wjert, 2024, Vol. 10 Issue 10, 61-108. Original Article ISSN 2454-695X

World Journal of Engineering Research and Technology
WJERT
www.wjert.org
SJIF Impact Factor: 7.029

PREDICTION OF DAILY SOLAR RADIATION IN SOUTHEASTERN NIGERIA USING EXTREME GRADIENT BOOSTING MODEL

[1]*Isaac Raphael Okafor, [2]Egarievwe Stephen, [3]Guangming Chen, [4]Garfield Jones, [5]Osuagwu Onyema and [6]Agbalagba Ezekiel

[1,2,3,4]Department of Industrial and Systems Engineering, Morgan State University, 1700 East Cold Spring Lane, Baltimore, Maryland, USA.

[5]Department of Electrical and Computer Engineering, Morgan State University, 1700 East Cold Spring Lane, Baltimore, Maryland, USA.

[6]Federal University of Petroleum Resources, Effurun, Delta State, Nigeria.

Article Received on 05/08/2024 **Article Revised on 25/08/2024** **Article Accepted on 15/09/2024**

***Corresponding Author**
Isaac Raphael Okafor
Department of Industrial and Systems Engineering, Morgan State University, 1700 East Cold Spring Lane, Baltimore, Maryland, USA.

ABSTRACT

The prediction of solar radiation (SR) is important for effective and reliable design, operation and optimization of solar energy systems, as well as for agricultural and environmental applications. Conversely, in the developing countries like Nigeria-southeast geopolitical zone, long-standing records of global solar radiation data are not available in many places because of the cost and maintenance of the measuring instruments. As a result, the current interest in the utilization of solar energy potential have made researchers to continue exploring alternative methods such as empirical models to predict SR using meteorological and geographic parameters. Due to the lack of daily solar radiation data, researchers like Dada and Okogbu have developed methods to estimate daily values from monthly averages using Fourier series, which is essential for accurate solar energy assessments (Dada & Ec 2016). In this study, two (2) machine learning models such as the gradient boosting (GBoost) and Extreme gradient boosting (XGB) regressors are investigated for real time ground-based measured solar radiation prediction in **s**outheastern Nigeria. Daily solar radiation parameters data such as temperature, relative humidity, precipitation, surface pressure, wind speed and wind direction were collected from the

National Aeronautics and Space Administration (NASA) Prediction of Worldwide Energy Resource (POWER) for 10years ranging from 2012 to 2021 and used as training/testing data while developing the model. The models performance was evaluated and compared based on the statistical evaluators such as root mean squared error (RMSE), normalized root mean squared error (nRMSE), and coefficient of determination (R^2). The two models were found significant in the prediction of solar radiation for the region based on the statistical indicators. Out of the two significant models, XGB regressor model emerged as the best /or most efficient model for the region with maximum coefficient of determination (R^2) and minimum RMSE and NRMSE respectively. The results showed that Ebonyi state has XGB, R^2 of 97.2%; GB, R^2 of 90.2% Anambra state has XGB R^2 of 95.9%; GB, R^2 of 84.1%; Imo state has XGB, R^2 of 95.2%; GB, R^2 of 83.1%; Abia state has XGB, R^2 of 94.8%, GB, R^2 of 82.4% and Enugu state has XGB, R^2 of 93.9%; GB, R^2 of 83.2%. The average nRMSE of % of Performance of XGB over GB in the five Southeastern States was found to be 95.8% and the average RMSE of % of Performance of XGB over GB was found to be 88.49%. The result is an indication that XGB model is of best fit for the prediction of solar radiation in the Southeastern Nigeria.

KEYWORDS: Prediction of Solar Radiation, Extreme Gradient Boosting Model, Statistical Evaluators, Southeastern Nigeria, NASA.

Nomenclature

AI = Artificial Intelligence

ANFIS = Adaptive Neuro-fuzzy Inference System

PSO-ANFIS = Particle Swarm Optimization- Adaptive Neuro-fuzzy Inference System

ANN = Artificial Neural Network

DNI = Direct Normal Irradiation

DSR = Daily (Diurnal) Solar Radiation

ET = Extratrees Regressor

Extratrees = Extreme Randomized Trees

GB = Gradient Boosting

GR = Gaussian Process Regressor

GHI = Global Horizontal Irradiation

GWO = Grey Wolf Optimization

KNN = K- Nearest Neighbors

LSTM = Long- Short Term Memory

MSE = Mean Squared Error

MFO = Moth Flame Optimization

ML = Machine Learning

MLP = Multilayer Perceptron

NASA = National Aeronautics and Space Administration

nRMSE = Normalized Root Mean Squared Error

PPT Precipitation

POWER = Prediction of Worldwide Energy Resource

PS = Surface Pressure

PSO = Particle Swarm Optimization

PV = Photovoltaic

RF = Random Forest

RH = Relative Humidity

RMSE = Root Mean Squared Error

R^2 = Coefficient of Determination

SR = Solar Radiation

SVM = Support Vector Machine

SVR = Support Vector Regressor

T = Temperature

WS = Wind Speed

W/m^2 = Watt per square meter

WT = Wavelet Transform

WT-ANFIS = Wavelet Transform- Adaptive Neuro-fuzzy Inference System

XGB = Extreme Gradient Boosting

XGB-MFO = Extreme Gradient Boosting- Moth Flame Optimization

1.0 INTRODUCTION

The idea of this study came into being as a result of increasing energy dependence and global rising energy costs, limited sources of the fossil fuels and its negative environmental impacts. The depletion of fossil energy sources, photochemical pollution, acid rain, emission of CO_2 and other hazardous greenhouse gases are some of the disadvantages of non-renewable energy (Salisu et al. 2020). These negative environmental impacts have given renewable energy opportunity to thrive and it is the fastest growing sector worldwide for energy generation. The utilization of renewable energy sources markedly diminishes the emission of greenhouse

gases, thereby playing a pivotal role in the alleviation of climate change and the safeguarding of environmental integrity (Constante & Erazo 2024) (Bashiru et al. 2024). It also serves to diversify the energy portfolio, thereby augmenting energy security and mitigating reliance on depleting fossil fuel resources (Constante & Erazo 2024) (Ryzhenkov & Burinova 2022). The integration of renewable energy sources such as solar, wind, and geothermal is associated with considerable economic advantages, including the generation of employment opportunities and the alleviation of poverty (Constante & Erazo 2024) (Ryzhenkov & Burinova 2022). It is evident that out of these renewable energy sources harnessed for electricity production, the most widely used source of energy is solar energy (Karthikeyan et al. 2014, Benmouiza, K & Cheknane, A.2013 in Salisu et al. 2019). Renewable energy resources have gained significant importance in the 21st century as researchers are striving hard to make a pollution free environment and sustainable energy provision (Inayat & Raza, 2019 in Salisu et al. 2020). The generation of energy from renewable energy sources is a desirable measure to save the earth from environmental degradation that results from the use of fossil fuel. Solar energy is a clean and environmentally friendly source of energy that is inexhaustible in nature. With the current utilization of solar energy across the globe, there still exists the problem of solar radiation data availability that serves as a barrier to an effective solar power project in places where these data are not available (Liu et al. 2012 in Salisu et al. 2019). The solar radiation data are however not easily available for many countries (Chineke, 2002; Samani and Pessarkli, 1986; Chineke, 2008; Chiemeka and Chineke, 2009 in Nwokocha et al., 2009). This data is a basic requirement to utilizing solar energy economically. Such data are required in many applications ranging from crop growth models, evapotranspiration estimates to the design of photovoltaic system, solar collector system, and solar crop drying systems and building sciences (Chineke et al., 1999; Chineke et al., 2007; Okoro et al., 2008; Samani and Pessarkli, 1986 in Nwokocha et al., 2009).

According to Yadav & Chandel (2013), solar radiation is an important pa-rameter for solar energy research but is not available for most of the sites due to non-availability of solar radiation measuring equipment at the meteorological stations. Therefore, it is essential to predict solar radiation for a location using several climatic variables. These variables are sunshine duration, maximum ambient temperature, relative humidity, latitude, longitude, day of the year, daily clear sky global radiation, total cloud cover, temperature, clearness index, altitude, months, average temperature, average cloudiness, average wind velocity, atmospheric pressure, reference clearness index, mean diffuse radiation, mean beam radiation month, extra-

terrestrial radiation, evaporation and soil temperature (Yadav & Chandel, 2013).

According to Salisu, et al. (2019), countries in Sub-Saharan Africa including Nigeria do not have the state of the art equipment for efficient meteorological data measurement such as solar radiation due to the high cost of equipment and its maintenance cost. Without adequate knowledge of solar radiation data, it will be uneasy to have an efficient solar power system design. Therefore, for an effective solar power design, there is a need for accurate solar radiation data prediction (Salisu. et al., 2019). For this reason, enormous models have been developed to predict the solar radiation data in different parts of the world where such data is unavailable. Modeling is an economical and essential tool for the estimation of solar radiation, and the accuracy of such models depend on the quality of the data used. According to Akpabio et al., 2005; Ibrahim, 1985 in Nwokocha, et al., 2009, modeling is a better tool for the estimation of solar radiation at places where measurements are not available but have similar climatic conditions to measured locations, even though may be less accurate.

Empirical models, as well as artificial intelligence models, have been developed for this purpose. These models were developed using the available meteorological data available in those areas and have a correlation with solar radiation e.g. sunshine hours, minimum temperature, maximum temperature, average temperature, cloud cover, longitude, latitude (Maghrabi, 2009, Garg, H. and Garg, S. 1983 in Salisu, et al. 2019).

Empirical models have been employed by many scholars (Ajayi, et al. 2014; Besharat, et al. 2013; Falayi,.et al. 2008; Fagbenle, 1993 & Sambo, 1986) for solar radiation forecasting by utilizing various accessible meteorological data especially sunshine hours. For better accuracy, artificial intelligence (AI) method has been recently adopted and utilized by many scholars for solar radiation prediction (Chiteka, and Enweremadu, 2016; Hussain, and Al Alili, 2015; Landeras, et al. 2012; Olatomiwa, et al. 2015; Salisu, et al. 2017; Shamshirband, et al. 2016).

Researchers like Hassan, et al (2017 in Gurel et al, 2023) developed four different machine learning algorithms in modelling daily global solar radiation, which he finds out that ANFIS and MLP models gave similar results, but SVM comes to the fore according to these two models. Karasu, et al (2027 in Gurel et al, 2023) also developed the prediction of solar radiation based on machine learning methods. He used linear regression (LR) and Gaussian process regression (GR) models to predict daily global solar radiation and finds that GR

exhibited more successful performance in estimating the solar radiation data than the LR model. Fan, et al (2018 in Gurel et al, 2023) examined the comparison of support vector machine (SVM) and Extreme Gradient Boosting (XGBoost) for predicting daily global solar radiation and the statistical metric results demonstrate that the XGBoost model is better at estimating the daily global solar radiation than the SVM algorithm. Benali, et al (2019 in Gurel et al, 2023) forecasted the three components of solar irradiation (global horizontal, beam normal and diffuse horizontal) using artificial neural network (ANN) and random forest (RF) methods and he finds that RF had better forecasting results of solar irradiation as compared to SP, and ANN. Additionally, all models presented worse results in spring and autumn owing to the less reliable of data and high meteorological variability in these seasons.

In this present study, the efficiency of gradient boosting (GBoost) and XGBoost regressors were examined for solar radiation prediction in southeastern Nigeria. The models were developed independently to predict the ground measured solar radiation. Anshul Saini (2024) mentioned that gradient boosting is a method standing out for its prediction of speed and accuracy, particularly with large and complex datasets. Gradient boosting is excellent for various machine learning tasks, including regression and classification. It provides high predictive accuracy by combining weak learners in an ensemble, such as decision trees, to create a strong predictor. Gradient boosting handles complex relationships in data, works well with both numerical and categorical features, and is particularly useful for dealing with large-scale structured and unstructured data in real-world applications.

The optimization of hyperparameters profoundly augments the efficacy of the XGBoost algorithm. Empirical investigations have indicated that methodologies such as moth flame optimization (MFO) and grey wolf optimization (GWO) significantly improve the model's precision, resulting in elevated R^2 coefficients and diminished RMSE metrics for diverse categories of solar irradiance (Namrata et al., 2023).

Integrated models that amalgamate XGBoost with alternative computational approaches demonstrated enhanced predictive capabilities in a study conducted in Sub-Saharan Africa, increasing accuracy by as much as 44.5% in comparison to pre-existing methodologies (Goliatt & Yaseen, 2022). Mbah et al (2021) successfully applied XGBoost to forecast daily global solar radiation on tilted surfaces, demonstrating improved prediction accuracy with R^2 values as high as 0.9977 during training phases. XGBoost consistently outperformed traditional sky models in terms of accuracy, further validating the model's ability to estimate solar

radiation (Mbah et al., 2021).

Namrata et al (2023) investigated that the advanced machine learning system referred to as XGB-MFO, which has been finely tuned for hyperparameters in Extreme Gradient Boosting, outshined competing models during the study, hitting impressive R^2 scores of 0.9337 for Global Horizontal Irradiance (GHI), 0.9011 for Diffuse Horizontal Irradiance (DHI), and 0.8744 for Direct Normal Irradiance (DNI). Also, the XGB-MFO model reflected the least Root Mean Square Error (RMSE) metrics of 76.29 Wm^{-2} for GHI, 41.90 Wm^{-2} for DHI, and 95.94 Wm^{-2} for DNI, indicating its exceptional proficiency in solar radiation intensity prediction.

In this research, the Long Short-Term Memory (LSTM) based deep learning model achieved a determination coefficient (R^2) of 0.79 during training and 0.78 during testing. The datasets used for training and testing yielded RMSE figures of 0.46 and 0.47, showcasing the capability of the LSTM deep learning model in predicting daily solar radiation (2023).

According to Sruthi (2024), random forest is a widely-used machine learning algorithm developed by Leo Breiman and Adele Cutler, which combines the output of multiple decision trees to reach a single result. Its ease of use and flexibility have fueled its adoption, as it handles both classification and regression problems. Random forest is a machine learning algorithm that creates an ensemble of multiple decision trees to reach a singular, more accurate prediction or result (Niklas, 2024).

The aim of the study is to accurately determine the best model in evaluating daily solar radiation in **southeastern Nigeria** by comparing Gradient Boosting (GBoost) model, and XGB regressor models. Daily solar sunshine data for 10 years ranging from 2012-2021 collected from the National Aeronautics and Space Administration (NASA was used as training/testing data while developing the model. The objective of this study is to assess the efficiency of the models by comparing model results based on statistical parameters such as minimum root mean square error (RMSE) and normalized root mean square error (nRMSE) and maximum coefficient of determination (R^2) in evaluating daily solar radiation (DSR) in **southeastern Nigeria.**

Many studies have been carried out for monthly and annual solar radiation prediction using Artificial Neural Network (ANN) model but no studies have been carried out on daily

prediction of solar radiation in **southeastern Nigeria** using gradient boosting (GBoost) model, and XGB regressor model. There exists one research on daily prediction of solar radiation using artificial neural network in North West Nigeria, but no research has been carried out in the southern Nigeria especially southeastern region. The prediction is done using the meteorological data available such as air temperature, relative humidity, precipitation, surface pressure, wind speed, and wind direction. These are considered as inputs to the developed models.

2.0 MATERIALS AND METHODS

2.1 Study Area

The Southeastern Nigeria is one of the six geopolitical zones of Nigeria representing both a geographic and political region of the country's. It comprises five states – Abia, Anambra, Ebonyi, Enugu, and Imo. It is commonly known as Igboland. Although the Southeastern Nigeria is the smallest geopolitical zone, it contributes greatly to the Nigerian economy due to oil and natural gas reserves along with a growing industrialized economy. The population of Southeastern Nigeria is about 22 million people. Southeastern Nigeria's demographic composition stands out for its marked density, particularly in the Igbo territories of Imo and Anambra, where population pressures significantly impact land utilization and socioeconomic conditions.

Southeastern Nigeria, notably the Igbo regions of Imo and Anambra, demonstrates some of the most elevated population densities within the nation, with Imo state recording 644.3 individuals per square mile and Anambra state 477.8 individuals per square mile (Ao 1982). It has large number of small and medium scale enterprises. As a result of this, the region needs better and regular electricity supply in order to sustain its businesses.

A study using Long Short-Term Memory (LSTM) models analyzed solar energy potential in five southeastern cities: Abakaliki, Awka, Enugu, Owerri, and Umuahia and Enugu was identified as having the highest solar energy potential, averaging 6.25 kWh/day, indicating a strong case for photovoltaic (PV) power systems in the region (Ikemba et al., 2024).

According to Ogbodo & Okeke (2022) southeastern Nigeria has a total landmass of about 28,987 km^2. Forest reserves cover approximately 1,335.42 km^2, or 5% of the total landmass, within this area. This is significantly below the Food and Agriculture Organization's recommended 25% forest cover threshold, indicating a need for increased conservation efforts.

Over the years, southeastern Nigeria has experienced significant urban growth, with states like Anambra and Enugu showing substantial increases in built-up areas. For instance, Anambra's built-up area expanded from 377.78 km^2 in 1986 to 1,795 km^2 in 2018, while Enugu's grew from 1,866.31 km^2 to 3,353.49 km^2 in the same period. This urban expansion is characterized by infilling, edge expansion, and outlying growth, necessitating effective urban growth management and land-use planning (Sampson et al., 2020). Land size of the southeastern Nigeria are shown in Table 1.

Map of the states in the Southeastern Nigeria is shown in figure 1.

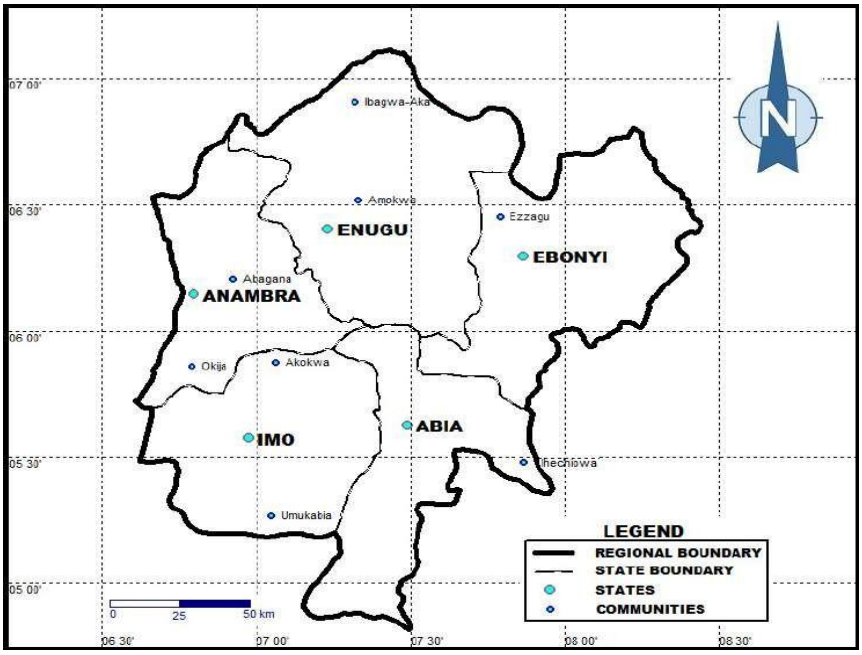


Figure 1: Map of the States in the Southeast Nigeria.

Source: Adapted from Map of Southeast Nigeria.

Table 1: Land Areas of the Southeast States.

States	Capital	Area in square kilometres
Abia	Umuahia	6,320
Anambra	Awka	4,844
Ebonyi	Abakaliki	5,935
Enugu	Enugu	7,161
Imo	Owerri	5,530

Source. Adapted from Legit. Nigeria (South East states in Nigeria: List, map, languages spoken and more - Legit.ng)

In this study, the solar radiation potentials for the states in the southeastern Nigeria are Abia state (2.30-16.60) W/m^2, Anambra state (3.0-16.90) W/m^2, Ebonyi state (2.92-17.70) W/m^2,

Enugu state (2.35-17.38) W/m^2 and Imo state (2.30-16.60) W/m$^{2.}$ From these solar radiation potentials. It is observed that Abia state and Imo state have the same solar potential. Between the two machine learning models developed for the prediction of solar radiation in the southeastern Nigeria, XGB regressor model emerged as a better (more significant) model with the statistical performance indicators as Abia state (R^2 = 0.948, RMSE = 0.2817. nRMSE = 0.0427), Anambra state (R^2 = 0.959, RMSE = 0.2848. nRMSE = 0.0359), Ebonyi state (R^2 = 0.972, RMSE = 0.3802. nRMSE = 0.0306), Enugu state (R^2 = 0.939, RMSE = 0.2829. nRMSE = 0.0370), Imo state (R^2 = 0.952, RMSE = 0.2799. nRMSE = 0.0388).

2.2 Data Collection and Analysis

We collected meteorological data for a 10-year period from 2012 to 2021 from the National Aeronautics and Space Administration (NASA) to carry out this study. We used daily solar radiation data as the output. We used other meteorological data as the inputs, which included the daily air temperature (T), relative humidity (RH), precipitation (PPT), surface pressure (PS), wind speed (WS), and wind direction (WD). The data obtained were trained and tested using K-fold cross validation to avoid over fitting, under fitting and bias rather than dividing the data into two: (i) training data and (ii) testing data, as for instance, 2012-2019 (70%) data training and 2020-2022 (30%) data testing. Figure 2 shows various input parameters used in the prediction of daily solar radiation in the Southeastern Nigeria.

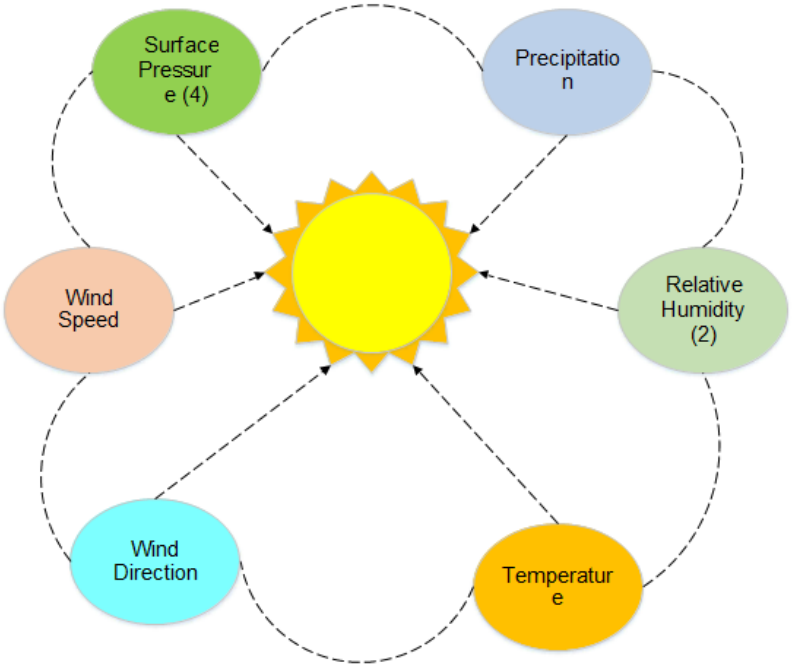


Figure 2: Various input parameters used for the prediction of daily solar radiation data in Southern Nigeria.

2.3 Machine Learning Models

A machine learning model is defined as a mathematical model with several parameters that need to be learned from the data. By training a model with existing data, we can fit the model parameters. When we talk about the machine learning model, we talk about how well it performs and its accuracy which is known as prediction errors. A model is said to be a good machine learning model if it generalizes any new input data from the problem domain in a proper way. This will help to make predictions about future data that the data model has never seen.

2.4 Regression in Machine Learning

Regression is a statistical approach used to analyze the relationship between a dependent variable (target variable) and one or more independent variables (predictor variables). It is also a supervised machine learning technique, used to predict the value of the dependent variable for new, unseen data. It models the relationship between the input features and the target variable, allowing for the estimation or prediction of numerical values.

It breaks the relationship between dependent and independent variables, enabling predictions through various regression models. It is targeted towards determining the most suitable function that characterizes the connection between these variables. It tries to find the best-fitting model, which can be utilized to make predictions or draw conclusions.

2.4.1 Types of Regression

There are three main types of regression:

1. Simple Regression

Simple linear regression is used when there is only a single independent variable. It is also used to predict a continuous dependent variable based on a single independent variable.

2. Multiple Regression

Multiple linear regression is used when there are multiple independent variables. It is also used to predict a continuous dependent variable based on multiple independent variables.

3. Nonlinear Regression

It shows that the relationship between the dependent variable and independent variable(s) follows a nonlinear pattern and also provides flexibility in modeling a wide range of functional forms.

2.4.2 Types of Regression Algorithms

There are different types of regression algorithms. Among which are

2.4.2.1 Gradient Boosting and Gradient Boosting Decision Trees

Gradient Boosting is a machine learning algorithm, used for both classification and regression problems. It works on the principle that many weak learners (e.g., shallow trees) can together make a more accurate predictor. Gradient boosting works by building simpler (weak) prediction models sequentially where each model tries to predict the error left over by the previous model (Shruti, 2023). A weak learning model is a model that exhibits either high bias or high variance. It does not perform well on testing sets due to the bias and variance, thus making it not suitable as standalone predictor. Bias refers to the error introduced by approximating a real-world problem with a simplified model. Bias indicates poor performance of the data by the models. For example, high bias implies that the model oversimplifies the underlying complexity, leading to systematic errors. On the other hand, variance signifies the model's sensitivity to fluctuations in the training data. For example, high variance results in overfitting, where the model fits the training data too closely but fails to generalize well to unseen data. Overfitting implies that data training is good and data testing is bad while underfitting signifies that both data training and testing are bad. Overfitting and underfitting are majorly responsible for the poor performances of the machine learning algorithms.

It is important to also understand ensemble learning; ensemble learning combines multiple models like the weak learners to achieve better results. It can be hypothesized that when weak models are correctly combined, we can obtain more accurate and robust models.

In general, ensemble learning is a model that makes predictions based on a number of different models. By combining a number of different models, an ensemble learning tends to be more flexible (less bias) and less data sensitive (less variance). The two most popular ensemble learning methods are bagging and boosting.

Bagging and boosting are ensemble methods used to enhance model performance. It is important to note that weak learners might not performance individually, but their collective strength lies in their unity within the ensemble methods. Generally speaking, ensemble learning is a model that makes predictions based on a number of different models, for instance, by combining a number of different models, an ensemble learning tends to be more flexible (less bias) and less data sensitive (less variance). The two most popular ensemble learning methods are bagging and boosting.

Bagging, also known as Bootstrap, is an ensemble learning method that involves training a bunch of models in parallel way. Each model learns from a random subset of the data, where the dataset is same size as original but is randomly sampled with replacement (bootstrapped). Boosting is an ensemble learning method that involves training a bunch of models sequentially. Each model learns from the mistakes of the previous model. That is, the subsequent models try to explain and predict the error left over by the previous model.

Bagging involves training a bunch of models in parallel way. Each model learns from a random subset of the data, where the dataset is same size as original but is randomly sampled with replacement (bootstrapped). The Process of Bagging algorithm is shown in figure 3.

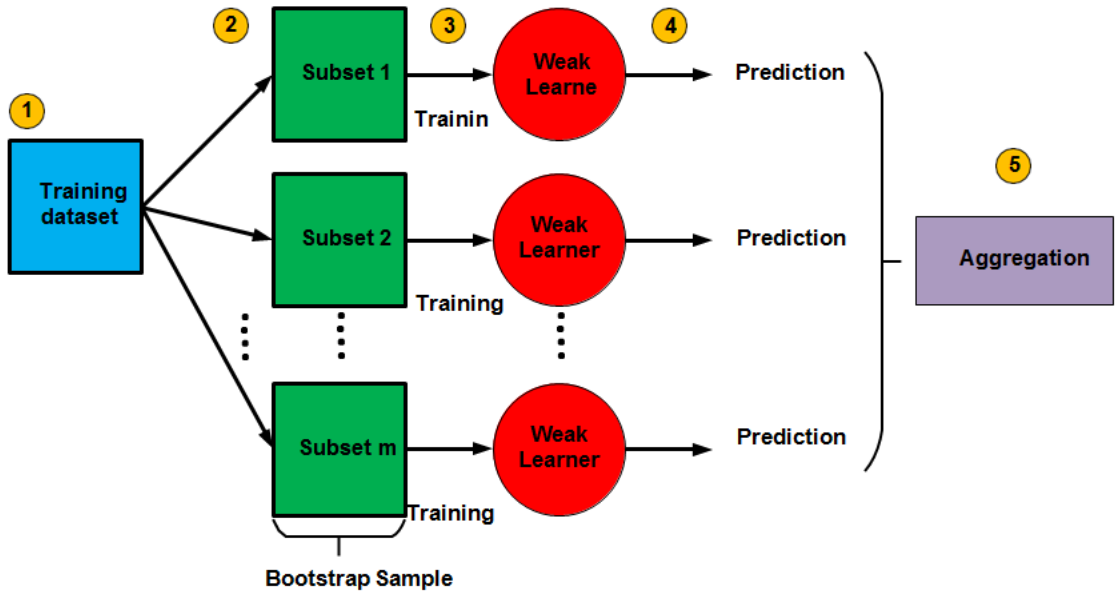


Figure 3: The Process of Bagging Algorithm.

The steps of bagging are as follows

1. We have an initial training dataset containing n-number of instances.
2. We create a m-number of subsets of data from the training set. We take a subset of N sample points from the initial dataset for each subset. Each subset is taken with replacement. This means that a specific data point can be sampled more than once.
3. For each subset of data, we train the corresponding weak learners independently. These models are homogeneous, meaning that they are of the same type.
4. Each model makes a prediction.
5. The predictions are aggregated into a single prediction. For this, either max voting or averaging is used (Mbali Kalirane, 2024).

The application of bagging is found in random forests. Random forests are a parallel combination of decision trees. Each tree is trained on random subset of the same data and the results from all trees are averaged to find the classification.

Bagging is designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression. It decreases the variance and helps to avoid overfitting. It is usually applied to decision tree methods. Bagging is a special case of the model averaging approach. Bagging algorithm or in shorthand form Bootstrap aggregate has B as individual decision trees which can now be aggregated using the formula below:

$$fbag(x) = \frac{1}{B}\sum_{b=1}^{B} fbag(x) \text{----------} (1)$$

Any new prediction will now be an average of all the predictions from the B small decision trees. Bagging helps to reduce variance of the overall cluster model and provides accuracy improvements over the individual models.

Boosting is a potent technique that has gained traction due to its capacity to improve model performance. Boosting is a well-known ensemble learning strategy that combines the predictions of numerous base models to produce a more robust overall model. Boosting is a supervised machine learning strategy that combines the predictions of multiple weak models (base models) to generate a powerful ensemble model. It is a machine learning strategy that combines numerous weak learners into strong learners to increase model accuracy. Boosting, as opposed to classic ensemble approaches like bagging or averaging, focuses on successively training the basic models in a way that emphasizes misclassified samples from prior iterations. The goal of boosting is to prioritize samples that are incorrectly categorized in previous iterations, allowing the model to learn from its mistakes and improve its performance iteratively. Boosting involves training a sequence of models, where each subsequent model focuses on the errors made by the previous model. The predictions are combined using a weighted voting scheme. Boosting works on the principle of improving mistakes of the previous learner through the next learner. In boosting, weak learners (example: decision trees with only the stump) are used which perform only slightly better than a random chance. Boosting focuses on sequentially adding up these weak learners and filtering out the observations that a learner gets correct at every step. The Process of Boosting is shown in figure 4.

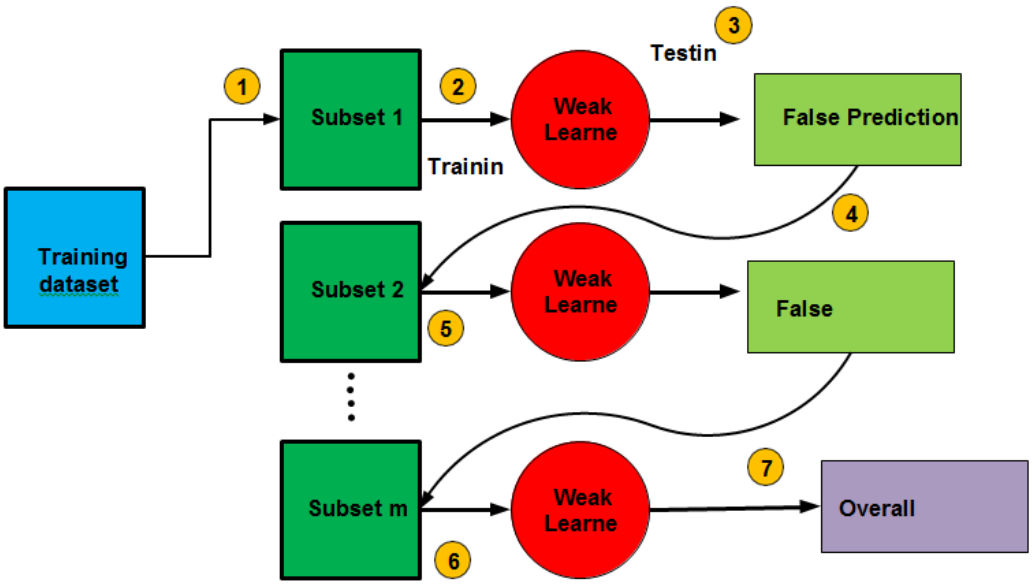


Figure 4: The Process of Boosting.

From figure 4, Boosting works with the following steps:

1. We sample m-number of subsets from an initial training dataset.
2. Using the first subset, we train the first weak learner.
3. We test the trained weak learner using the training data. As a result of the testing, some data points will be incorrectly predicted.
4. Each data point with the wrong prediction is sent into the second subset of data, and this subset is updated.
5. Using this updated subset, we train and test the second weak learner.
6. We continue with the following subset until the total number of subsets is reached.
7. We now have the total prediction. The overall prediction has already been aggregated at each step, so there is no need to calculate it (Mbali Kalirane, 2024).

Advantages of Boosting

The following are some of the benefits of boosting in machine learning:

1. Reduced Overfitting: Boosting algorithms, such as AdaBoost, are designed to reduce overfitting by adjusting weights dynamically based on the errors of prior models, which helps to generalize better on unseen data.

2. Improved Performance: Because boosting combines the predictions of any base models, it effectively reduces bias and variance, resulting in more accurate and robust predictions.

3. Ability to Handle Complex Data: Boosting can handle complicated data patterns, including non-linear correlations and interactions, making it appropriate for a wide range of machine learning applications such as classification, regression, and ranking.

4. **Robustness to Noise:** When compared to other machine learning techniques, boosting is less vulnerable to noise in training data since it focuses on misclassified samples and gives greater weights to them, effectively reducing the impact of noisy samples on final predictions.
5. **Flexibility**: Boosting algorithms are versatile and can be employed with a variety of base models and loss functions, allowing for customization and adaptation to various problem domains.
6. **Interpretability:** While boosting models are frequently referred to as **"black-box"** models, they can nevertheless provide some interpretability through feature importance rankings, which can aid in understanding the relative value of various features in the prediction process (Brijesh Soni, 2023).

The application of boosting is found in Gradient Boosting Decision Trees. The boosting algorithm is described in the equation below

$$f(x) = \sum_{b=1}^{B} \alpha f b(x) \quad (2)$$

Where ensemble function was initially set to 0

$$f(x) = 0 \quad (3)$$

$$ri = yi \quad (4)$$

The initial ensemble method is set to zero (0) and the residuals are just the true class labels of the data points.

Each of the model in the summation tries to improve on the performance of the previous model by trying to classify the points that the previous one was not able to.

α represents the shrinkage parameter and each function inside the summation is a weak learner and we train at most B models and sum them together along with their alphas to get the final ensemble model.

$$rnew = rold - \alpha f b(xi) \quad (5)$$

Where r represents the residual.

One of the first boosting algorithms developed is Adaboost. AdaBoost is one of the most extensively used boosting algorithms. It gives weights to each data point in the training set based on the accuracy of prior models, and then trains a new model using the updated weights. AdaBoost is very useful for classification tasks.

Gradient boosting improvised upon some of the features of Adaboost to create a stronger and more efficient algorithm. Adaboost uses decision stumps as weak learners. Decision stumps are decision trees with only one single split. It also attached weights to observations, adding more weight to difficult-to-classify observations and less weight to those that are easy to classify.

In gradient boosting decision trees, many weak learners are combined to come up with one strong learner. The weak learners are the individual decision trees. All the trees are connected in series and each tree tries to minimize the error of the previous tree. Due to this sequential connection, boosting algorithms are usually slow to learn (controllable by the developer using the learning rate parameter), but also highly accurate.

Generally speaking, in statistical learning, models that learn slowly perform better. The weak learners are fit in such a way that each new learner fits into the residuals of the previous step so as the model improves. The final model adds up the result of each step and thus a stronger learner is eventually achieved. Gradient boosting decision tree is shown in figure 5.

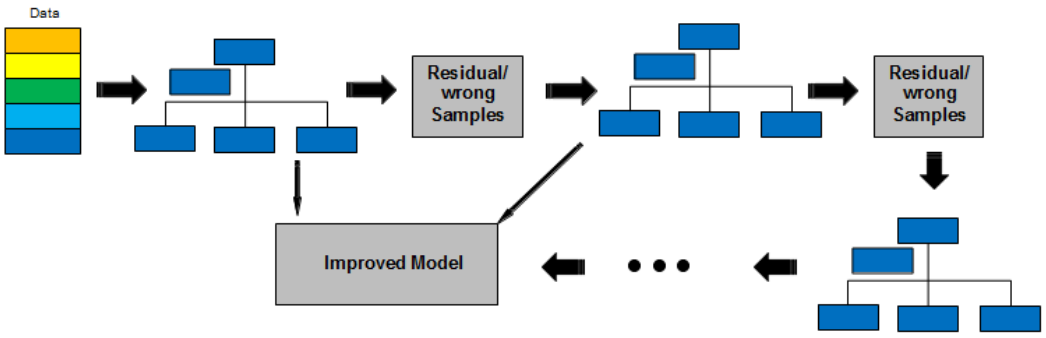


Figure 5: Gradient Boosting Decision Trees.

A loss function is used to detect the residuals. For instance, mean squared error (MSE) can be used for a regression task and logarithmic loss (log loss) can be used for classification tasks. It is worth noting that existing trees in the model do not change when a new tree is added. The added decision tree fits the residuals from the current model.

A decision tree regression is a type of regression algorithm that builds a decision tree to predict the target value. A decision tree is a tree-like structure that consists of nodes and branches. Each node represents a decision, and each branch represents the outcome of that decision. The goal of decision tree regression is to build a tree that can accurately predict the target value for

new data points.

The difference between the prediction and the actual value is known as the residual (or in this case, pseudo residuals)

The Algorithm

Let's assume that the output model y when fit to only 1 decision tree, is given by:

$$y = A_1 + (B_1 * X) + e_1 \quad \text{----------(6)}$$

Where, e1 is the error residual from this decision tree.

In gradient boosting, we fit the consecutive decision trees on the residual from the last one. Therefore, when gradient boosting is applied to this model, the consecutive decision trees will be mathematically represented as

$$e_1 = A_2 + (B_2 * X) + e_2 \quad \text{----------(7)}$$

and

$$e_2 = A_3 + (B_3 * X) + e_3 \quad \text{----------(8)}$$

We stopped at 3 decision trees, but in an actual gradient boosting model, the number of learners or decision trees is much more. Combining the three equations, the final model of the decision tree will be given by

$$y = A_1 + A_2 + A_2 + (B_1 * x) + (B_2 * x) + (B_3 * x) + e_3 \quad \text{----------(9)}$$

In adaptive boosting (often called adaboost), we try to define an ensemble model as a weighted sum of L weak learners

$$sL(.) = \sum_{l=1}^{L} cl \times wl(.) \quad \text{----------(10)}$$

Where cl's are coefficients and wl's are weak learners.

In gradient boosting, the ensemble model we try to build is also a weighted sum of weak learners.

Other Equations and Formulae Associated with Gradient Boosting

The Gradient Boosting model builds an ensemble of trees sequentially, with each tree correcting the errors of its predecessor.

These are some of the key equations and formulae associated with the Gradient Boosting

model.

Initialize the Model

Initialize the model with a constant value, typically the mean of the target values y.

For Regression

$$F_0(x) = \frac{1}{N}\sum_{i=1}^{N} yi \quad \text{----------(11)}$$

Compute the Pseudo-Residuals

Calculate the pseudo-residuals, which are the negative gradients of the loss function with respect to the model's predictions.

For Regression

Mean Squared Error (MSE) loss

$$\tau_{im} = \frac{\partial L(\hat{yi}, F(xi))}{\partial F(xi)} = yi - F_{m-1}(xi) \quad \text{----------(12)}$$

Fit a Base Learner

Fit a Base Learner (e.g., a decision tree) to the pseudo- residuals

$$h_m(x) = \text{argmin}_h \sum_{i=1}^{N} (\tau im - h(xi))^2 \quad \text{----------(13)}$$

Update the Model

Update the Model by adding the fitted base learner, scaled by a learning rate ν For Regression

$$F_m(x) = F_{m-1}(x) + \nu h_m(x) \quad \text{----------(14)}$$

Where M is the total number of trees in the ensemble.

Final Model

The final model is the sum of the initial model and the scale base learners

$$F_M(x) = F_0(x) + \sum_{m=1}^{M} V h_m(x) \quad \text{----------(15)}$$

Prediction

Make Prediction using the final model. For Regression

$$\hat{y} = F_M(x) \quad \text{----------(16)}$$

These equations and steps describe the core mechanics of the Gradient Boosting algorithm, highlighting how it sequentially builds an ensemble of models to improve prediction accuracy.

Table 2: Differences Between Bagging and Boosting.

Features	Bagging	Boosting
Goal	• Reduces variance and prevent overfitting by combining multiple models trained on different subsets of the training data.	• Reduces both bias and variance by sequentially combining multiple weak learners to create a strong learner.
Process	• **Data Sampling**: Generates multiple training datasets by sampling with replacement from the	• **Sequential Training**: Models are trained sequentially, each new model focusing on the
	original dataset (bootstrap samples). • **Model Training**: Each model (usually the same type) is trained independently on a different bootstrap sample. • **Aggregation**: Predictions are aggregated (averaged for regression, majority voting for classification) to make the final prediction.	errors made by the previous models. • **Weight Adjustment**: Misclassified instances are given more weight so that subsequent models focus more on these difficult cases. • **Model Combination**: The final model is a weighted sum of the individual models, with more weight given to better-performing models.
Model Independence	• Models are trained independently and in parallel, without any dependency on each other's outcomes.	• Models are trained sequentially, with each model depending on the results of the previous ones.
Algorithm Examples	• Random Forest (an ensemble of decision trees).	• AdaBoost, Gradient Boosting Machines (GBM), XGBoost, LightGBM, and CatBoost.
Strengths	• Effective in reducing variance and improving model stability. • Simple to parallelize and implement.	• Effective at improving accuracy and handling complex data patterns. • Can reduce both bias and variance, leading to high-performance models
Weaknesses	• Does not focus on improving model bias. • May not significantly improve performance if the base learners are not high variance models.	• More complex to implement and train compared to bagging. • Prone to overfitting if not properly regularized. • Sequential nature makes parallelization more challenging.

2.4.2.2 XGBoost

Extreme Gradient Boosting (XGBoost) is a well-known and robust machine learning algorithm usually used for supervised learning tasks such as classification, regression, and ranking. It is based on gradient-boosting architecture and has gained popularity because of its high accuracy and scalability. It is very versatile and that is reason it can handle large datasets and manufacture complex data relationships. Extreme Gradient Boosting (XGBoost) is an advanced implementation of gradient boosting with several optimizations.

XGBoost is a type of supervised learning algorithm that can be used to make predictions on continuous numerical data. It is designed to be efficient, flexible, and portable, and it has become one of the most popular machine learning libraries in the world.

We employ XGBoost because it has plenty of useful features for handling regression tasks. Some of the Benefits and Attributes of XGBoost Model are as follows

1. **High speed and efficiency:** XGBoost is highly optimized and supports parallel processing, much faster than traditional gradient boosting methods. XGBoost Classifier is known for its accuracy and has been shown to outperform other machine learning algorithms in many predictive modeling tasks. It is designed to be computationally efficient and can quickly train models on large datasets.
2. **Handling non-linear relationships:** It can capture complex relationships between input features and target variables.
3. **Feature importance:** XGBoost allows for better feature selection and understanding of model behavior.
4. **Scalability:** It is highly scalable and can handle large datasets with millions of rows and columns.
5. **Flexibility:** It supports a variety of data types and objectives, including regression, classification, and ranking problems.
6. **Regularization:** It incorporates regularization techniques to avoid overfitting and improve generalization performance.
7. **Interpretability:** It provides feature importance scores that can help users understand which features are most important for making predictions.
8. **Open-source:** XGBoost Model is an open-source library that is widely used and supported by the data science community.

XGBoost can be used

- When you have large number of observations in training data.
- Number features < number of observations in training data.
- When data has mixture of numerical and categorical features or just numeric features.
- When the model performance metrics are to be considered. Other uses of XGBoost are

1. It utilizes decision trees as base learners and employs regularization techniques to enhance model generalization.
2. It is known for its high computational efficiency, feature importance analysis, and handling of missing values
3. XGBoost is widely used for tasks such as regression, classification, and ranking. Some of the unique features that make XGBoost interesting are

- **Regularization:** XGBoost has an option to penalize complex models through both L1 and L2 regularization. Regularization helps in preventing overfitting. L1 and L2 regularization penalties can be implemented on leaf weight values to slow down learning and prevent overfitting.
- **Handling sparse data:** Missing values or data processing steps like one-hot encoding make data sparse. XGBoost Classifier incorporates a sparsity-aware split finding algorithm to handle different types of sparsity patterns in the data.
- **Weighted quantile sketch:** Most existing tree based algorithms can find the split points when the data points are of equal weights (using quantile sketch algorithm). However, they are not equipped to handle weighted data. XGBoost has a distributed weighted quantile sketch algorithm to effectively handle weighted data.

To find how good the prediction is, we calculate the loss function, by using the formula,

$$L(yi, pi) = \frac{1}{2}(yi - pi)^2 \text{----------(17)}$$

In general,

$$\sum_{i=1}^{n} L(yi, pi) = \frac{1}{2}(yi - pi)^2 \text{----------(18)}$$

XGBoost uses the loss function to build trees by minimizing the below equation

$$\sum_{i=1}^{n} L\,(\text{yi}, \text{pi}) + \frac{1}{2}\lambda O^2{}_{\text{v}} \text{----------(19)}$$

Where O_V = Output value

The first part of the equation is the loss function and the second part of the equation is the regularization term and the ultimate goal is to minimize the whole equation. For optimizing

output value for the first tree, we write the equation as follows, replace p(i) with the initial predictions and output value and let lambda = 0 for simpler calculations. Now the equation looks like,

$$\sum_{i=1}^{n} L\ (yi,\ pi^{o} + O_v) + \frac{1}{2}\lambda O^2{}_v \text{ ----------(20)}$$

Putting $\lambda = 0$, we have

$$\sum_{i=1}^{n} L\ (yi,\ pi^{o} + O_v) \text{ ----------(21)}$$

XGBoost uses Second-Order Taylor Approximation for both classification and regression. The loss function containing output values can be approximated as follows

$$L(yi,\ pi^{o} + O_v) = L(y, pi) + \left[\frac{d}{dpi} l(y, pi)\right] Ov + \frac{1}{2}\left[\frac{d2}{dp2i} l(y, pi)\right] O^2{}_v \text{ ----------(22)}$$

The first part is Loss Function, the second part includes the first derivative of the loss function and the third part includes the second derivative of the loss function. The first derivative is related to Gradient Descent. Hence, XGBoost uses 'g' to represent the first derivative and the second derivative is related to Hessian, so it is represented by 'h' in XGBoost. Putting the same in the equation, we have

$$L(y,\ pi\ + O_v) = L(y, pi) + gOv + \frac{1}{2} hO^2{}_v \text{ ----------(23)}$$

Expanding the summation, we have

$$\sum_{i=1}^{n} L\ (yi,\ pi^{o} + O_v) + \frac{1}{2}\lambda O^2{}_v \text{ ----------(24)}$$

$$L(y_1,\ pi^{o}\ + O_v) + L(y_2,\ p_2{}^{o}\ + O_v) + \text{---} + L(y_n,\ p_n{}^{o}\ + O_v) + \frac{1}{2}\lambda O^2{}_v \text{ ----------(25)}$$

Plug in Taylor Approximation, we have

$$L(y_1,\ pi^{o}\ ) + g_1 O_v + \frac{1}{2} h1 O^2{}_v + L(y_2,\ p_2{}^{o}\ ) + g_2\ O_v) + \frac{1}{2} h2 O^2{}_v + \text{-----} + L(y_n,\ p_n{}^{o}\ ) + g_n O_v) + \frac{1}{2} hn O^2{}_v + \frac{1}{2}\lambda O^2{}_v \text{ ----------(26)}$$

This is the approximated equation.

We remove the terms that do not contain the output value term, and then minimize the remaining function by following steps:

- Take the derivative w.r.t output value.
- Set derivative equals 0 (solving for the lowest point in parabola)
- Solve for the output value.
- g(i) = negative residuals
- h(i) = number of residuals

$$O_v = \frac{-(g1+g2+\cdots+gn)}{(h1+h2+\cdots+hn+\lambda)} \quad (27)$$

$$gi = \frac{d}{dpi}\left(\frac{1}{2}(yi-pi)2\right) = -(yi-pi) \quad (28)$$

$$hi = \frac{d2}{dpi2}\left(\frac{1}{2}(yi-pi)2\right) = 1 \quad (29)$$

$$O_v = \frac{(y1-p1)+(y2-p2)+\cdots+(yn-n)}{(1+1+\cdots+1+\lambda)} \quad (30)$$

$$O_v = \frac{Sum\ of\ residuals}{Number\ of\ residuals+\lambda} \quad (31)$$

This is the output value formula for XGBoost in Regression. It gives the x-axis coordinate for the lowest point in the parabola.

Other Equations and Formulae Associated with Extreme Gradient Boosting (XGBoost)

Objective Function

$$\text{Obj}(\theta) = \sum_{n=1}^{n} L(\text{yi}, \dot{y}\text{i}) + \sum_{k=1}^{K} \Omega(\text{f}_\text{k}) \quad (32)$$

Where L is the loss function, Ω is the regularization term, yi are the true values, and $\dot{y}$ are the predicted values.

Regularization Term

The regularization term controls the complexity of the model:

$$\Omega(\text{f}) = \gamma\text{T} + \frac{1}{2}\lambda\sum_{j=1}^{T} w^2{}_\text{j} \quad (33)$$

Where T is the number of leaves, wj are the leaf weights, γ and λ are regularization parameters.

Second-Order Approximation

XGBoost uses a second-order Taylor expansion to approximate the loss function

$$\text{Obj}^{(t)} \approx \sum_{n=1}^{n}[gi\text{f}_\text{t}(\text{xi}) + \frac{1}{2}\text{h}_\text{i}\text{f}_\text{t}(\text{xi})^2] + \Omega(\text{f}_\text{t}) \quad (34)$$

Where $g_i = \frac{\partial L(yi,\dot{y}i}{\partial \dot{y}i}$ and $h_i = \partial^2 L(yi, \dot{y}i)/\partial \dot{y}^2{}_i$ are the first and second derivatives of the loss function

Split Finding

$$\text{Gain} = \frac{1}{2}\left[\frac{\sum_{i=IL} g^2\ i}{\sum_{n=IL} hi+\lambda} + \frac{\sum_{i=Ig} g^2\ i}{\sum_{n=IR} hi+\lambda} - \frac{\sum_{i=I} g^2\ i}{\sum_{n=I} hi+\lambda}\right] - \gamma \quad (35)$$

Where I is the set of instances and I_L and I_h are the sets of instances in the left and right branches after the split.

Leaf Weight Calculation

The optimal weight of a leaf is calculated as

$$W_j = -\frac{\sum_{i=Ij} gi}{\sum_{n=Ij} hi+\lambda} \text{ ----------(36)}$$

Prediction

The final prediction for a data point x is the sum of the predictions from all the trees

$$\hat{y} = \sum_{t=1}^{T} ft(x) \text{ ----------(37)}$$

Table 3: Differences between XGBoost and Gradient Boosting.

S/N	Features	XGBoost	Gradient Boosting
1	Speed	XGBoost is significantly faster than traditional Gradient Boosting due to its parallel processing and other optimizations.	Gradient boosting speeds less compared to XGBoost
2	Regularization	XGBoost includes regularization (L1 and L2), which helps in controlling overfitting	Traditional Gradient Boosting typically does not
3	Handling Missing Values	XGBoost has built-in mechanisms to handle missing values efficiently	Gradient boosting does not have such capacity
5.	When to Use/Practical Considerations	Use XGBoost for larger datasets, more complex problems, and when you need high performance and efficiency. It is particularly useful in competitive machine learning scenarios such as Kaggle competitions.	Use Gradient Boosting when you have smaller datasets, less computational power, or when the problem is simple and does not require extensive tuning and optimization.

2.5 Hyperparameter and Hyperparameter tuning in Machine Learning

A hyperparameter is a parameter that specifies details of the learning process in a machine learning model. Hyperparameters control various aspects of training and optimization, unlike regular parameters which determine the model itself. A hyperparameter is a kind of parameter that cannot be directly learned from the regular training process. They are usually fixed before the actual training process begins. These parameters express important properties of the model such as its complexity or how fast it should learn. Examples of hyperparameters include the learning rate, batch size, and choice of optimizer. Hyperparameters are key parts of learning algorithms which effect the performance and accuracy of a model. They play a vital

role in shaping model behavior, and their careful selection is essential for effective machine learning. There are two types of hyperparameters. These include:

2.5.1 Model Hyperparameters

These are specific to the model architecture and cannot be inferred during training. For instance, the topology and size of a neural network are model hyperparameters.

2.5.2 Algorithm Hyperparameters

These affect the learning process and can vary across different algorithms. Examples include the learning rate and batch size.

It is important to note that learning rate and n-estimators are the two critical hyperparameters for gradient boosting decision trees.

Learning rate controls how fast the model learns. This is done by multiplying the error in previous model with the learning rate and then use that in the subsequent trees. Therefore, the lower the learning rate, the slower the model learns. Each tree added modifies the overall model. The advantage of slower learning rate is that the model becomes more robust and efficient and avoids overfitting. In statistical learning, models that learn slowly perform better. Though, learning slowly comes at a cost. It takes more time to train the model which brings about the concept of n-estimator.

n-estimator is the number of trees used in the model. If the learning rate is low, we need more trees to train the model. But, we need to be very careful at selecting the number of trees. It creates a high risk of overfitting to use too many trees. n-estimator is a crucial hyperparameter in the context of random forests. It represents the number of decision trees within the random forest classifier. A random forest is essentially an ensemble of multiple decision tree classifiers. Each tree is trained on a different subset of the dataset, and their predictions are combined to improve overall accuracy and mitigate overfitting. In general, n-estimators determine the number of trees in the forest, impacting both model accuracy and computational resources. It's a key parameter to fine-tune when building random forest models. In gradient boosting decision trees, the addition of too many trees will cause overfitting unlike in random forests, the addition of too many trees will not cause overfitting. In random forest, the accuracy of the model does not improve after a certain point but no problem of overfitting is encountered. On the other hand, in gradient boosting decision trees, we have to be careful about the number of trees we select, because having too many weak learners in the model may lead to overfitting of data.

Therefore, gradient boosting decision trees require very careful tuning of the hyperparameters. Hyperparameter tuning is the process of selecting the optimal values for a machine learning model's hyperparameters. The goal of hyperparameter tuning is to find the values that lead to the best performance on a given task.

Hyperparameters are settings that control the learning process of the model, such as the learning rate, the number of neurons in a neural network, or the kernel size in a support vector machine. They are configuration variables that are set before the training process of a model begins. They control the learning process itself, rather than being learned from the data. In other words, they control the learning process of a machine learning model. They are often used to tune the performance of a model, and they have a significant impact on the model's accuracy, generalization, and other metrics. Hyperparameters are distinct from model parameters, which are the weights and biases that are learned from the data.

Hyperparameters in XGBoost

The following essential XGBoost hyperparameters need to be adjusted:

1. Learning rate: This hyperparameter determines the step size taken by the optimizer during each iteration of training. A larger learning rate can lead to faster convergence, but it may also increase the risk of overfitting. A smaller learning rate may result in slower convergence but can help prevent overfitting.
2. n-estimators: This hyperparameter determines the number of boosting trees to be trained. A larger number of trees can improve the model's accuracy, but it can also increase the risk of overfitting. A smaller number of trees may result in lower accuracy but can help prevent overfitting.
3. Max-depth: This hyperparameter determines the maximum depth of each tree in the ensemble. A larger max-depth can allow the trees to capture more complex relationships in the data, but it can also increase the risk of overfitting. A smaller max-depth may result in less complex trees but can help prevent overfitting.
4. Min-child-weight: This hyperparameter determines the minimum sum of instance weight (hessian) needed in a child node. A larger min-child-weight can help prevent overfitting by requiring more data to influence the splitting of trees. A smaller min-child-weight may allow for more aggressive tree splitting but can increase the risk of overfitting.
5. Subsample: This hyperparameter determines the percentage of rows used for each tree construction. A smaller subsample can improve the efficiency of training but may reduce

the model's accuracy. A larger subsample can increase the accuracy but may make training more computationally expensive.

Advantages of Hyperparameter tuning

The advantages of hyperparameter tuning includes:

- Improved model performance
- Reduced overfitting and underfitting
- Enhanced model generalizability
- Optimized resource utilization
- Improved model interpretability

Disadvantages of Hyperparameter tuning

The disadvantages of hyperparameter tuning includes:

- Computational cost
- Time-consuming process
- Risk of overfitting
- No guarantee of optimal performance
- Requires expertise

Applications of Hyperparameter Tuning

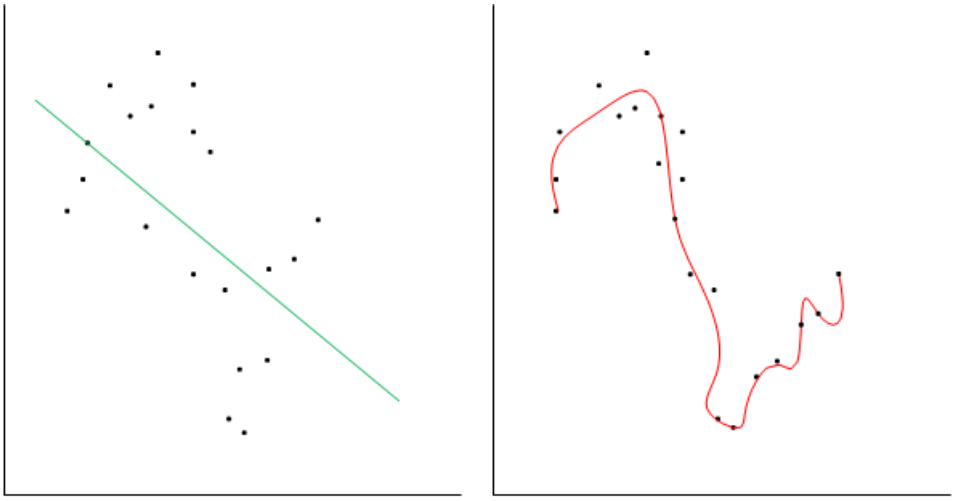
Hyperparameter tuning can be useful in the following:

- Model Selection: This involves choosing the right model architecture for the task
- Regularization Parameter Tuning: This involves controlling model complexity for optimal performance
- Feature Preprocessing Optimization: This involves enhancing data quality and model performance
- Algorithmic Parameter Tuning: This involves adjusting algorithm-specific parameters for optimal results

2.6 Bias and Variance in Machine Learning

Bias is the error due to overly simplistic assumptions in the learning algorithm. These assumptions make the model easier to comprehend and learn but might not capture the underlying complexities of the data. It is the error due to the model's inability to represent the true relationship between input and output accurately. When a model has poor performance

both on the training and testing data, it means high bias because of the simple model, indicating underfitting. In other word, bias indicates underfitting. Variance, on the other hand, is the error due to the model's sensitivity to fluctuations in the training data. It is the variability of the model's predictions for different instances of training data. High variance occurs when a model learns the training data's noise and random fluctuations rather than the underlying pattern. As a result, the model performs well on the training data but poorly on the testing data, indicating overfitting. Variance indicates overfitting. A high-bias model results from not learning data well enough. It is not related to the distribution of the data. Hence future predictions will be unrelated to the data and thus incorrect. The resultant effect of high bias is underfitting. A high variance model results from learning the data too well. It varies with each data point. Hence it is impossible to predict the next point accurately. The resultant effect of high variance is overfitting. Plots of Bias and Variance are shown in figures 6.



High bias, low variance **Low bias, high variance**

Figure 6: Bias and Variance.

2.7 Underfitting and Overfitting in Machine Learning

Underfitting occurs in a machine learning algorithm (or statistical model) when a model is too simple to capture data complexities. Underfitting represents the inability of the model to learn the training data effectively, which result in poor performance both on the training and testing data. In simple terms, an underfit model is inaccurate, especially when applied to new, unseen examples. It mainly happens when we use very simple model with overly simplified assumptions. To address underfitting problem of the model, we need to use more complex models, with enhanced feature representation, and less regularization. It is important to note that the underfitting model has high bias and low variance.

Underfitting occurs in machine learning for the following reasons when:

a. The model is too simple; thus it may not be capable to represent the complexities in the data.
b. The input features which is used to train the model is not the adequate representations of underlying factors influencing the target variable.
c. The size of the training dataset used is not enough.
d. Excessive regularization is used to prevent the overfitting, which constraint the model to capture the data well.
e. Features are not scaled.

Underfitting can be minimized by adopting the following techniques:

a. Increasing the model complexity.
b. Increasing the number of features, performing feature engineering.
c. Removing noise from the data.
d. Increasing the number of times of training (iteration) or increasing the duration of training to get better results.

Overfitting occurs in statistical model when the model does not make accurate predictions on testing data. When a model gets trained with so much data, it starts learning from the noise and inaccurate data entries in the data set. When testing with test data results in high variance, then the model does not categorize the data correctly, because of too many details and noise.

The causes of overfitting are the non-parametric and non-linear methods because these types of machine learning algorithms have more freedom in building the model based on the dataset and therefore they can really build unrealistic models. A solution to avoid overfitting is using a linear algorithm if we have linear data or using the parameters like the maximal depth if we are using decision trees. Simply put, overfitting is a problem where the evaluation of machine learning algorithms on training data is different from unseen data.

Overfitting occurs in machine learning for the following reasons when:

a. High variance and low bias.
b. The model is too complex.
c. The size of the training data.

Overfitting can be minimized by adopting the following techniques

a. Improving the quality of training data. This reduces overfitting by focusing on meaningful patterns, mitigate the risk of fitting the noise or irrelevant features.
b. Increasing the training data. This can improve the model's ability to generalize to unseen data and reduce the likelihood of overfitting.
c. Reducing model complexity.
d. Early stopping during the training phase (have an eye over the loss over the training period as soon as loss begins to increase stop training).
e. Ridge Regularization and Lasso Regularization.
f. Use dropout for neural networks to tackle overfitting. Underfitting and Overfitting plots are shown in figure 7.

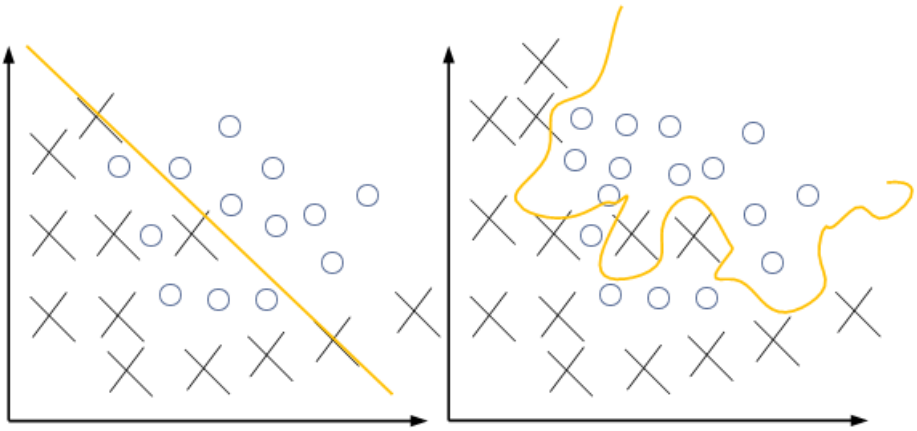


Figure 7: Underfitting and Overfitting Model plots.

2.8 Good Fit in a Statistical Model and K-Fold Cross Validation

A model is said to be in good fit if it makes predictions with zero error on the data. This can be achieved at a spot between overfitting and underfitting. Good fit happens over passage of time while the model is learning from the training dataset. As it keeps on learning, the error for the model on the training and testing data will keep on decreasing. If it will learn for too long, the model will become more prone to overfitting due to the presence of noise and less useful details. Therefore, the performance of the model will decrease. In order to get a good fit, it will be important to stop at a point just before where the error starts increasing. At this point, the model is said to have good skills in training datasets as well as the unseen testing dataset.

Cross-validation is a statistical method primarily used in machine learning to estimate the skill of a machine learning model on unseen data. It is commonly used in applied machine learning to compare and select a model for a given predictive modeling problem because it is easy to understand, easy to implement, and results in skill estimates that generally have a lower bias than other methods. K-fold cross-validation is a technique for evaluating predictive models. However, the value of k depends on the size of the dataset. A good standard values for k in k-fold cross-validation are 5 and 10. However, the value of k depends on the size of the dataset. For example, for small datasets, we can use higher values for k. Larger values of k will increase the runtime of the cross-validation algorithm and the computational cost. When a specific value for k is chosen, it may be used in place of k in the reference to the model, such as k=10 becoming 10-fold cross-validation: 10% of the test set is held back each time. When k=5, 20% of the test set is held back each time.

K-fold cross-validation is designed and needed for hyperparameter tuning. The reason behind fitting the best model to the whole training set after k-fold cross-validation is to provide more training samples to the learning algorithm of the best model. This usually results in a more accurate and robust model.

The general procedure for K-fold cross-validation is as follows:

1. Shuffle the dataset randomly.
2. Split the dataset into k groups
3. For each unique group:

- Take the group as a hold out or test data set
- Take the remaining groups as a training data set
- Fit a model on the training set and evaluate it on the test set
- Retain the evaluation score and discard the model

4. Summarize the skill of the model using the sample of model evaluation scores

Most importantly, each observation in the data sample is assigned to an individual group and stays in that group for the duration of the procedure. This means that each sample is given the opportunity to be used in the hold out set 1 time and used to train the model k-1 times.

This approach involves randomly dividing the set of observations into k groups, or folds, of approximately equal size. The first fold is treated as a validation set, and the method is fit on the remaining k-1 folds. The Life Cycle of K-Fold Cross-Validation is shown in figure 8.

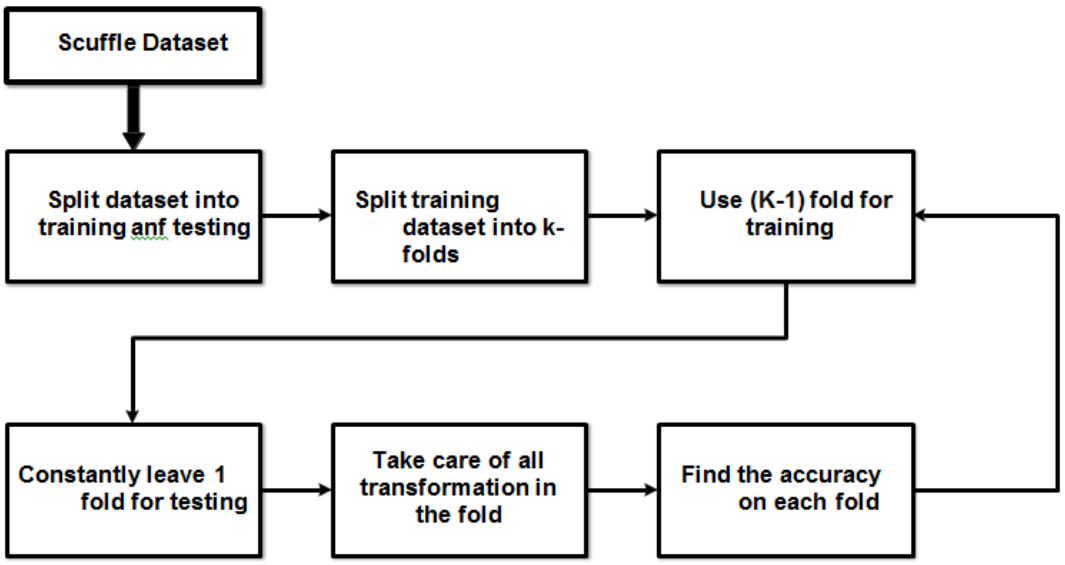


Figure 8: The Life Cycle of K-Fold Cross-Validation.

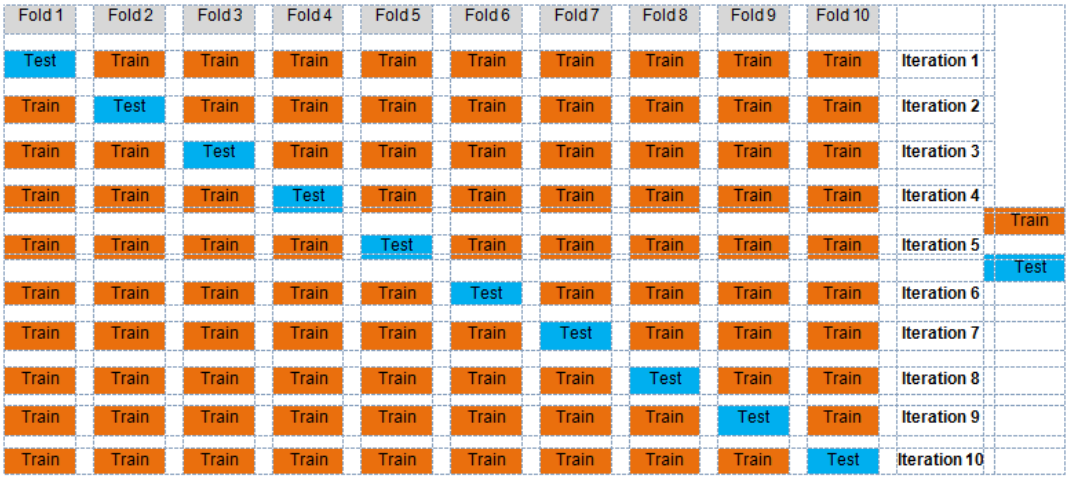


Figure 9: The general procedure for K-fold cross-validation when K=10.

From figure 13, K=10, it means that dataset is divided into 10 folds. One-fold run is the Train and the other run is the Test. During each run, one fold is considered for testing and the rest give an idea of the flow of the fold-defined size.

3. MODEL DEVELOPMENT

The prediction methodology that powers the model development involves the following key steps:

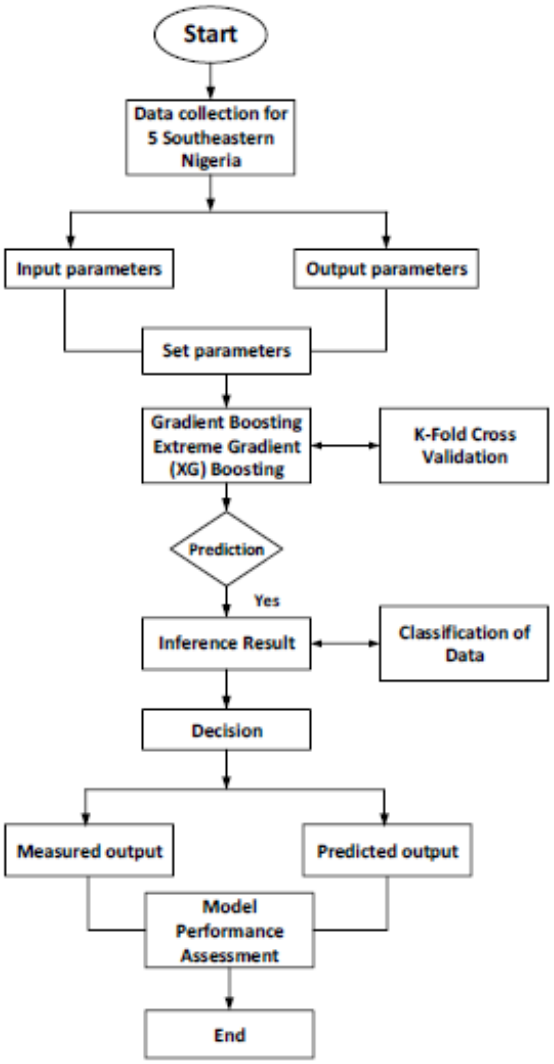
1. Selection of the variables to predict such as global. Annual, Monthly, Daily, Diffuse or Beam solar radiation on ground horizontal or tilted surfaces.

2. Selection of the input parameters and data collection e.g Temperature (T), Relative humidity, Precipitation (PPT), Surface pressure (PS). Wind speed (WD), and Wind direction

(WD).

3. Definition of training and testing sets with K-Fold Cross Validation
4. Development of models and training phase assessment.
5. Determination and calculation of prediction errors with statistical indicators such as R^2, RMSE, and nRMSE.
6. Comparisons and final choice of the best models following statistical indicators such as maximum coefficient of determination, R^2 and minimum RMSE and nRMSE.

The prediction is performed using Gradient Boosting and Extreme Gradient Boosting (XGBoost) while the K-Fold Cross Validation are used in improving the performance of the XGBoost model. The block diagram of the model development procedures is shown in figure 10.



Figures 10: Block Diagram of the Model Development.

3.1 Gradient Boosting and XGBoost Development

The data set used for this study is presented in an excel spreadsheet and saved as CSV file with one column of years 2012-2021, one column of months January-December, one column of days, six columns of input parameters and one column of output parameter. For the six input parameters, #1 represents temperature, #2 denotes relative humidity, #3 depicts precipitation, #4 signifies surface pressure, #5 symbolises wind speed and #6 characterises wind direction. The input and output parameters is represented in figure 12.

[4]:

	YEAR	MONTH	DAY	1	2	3	4	5	6	SOL RAD
0	2012	1	1	24.12	75.56	0.23	100.13	2.94	89.56	11.77
1	2012	1	2	22.97	65.81	0.00	100.16	2.73	95.50	11.89
2	2012	1	3	22.76	68.44	0.00	100.12	2.51	112.94	10.94
3	2012	1	4	22.14	65.25	0.02	100.09	3.41	31.81	10.85
4	2012	1	5	22.72	70.19	0.18	100.16	1.79	32.62	10.23
...	...	...	...	...	...	...	...	...	...	...
3648	2021	12	27	26.45	83.88	0.10	99.99	1.51	251.12	10.68
3649	2021	12	28	24.47	78.50	0.71	100.10	1.62	155.25	10.17
3650	2021	12	29	23.95	72.50	0.31	100.08	2.34	229.25	8.53
3651	2021	12	30	24.54	72.62	0.00	99.98	2.44	165.38	12.62
3652	2021	12	31	24.76	73.56	3.09	100.03	1.80	231.75	11.17

3653 rows × 10 columns

Figures 11: Input and Output Parameters in Xsl in Python.

The range of input and output parameter values is found in figure 12

[5]:

	YEAR	MONTH	DAY	1	2	3	4	5	6	SOL RAD
count	3653.000000	3653.000000	3653.000000	3653.000000	3653.000000	3653.000000	3653.000000	3653.000000	3653.000000	3653.000000
mean	2016.499589	6.522310	15.731454	25.640397	86.129689	4.973321	100.053575	2.346919	215.311705	11.038429
std	2.873056	3.449345	8.802064	1.114471	7.471530	7.818522	0.139971	0.603206	44.255913	2.069756
min	2012.000000	1.000000	1.000000	19.540000	49.060000	0.000000	99.640000	0.810000	12.810000	2.300000
25%	2014.000000	4.000000	8.000000	24.930000	84.810000	0.420000	99.930000	1.910000	205.620000	9.910000
50%	2016.000000	7.000000	16.000000	25.670000	88.190000	2.630000	100.050000	2.310000	222.810000	11.240000
75%	2019.000000	10.000000	23.000000	26.440000	90.500000	6.270000	100.170000	2.720000	239.060000	12.450000
max	2021.000000	12.000000	31.000000	28.890000	96.000000	105.920000	100.560000	5.310000	327.750000	16.600000

Figures 12: Range of input and output parameter values.

3.2 Model Performance Assessment

In this present study, the models performance is analyzed using the following statistical indicators:

1. Coefficient of Determination: R^2 is an empirical statistic that helps clarify how suitably a model fits with the real-world data observed. Its range is confined between 0 and 1. An

increase in this statistic signifies an improvement in the model's predictive efficacy. Hence, a value of R^2 that approaches unity indicates a superior capacity of the model to estimate the dependent variable. This metric shows the extent of variability in the response variable that is elucidated by the influencing variables. It is expressed mathematically as:

$$R^2 = 1 - \frac{\sum (ya - yp)2}{\sum (ya - yp)} \text{----------(38)}$$

Where n = number of observations ya = Actual value

yp = Predicted value

yp bar = Mean predicted values

2. Root-Mean-Square Error (RMSE): The RMSE represents a quantitative approach utilized to evaluate the precision of a model in forecasting target values. It elucidates insights regarding the immediate performance by systematically comparing the actual deviations between the estimated predictions and the observed measurements. To assess the performance of a model in machine learning, we compute the root mean square of the variations that have developed between the observed test results and the expected outputs. It is expressed mathematically as: For a wide set of values, RMSE is defined as follows:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (yi,e - yi,m)}{n}} \text{----------(39)}$$

Where n = number of observations $Y_{i,e}$ = The ith estimated value

$Y_{i,m}$ = The ith measured value

3. Normalized Root-Mean-Square Error (nRMSE): The procedure to achieve the nRMSE includes the division of the Root Mean Square Error (RMSE) by the mean of the observed data. Individuals occasionally denote nRMSE as Relative Root Mean Square Error (rRMSE) or as Root Mean Square Standardized (RMSS).

$$nRMSE = \frac{RMSE}{E(Yi\ cap)} \text{----------(40)}$$

$$nRMSE = \frac{RMSE}{ymax - ymin} \text{----------(41)}$$

$$nRMSE = \frac{1}{E\ (Yi\ cap)} \cdot \sqrt{\frac{\sum_{i=1}^{n} (Yi - Yi\ cap)2}{n}} \text{----------(42)}$$

Where n = number of data pairs (predicted and measured values) in testing set Yi cap and Yi are the i-th measured (actual) and predicted values

E (Yi cap) is the mean of the measured values

Lower values of RMSE and nRMSE signify good performance while higher values of R^2 signify good model performance

4. RESULTS AND DISCUSSION

The scatter plots for each of the model development are showed in figures 13-17. These plots indicate that the performance assessment for each of the models with respect to the measured and the predicted solar radiation are of good fit. It also implies that the models made predictions with minimal error on the data for the 5 southeast states Nigeria. The evaluation of the models performance in relation to both observed and predicted solar radiation indicated a strong level of correspondence.

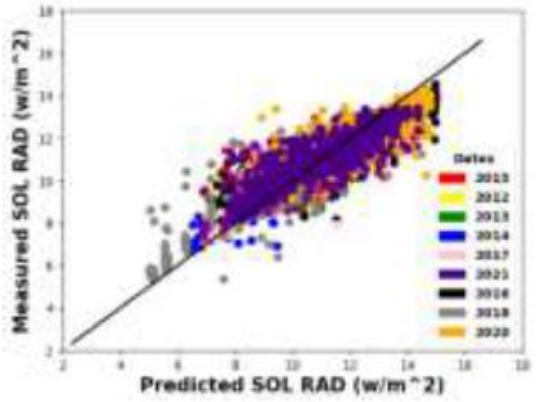
The phenomenon of optimal model fit transpires progressively as the algorithm assimilates knowledge from the training dataset. As the learning process continues, the discrepancy in performance metrics for both the training and testing datasets will consistently diminish. However, should the learning process extend excessively, the model may become increasingly susceptible to the pitfalls of overfitting, exacerbated by extraneous noise and superfluous information. Hence, the capability of the model is probably going to lessen. To attain an optimal fit, it is critical to terminate the training process at a juncture immediately prior to the onset of error escalation. At this critical juncture, the model is deemed proficient in its ability to generalize to both the training datasets and the previously unexamined testing datasets. The evaluation of optimal fit performance is facilitated through the implementation of K-Fold Cross Validation, which systematically and randomly allocates data for both training and testing purposes throughout a designated timeframe.

```
kf = KFold(n_splits=10, shuffle=True, random_state=22)
models = Parallel(n_jobs=-1)(delayed(fit_model)(X[train_index], y[train_index],GradientBoostingRegressor()) for train_index, _ in kf.split(X, y))
```

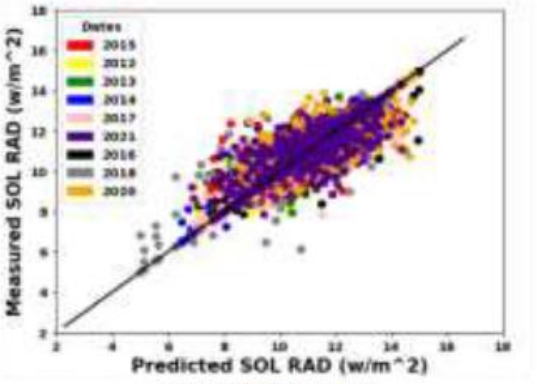
Table 4 shows that XGBoost is the best model for the prediction of solar radiation in the southeast Nigeria among the competing models based on the statistical indicators. It consistently has maximum coefficient of determinations (R^2) and consistently has the minimum RMSE and nRMSE respectively for each of the states. For examples, the result also showed that Ebonyi state have the highest prediction accuracy with XGBoost, R^2 of 0.972, secondly, Anambra state with XGBoost, R^2 of 0.959, thirdly, Imo state with XGBoost, R^2 of 0.952, fourthly, Abia state with XGBoost, R^2 of 0.948 and lastly, Enugu state with XGBoost, R^2 of 0.939. Figure 18 shows the comparison of the XGBoost and GB models performance

evaluation in Southeastern Nigeria.

4.1 Abia State and Model Development



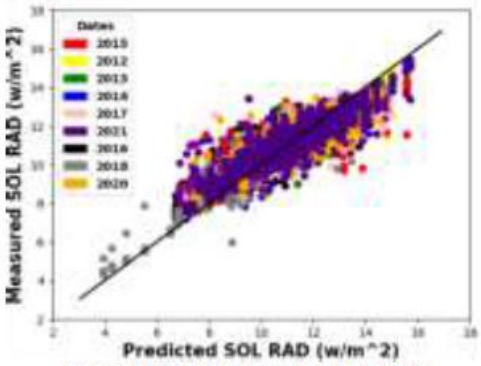
Abia State Gradient Boosting Scatter Plot



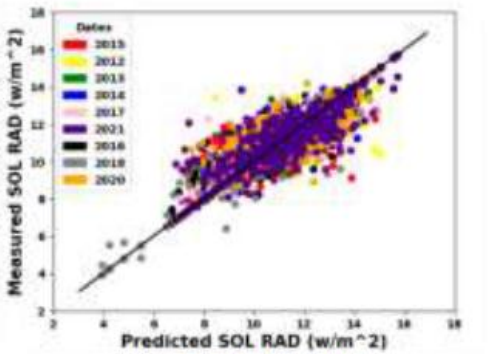
Abia State XGB Scatter Plot

Figures 13: Abia state GB and XGBoost Scatter Plots.

4.2 Anambra State and Model Development



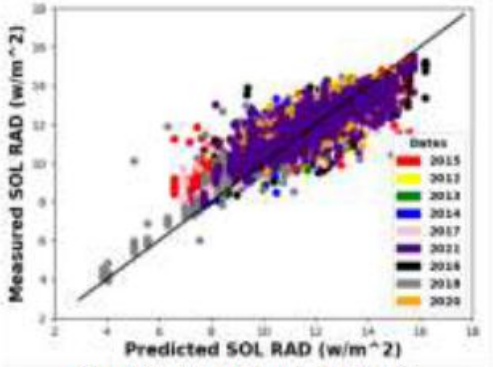
Anambra State Gradient Boosting Scatter Plot



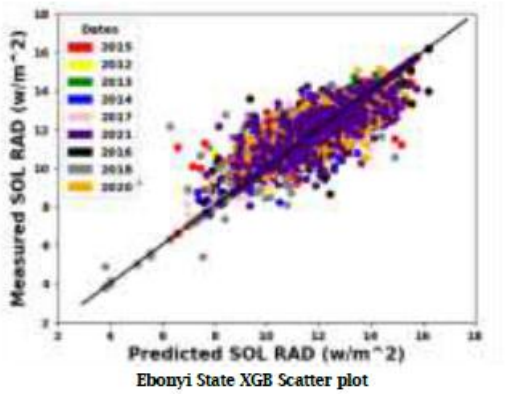
Anambra State XGB Scatter Plot

Figures 14: Anambra state GB and XGBoost Scatter Plots.

4.3 Ebonyi State and Model Development



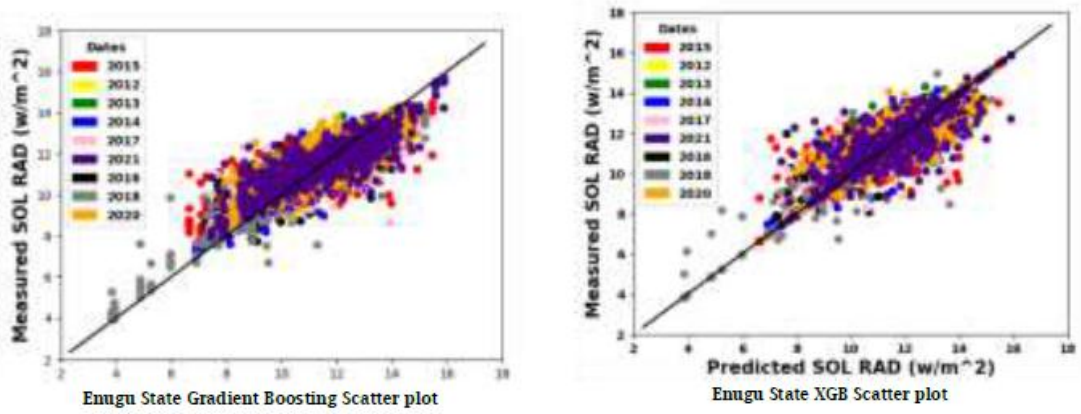
Ebonyi State Gradient Boosting Scatter plot



Ebonyi State XGB Scatter plot

Figures 15: Ebonyi state GB and XGBoost Scatter Plots.

4.4 Enugu State and Model Development

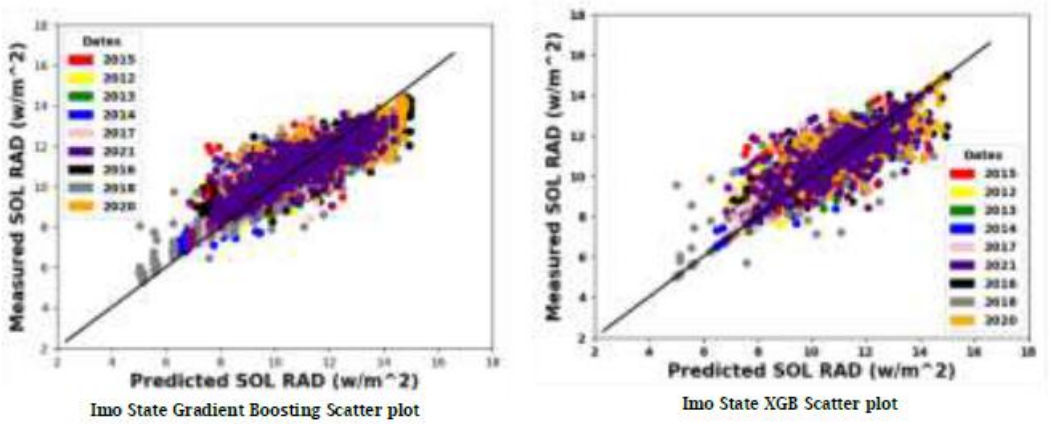


Enugu State Gradient Boosting Scatter plot

Enugu State XGB Scatter plot

Figures 16: Enugu state GB and XGBoost Scatter Plots.

4.5 Imo State and Model Development



Imo State Gradient Boosting Scatter plot

Imo State XGB Scatter plot

Figures 17: Imo state GB and XGBoost Scatter Plots.

Table 4: Comparisons of the Performance metrics of Gradient Boosting (GB) and Extreme Gradient Boosting (XGBoost) models for the Southeastern Nigeria.

Location	Model	Statistical Parameters		
		R^2	RMSE	nRMSE
Abia	GB	0.824	0.5378	0.0780
	XGBoost	0.948	0.2817	0.0427
Anambra	GB	0.841	0.5411	0.0723
	XGBoost	0.959	0.2838	0.0359
Ebonyi	GB	0.902	0.6818	0.0580
	XGBoost	0.972	0.3802	0.0306
Enugu	GB	0.832	0.5449	0.0767
	XGBoost	0.939	0.2829	0.0370
Imo	GB	0.831	0.5289	0.0769
	XGBoost	0.952	0.2799	0.0388

Figure 18 shows the comparison of the XGBoost and GB models performance evaluation in Southeastern Nigeria.

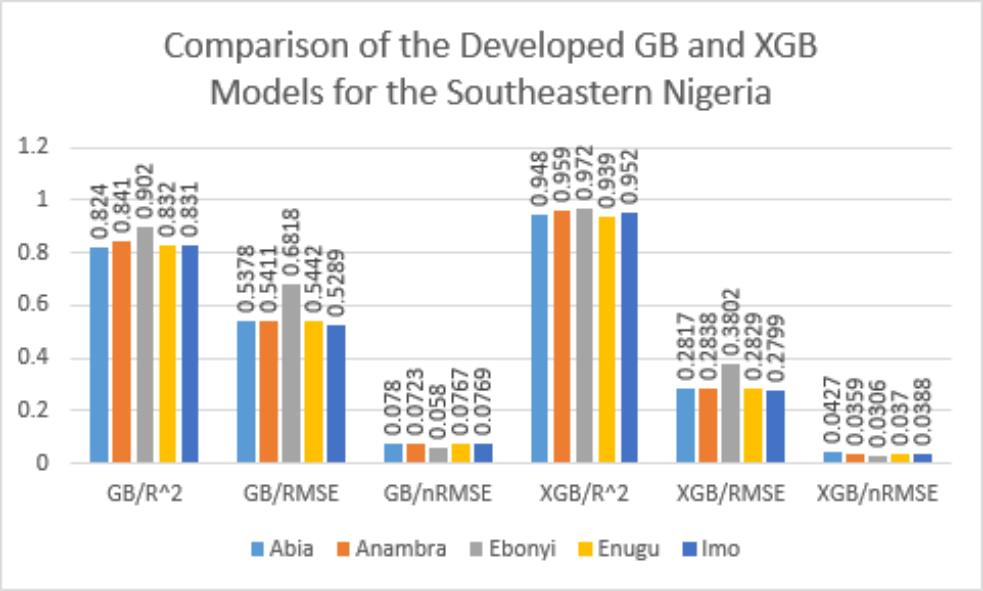


Figure 18: Comparison of the Performance Evaluation of Gradient Boosting and Extreme Gradient Boosting (XGBoost) Models for the Southeastern Nigeria.

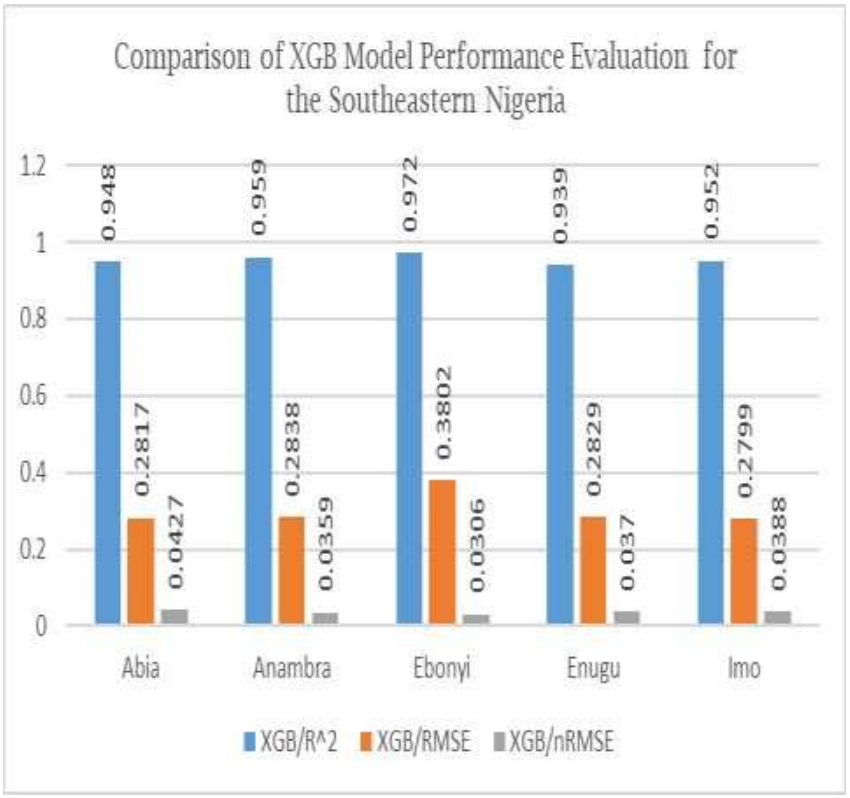


Figure 19: Comparison of XGBoost Model Performance Evaluation for the Southeastern Nigeria.

4.6 Percentage of Performance of Extreme Gradient Boosting (XGB) Model over Gradient Boosting (GB) Model using the Normalized Root Mean Square Error (nRMSE) and Root Mean Square Error (RMSE) by the Southeastern States

Normalized Root Mean Square Error (nRMSE)

Abia State

% of Performance of XGBoost over GB in Abia State is given as $\frac{0.0780-0.0427}{0.0427}x100 = 82.67\%$

Anambra State

% of Performance of XGBoost over GB in Anambra State is given as $\frac{0.0723-0.0359}{0.0359}x100 = 101.40\%$

Ebonyi State

% of Performance of XGBoost over GB in Ebonyi State is given as $\frac{0.0580-0.0306}{0.0306}x100 = 89.54\%$

Enugu State

% of Performance of XGBoost over GB in Enugu State is given as $\frac{0.0767-0.0370}{0.0370}x100 = 107.29\%$

Imo State

% of Performance of XGBoost over GB in Imo State is given as $\frac{0.0769-0.0388}{0.0388}x100 = 98.19\%$

Average nRMSE of % of Performance of XGBoost over GB in the 5 Southeastern States

$$\frac{82.67 + 101.40 + 89.54 + 107.29 + 98.19}{5} = 95.82\%$$

Root Mean Square Error (RMSE)

Abia State

% of Performance of XGBoost over GB in Abia State is given as $\frac{0.5378-0.2817}{0.2817}x100 = 90.91\%$

Anambra State

% of Performance of XGBoost over GB in Anambra State is given as $\frac{0.5411-0.2838}{0.2838}x100 = 90.66\%$

Ebonyi State

% of Performance of XGBoost over GB in Ebonyi State is given as $\frac{0.6818-0.3802}{0.03802}x100 = 79.32\%$

Enugu State

% of Performance of XGBoost over GB in Enugu State is given as $\frac{0.5449-0.2829}{0.2829}x100 = 92.61\%$

Imo State

% of Performance of XGBoost over GB in Imo State is given as $\frac{0.5289-0.2799}{0.2799}x100 = 88.96\%$

Average RMSE of % of Performance of XGB over GB in the 5 Southeastern States

$$\frac{90.91 + 90.66 + 79.32 + 92.61 + 88.96}{5} = 88.49\%$$

CONCLUSION

The implementation of the Extreme Gradient Boosting (XGBoost) technique to model daily solar radiation is notably significant across renewable energy, agriculture, environmental observation, and urban growth. XGBoost is esteemed for its superior accuracy, proficiency in handling intricate data interrelations, and resilience to overfitting, rendering it an exemplary choice for dependable predictive analytics.

XGBoost is typically preferred over conventional gradient boosting algorithms due to its superior efficacy regarding computational speed, regularization techniques, management of sparse datasets, and adaptability in model optimization. It provides a more effective and scalable approach, particularly when dealing with extensive and intricate datasets, while concurrently mitigating the potential for overfitting. These benefits render XGBoost an influential and preferred option for numerous practitioners in the field of machine learning, especially within highly competitive data science contexts.

In this research, the better model between gradient boosting (GB) and extreme gradient boosting (XGBoost) in predicting solar radiation in southeastern Nigeria is the XGBoost model. It consistently has the maximum coefficient of determination, R^2, and consistently has the minimum root mean square error (RMSE) and normalized root mean square error (nRMSE), respectively, for the five southeastern Nigerian states. The performance assessment for each of the models with respect to the measured and predicted solar radiation is a good fit. The XGBoost model has the best fit when compared to the gradient boosting model for the prediction of solar radiation in southeastern Nigeria. Based on the results of the performance assessments shown in Table 4 and Figure 18, respectively, Ebonyi State is predicted to have the highest XGBoost prediction accuracy with an R^2 of 0.972. The average nRMSE of the performance of XGBoost over GB in the 5 Southeastern States is 95.82%, and the average RMSE of the performance of XGB over GB in the 5 Southeastern States is 88.49%.

Namrata et al (2023) used XGB-MFO and found impressive R^2 scores of 0.9337 for Global Horizontal Irradiance (GHI), 0.9011 for Diffuse Horizontal Irradiance (DHI), and 0.8744 for Direct Normal Irradiance (DNI).

According to Mbah et al (2021), the XGBoost algorithm, created to predict daily global solar radiation on sloped surfaces, showed remarkable effectiveness in its forecasting abilities. While training, it noted an R^2 value of 0.9977, an RMSE measuring 1.6988, and a MAE of 1.081. The testing phase clearly showcased its impressive performance, highlighted by an R^2 value of 0.9934, an RMSE of 2.8558, and an MAE of 2.033, which emphasizes its skill in forecasting solar radiation.

FUTURE RESEARCH

Future research should focus on some key areas in order to enhance and expand the modeling of daily solar radiation using the XGBoost machine learning algorithm. These include.

Data Quality and Availability

Advanced Data Collection

Employ cutting-edge (innovative) data acquisition techniques, including the utilization of pyranometer sensors and microcontrollers, to systematically collect high-fidelity solar radiation data in an efficient and economically viable manner. This methodology has the potential to alleviate financial constraints and enhance the precision of data utilized for model training (Nadeem et al. 2024).

Handling Missing Data

Formulate comprehensive methodologies for addressing the issue of absent data, including basic mean imputation and Bayesian optimization, to guarantee the accuracy of models despite the incompleteness of the data (Halima et al. 2024).

Hybrid Model Integration

Wavelet Transform Integration

Incorporate the wavelet technique with the XGBoost framework to refine the precision of the predictive model. Hybrid methodologies, such as Support Vector Regression combined with Wavelet Transform (SVR-WT), have demonstrated superior efficacy compared to standalone models, indicating that analogous incorporation with XGBoost may produce enhanced outcomes (Küçüktopcu et al. 2024).

Other future research/investigations pertaining to the modeling of daily solar radiation utilizing the Extreme Gradient Boosting (XGBoost) machine learning algorithm may concentrate on several domains for enhancement and expansion.

These domains might include the blending of additional weather and ecological factors, the fusion of XGBoost with other methods, the enhancement of time and space generalization, the integration of uncertainty assessment and clarity, the fine-tuning of hyperparameters customized for particular areas or circumstances, the creation of real-time platforms and IoT solutions, the analysis of climate shifts and severe weather events, the comparison with newly emerging techniques, and the formation of extensive worldwide datasets for forecasting solar energy.

The integration of additional meteorological and environmental variables, including aerosols, air quality indices, and atmospheric contaminants, has the potential to furnish greater context

for solar radiation predictions, particularly in locations characterized by considerable pollution or fluctuating atmospheric conditions. Experts might evaluate hybrid modeling approaches, incorporating the synergy of XGBoost with alternative algorithms or ensemble systems, to effectively grasp both spatial and temporal dynamics connected to solar radiation. The use of uncertainty evaluation strategies in the XGBoost architecture could support the development of more dependable outcomes, especially when dealing with input data that exhibits noise or partial information. Furthermore, enhancing the interpretability of XGBoost models could significantly bolster the reliability of solar energy systems in adverse conditions. Through a concentrated focus on these domains, forthcoming research endeavors may substantially improve the accuracy and generalization of solar radiation predictions.

REFERENCES

1. Ajayi, O. et al. New model to estimate daily global solar radiation over Nigeria, 2014.
2. Sustainable Energy Technologies and Assessments, 5: 28-36.
3. Akpabio, L. E., Udo, S. O. and Etuk. S. E. Modeling global solar radiation for a tropical location: Onne, Nigeria. Turk J. Physics, 2005; 29: 63-68.
4. Anshul, S. Gradient Boosting Algorithm: A Complete Guide for Beginners, 2024.
5. Bashiru, O., Ochem, C., Enyejo, L. A., Manuel, H.N.N., Adeoye, T. O., 2024.
6. The crucial role of renewable energy in achieving the sustainable development goals for cleaner energy. Global Journal of Engineering and Technology Advances, doi: 10.30574/gjeta.2024.19.3.0099
7. Benali, L., Notton, G., Fouilloy, A., Voyant, C., Dizene, R. Solar radiation forecasting using artificial neural network and random forest methods: application to normal beam, horizontal diffuse and global components, Renew. Energy, 2019; 132: 871–884.
8. Benmouiza, K. and Cheknane, A. Forecasting hourly global solar radiation using hybrid k-means and nonlinear autoregressive neural network models. Energy Conversion and Management, 2013; 75: 561-569.
9. Besharat, F. et al. Empirical models for estimating global solar radiation: A review and case study," Renewable and Sustainable Energy Reviews, 2013; 21: 798-821.
10. Brijesh, S. (2023). Understanding Boosting in Machine Learning: A Comprehensive Guide. Chiemeka, I. U. and Chineke, T. C. Evaluating the global solar energy potential at Uturu, Nigeria. International Journal of Physical Sciences, 2009; 4(3): 115-119.
11. Chineke, T. C. A robust method of evaluating the solar energy potential of a data sparse site. The Physical Scientist, 2002; 1: 59-69.

12. Chineke, T. C. Equations for estimating global solar radiation in data sparse regions. Renewable Energy, 2008; 33(4): 827-831.
13. Chineke T. C., Okoro U. K. and Igwiro C. E. The imperatives of solar photovoltaic as viable options for rural electrification in Nigeria. International Journal of Natural and Applied Sciences, 2007; 3(2): 193-199.
14. Chiteka, K. and Enweremadu, C. Prediction of global horizontal solar irradiance in Zimbabwe using artificial neural networks. Journal of Cleaner Production, 2016; 135: 701-711.
15. Constante, D. M. A. and Erazo, C. R. R. C. The role of renewable energies in the transition to a sustainable energy model: Challenges and opportunities. Journal of business and entrepreneurial studies, 2024. doi: 10.37956/jbes. v8i3.372
16. Dada, B., and Okogbu, Ec. Estimating Daily Solar Radiation from Monthly Values Over Selected Nigeria Stations for Solar Energy Utilization. Journal of Fundamentals of Renewable Energy and Applications, 2016. doi: 10.4172/2090-4541.1000240
17. Fagbenle, R. L. Total solar radiation estimates in Nigeria using a maximum-likelihood quadratic fit. Renewable Energy, 1993; 3: 813-817.
18. Falayi, E.et al. Empirical models for the correlation of global solar radiation with meteorological data for Iseyin, Nigeria. International Journal of Physical Sciences, 2008; 3: 210-216.
19. Fan, J., Wang, X., Wu L., Zhou, H., Zhang, F., Yu, X. et al. Comparison of Support Vector Machine and Extreme Gradient Boosting for predicting daily global solar radiation using temperature and precipitation in humid subtropical climates: a case study in China, Energy Convers. Manag., 2018; 164: 102–111.
20. Garg, H. and Garg, S. Prediction of global solar radiation from bright sunshine hours and other meteorological data," Energy Conversion and Management, 1983; 23: 113- 118.
21. Goliatt, L., Zaher, Mundher, Yaseen. Development of a hybrid computational intelligent model for daily global solar radiation prediction. Expert systems with applications, 2022. doi: 10.1016/j.eswa.2022.118295
22. Gurel, A.E, Agbulut, U., Bakir, H., Ergun, A., Yildiz, G. A state of art review on estimation of solar radiation with various models. Heliyon at Science Direct., 2023.
23. Halima, D., Camel, T., Djelloul, B. Enhancing ANN Model Perfomance to be used for Solar Radiation Prediction, 2024. doi: 10.1109/sesai61023.2024.10599434
24. Hassan, M.A. Khalil, A. Kaseb, S. Kassem, M.A. Potential of four different machine-learning algorithms in modeling daily global solar radiation, Renew. Energy, 2017; 111:

52–62.

25. Hussain, S. and Al Alili, A. Soft computing approach for solar radiation prediction over Abu Dhabi, UAE: A comparative analysis. Smart Energy Grid Engineering (SEGE), IEEE International Conference on, 2015; 1-6.
26. Ibrahim, S. M. A. Predicted and measured global solar radiation in Egypt. Solar Energy, 1985; 35: 185.
27. Ikemba, S., Song-hyun, K., Temiloluwa, O, Scott., Daniel, R., E., Ewim., S.M., Abolarin., Akeeb, Adepoju, Fawole. Analysis of solar energy potentials of five selected south-east cities in Nigeria using deep learning algorithms, 2024. doi: 10.1186/s40807-023- 00096-7
28. Karasu, S. Altan, A. Sarac, Z. Hacioglu, R. Prediction of solar radiation based on machine learning methods, J. Cogn. Syst., 2017; 2(1): 16–20.
29. Karthikeyan, S., Solomon, G.R., Kumaresan, V., Velraj, R. Parametric studies on packed bed storage unit filled with PCM, 2014.
30. Encapsulated spherical containers for low temperature solar air heating applications. Energy Conversion and Management, 78: 74-80.
31. Landeras, G., Lopez, J.J., Kisi, O., Shiri, J. Comparison of Gene Expression Programming with neuro-fuzzy and neural network computing techniques in estimating daily incoming solar radiation in the Basque Country (Northern Spain). Energy Conversion and Management, 2012; 62: 1-13.
32. Liu, J., H.W., Chen, D., Yu, Q., Wu, D., Haginoya, S. Observation and calculation of the solar radiation on the Tibetan Plateau. Energy Conversion and Management, 2012; 57: 23-32.
33. Maghrabi, A. Parameterization of a simple model to estimate monthly global solar radiation based on meteorological variables, and evaluation of existing solar radiation models for Tabouk, Saudi Arabia. Energy conversion and management, 2009; 50: 2754-2760.
34. Mbali, K. Ensemble Learning in Machine Learning: Bagging, Boosting and Stacking. Mbah, O.M., Chioma, Ifeyinwa, Madueke., Umunakwe, R., Agba. M.N. (2021). Extreme Gradient Boosting: A Machine Learning Technique for Daily Global Solar Radiation, 2024.
35. Forecasting on Tilted Surfaces. Журнал інженерних наук, doi: 10.21272/jes.2022.9(2). e1.
36. Namrata, K., Kumar, M., Kumar, N. Data-Driven Hyperparameter Optimized Extreme Gradient Boosting Machine Learning Model for Solar Radiation Forecasting. Advances in

Electrical and Electronic Engineering, 2023. doi: 10.15598/aeee. v20i4.4650

37. Niklas, D. Random Forest: A Complete Guide for Machine Learning. All you need to know about the random forest model in machine learning, 2024.
38. Nwokocha, C., Chineke, T., Nwofor, O., & Okoro, U. Estimation of solar radiation in Southeastern Nigeria, 2009; 5: 223–228.
39. Ogbodo, J. A., Okeke, F. Spatial analysis of southeastern forest reserves in Nigeria using open geospatial data. African Geographical Review, 2022. doi: 10.1080/19376812.2021.2019069
40. Okore, Ao. Population distribution, land, and livelihood in South-Eastern Nigeria/La repartition de la population, les terres et le mode de vie en Nigeria du Sud-Est., 1982.
41. Okoro U. K., Chineke, T. C., Nwofor, O. K. and Ibe, A. Evapotranspiration levels at Owerri In the Niger Delta Nigeria. International Journal of Natural and Applied Sciences, 2008; 4(2): 168- 173.
42. Olatomiwa, L., Mekhilef, S., Shamshirband, S., Mohammadi, K., Petkovic, D., Sudheer, Ch. A support vector machine–firefly algorithm-based model for global solar radiation prediction. Solar Energy, 2015; 115: 632-644.
43. Olatomiwa, L., Mekhilef, S., Shamshirband, S., Petkovic, D. Adaptive neuro-fuzzy approach for solar radiation prediction in Nigeria. Renewable and Sustainable Energy Reviews, 2015; 51: 1784-1791.
44. Ryzhenkov, A., and Burinova, L. Development of renewable energy sources and their importance for Russia's transition to the standards of a "green" economy. Izvestiâ Saratovskogo universiteta. Novaâ seriâ, 2022. doi: 10.18500/1994-2540-2022-22-4-432-439
45. Salisu, A, Aminu, S. Z, Muhammad, I and Mohammed, A. A. An Artificial Neural Network Model for Estimating Daily Solar Radiation in Northwest Nigeria. FUOYE Journal of Engineering and Technology (FUOYEJET), 2020; 5(2). September 2020 ISSN: 2579-0625 (Online), 2579-0617.
46. Salisu, S., Mustafa, M.W., and Mustapha, M. Predicting Global Solar Radiation in Nigeria Using Adaptive Neuro-Fuzzy Approach, in International Conference of Reliable Information and Communication Technology, 2017; 513-521.
47. Salisu, S, Mustafa, M. W, Mustapha, M and Mohammed, O. O. Solar Radiation Forecasting in Nigeria Based on Hybrid PSO-ANFIS and WT- ANFIS Approach. International Journal of Electrical and Computer Engineering, 2019; 9(2): 3916- 3926. ISSN: 2088-8708.

48. Samani Z. A. and Pessarkli, M. Estimating potential crop evapotranspiration with minimum data in Arizona, Trans. ASAE, 1986; 29: 522-524.
49. Sambo, A. Empirical models for the correlation of global solar radiation with meteorological data for northern Nigeria. Solar & wind technology, 1986; 3: 89-93.
50. Sampson, I., Bankole, A.S., Innocent, W. I. Spatio Temporal Analysis of Land Use Changes in South-Eastern States, Nigeria. International journal of scientific and research publications, 2020. doi: 10.29322/IJSRP.10.04. 2020.P10060
51. Shamshirband, S., Mohammadi, K., Khorasanizadeh, H., Yee, P.L., Lee, M., Petkovic, D., Zalnezhad, E. Estimating the diffuse solar radiation using a coupled support vector machine–wavelet transform model. Renewable and Sustainable Energy Reviews, 2016; 56: 428-435.
52. Sruthi, E. R. (2024). Understand Random Forest Algorithms with Examples (Updated 2024)/ Yadav, A. K., & Chandel, S. S. Solar radiation prediction using Artificial Neural Network techniques: A review. Renewable and sustainable energy reviews, 2014; 33: 772- 781.