features of diagnostics of technical systems using multiclass classification. *J. Commun. Technol. Electron.* **2025**, *70*, 232–237.
23. Zhang, L.; Zhao, Y.; Guo, Y.; Hu, T.; Xu, X. Research on wind turbine icing prediction data processing and accuracy of machine learning algorithm. *Renew. Energy* **2024**, *237*, 121566.
24. Kovalnogov, V.; Fedorov, R.; Klyachkin, V.; Generalov, D.; Kuvayskova, Y.; Busygin, S. Applying the random forest method to improve burner efficiency. *Mathematics* **2022**, *10*, 12.
25. Xu, J.; Tan, W.; Li, T. Predicting fan blade icing by using particle swarm optimization and support vector machine algorithm. *Comput. Electr. Eng.* **2020**, *87*, 106751.
26. Xiong, C.; He, Z.; Zhang, Y.; Huang, S. Research on Icing Status Forecasting of Wind Turbine Blades Based on Machine Learning. In Proceedings of the 2021 6th Asia Conference on Power and Electrical Engineering (ACPEE), Chongqing, China, 8–11 April 2021; pp. 508–513.
27. Li, S.; Peng, Y.; Bin, G. Prediction of Wind Turbine Blades Icing Based on CJBM With Imbalanced Data. *IEEE Sens. J.* **2023**, *23*, 19726–19736.
28. He, H.; Garcia, E.A. Learning from Imbalanced Data. *IEEE Trans. Knowl. Data Eng.* **2009**, *21*, 1263–1284.
29. Liu, F.; Wang, X.; Pu, X. Wind Turbine Blade Icing Detection Method Based on SMOTE-SA-CNN. In Proceedings of the 2025 8th International Conference on Artificial Intelligence and Big Data (ICAIBD), Chengdu, China, 23–26 May 2025; pp. 403–407.
30. Zhang, Y.; Deng, L.; Wei, B. Imbalanced Data Classification Based on Improved Random-SMOTE and Feature Standard Deviation. *Mathematics* **2024**, *12*, 1709.
31. Ge, Y.; Yue, D.; Chen, L. Prediction of Wind Turbine Blades Icing Based on MBK-SMOTE and Random Forest in Imbalanced Data Set. In Proceedings of the 2017 IEEE Conference on Energy Internet and Energy System Integration (EI2), Beijing, China, 26–28 November 2017; pp. 1–6.
32. Diallo, M.; Xiong, S.; Emiru, E.D.; Fesseha, A.; Abdulsalami, A.O.; Elaziz, M.A. A Hybrid MultiLayer Perceptron Under-Sampling with Bagging Dealing with a Real-Life Imbalanced Rice Dataset. *Information* **2021**, *12*, 291.
33. Lü, Y.; Hu, Q.; Hu, Z.; Wu, Y.; Lin, H. Diagnosis and Interpretability Study of Wind Turbine Blade Icing under Consideration of Sample Imbalance Conditions. *Trans. China Electrotech. Soc.* **2025**, *40*, 3667–3679.
34. Zhang, Y.; Zhang, Y.; Zhang, Y.; Li, H.; Yan, L.; Wen, X.; Wang, H. Multi-class data augmentation and fault diagnosis of wind turbine blades based on ISOMAP-CGAN under high-dimensional imbalanced samples. *Renew. Energy* **2025**, *243*, 122609.
35. Safi, S.K.; Gul, S. An Enhanced Tree Ensemble for Classification in the Presence of Extreme Class Imbalance. *Mathematics* **2024**, *12*, 3243.
36. Ghasemkhani, B.; Kut, R.A.; Birant, D.; Yilmaz, R. Balanced Hoeffding Tree Forest (BHTF): A Novel Multi-Label Classification with Oversampling and Undersampling Techniques for Failure Mode Diagnosis in Predictive Maintenance. *Mathematics* **2025**, *13*, 3019.

37. Qi, L.; Liu, M.; Cheng, X.; Shi, F.; Liu, X.; Chen, S. Prototype-based Heterogeneous Federated Learning for Blade Icing Detection in Wind Turbines with Class Imbalanced Data. *arXiv* **2025**, arXiv:2503.08325.
38. Zhang, J.; Li, Y.; Liu, B.; Chen, H.; Zhou, J.; Yu, H.; Qin, B. A Broad TSK Fuzzy Classifier with a Simplified Set of Fuzzy Rules for Class-Imbalanced Learning. *Mathematics* **2023**, *11*, 4284.
39. Cai, C.; Guo, J.; Song, X.; Zhang, Y.; Wu, J.; Tang, S.; Jia, Y.; Xing, Z.; Li, Q. Review of Data-Driven Approaches for Wind Turbine Blade Icing Detection. *Sustainability* **2023**, *15*, 1617.
40. Menze, B.H.; Kelm, B.M.; Masuch, R.; Himmelreich, U.; Bachert, P.; Petrich, W.; Hamprecht, F.A. A comparison of random forest and its Gini importance with standard chemometric methods for the feature selection and classification of spectral data. *BMC Bioinform.* **2009**, *10*, 213.
41. Strobl, C.; Boulesteix, A.-L.; Zeileis, A.; Hothorn, T. Bias in Random Forest Variable Importance Measures: Illustrations, Sources and a Solution. *BMC Bioinform.* **2007**, *8*, 25.
42. Fisher, A.; Rudin, C.; Dominici, F. All Models Are Wrong, but Many Are Useful: Learning a Variable's Importance by Studying an Entire Class of Prediction Models Simultaneously. *J. Mach. Learn. Res.* **2019**, *20*, 1–81.
43. Kupidura, P.; Kępa, A.; Krawczyk, P. Comparative analysis of the performance of selected machine learning algorithms depending on the size of the training sample. *Rep. Geod. Geoinform.* **2024**, *118*, 53–69.
44. Gulati, J. Logistic vs SVM vs Random Forest: Which One Wins for Small Datasets? MachineLearningMastery.com 2025. Available online: https://machinelearningmastery.com/logistic-vs-svm-vs-random-forest-which-one-wins-for-small-datasets/ (accessed on 8 July 2026).
45. Probst, P.; Wright, M.N.; Boulesteix, A.-L. Hyperparameters and tuning strategies for random forest. *WIREs Data Min. Knowl. Discov.* **2019**, *9*, e1301.
46. Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794.
47. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780.
48. Bergstra, J.; Bengio, Y. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.* **2012**, *13*, 281–305.
49. Batista, G.E.A.P.A.; Prati, R.C.; Monard, M.C. A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explor. Newsl.* **2004**, *6*, 20–29.
50. Kuvayskova, Y.; Izyurova, A. Application of Sampling Algorithms for Sample Balancing in Diagnostics of Technical Object Condition. In Proceedings of the 2025 International Russian Smart Industry Conference (SmartIndustryCon), Sochi, Russia, 24–28 March 2025; pp. 670–674.
51. Grandini, M.; Bagli, E.; Visani, G. Metrics for Multi-Class Classification: An Overview. *arXiv* **2020**, arXiv:2008.05756.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.