Joinforcs: Journal of informatics and information system
Volume 1, (1) 2026, p. 14-20
E-ISSN : XXXX-XXXX

Comparative Analysis of Decision Tree, Random Forest, and X-Gradient Boost Algorithms for Weather Prediction in Seattle

Bagus Aditya Saputra [1], Khairul Ikhsan [2], Akbar Rizky Dio Saputra [3], Muhammad Galih Arifin [4]
Informatics Study Program, STMIK AMIKOM Surakarta, Sukoharjo, Indonesia
[1] bagus.10487@mhs.amikomsolo.ac.id, [2] khairul.10356@mhs.amikomsolo.ac.id[3] akbar.10355@mhs.amikomsolo.ac.id, [4] muhammad.10368@mhs.amikomsolo.ac.id

Article Info

Article History:

Received Dec 16, 2025
Revised Jan 9, 2026
Accepted Jan 23, 2026

Keywords:
Decision Tree, Machine learning, Random Forest, X-Gradient Boost, Weather prediction

Abstrak

Weather prediction plays an essential role in supporting various sectors such as agriculture, transportation, and urban planning, particularly in regions with highly dynamic weather conditions like Seattle. This study aims to analyze and compare the performance of three machine learning algorithms Decision Tree, Random Forest, and X-Gradient Boost in predicting weather conditions using historical meteorological data. The dataset was obtained from Kaggle and includes several key attributes, such as precipitation, humidity, air pressure, temperature, and wind speed. The research methodology consists of data collection, preprocessing, model training, and performance evaluation using accuracy, precision, recall, and F1-score metrics. Model implementation and experimentation were conducted using the Google Colab platform, with hyperparameter tuning applied through GridSearchCV to optimize model performance. The experimental results indicate that the X-Gradient Boost algorithm achieved the highest accuracy of 84%, followed by Random Forest with 83.96% and Decision Tree with 83% after tuning. Based on these results, X-Gradient Boost is identified as the most effective algorithm for weather prediction in this study. These findings are expected to contribute to the development of more accurate and reliable machine learning-based weather forecasting systems.

1. Introduction

Weather is defined as the atmospheric state at a specific time and within a relatively narrow and short-term geographical area[1]. It plays a crucial role in various aspects of human life, spanning from daily activities to industrial sectors such as agriculture, aviation, and urban planning. The city of Seattle, widely known for its high precipitation and rapid weather fluctuations, requires an accurate weather prediction system to assist the public and the government in making better-informed decisions. Currently, weather factors significantly affect human life, particularly in the agricultural and aviation sectors, among others.

However, predicting the weather is highly challenging due to the vast number of interacting variables that dictate atmospheric conditions. These variables such as atmospheric pressure, wind speed, precipitation, temperature, and other atmospheric phenomena are frequently observed only within short timeframes, which can be formulated into several components[2]. Traditional weather forecasting systems heavily rely on conventional statistical methods and physical atmospheric models. Unfortunately, these approaches exhibit limitations when handling highly complex weather data, primarily because multiple variables interact dynamically in a short amount of time. Consequently, machine learning-based approaches have been increasingly adopted in meteorology to enhance prediction accuracy.

Several key challenges must be addressed in developing effective weather forecasts. First, it demands diverse data sources, including direct observations, satellite cloud imagery, and radar scan outputs. Second, traditional forecasting remains highly dependent on the knowledge and subjective expertise of individual forecasters, which can result in inconsistent predictions from one professional to another[3]. One machine learning method that has proven highly effective for weather classification is the Random Forest Classifier. Random Forest is an ensemble learning algorithm utilized for classification and prediction tasks[4]. The primary advantages of ensemble learning include its versatility for both classification and regression tasks,

its capability to achieve high accuracy, and its suitability for high-dimensional, large-scale datasets[5]. This method operates by constructing and combining multiple decision trees to mitigate overfitting and enhance prediction stability. Numerous studies have demonstrated that Random Forest can deliver superior accuracy compared to alternative machine learning methods, particularly in historical data-driven weather predictions.

To evaluate and ensure optimal results, Decision Tree, Random Forest, and X-Gradient Boost (XGBoost) algorithms are incorporated into this comparative study to determine which method yields the highest performance. XGBoost is an ensemble algorithm based on the Gradient Boosting Decision Tree framework[6]. This algorithm constructs sequential decision trees to systematically correct errors made by preceding trees.

Previous research has extensively implemented Random Forest Classifiers across various weather classification scenarios. For instance, studies using ensemble learning approaches successfully improved the accuracy of predicting atmospheric conditions based on parameters like temperature, humidity, air pressure, wind speed, and precipitation. Additionally, oversampling techniques such as the Synthetic Minority Over-sampling Technique (SMOTE) have proven effective in addressing class imbalances in weather datasets, which frequently pose challenges in historical data-driven predictions.

Furthermore, prior research by Apriliah et al. demonstrated that the Random Forest algorithm exhibits exceptional classification performance on complex datasets with numerous attributes, achieving an accuracy above 97% in medical prediction cases[7]. This indicates that Random Forest is robust in handling non-linear patterns and multivariate data. In terms of comparative analysis, Nugraha demonstrated that ensemble algorithms like XGBoost outperform conventional classification algorithms when applied to cardiovascular disease data[8]. This reinforces the notion that ensemble learning holds a distinct advantage in boosting model stability and accuracy.

In the specific context of weather and precipitation data, Decision Tree-based classification has also been widely adopted. A study by Hasanah et al. utilized the Decision Tree algorithm within a CRISP-DM framework to predict flood-prone rainfall, securing an accuracy of 89.4%[9]. Their findings showcase that Decision Trees offer excellent interpretability when classifying weather data based on meteorological parameters. Moreover, Decision Trees have shown strong efficacy across diverse historical data classification tasks; Qisthiano et al. applied it to predict student graduation rates and achieved an accuracy of 87.93%[10]. This success further underscores that Decision Trees perform well on large datasets with multi-attribute variables.

Based on these prior studies, it is evident that Decision Tree, Random Forest, and XGBoost algorithms possess strong capabilities in resolving classification problems within complex datasets. Therefore, this study aims to systematically analyze and compare the performance of Decision Tree, Random Forest, and XGBoost algorithms for weather prediction in Seattle. Through this rigorous comparison, the study intends to identify the most optimal algorithm among the three.

2. Research Methodology

The Research Methodology section systematically explains the steps used to address the research problem. This research incorporates several structured stages to arrive at a definitive conclusion. The systematic workflow begins with Data Collection, followed by Data Preprocessing, Data Analysis, Data Presentation, and culminates in Performance Analysis.

2.1 Data Collection

This stage involves gathering the required dataset from open source platform repositories, specifically via the Kaggle.com website. The retrieved dataset consists of raw, unprocessed historical records containing attributes such as location, date, precipitation, humidity, maximum and minimum temperatures, wind speed, and the corresponding weather condition labels.

Joinforcs: Journal of informatics and information system
Volume 1, (1) 2026, p. 14-20
E-ISSN : XXXX-XXXX

2.2 Data Preprocessing

Tables Following data acquisition, the next crucial step is Data Preprocessing. At this stage, the raw dataset is cleaned and transformed to ensure it is structurally sound for analysis. The primary operations include removing missing values or duplicate records, dropping irrelevant columns that do not contribute to the analytical objectives, performing data normalization, and encoding categorical text attributes into numerical formats.

2.3 Data Analysis

During this phase, an exploratory and structural analysis is performed on the Seattle weather dataset to identify and evaluate the suitability of the selected machine learning models. Based on literature and empirical trends, Decision Tree, Random Forest, and X-Gradient Boost are selected as highly effective algorithms widely recognized for classification tasks across various predictive domains, including meteorology and healthcare.

2.4 Data Presentation

The evaluation and performance results derived from each algorithm are presented using intuitive and structured data visualizations. Presenting the metrics visually ensures that the conveyed information is highly interpretable, allowing for quick insights and serving as a reliable foundation for data-driven decision-making.

2.5 Performance Analysis

The final stage focuses on assessing and evaluating the performance of the trained models, which includes measuring accuracy, error rates, precision, recall, F1-score, and other relevant evaluation matrices depending on the specific algorithm. The ultimate objective is to validate, verify, and compare the results to identify the most optimal algorithm among Decision Tree, Random Forest, and X-Gradient Boost for predicting weather conditions in Seattle.

3. Results and Discussion

This research was conducted using Google Colab, a cloud-based environment providing GPU acceleration to expedite computational workloads during machine learning model training to run the three algorithms, namely Decision Tree, Random Forest, and X-Gradient Boost. The data manipulation and computations were implemented using the pandas library for dataset management and numpy for handling numerical array operations

3.1 Loading the Dataset

The initial step involved establishing a connection to the data source and displaying a subset of the dataset to verify that the system read the structure properly, show in Figure 1.

```
import pandas as pd

# Path ke file di Google Drive
file_path = "/content/drive/My Drive/seattle-weather.csv"

# Membaca dataset
df = pd.read_csv(file_path)

# Menampilkan beberapa baris pertama
df.head()
```

Figure 1. Connecting Dataset

3.2 Importing Libraries

Once the dataset was successfully loaded, the necessary software libraries required for data execution and modeling were imported. The implementation leverages numpy frameworks to run the three algorithms to be tested, show in Figure 2.

Joinforcs: Journal of informatics and information system
Volume 1, (1) 2026, p. 14-20
E-ISSN : XXXX-XXXX


```
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
```

Figure 2. Import Library

3.3 Data Preprocessing Implementation

In the next stage, data preprocessing was performed to ensure the dataset was undamaged and ready for analysis. Researchers converted categorical data to numerical data and split the dataset into two parts for training and testing, shown in Figure 3.

```
df["weather"] = LabelEncoder().fit_transform(df["weather"])
X = df.drop(columns=["date", "weather"])  # Fitur
y = df["weather"]  # Target

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figure 3. Data preprocessing

3.4 Model Execution

Following preprocessing, the chosen machine learning algorithms were initialized and trained. Using the designated training and testing subsets, baseline models were executed; for instance, the Random Forest model was initially configured with an n_estimators parameter of 100 and a random_state of 42, shown in Figure 4.

```
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
```

Figure 4. Running the Random Forest Algorithm

3.5 Hyperparameter Tuning

To optimize model performance and extract the best possible results, hyperparameter tuning was conducted via GridSearchCV to determine the parameters with the best results. The grid search configuration systematically evaluated combinations of various hyperparameters, including: n_estimators (the number of trees), max_depth (the maximum depth of each tree), min_samples_split (the minimum samples required to split an internal node), min_samples_leaf (the minimum samples required at a leaf node), and max_features (the number of features considered for the best split to enhance model diversity), shown in Figure 5..

```
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestClassifier

# Definisikan parameter yang akan dicoba
param_grid = {
    'n_estimators': [100, 200, 300],  # Coba 100, 200, 300 pohon
    'max_depth': [10, 20, None],  # Coba batas kedalaman 10, 20, atau tanpa batas
    'min_samples_split': [2, 5, 10],  # Minimum sampel untuk split
    'min_samples_leaf': [1, 2, 4],  # Minimum sampel di setiap daun
    'max_features': ['auto', 'sqrt']  # Jumlah fitur yang dipilih untuk setiap pohon
}

# Inisialisasi model
rf = RandomForestClassifier(random_state=42)

# Gunakan GridSearchCV untuk mencari kombinasi terbaik
grid_search = GridSearchCV(estimator=rf, param_grid=param_grid,
                           cv=5, n_jobs=-1, verbose=2)

# Latih model dengan Grid Search
grid_search.fit(X_train, y_train)

# Tampilkan parameter terbaik
print("Parameter terbaik:", grid_search.best_params_)
```

Figure 5. Tuning on Parameter

Joinforcs: Journal of informatics and information system
Volume 1, (1) 2026, p. 14-20
E-ISSN : XXXX-XXXX

3.6 Performance Comparison Before and After Tuning

After identifying the optimal parameters for each model, a comparative analysis was performed against their respective baseline configurations. This evaluation was applied across all three algorithms to determine the most effective model for weather prediction using the Seattle dataset.

The experimental results demonstrate the following performance variations:

a. Random Forest: The algorithm experienced a notable predictive boost, with its accuracy increasing by 2.39%, rising from a baseline score of 81.57% to 83.96% post-tuning. This proves that hyperparameter tuning successfully optimizes the decision paths within the ensemble, shown in Figure 6.

```
y_pred = rf_model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f"Akurasi Model: {accuracy * 100:.2f}%")
```

```
Akurasi Model: 81.57%
```

```
# Prediksi pada data uji
y_pred = rf_best.predict(X_test)

# Hitung akurasi
accuracy = accuracy_score(y_test, y_pred)
print(f"Akurasi setelah tuning: {accuracy * 100:.2f}%")
```

```
Akurasi setelah tuning: 83.96%
```

Figure 6. Random Forest Accuracy Results

b. Decision Tree: This model showed a substantial improvement, where its accuracy jumped by 9%, surging from 74% in its baseline state to 83% after tuning. This significant gain highlights that tuning can provide an increase in the accuracy results of the Decision Tree algorithm, shown in Figure 7.

```
Akurasi: 0.744874715261959
Laporan Klasifikasi:
              precision    recall  f1-score   support

           0       0.05      0.07      0.06        14
           1       0.24      0.28      0.26        32
           2       0.90      0.90      0.90       192
           3       0.33      0.38      0.35         8
           4       0.78      0.73      0.76       193

    accuracy                           0.74       439
   macro avg       0.46      0.47      0.47       439
weighted avg       0.76      0.74      0.75       439
```

```
=== HASIL TUNING DECISION TREE ===
Parameter terbaik:
 {'criterion': 'entropy', 'max_depth': 5, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 10}
Akurasi rata-rata (cross-validation): 0.8542
Akurasi pada data uji: 0.8314

--- Laporan Klasifikasi ---
              precision    recall  f1-score   support

           0       0.25      0.07      0.11        14
           1       0.00      0.00      0.00        32
           2       0.96      0.91      0.93       192
           3       0.00      0.00      0.00         8
           4       0.76      0.98      0.86       193

    accuracy                           0.83       439
   macro avg       0.39      0.39      0.38       439
weighted avg       0.76      0.83      0.79       439
```

Figure 7. Decision Tree Accuracy Results

Joinforcs: Journal of informatics and information system
Volume 1, (1) 2026, p. 14-20
E-ISSN : XXXX-XXXX

c. X-Gradient Boost: The gradient boosting model also demonstrated enhanced performance, gaining a 2% increase in accuracy, shifting from 82% to a peak of 84% post-tuning. This underscores that tuning can provide an increase in the accuracy results of the X-Gradient Boost algorithm, shown in Figure 8.

```
# Evaluasi performa
print("Akurasi:", accuracy_score(y_test, y_pred))
print("Laporan Klasifikasi:\n", classification_report(y_test, y_pred))
```

```
Akurasi: 0.8246013667425968
Laporan Klasifikasi:
              precision    recall  f1-score   support

           0       0.25      0.07      0.11        14
           1       0.43      0.28      0.34        32
           2       0.94      0.91      0.93       192
           3       0.33      0.25      0.29         8
           4       0.79      0.91      0.84       193

    accuracy                           0.82       439
   macro avg       0.55      0.48      0.50       439
weighted avg       0.80      0.82      0.81       439
```

```
best_model = GradientBoostingClassifier(**random_search.best_params_)
best_model.fit(X_train, y_train)

# Evaluasi Model
y_pred = best_model.predict(X_test)
from sklearn.metrics import accuracy_score
print("Akurasi Model setelah Tuning:", accuracy_score(y_test, y_pred))
```

```
Akurasi Model setelah Tuning: 0.8428246013667426
```

Figure 8. X-Gradient Boost Accuracy Results

In summary, the comparative analysis validated that hyperparameter tuning provided performance improvements across all tested machine learning frameworks. Ultimately, X-Gradient Boost and Random Forest emerged as the best-performing models, producing optimal predictive capabilities for this data, as shown in the following figure 9.

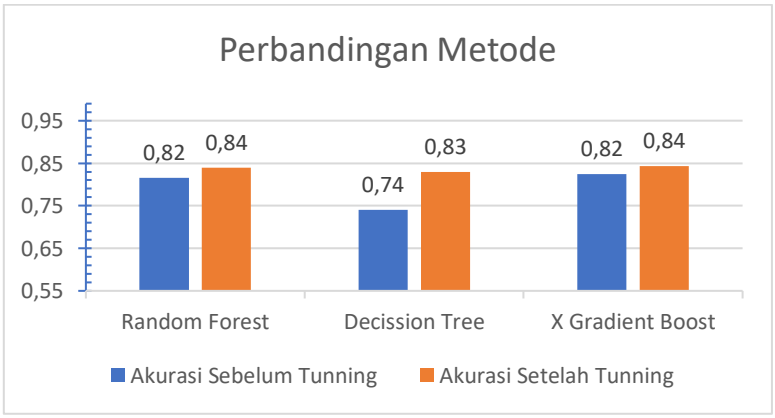

Figure 9. Accuracy Comparison Chart

4. Conclusion

Based on the experimental analysis of the three machine learning algorithms Decision Tree, Random Forest, and X-Gradient Boost can be concluded that all three methods are highly capable of classifying weather conditions for the Seattle dataset with a robust level of accuracy. Among the models evaluated, X-Gradient Boost and Random Forest achieved the highest overall accuracy of 84% and 83.96% respectively, followed by Decision Tree at 83% after optimization. The application of hyperparameter tuning via the

GridSearchCV method proved highly effective in elevating model performance; it expanded the accuracy of the Decision Tree by 9%, Random Forest by 2.39%, and X-Gradient Boost by 2%. For future research, the system can be expanded by integrating additional classification algorithms for a broader comparative scope, extending the historical observation periods, or utilizing deep learning architectures to achieve even higher forecasting precision.

REFERENCES

[1] M. Yulianto, D. Afriyantari, and P. Putri, "Pengembangan Game Edukasi Pengenalan Iklim Dan Cuaca Untuk Siswa Kelas III Sekolah Dasar," vol. 20, no. 02, pp. 143–148, 2020.

[2] A. M. Siregar, "Klasifikasi Untuk Prediksi Cuaca Menggunakan Esemble Learning," *Petir*, vol. 13, no. 2, pp. 138–147, 2020, doi: 10.33322/petir.v13i2.998.

[3] G. A. Rakhmat and W. Mutohar, "Prakiraan Hujan menggunakan Metode Random Forest dan Cross Validation," *MIND (Multimedia Artif. Intell. Netw. Database) J.*, vol. 8, no. 2, pp. 173–187, 2023, [Online]. Available: https://doi.org/10.26760/mindjournal.v8i2.173-187

[4] S. Joses, D. Yulvida, and S. Rochimah, "JOURNAL OF APPLIED COMPUTER SCIENCE AND TECHNOLOGY (JACOST) Pendekatan Metode Ensemble Learning untuk Prakiraan Cuaca menggunakan Soft Voting Classifier," vol. 5, no. 1, pp. 72–80, 2024.

[5] F. Hamami and I. A. Dahlan, "Klasifikasi Cuaca Provinsi Dki Jakarta Menggunakan Algoritma Random Forest Dengan Teknik Oversampling," *J. Teknoinfo*, vol. 16, no. 1, p. 87, 2022, doi: 10.33365/jti.v16i1.1533.

[6] S. Elina, H. Yulianti, O. Soesanto, and Y. Sukmawaty, "Penerapan Metode Extreme Gradient Boosting (XGBOOST) pada Klasifikasi Nasabah Kartu Kredit," vol. 4, no. 1, pp. 21–26, 2022.

[7] W. Apriliah *et al.*, "Prediksi Kemungkinan Diabetes pada Tahap Awal Menggunakan Algoritma Klasifikasi Random Forest," vol. 10, pp. 163–171, 2021.

[8] W. Nugraha, "Prediksi penyakit jantung cardiovascular menggunakan model algoritma klasifikasi," 2021.

[9] M. A. Hasanah, S. Soim, and A. S. Handayani, "Implementasi CRISP-DM Model Menggunakan Metode Decision Tree dengan Algoritma CART untuk Prediksi Curah Hujan Berpotensi Banjir," vol. 5, no. 2, 2021.

[10] P. A. Prayesy and I. Ruswita, "G-Tech : Jurnal Teknologi Terapan," vol. 7, no. 1, pp. 21–28, 2023.

[11] L. Mdegela, E. Municio, Y. De Bock, E. Luhanga, J. Leo, and E. Mannens, "for Kikuletwa River Floods," *Water (Switzerland)*, vol. 15, no. 6, pp. 1–14, 2023, [Online]. Available: https://www.mdpi.com/2073-4441/15/6/1021?utm_campaign=releaseissue_waterutm_medium=emailutm_source=releaseissueutm_term=titlelink24

[12] F. W. Mugware, C. Sigauke, and T. Ravele, "Evaluating Wind Speed Forecasting Models: A Comparative Study of CNN, DAN2, Random Forest and XGBOOST in Diverse South African Weather Conditions," *Forecasting*, vol. 6, no. 3, pp. 672–699, 2024, doi: 10.3390/forecast6030035.

[13] M. Putra, M. S. Rosid, and D. Handoko, "High-Resolution Rainfall Estimation Using Ensemble Learning Techniques and Multisensor Data Integration," *Sensors*, vol. 24, no. 15, 2024, doi: 10.3390/s24155030.

[14] K. Shin, K. Kim, J. J. Song, and G. W. Lee, "Classification of Precipitation Types Based on Machine Learning Using Dual-Polarization Radar Measurements and Thermodynamic Fields," *Remote Sens.*, vol. 14, no. 15, 2022, doi: 10.3390/rs14153820.

[15] Z. A. Dwiyanti and C. Prianto, "Prediksi Cuaca Kota Jakarta Menggunakan Metode Random Forest," *J. Tekno Insentif*, vol. 17, no. 2, pp. 127–137, 2023, doi: 10.36787/jti.v17i2.1136.