International Journal of Computer, Network Security and Information System (IJCONSIST) ISSN: 2686-3480
Vol. 7 No. 2, Maret 2026, pp. 1-8

Performance Evaluation of XGBoost and Random Forest Models in Visibility Prediction at Juanda Airport

Ananda Amelia Pramaisita
Department of Mathematics, Faculty of Science & Technology
State Islamic University of Sunan Ampel Surabaya
nandaa.sita@gmail.com

Nurissaidah Ulinnuha*
Department of Mathematics, Faculty of Science & Technology
State Islamic University of Sunan Ampel Surabaya
nuris.ulinnuha@uinsa.ac.id

Yuniar Farida
Department of Mathematics, Faculty of Science & Technology
State Islamic University of Sunan Ampel Surabaya
yuniar_farida@uinsa.ac.id

Addien Haniefardy
Informatics
Universitas Pembangunan Nasional "Veteran" Jawa Timur
haniefardy.id@upnjatim.ac.id

***Abstract*— Meteorological visibility forecasting plays an essential role in aviation safety and operational decision-making, particularly under adverse weather conditions that may disrupt airport activities. Although ensemble machine learning methods such as Random Forest and XGBoost have demonstrated promising performance in various forecasting applications, comparative studies focusing on univariate visibility forecasting in aviation environments remain limited. This study aims to evaluate and compare the predictive performance of Random Forest and XGBoost for hourly visibility forecasting at Juanda Airport, Indonesia. The study utilized one year of hourly visibility observations obtained from BMKG Juanda Station. Data preprocessing included missing-value imputation, normalization, lag-feature generation, and chronological train-test splitting. Hyperparameter optimization was performed using GridSearchCV with TimeSeriesSplit cross-validation. Model performance was evaluated using Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE), and computational processing time. The results indicate that both models successfully captured temporal visibility patterns; however, Random Forest achieved slightly superior performance with an MAE of 808.54, RMSE of 1,312.64, MAPE of 21.09%, and a processing time of 1.02 seconds, compared with XGBoost, which obtained an MAE of 808.81, RMSE of 1,323.12, MAPE of 21.47%, and a processing time of 1.36 seconds. These findings suggest that Random Forest provides a more efficient and reliable approach for operational visibility forecasting, while XGBoost remains a competitive alternative for applications requiring greater sensitivity to short-term fluctuations.**

Keywords—Prediction, Time series, Visibility, XGBoost, Random Forest

© 2026 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution ShareAlike (CC BY SA) license (https://creativecommons.org/licenses/by-sa/4.0/).

I. INTRODUCTION

Visibility represents a significant factor in the field of aviation, as it has a direct impact on flight safety, operational smoothness, and efficiency [1]. Reduced visibility resulting from fog, heavy precipitation, or other severe weather phenomena can result in delays, cancellations, and even rerouting of flights [2]. For instance, in February 2025, nine flights at Juanda Airport were rerouted due to extreme weather conditions that resulted in reduced runway visibility [3]. This situation underscores the importance of precise visibility forecasting, enabling airport management and airlines to make informed decisions that minimize associated risks and financial losses. In the context of aviation, visibility is essential in assessing the viability of flights. Generally, a flight may be deemed viable if visibility is within a minimum range of 800 meters to more than 5,000 meters, contingent upon the specific flight phase and the nature of the airport operations being conducted [4].

To address such problems, machine learning techniques like Random Forest have increasingly been employed in predicting visibility. It is an ensemble technique consisting of multiple decision trees, whose ensemble results in robust prediction outcomes. Numerous research studies have demonstrated its success in a wide range of applications. Recent research in the field of visibility prediction has yielded significant contributions. For example, a study utilized a backpropagation neural network that incorporated weather variables and achieved a level of prediction accuracy above 93% [5]. Moreover, a study at Patna Airport in India investigated the prediction of fog and thick fog using a series of machine learning algorithms, including Random Forest and XGBoost. The study concluded that a stacked ensemble method proved superior to individual algorithms in obtaining more accurate early warning results [6]. Another study at Surabaya's Juanda International Airport developed a system for predicting visibility based on a modified version of backpropagation, achieving an accuracy level of 93.22%

when fine-tuned using a learning rate of 0.001 over 24 hidden layers [7].

Unlike multivariate forecasting approaches that require multiple meteorological variables, this study focuses on univariate visibility forecasting using historical visibility observations only. We opted for this approach to assess how precisely temporal patterns can be predicted using visibility measurement numbers independently of meteorological parameter inputs into the ensemble machines. In addition, visibility measurements are always conducted at most of the meteorological stations, thus making the suggested methodology more implementable.

Even though there are numerous studies evaluating the performance of Random Forest and XGBoost approaches in different cases, very few studies exist comparing the performance of those two algorithms when it comes to univariate visibility prediction. Although both algorithms are types of ensemble learning techniques, however, Random Forest and XGBoost differ in terms of learning approaches. In particular, Random Forest applies a bagging learning approach, which is characterized by the benefits of stability, overfitting avoidance, and efficiency, whereas XGBoost utilizes boosting, hence increasing accuracy while improving bad predictions, but demanding much more computation resources. This gap will be addressed in our research, whereby a systematic comparison of the above approaches will be made using identical preprocessing, hyperparameters tuning, and evaluation approaches. The value of the paper is in the empirical investigation of the added benefits of boosting learning compared to the bagging one.

II. METHODOLOGY

The stages in research methodology begin with data acquisition by BMKG Juanda, followed by data pre-processing, dataset splitting, model training, and model performance evaluation, as shown in Fig. 1. These stages are performed in sequence to ensure the effective quality and credibility of the predicted results.

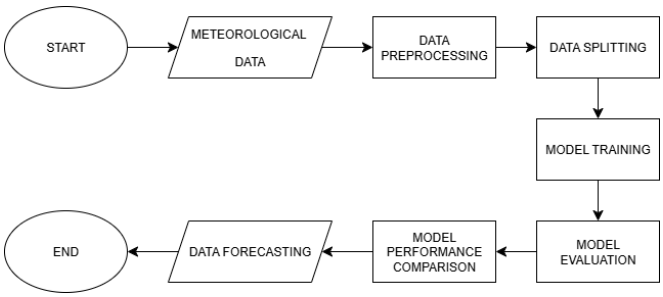

Fig. 1. Research Stages

A. Data Collection

This study uses hourly visibility observations obtained from the Meteorology, Climatology, and Geophysics Agency (BMKG) station at Juanda, Sidoarjo. The dataset comprising 8,784 observations was collected over a one-year period, from January 1, 2024, to January 1, 2025, on an hourly basis for each dataset. The key variable for the purpose of this research is visibility, measured at each designated time of recording.

B. Data Preprocessing

In data analysis, raw data are often collected from various sources and require preprocessing before further analysis. Therefore, preprocessing is required to improve data quality and prepare the dataset for model development. In this phase the data quality is increased through processes such as filling in the empty cells, removing anomalies, and reconciling differences in the data to make it more consistent. This step is very important as it improves model accuracy in forecasting and other analysis from the data. The data preprocessing steps in this research include addressing imputed missing data, data normalization, and feature engineering including the creation of lag variables and the removal of newly created missing data.

C. Data Splitting

The pre-processed data set was split chronologically into training and testing sets in the ratio of 80% and 20%, respectively, without random shuffle so as to maintain the temporal relationship between the data points. The training set was employed in developing the model and conducting the optimization of the hyperparameters, whereas the test set was employed only in evaluation [8].

On the other hand, the test data set assesses the capability of the model to forecast data points it has never seen before. Usually, the data set aside for testing is the least, around 20% to 30%, so the model can be assessed using data fully independent of the training. This assessment of the testing data can determine to what degree the model can make generalized predictions about new data, as well as observe the model's predictions for signs of overfitting. As a result, the objectivity of the model's predictive results can be evaluated.

D. Model Training

1) Model Hyperparameter Tuning

This is the stage at which we adjust parameters to arrive at the perfect model configuration through Cross-Validation combined with the Grid Search approach. This employs the GridSearchCV function to perform a structured and thorough examination of the different combinations of values given for each of the hyperparameters [8] in a model such as in the case of Random Forest and XGBoost. A model’s performance is assessed through a Grid Search for each of the possible values for a given parameters set along with the TimeSeriesSplit which gives a more true characterization of the data in a time series structure.This study aims to optimize predictive capacity by determining the parameters best suited for use in a specific study area, based on a set of meteorological visibility data.

2) Random Forest

Random Forest Regression is an ensemble learning method that combines multiple decision trees to improve prediction accuracy and reduce overfitting. Each tree is

trained using a bootstrap sample of the training data, and the final prediction is obtained by averaging the predictions from all trees [9]. The prediction generated by the Random Forest model is expressed as [10]:

$$\hat{y} = \frac{1}{B}\sum_{b=1}^{B} T_b(x) \quad (1)$$

where:
$\hat{y}$ = predicted visibility value,
B = total number of decision trees
$T_b(x)$ = prediction produced by the b-th decision tree
x = input feature vector.

To determine the optimal split at each node, Random Forest Regression commonly minimizes the Mean Squared Error (MSE), defined as [10]:

$$MSE = \frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y_i})^2 \quad (2)$$

where:
n = number of observations
y_i = actual visibility value
$\hat{y_i}$ = predicted visibility value

3) Extreme Gradient Boosting (XGBoost)

XGBoost is an ensemble learning algorithm based on gradient boosting that constructs decision trees sequentially. Each new tree is trained to minimize the prediction errors produced by the previous trees [11]. The prediction generated by XGBoost is formulated as [12]:

$$\hat{y_i} = \sum_{k=1}^{K} f_k(x_i) \quad (3)$$

where:
$\hat{y_i}$ = predicted value for observation i
K = number of trees
f_k = prediction function of the k-th tree
x_i = input feature vector for observation i

The objective function optimized by XGBoost is [12]:

$$Obj = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \quad (4)$$

where:
Obj = objective function
$l(y_i, \hat{y}_i)$ = loss function measuring prediction error
y_i = actual value
$\hat{y}_i$ = predicted value
$\Omega(f_k)$ = regularization term for controlling model complexity

The regularization function is defined as [12]:

$$\Omega(f_k) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \quad (5)$$

where:
T = number of leaves in a tree
w_j = weight of leaf j
γ = penalty parameter for the number of leaves
λ = L2 regularization coefficient

For regression problems such as visibility forecasting, XGBoost typically employs the squared error loss function [13]:

$$l(y_i, \hat{y}_i) = (y_i - \hat{y_i})^2 \quad (6)$$

E. Model Evalution

The developed model was evaluated using Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE) to measure forecasting accuracy. These metrics were selected to assess prediction accuracy from complementary perspectives, including average absolute deviation, sensitivity to large prediction errors, and relative percentage error. Lower values of RMSE, MAE, and MAPE indicate better predictive accuracy [7]. The evaluation metrics are calculated using Equations (7)– (9).

$$RMSE = \sqrt{\sum_{i=1}^{n} \left(\frac{\hat{y}_i - y_i}{n}\right)^2} \quad (7)$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |\hat{y}_i - y_i| \quad (8)$$

$$MAPE = \sum_{i=1}^{n} \left|\frac{\hat{y}_i - y_i}{y_i}\right| \times 100 \quad (9)$$

where $\hat{y}_i$ is the predicted value for the i-th observation, while y_i is the actual observed value for the i-th observation. We are left with n for the number of observations or data points

III. Result and Discussion

This section presents the results of data preprocessing, model development, hyperparameter optimization, model evaluation, and seven-day-ahead visibility forecasting. The performance of Random Forest and XGBoost is compared using forecasting accuracy metrics and computational efficiency to identify the most suitable model for operational visibility prediction.

A. Data Preprocessing

1) Initial Data Description

In regard to the data for this project, as shown in TABLE 1, there are 8,784 data rows for analysis, split into 3 variables. This data covers January 1, 2024, to January 1, 2025. As this project centers around univariate time-series analysis, there will be specific focus on the column with the headers marked as (UTC), as this Data will be used to form the time identifier, and Visibility, as this is the target variable. Accurate and reliable visibility data are essential for developing a robust forecasting model.

2) Descriptive Statistics of the Data

Based on TABLE 2, visibility values ranged from 200 m to 80,000 m, indicating substantial variability in atmospheric conditions throughout the observation period. The mean visibility was 7,933 m, while the median was 8,000 m, suggesting a relatively balanced distribution with no strong skewness. The standard deviation of 2,417 meters suggests there is quite a bit of variation around the mean visibility value. In addition to this, some observations had a visibility less than 550 meters, indicating dangerous operating conditions. It can therefore be concluded that this data has normal as well as abnormal visibility readings.

TABLE 1. VISIBILITY DATA

Date	Time (UTC)	Visibility
01/01/2024	01:00	5,000
01/01/2024	02:00	10,000
01/01/2024	03:00	10,000
01/01/2024	04:00	10,000
…	…	…
31/12/2024	20:00	5,000
31/12/2024	21:00	5,000
31/12/2024	22:00	8,000
31/12/2024	23:00	8,000

TABLE 2. Descriptive Statistics

Statistic	Visibility
Count	8,784
Mean	7,933
Std	2,417
Min	200
25%	6,000
50%	8,000
75%	10,000
Max	80,000

3) Handling Missing Values in Visibility

TABLE 3 contains the findings of the first inspection which shows some data entries have missing values. In particular, the column Visibility that is the target of prediction shows that 61 entries are missing.

TABLE 3. MISSING VALUES BEFORE INTERPOLATION

Date	Time (UTC)	Visibility
03/01/2024	13:00:00	NaN
03/01/2024	14:00:00	NaN
03/01/2024	15:00:00	NaN
03/01/2024	16:00:00	NaN
…	…	…
03/12/2024	14:00:00	NaN
12/12/2024	06:00:00	NaN
15/12/2024	13:00:00	NaN
01/01/2025	00:00:00	NaN

TABLE 4 will be used to fill in the gaps for the first round of analysis. The gaps in the visibility column will be filled using linear interpolation since that column is the primary focus of the time series forecasting analysis. Linear interpolation will fill in the gaps without dramatically altering the local patterns or the trends in the data. This will allow for better estimation of missing data of the visibility time series by using information from data points in close temporal proximity. The completed visibility time series is continuous and representative, as shown in TABLE 5.

TABLE 4. Missing Values After Interpolation

Date	Time (UTC)	Visibility
03/01/2024	13:00:00	5,000.00
03/01/2024	14:00:00	5,000.00
03/01/2024	15:00:00	5,000.00
03/01/2024	16:00:00	5,000.00
…	…	…
03/12/2024	14:00:00	7,500.00
12/12/2024	06:00:00	6,500.00
15/12/2024	13:00:00	6,000.00
01/01/2025	00:00:00	8,000.00

4) Data Normalization

TABLE 5 holds normalized visibility data ready for scaling for machine learning modeling. This means every visibilty value has been scaled with all other values so that no data feature disproportionately represents the segment of data being analyzed.

TABLE 5. Normalization of Visibility Data

Data After Normalization		
Date	**Time (UTC)**	**Visibility**
01/01/2024	01:00	0.06015
01/01/2024	02:00	0.12281
01/01/2024	03:00	0.12281
01/01/2024	04:00	0.12281
…	…	…
31/12/2024	20:00	0.06015
31/12/2024	21:00	0.06015
31/12/2024	22:00	0.09774

Data After Normalization		
Date	**Time (UTC)**	**Visibility**
31/12/2024	23:00	0.09774

5) Feature Engineering

Following feature engineering by adding lag features (lag_1, lag_2, lag_3), each visibility observation is now supplemented with visibility values from one, two, and three hours earlier. Including these features enables the model to learn from historical patterns in the data and is expected to improve prediction accuracy. As shown in TABLE 6, the dataset retained its completeness after creating the lag features. After generating lag features, the first three observations contained undefined lag values and were therefore removed prior to model training.

Subsequently, the data, which had been enriched with lag features and cleaned of missing values, were divided into two main parts: a training set comprising 80% and a testing set containing 20%. The splitting was carried out without shuffling to preserve the sequence of events, hence the features of the time series and the reliability of the testing.

TABLE 6. Data After Adding Lag Features

No	Visibility	lag_1	lag_2	lag_3
1	5,000	NaN	NaN	NaN
2	10,000	5,000	NaN	NaN
3	10,000	10,000	5,000	NaN
4	10,000	10,000	10,000	5,000
…		…	…	…
8781	5,000	5,000	6,000	6,000
8782	5,000	5,000	5,000	6,000
8783	8,000	5,000	5,000	5,000
8784	8,000	8,000	5,000	5,000

B. Model Evaluation and Comparison

1) Modeling and Hyperparameter Tuning

In this case, the Random Forest and XGBoost methods are employed for predicting the values of visibility based on the processed data. Initially, both models were trained with the parameters by default. Then, these two models were tuned by adjusting their parameters through GridSearchCV. At this stage, we seek the parameter values that will ensure the effectiveness of the model predictions.

In order to use a cross-validation technique that will be suitable for time series analysis, we decided to use TimeSeriesSplit cross-validator, where the order of events in data will be preserved for each fold. This way, we will be able to get reliable results from the tuning and will be able to see whether our model can successfully predict the future based on the past information. If the models are not tuned properly, we will not obtain a reliable model that can be used for predicting on the test set. Parameters for Random Forest and XGBoost are listed in TABLE 7.

The hyperparameters are crucial in managing the complexity of the model and predictive performance. In case of the Random Forest model, the hyperparameters n_estimators and max_depth help in specifying the number of trees and the maximum depth of each tree, respectively. On the other hand, the hyperparameters min_samples_split and min_samples_leaf help in specifying the criteria for splitting nodes and creating leaves to avoid overfitting. Additionally, the hyperparameters max_features and bootstrap are critical when determining the variability of trees within the model. In case of the XGBoost model, the hyperparameters n_estimators and max_depth manage the complexity of the model while learning_rate specifies the impact that each tree will have on the overall model through the boosting process. In addition, the hyperparameters subsample and colsample_bytree are instrumental in reducing overfitting.

TABLE 7. Hyperparameter Random Forest & XGBoost

Model	Parameters	Range
Random Forest	n_estimators	[100, 200, 300]
Random Forest	max_depth	[5, 10, 20, None]
Random Forest	min_samples_split	[2, 5, 10]
Random Forest	min_samples_leaf	[1, 2, 4]
Random Forest	max_features	['auto', 'sqrt']
Random Forest	bootstrap	[True, False]
XGBoost	n_estimators	[100, 200, 300]
XGBoost	learning_rate	[0.01, 0.05, 0.1]
XGBoost	max_depth	[3, 5, 7, 10]
XGBoost	subsample	[0.8, 1.0]
XGBoost	colsample_bytree	[0.8, 1.0]

2) Model Training

Prior to the training of the final model, a thorough process of hyperparameter tuning was carried out for two types of algorithms, namely Random Forest and XGBoost, where some specific parameter combinations were selected that gave the highest performance level. The results of the process are demonstrated in TABLE 8 and TABLE 9, presenting the variations in hyperparameter combinations and validation score. As a result, the final model is likely to predict the visibility more accurately.

With regard to the hyperparameter tuning process carried out in TABLE 8 and TABLE 9, different values for each hyperparameter were applied to get the ideal settings of Random Forest and XGBoost regression models. For

example, parameters such as max_depth, min_samples_leaf, min_samples_split, and n_estimators were experimented with in case of Random Forest. From the results of tuning for Random Forest, it is found that max_depth = 10, min_samples_leaf = 1, min_samples_split = 10, and n_estimators = 300 had the minimum MAE, which is 808.54. It means that a model, where the tree depth is not too big and the number of estimators is quite high, will be capable of finding correlations within granular data, which would prevent overfitting.

TABLE 8. The outcome for Hyperparameter tuning for the XGBoost model

Colsample by tree	Learning rate	Max depth	n estimators	Subsample	MAE
0.8	0.01	3	100	0.8	1204.990364
0.8	0.01	3	100	1.0	1202.478416
0.8	0.01	3	200	0.8	979.323510
...	...	...	...	...	...
0.8	0.1	3	100	1.0	822.230582
0.8	**0.1**	**3**	**200**	**0.8**	**808.814508**
0.8	0.1	3	200	1.0	822.167619
...	...	...	...	...	...
1.0	0.1	10	100	0.8	867.359034
1.0	0.1	10	100	1.0	880.301701
1.0	0.1	10	200	0.8	867.359048

TABLE 9. Hyperparameter Tuning Results for the Random Forest Model

Max depth	Min sample leaf	min sample Split	n estimators	MAE
5	1	2	100	855.707895
5	1	2	200	854.036860
5	1	2	300	847.163462
…	…	…	…	…
10	1	10	200	811.630785
10	**1**	**10**	**300**	**808.543026**
10	2	2	100	826.927375
…	…	…	…	…
20	4	10	100	829.709729
20	4	10	200	829.411381
20	4	10	300	826.337523

In case of XGBoost, the following parameters were considered: colsample_bytree, learning_rate, max_depth, n_estimators, and subsample. These parameters were found at their optimal configuration when colsample_bytree = 0.8, learning_rate = 0.1, max_depth = 3, n_estimators = 200, and subsample = 0.8, at which an MAE of 808.81 was achieved. The findings above illustrate that an XGBoost model, where tree depth is relatively small but learning rate is not too low, will be able to optimize learning efficiently. Hence, both models have been optimized in such a way that the MAE value for the best setting is lower than for the others.

3) Model Evaluation

The results in TABLE 10 indicate that Random Forest and XGBoost had comparable levels of accuracy. Specifically, the former had MAE of 808.54, RMSE of 1,312.64, and MAPE of 21.09%, while the latter showed MAE of 808.81, RMSE of 1,323.12, and MAPE of 21.47%. The margin difference between the two models' performance can be explained by their capacity for effective learning of temporal dependencies between visibility records.

TABLE 10. Summary of Evaluation Results of All Models

Model	Processing Time (seconds)	MAE	RMSE	MAPE
XGBoost	1.361033	808.81450	1,323.11870	21.4715
Random Forest	1.016363	808.54302	1,312.63944	21.0938

Despite the performance difference being minor, the Random Forest model demonstrated better results in all three metrics considered in this work. This might be due to the relative stability of the visibility dataset in combination with lag-based features, which is favorable for using the bagging algorithm used in the Random Forest model. Namely, by building multiple decision trees and taking the average of their outputs, the Random Forest model reduces variance and produces accurate predictions while being less complex than other algorithms.

On the contrary, the boosting technique used by XGBoost implies adding new decision trees to address prediction errors in the previous trees. While this method is usually beneficial when analyzing highly non-linear data, it proved to be unnecessary in the visibility forecasting case. The almost negligible differences between MAE and RMSE values

indicate that the underlying relationships in the data can still be effectively captured by both methods.

In terms of computations, Random Forest proved to be faster as it took only 1.02 seconds, as opposed to 1.36 seconds required by XGBoost. Though this difference in computational times is less than one second, it becomes increasingly significant once the prediction system is operational and needs to execute the model repeatedly. These results agree with existing studies [14] that have found that Random Forest provides comparable predictive performance with less computational complexities than boosting algorithms. Thus, by taking into consideration the above-discussed criteria of model prediction accuracy, efficiency, and simplicity, we conclude that Random Forest is the preferable model for visibility forecasting at Juanda Airport.

C. Seven-Day Ahead Visibility Forecast Results

1) Comparison of Model Predictions with Actual Data

Fig. 2 shows a comparison of visibility predictions made by the two most efficient models, Random Forest (RF) and XGBoost (XGB), with the true value of visibility in the test dataset. In general, both the models have been able to capture the time trends that are seen in the visibility series and daily variations quite effectively. In instances when the actual value of visibility is higher, the predictions made using both these models will be close to each other; similarly, in situations when there is a decrease in the value of visibility, both these models would predict the trend correctly.

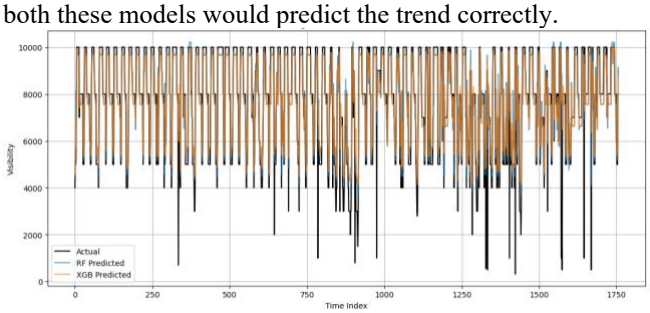

Fig. 2. Comparison of Actual and Predicted

There is a difference between the predicted value and the actual values, especially when changes or decreases in visibility happen sharply. Both Random Forest and XGBoost would fail to capture some very low visibility values, thus not predicting the lowest visibility points accurately. Even though XGBoost seems to respond to the changes better than Random Forest, there is still a difference at certain extremely low points due to the fact that both these models average the values while working with target metrics.

2) Forecasting Results of the Best Model

The TABLE 11 shows the visibility forecasts for seven days ahead from the optimized Random Forest and XGBoost methods. The forecasts produced by the two models exhibit visibility values that stay within operational boundaries during the entire period of prediction. It is clear from the results shown below that there is no expected degradation of visibility.

TABLE 11. Forecasting Results for Each Model

Datetime	XGB_Forecast	RF_Forecast
01/01/2025 00:00	8,679.328000	8,834.582568
01/01/2025 01:00	8,144.883300	7,783.046735
01/01/2025 02:00	5,658.150400	6,369.770645
01/01/2025 03:00	8,034.175000	9,093.255540
…	…	…
07/01/2025 20:00	8,263.840000	7,728.722849
07/01/2025 21:00	6,195.913600	5,548.335444
07/01/2025 22:00	5,587.177200	6,208.898905
07/01/2025 23:00	8,197.979000	7,728.722849

As regards the difference between the two methods, it becomes evident through the variation in forecasts from one hour to another. XGBoost method displays higher variation since it reacts more vigorously to changes in the historical values. On the contrary, Random Forest exhibits lower variation because it performs better at noise filtering than the first method. Convergence in forecast values in the end period implies the identification of the same pattern in the series. This observation strengthens the conclusion that the dominant visibility dynamics in the dataset can be captured effectively by either model, although Random Forest provides more stable forecasts and slightly better predictive performance.

Based on Fig. 3, unlike the Random Forest model, which shows a much more even and smooth output trend, XGBoost shows sharper spikes and fluctuations during the 7-day forecast period provided in the output. This difference indicates that XGBoost captures daily changes in the dataset better than Random Forest, which appears to filter out noise in the data and provide conservative predictions. This highlights the importance of tailoring the model to the specific needs of the use case, whether one is looking for forecast stability or forecasts that reflect more irregular behavior in the available visibility information dataset.

From the seven-day-ahead forecast values shown in TABLE 11, all values for flights in all hours of operation are much greater than the minimum threshold level for flight operability set between 800 and 1,000 meters in keeping with standard commercial aviation operations.

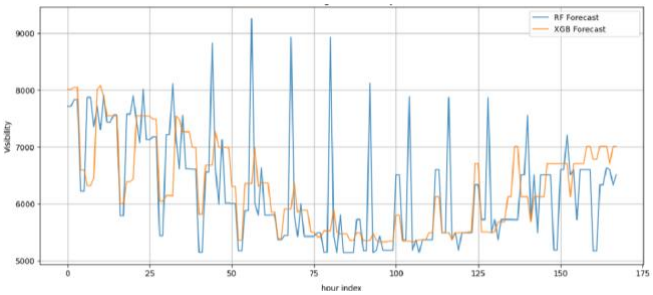

Fig. 3. Visualization of the Seven-Day-Ahead Forecast

IV. CONCLUSION

This study evaluated the performance of Random Forest and XGBoost for univariate visibility forecasting using hourly observations from Juanda Meteorological Station. From the results, we can see that both machine learning algorithms successfully capture dominant temporal dynamics in the dataset, resulting in satisfactory predictions. Nevertheless, Random Forest was more efficient in terms of computational speed, while also yielding better results. This indicates that boosting, although a relatively complicated method, failed to yield any improvements for the considered dataset, which makes the use of Random Forest a reasonable solution for visibility forecasting problems.

At the same time, the study is rather constrained because it only uses one input variable in the model and considers a single meteorological station. Further research may consider building multivariate forecasting models based on the inclusion of other parameters, such as humidity, temperature, and wind speed. In addition, more complex deep learning algorithms can be used for forecasting purposes.

V. REFERENCES

[1] D. G. A. Mahendra, I. Gst. Ngr. S. Adiputra, G. Indrawan, and I. M. A. O. Gunawan, "Evaluasi Pengalaman Pengguna Menggunakan Metode User Experience Questionnaire (UEQ) pada Automated Weather Observing System (AWOS) di Stasiun Meteorologi I Gusti Ngurah Rai," *Explore Jurnal Sistem Informasi dan Telematika*, vol. 15, no. 2, p. 256, 2024, doi: 10.36448/jsit.v15i2.4069.

[2] M. Musfiroh, D. C. R. Novitasari, P. K. Intan, and G. G. Wisnawa, "Penerapan Metode Principal Component Analysis (PCA) dan Long Short-Term Memory (LSTM) dalam Memprediksi Curah Hujan Harian," *Building of Informatics, Technology and Science (BITS)*, vol. 5, no. 1, 2023, doi: 10.47065/bits.v5i1.3114.

[3] Detik.com, "9 Jadwal Penerbangan Dialihkan dari Bandara Juanda Buntut Cuaca Buruk," 2025. [Online]. Available: https://www.detik.com/jatim/berita/d-7851330/9-jadwal-penerbangan-dialihkan-dari-bandara-juanda-buntut-cuaca-buruk

[4] J. Anjani, "Prediction of Air Temperature on Runway 10 Juanda Airport Using Hybrid LSTM," *Indonesian Journal of Electrical and Electronics Engineering (INAJEEE)*, vol. 7, no. 2, pp. 50–58, 2024.

[5] M. R. K. Hakim and A. Fanani, "Deteksi Jarak Pandang Aman Sebagai Acuan Untuk Keselamatan Penerbangan dengan Menggunakan Metode Backpropagation," *Jurnal RESISTOR (Rekayasa Sistem Komputer)*, vol. 1, no. 2, pp. 94–99, 2018.

[6] L. S. Moonlight, B. B. Harianto, A. Musadek, M. M. Sukma, and T. Arifianto, "Airport Visibility Prediction System to Improve Aviation Safety," in *International Conference on Advance Transportation, Engineering, and Applied Science (ICATEAS 2022)*, Atlantis Press, 2023, pp. 199–210.

[7] A. Shankar, B. C. Sahana, and S. P. Singh, "Prediction of Low-Visibility Events by Integrating the Potential of Persistence and Machine Learning for Aviation Services," *Mausam*, vol. 75, no. 4, pp. 977–992, 2024.

[8] H. Tao *et al.*, "Training and Testing Data Division Influence on Hybrid Machine Learning Model Process: Application of River Flow Forecasting," *Complexity*, vol. 2020, pp. 1–22, 2020, doi: 10.1155/2020/8844367.

[9] R. N. Fadhila, N. Ulinnuha, and D. Yuliati, "Implementation of Random Forest Method with Information Gain Selection and Hyperparameter Tuning for Alzheimer's Disease Classification," *Lontar Komputer : Jurnal Ilmiah Teknologi Informasi*, vol. 16, no. 1, pp. 1–13, Oct. 2025, doi: 10.24843/LKJITI.2025.v16.i01.p01.

[10] M. B. Akram *et al.*, "Improving time series forecasting for extrusion process monitoring using a hybrid SARIMA & Random Forest model," *J. Process Control*, vol. 161, p. 103698, May 2026, doi: 10.1016/j.jprocont.2026.103698.

[11] S. F. Aqilah Khansa, N. Ulinnuha, and W. D. Utami, "Grid Search and Random Search Hyperparameter Tuning Optimization in Xgboost Algorithm for Parkinson's Disease Classification," *BAREKENG: Jurnal Ilmu Matematika dan Terapan*, vol. 19, no. 3, pp. 1609–1624, Jul. 2025, doi: 10.30598/barekengvol19iss3pp1609-1624.

[12] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794. doi: 10.1145/2939672.2939785.

[13] Y. Wang, X. Li, H. Zhang, and others, "Time Series Analysis of Hemorrhagic Fever with Renal Syndrome in Mainland China by Using an XGBoost Forecasting Model," *BMC Infect. Dis.*, vol. 21, no. 839, 2021, doi: 10.1186/s12879-021-06503-y.

[14] G. Dudek, "A Comprehensive Study of Random Forest for Short-Term Load Forecasting," *Energies (Basel).*, vol. 15, no. 20, p. 7547, Oct. 2022, doi: 10.3390/en15207547.