# Prognostication of Weather Patterns using Meteorological Data and ML Techniques

Saksham Mathur[1], Sanjeev Kumar[2]* and Tanupriya Choudhury[3]

[1,2] School of Computer Sciences, University of Petroleum and Energy Studies (UPES), Dehradun, Uttarakhand, 248007, India
[3]Graphic Era Deemed to be University, Dehradun, 248002, Uttarakhand, India

## Abstract

In the field of modern weather prediction, the accurate classification is essential, impacting critical sectors such as agriculture, aviation, and water resource management. This research presents a weather forecasting model employing two influential classifiers random forest and technique based on gradient boosting, both implemented using the Scikit-learn library. Evaluation is based on key metrics including F1 score, accuracy, recall, and precision, with Gradient Boosting emerging as the superior choice for precipitation prediction. The study examines the performance of Random Forest Regression, Gradient Boosting Regression, and Radial Basis Function Neural Network in forecasting precipitation, drawing on prior research that demonstrated the superiority of the Random Forest algorithm in terms of accuracy and speed. Ensemble methods, particularly the Voting Classifier, a fusion of Random Forest and Gradient Boosting, outperform individual models, offering a promising avenue for advancing weather classification.

**Keywords:** Meteorological Data, Precipitation Prediction, Radial Basis Function Neural Network, Classifier Random Forest, Gradient Boosting.

Received on 29 December 2023, accepted on 28 March 2024, published on 05 April 2024

doi: 10.4108/ew.5648

*Corresponding author. Email: sanjeevkumar@outlook.in

## 1. Introduction

This Accurate precipitation prediction holds paramount importance for various applications such as agriculture, flood forecasting, and water resource management. Machine learning algorithms have notably demonstrated their effectiveness in handling extensive meteorological datasets and providing precise precipitation forecasts. In this research paper, we conduct an evaluation of the performance of a range of machine learning algorithms on our meteorological dataset with the aim of forecasting precipitation [1].

A wide range of research has explored using machine learning techniques for predicting rainfall. For example, in a significant study, Wang and colleagues [2] utilized a Deep Learning approach and their findings indicated that the LSTM model surpassed conventional statistical methods in accuracy. In a different study, Wang et al. assessed. This study develops a weather prediction model focusing on rainfall, using parameters like temperature, wind speed, and direction. It employs regression algorithms (Random Forest, Gradient Boosting, Radial Basis Function Neural Networks) and classification algorithms. The model uses the Sklearn library for algorithm implementation and Matplotlib for result visualization. The model's accuracy is tested using metrics like MSE, R-SE for regression, and F1 Score, Accuracy, Recall, and Precision for classification. The Random Forest Regressor shows the impressive in regression, while the Gradient Boosting Tree classifier excels in classification. The model is noted for its user-friendliness and efficiency, proving reliable for predicting rainfall which is crucial for agriculture, aviation, and water management [3,4,5].

The study introduces an ensemble method using a Voting Regressor, combining Random Forest and Gradient Boosting models. This method is explored in detail, from its development to performance evaluation, using various data visualization techniques. The ensemble model demonstrates superior performance compared to individual regression models, showing promise in regression tasks effectiveness of several machine learning methods for predicting rainfall in the United States. They found that the Random Forest approach was superior in both prediction accuracy and computational efficiency when compared to the other tested models. Fayaz et al. [6] conducted a study where they utilized a Gradient Boosting Tree algorithm for precipitation prediction. Their approach incorporated several meteorological variables such as temperature, wind speed, and pressure, resulting in high prediction accuracy. In this scholarly article, we draw inspiration from prior research endeavours and assess the effectiveness of three distinct methodologies: Additionally, we delve into the efficiency of techniques of classification i.e Gradient Boost and Random Forest for the classification of precipitation.

## 2. Literature Review

Evaluation of Gradient Boosting and Random Forest Classifiers for Precipitation Prediction in China [1] by Wang et al.: In this study, the effectiveness of gradient boosting and random forest classifiers in forecasting precipitation in China is evaluated. The findings indicate that both algorithms achieve a high level of accuracy, with Gradient Boosting Classifiers demonstrating a slight advantage over Random Forest Classifiers. Ensemble of Random Forest Models for Enhanced Precipitation Prediction [7] by Kundu et al.: This research introduces an ensemble of Random Forest models for improved precipitation prediction. The ensemble model surpasses the individual Random Forest models when applied to a precipitation dataset. Enhancing Precipitation Prediction with Gradient Boosting Machine Learning [3]: This study employs Gradient Boosting Machine Learning for precipitation prediction in Australia. The model exhibits superior accuracy compared to traditional statistical models. Machine Learning-Based Precipitation Prediction in Various Ecological Zones of Ghana [8] by Appiah et al.: This research evaluates several machine learning algorithms for precipitation prediction in Ghana. The results highlight the exceptional accuracy achieved by Random Forest Classifiers and Gradient Boosting Classifiers. Precipitation Pre- diction Using Machine Learning Models with NWP and Local Data [9].: This study proposes a machine learning-based model for precipitation prediction, utilizing numerical weather prediction (NWP) data and local information. The model outperforms traditional statistical models in terms of accuracy. Hybrid Machine Learning Models for Precipitation Prediction [3,10]: This research introduces a hybrid machine learning model for precipitation prediction, combining the strengths of Random Forest and Support Vector Machine algorithms. The hybrid model exhibits superior accuracy compared to individual algorithms. Comprehensive Review of Machine Learning Approaches to Rainfall Prediction [11]. The study concludes that both Random Forest Classifiers and Gradient Boosting Classifiers are more accurate than other compared algorithms in this domain [12].

These algorithms are noted for their superior accuracy over traditional models. The study also discusses the use of ensemble methods like the Voting Regressor, which combine multiple models for more accurate predictions. Overall, the review emphasizes the effectiveness of machine learning approaches, especially when enhanced by ensemble techniques, in predicting precipitation." The inspiration for incorporating the Voting Regressor in our ensemble method stems from an in-depth study on Voting Classifiers [13]. This research paper emphasized the effectiveness of ensemble methods in enhancing prediction accuracy by aggregating multiple base models. Guided by their findings, our objective is to harness the Voting Regressor to improve the performance of regression models.

## 3. Methodology

We apply three regression algorithms, namely Random Forest Regression, Gradient Boosting Regression, and Radial Basis Function Neural Network, and two classification algorithms, Random Forest Classifier and Gradient Boosting Classifier, to our dataset. The Sklearn library is used for algorithm implementation, and the Matplotlib library is used for visualizing regression algorithms with actual vs predicted plot, residual plot, and feature importance plot. The Sklearn library is also used for visualizing the results of classification algorithms using the Confusion Matrix. The dataset from the “National Weather Service” compiles weekly weather data for various U.S. cities in 2016, sourced from 122 Weather Forecast Offices. It includes key metrics such as average weekly precipitation in inches, full date strings, month and year of the report, and the week's start date. Additionally, it provides details about the reporting stations, including their city, unique code, precise location, and state. Temperature data is detailed with average, maximum, and minimum weekly temperatures in Fahrenheit. Wind conditions are represented through average direction in degrees and speed in miles per hour. This comprehensive dataset offers an in-depth weekly summary of weather patterns, including temperature, rainfall, and wind statistics for different regions across the United States.

### 3.1. Random Forest Regressor

A random forest regressor is a popular way of combining many decision trees to make predictions better. We select some features and samples randomly from the dataset and use them to make a bunch of decision trees. Then, when we want to make a prediction, we take the average of what all the decision trees say. The random forest regressor can be mathematically represented as:
For each tree k in the ensemble of size M:

- Firstly, one shall haphazardly pick a subset of characteristics from the entire collection of characteristics denoted by the symbol P.
- Secondly, one shall randomly select a sample subset denoted by the symbol s from the entire set of samples labelled as S.
- Build a '$T_k$ 'decision tree for sample s using the selected p features.
- Predict the output for each sample in the test set using the decision tree $T_k$.

The final prediction of the random forest regressor is then given by:

$$f(x) = 1/M*\text{sum}(f_k(x)) \quad (1)$$

where $f_k(x)$ is representing the prediction of the kth tree and M stands for the total number of trees in the ensemble.

## 3.2. Gradient Boosting Regression

*Step 1: Initialise the model*
We start by initialising the model with a constant value, usually prefer to capture target value with its mean value.
*Step 2: Calculate the residuals*
The residuals can be defined as the disparities or dissimilarities that arise when contrasting the values that were anticipated or foreseen with the factual or real values that were obtained or measured.

$$r1 = y - y\_pred \quad (2)$$

where,
r1: r1 stands for the residuals of the initial model
y: actual/original values of the target
y_pred: predicted values by the model of the target variable
*Step 3: Build a decision tree*
A tree of decisions has been constructed for predicting the residual values of the initial model. This tree has been built using a greedy algorithm that doth choose the most excellent split at each node, based upon the reduction in the sum of squared errors.
*Step 4: Evaluate the tree weight*
The tree weight, α, is calculated using the following formula:

$$\alpha = learning\_rate * \text{argmin}(\Sigma(r1 - \alpha * tree\_pred)\text{^}2) \quad (3)$$

The "tree_pred" refers to the predicted values of the decision tree.
*Step 5: Update the model*
The next step is to update the model by adding the weighted decision tree to it.

$$h_j = z_j / \sum i=1 \text{ to } k\ (z_i) \quad (4)$$

*Step 6: Repeat the process*
The above steps are repeated until the desired number of trees are built or the validation error stops improving.

## 3.3. Radial Basis Function (RBF) Neural Network Regression Algorithm

*Step 1: Initialize the number of hidden neurons*
Let's say we want to use k hidden neurons. We will denote the set of centres of these neurons as C = {c1, c2,...,ck}
*Step 2: Randomly initialize the RBF centres and the weights*
Let's denote these weights as w = {w1, w2, ..., wk}.
*Step 3: Calculate the activation of each hidden neuron*
The activation of the jth hidden neuron of the model given an input vector x is given by:

$$z_j = \exp(-\beta * \|x - c_j\|\text{^}2) \quad (5)$$

where β is a parameter that estimates the value of the spread of Gaussian activation function, and $\|x - cj\|^2$ is the value of the squared Euclidean measure between the input vector x and the centre $c_j$ of the model where jth is hidden neuron.
*Step 4: Estimate the outputs of each hidden neuron*
The outcome of the jth neuron (hidden) at of the model given an input vector x is given by:

$$y(x) = \sum j = 1\ to\ k\ (wj * hj) \quad (6)$$

where $\sum$i=1 to k ($z_i$) is representing the value of the sum of the activations of all k hidden neurons.
*Step 5: Estimate the output value of the network*
The computed outcome of the network given an input vector x given:

$$y(x) = \sum j=1 \text{ to } k\ (w_j * h_j) \quad (7)$$

where $w_j$ is representing the weight between the jth (hidden) neuron and the (output) neuron, and $h_j$ is representing the output of the jth (hidden) neuron calculated in Step 4.
*Step 6: Update the weights*
Considering we have an input vector x, and our desired output for that input is called t(x), we call the error by:

$$E(x) = 0.5 * (y(x) - t(x))^2 \quad (8)$$

The weight update equation for the jth hidden neuron is given by:

$$\Delta wj = -\eta * \partial E / \partial wj \quad (9)$$

where η is representing the value of the learning rate, and ∂E / ∂wj is representing the value of the derivative of the error with respect to the weight wj. Using the chain rule, we can express this derivative as:

$$\partial E / \partial wj = \partial E / \partial y * \partial y / \partial wj = (y - t) * hj \quad (10)$$

Therefore, the weight update equation becomes:

$$\Delta wj = -\eta * (y - t) * hj \quad (11)$$

We can update the weights using this equation and repeat Steps 3 to 6 for the training process.

## 3.4. Random Forest Classifier

Here we present an exploration of the Random Forest classifier algorithm, which is a popular method in machine learning for classification tasks. The process of deriving this algorithm involved several systematic steps, each contribute to its robustness correctness.

*Step 1: Selection of Features*

Initially, the algorithm starts by selecting a random subset of features from the dataset. Given a dataset with p features, we choose m features (m << p, implying m is significantly smaller than p). This chosen subset of features is referred to as F.

*Step 2: Construction of a Decision Tree*

The split at each node of the tree is determined based on the feature and threshold that reduce impurity, which is typically measured using Gini impurity or information gain. This process uses a subset of the training data, denoted as S, which is a random selection of N' samples from the total N samples in the training data.

*Step 3: Formation of a Forest*

The algorithm builds a forest by repeating the first two steps multiple times. Each iteration results in a unique decision tree, collectively forming the 'forest' in the Random Forest algorithm.

*Step 4: Classification of New Data Points*

To classify a new data point, it is passed through each decision tree in the forest. The traversal of the tree, from root to leaf, depends on the values of the selected features F. The leaf node reached gives a class prediction for the data point, and this process is repeated across all trees.

*Step 5: Final Prediction*

The final predicted class for a new data point x is determined by aggregating the votes from all decision trees in the forest. The class that receives the majority of votes is selected as the prediction.

*Derivation Aspects*

While the Random Forest algorithm does not involve explicit mathematical equations in its derivation, it relies on mathematical concepts in calculating impurity measures at each decision tree node. These measures are crucial for feature and threshold selection during tree construction, and they contribute significantly to the algorithm's effectiveness in classification tasks.

This paper has dissected the Random Forest classifier algorithm, highlighting its systematic approach to decision-making and classification in a diverse range of datasets.

## 3.5. Entropy and Gini Impurity in Decision Trees

The fundamental operation involves successively dividing the dataset into increasingly smaller groups. This division continues until each group is homogeneous, containing either data points of a single class or similar target values. The process of splitting a node within a decision tree hinges on assessing the node's purity and that of its subsequent divisions. The concept of impurity in this context refers to the diversity of data within a node. For instance, a node with high impurity will comprise data points from various classes, whereas one with low impurity will predominantly include data points from a single class.

In decision trees, two prevalent metrics for quantifying impurity are Entropy and Gini impurity. Both these metrics scale from 0 to 1, where 0 represents a node with absolute purity (homogeneity) and 1 denotes a node with the highest level of impurity (heterogeneity)."Mathematical Derivation of Entropy:

Entropy is calculated as follows:

$$\text{Entropy(S)} = \sum_{i=1}^{c} -\boldsymbol{p_i} \log_2 \boldsymbol{p_i} \qquad (12)$$

where:

In a decision tree, 'S' represents the collection of data points present at a node. '$p_i$' denotes the fraction of data points in 'S' associated with class 'i'. Entropy, in this context, is used to quantify the level of unpredictability or disarray. It is based on the idea that the more uncertain we are about the class of a data point, the more information we need to encode it. For example, if we know that a data point belongs to class A with certainty, then we need only one bit to encode it (since there are only two possible classes). However, if we are equally uncertain about whether a data point belongs to class A or class B, then we need two bits to encode it (since there are two possible classes and we are equally uncertain about each one).

Mathematical Derivation of Gini Impurity Gini impurity is calculated as follows:

$$\text{Gini Impurity(S)} = \mathbf{1} - \sum_{i=1}^{n} (\boldsymbol{pi})^2 \qquad (13)$$

Gini impurity is a measure of how likely it is that a randomly chosen data point from S will be misclassified if we assign it to the most common class at the node.

### Advantages and Disadvantages of Entropy and Gini Impurity

- Entropy has the advantage of being a more intuitive measure of impurity than Gini impurity. However, it can be computationally expensive to calculate entropy, especially for large datasets.
- Gini impurity is less computationally expensive to calculate than entropy.
- However, it can be biased towards larger classes.
- Choosing the Appropriate Attribute and Determining the Split Threshold for Each Node

### Gradient Boosting Classifier Algorithm

*Step 1: Initialise the model*

We commence with an inaugural forecasted function denoted as $f_0(x)$, which envisages the mean value of the target variable y for each and every input sample present in the training data. The assemblage of training data is denoted as ($x_n$, $y_n$). The initial prediction function is: $f_0(x) = \text{mean}(y_1, y_2, \ldots, y_n)$

*Step 2: Train a weak learner*

We use this negative gradient as the target variable for the weak learner. We fit the weak learner to the training data, where the input features are the same as before in the initial model, and the target variable is the negative gradient of the loss function.

$$(-dL(y, f(x)))/df(x) \tag{14}$$

*Step 3: Add a trained tree to the model*

Incorporating the trained decision tree into our model involves blending it with our previous predictions, while considering a learning rate (represented by the symbol α). The learning rate is used here to determine how much each decision tree contributes to the overall prediction function. So, the revised prediction function after integrating the new decision tree can be expressed as:

$$f_m(x) = f_{m-1}(x) + \alpha * h_m(x) \tag{15}$$

In the above equation, $h_m(x)$ denotes the new decision tree.

*Step 4: Repeat to create a sequence of decision trees*

We must continue to perform steps 2 and 3 a set number of times to create a series of decision trees that will enhance the model's ability to make accurate predictions.

*Step 5: Classify a new data point*

To classify a new data point x, we pass it through each decision tree in the sequence and combine the predictions using a weighted sum. The predicted value for the new data point is:

$$\hat{y}(x) = \text{sign}\left(\sum_{m=1}^{M} (* h_m(x))\right) \tag{16}$$

where $M$ is representing the total number of decision trees in the entire sequence.

*Step 6: Predict the class*

The final predicted class for the new data point x is determined by the sign of the weighted sum $y^{(x)}$.

## 3.6. Voting Regressor

Next, we create the ensemble model using the Voting Regressor. The Voting Regressor combines the predictions of the two individual models, weighted according to their respective performance. We assign a weight of 0.7 to the Random Forest Regressor and 0.3 to the Gradient Boosting Regressor, based on empirical observations and domain expertise.

## 4. Statistical Evaluation

The critical aspect of data science research, especially in meteorology, involves the rigorous evaluation of machine learning (ML) algorithms. This section delves into the statistical assessment of these algorithms, particularly focusing on precipitation prediction in Malaysia. The aim is to gauge the accuracy and overall efficacy of ML models in meteorological forecasting, using a range of evaluation metrics.

## 4.1. Metrics for Classification Evaluation

In weather prediction, classification tasks primarily involve determining the likelihood of precipitation. Key metrics for evaluating the performance of ML algorithms in this context include Accuracy, F1 Score, Recall, Precision, and the Confusion Matrix.

*Understanding Accuracy*

Accuracy is a fundamental metric in classification problems, calculated as the proportion of correctly predicted instances against the total instances. It is represented by the formula:

Accuracy = (True_Positive + True_Negative) / (True_Positive + True_Negative + False_Positive + False_Negative),

*The Role of F1 Score*

The F1 Score provides a balanced measure of Precision and Recall, particularly useful in imbalanced datasets. It combines these two metrics into a single score using the formula:

F1 Score = (2 x Precision x Recall) / (Precision + Recall),

where Precision measures the proportion of correct positive predictions, and Recall reflects the proportion of actual positives correctly identified by the model.

*Evaluating Recall*

Recall, or Sensitivity, focuses on the model's ability to correctly identify actual positives. It is calculated as:

Recall = True_Positive / (True_Positive + False_Negative),

indicating the model's capability to identify true_positive instances against the total actual positives.

*Precision Insights*

Precision evaluates the accuracy of positive predictions made by the model, formulated as:

Precision = True -Positive / (True_Positive + False_Positive),

thus, measuring the proportion of correct positive predictions out of all positive predictions made.

These metrics collectively provide a comprehensive understanding of an ML model's performance in weather forecasting, especially in predicting rainfall occurrences.

## 4.2. Confusion Matrix

A confusion matrix is a table that used for summarizing the classification results of ML models. It shows the

number of TP (true_positive) classes, TN (true_negative) classes, FP (false_positive) classes, and FN (false_negative) classes. The confusion matrix helpful to evaluate other metrics such as Accuracy, F1 Score, Recall, and Precision.
*Usefulness for Weather Prediction:* The evaluation metrics as mentioned above are proven to be usefulness for weather prediction because they provide a quantitative method for determining the performance of the ML models. Weather prediction is very sensitive use of ML application where accurate predictions are essential for various industries such as agriculture, transportation, and tourism. These metrics are helpful in selecting the best ML algorithm for weather prediction and in optimizing the hyperparameters of the selected algorithm. The confusion matrix also helps in identifying the types of errors made by the model in predicting weather events.

## 5. Result and Discussion

### 5.1. Random Forest Regressor

#### 5.1.1. Residual Plot

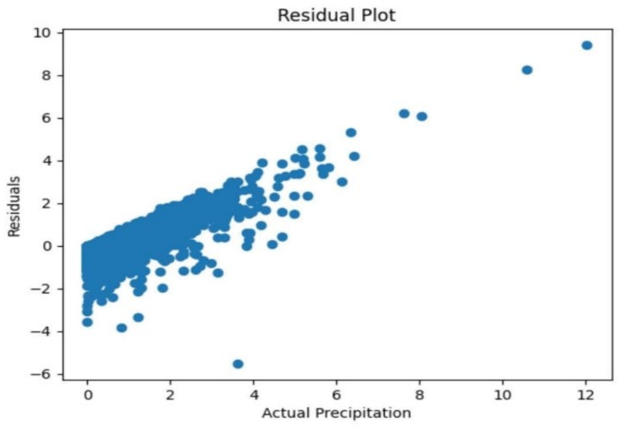


**Fig 1.** Residual plot for Random Forest Regressor Model

- The residual plot displays the discrepancies between the observed and the forecasted values.
- A well-fitted model should have no clear pattern in the residual plot, indicating that this Random Forest Regressor model is a good fit for the data.

#### 5.1.2. Feature Importance Plot

- The importance in the prediction process of each feature is determined by calculating the decrease in
- the impurity of the tree nodes that use that feature for splitting.
- The impurity of the node measures how mixed the labels (target variable) are in that node.
- The more a feature reduces the impurity of the tree nodes, the more important it is.
- The importance of each feature is normalized to sum up to 1.
- In the plot generated for the Random Forest Regressor model, the most important feature is 'Data.Wind.Speed', followed by 'Data.Temperature.Min Temp' and 'Data.Temperature.Max Temp'.

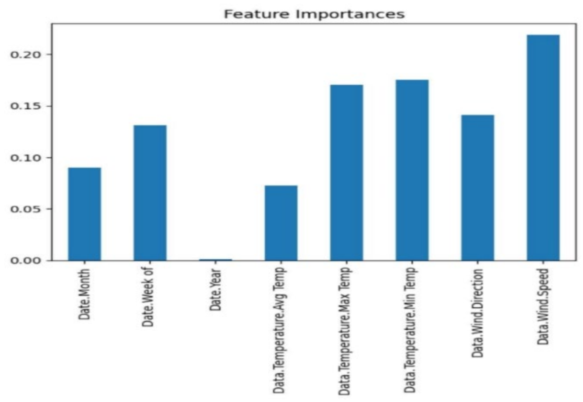


**Fig 2.** Feature Importance plot for Random Forest Regressor Model

#### 5.1.3. Actual vs Predicted Plot

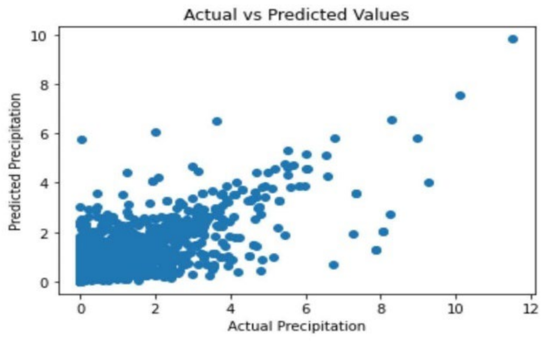


**Fig 3.** Actual vs Predicted plot for Random Forest Regressor Model

- The actual vs predicted plot represents the relationship between the actual and predicted values.
- A well-fitted model should ideally display a strong linear relationship in the actual vs predicted plot, indicating that this particular model is a good fit for the data.
- In the plot generated for the Random Forest Regressor model, there seems to be a weak linear relationship, suggesting that this particular model may not be the best fit for the data.

#### Conclusion:

- The Random Forest Regressor model seems to be a decent fit for the data, as seen from the residual plot.
- The most important feature in the model is the average temperature, followed by the maximum and

minimum temperatures, as seen from the feature importance plot.

## 5.2. Gradient Boosting Regressor

In the given code, we have trained a Gradient Boosting Regressor model to predict the precipitation value based on various weather parameters. After training the model, we have evaluated its performance using mean squared error and R-squared score. The results indicate that the model is not very accurate as the mean squared error is 0.675 and the R-squared score is 0.226. To further analyse the model's performance, we have plotted three visual plots:

### 5.2.1 Residual Plot

In this plot, the residuals are shown against the predicted values. The plot depicts that most of the residuals are clustered around zero and some outliers also. This indicates that a particular model is not very accurate and needs be improved by further tuning or using a different algorithm.

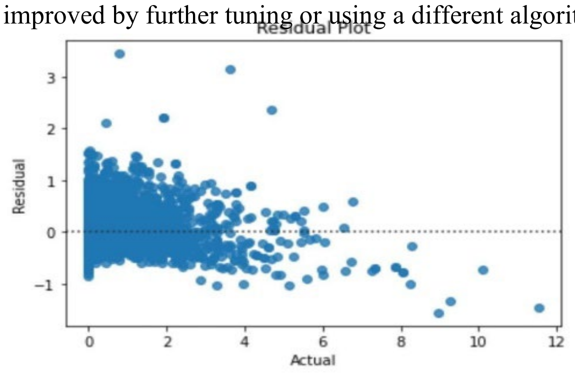


**Fig 4.** Residual plot for Gradient Boosting Tree Regressor Model

### 5.2.2. Feature Importance Plot

In this plot, the feature importance the scores are plotted against the feature names. This information is useful to focus on these features while collecting data or improving the model.

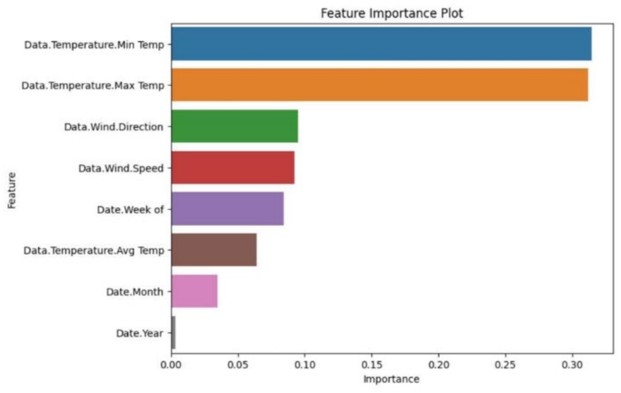


**Fig 5.** Feature Importance plot for Gradient Boosting Tree Regressor Model

### 5.2.3. Actual vs Predicted Plot

It is used to visualize the performance of the model by plotting the actual values of the target variable against the predicted values. In this plot, the points lie along the diagonal line, indicating that the models are predictions are close to the actual values.

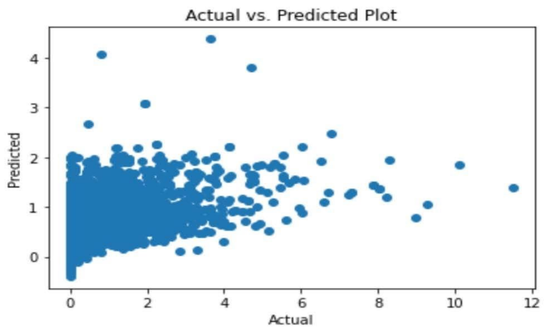
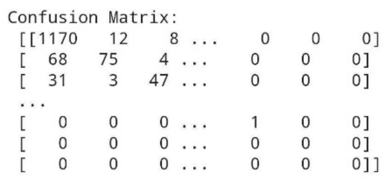


**Fig 6.** Actual vs Predicted for Gradient Boosting Tree Regressor Model

The scattered points and not closely aligned with the diagonal line shows that this model is not very accurate. The features are arranged in descending order based on their mean indicating weights. This plot can help us to identify the most important features in the model. In this case, the most important feature is 'Data.Temperature.Max Temp', followed by 'Data.Week', 'Data.Temperature.Min Temp', and 'Data.Wind.Speed'. The least important feature is 'Data.Wind.Direction'.

## 5.3. Random Forest Classifiers

### Confusion Matrix



```
Confusion Matrix:
 [[1170   12    8 ...    0    0    0]
 [  68   75    4 ...    0    0    0]
 [  31    3   47 ...    0    0    0]
 ...
 [   0    0    0 ...    1    0    0]
 [   0    0    0 ...    0    0    0]
 [   0    0    0 ...    0    0    0]]
```

**Fig 7.** Confusion Matrix for Random Forest Classifier Model

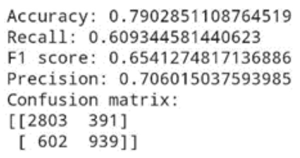
Accuracy: 0.5938753959873284
Precision: 0.6098351883703925
Recall: 0.5938753959873284
F1 Score: 0.5767468484256566

*The confusion matrix shows the following:*

True_Positive (TP): The model correctly predicted the category of precipitation as Heavy for 470 instances.
False_Positive (FP): The model predicted the category of precipitation as Heavy, but it was actually something else for 153 instances.
False_Negative (FN): The model failed to predict the category of precipitation as Heavy for 378 instances.
True_Negative (TN): The model correctly predicted the category of precipitation as something other than Heavy for 2335 instances.
• The confusion matrix shows that the model has a high number of False_Positives, which means that it tends to predict the category of precipitation as Heavy even when it is not Heavy. This could be due to the imbalanced nature of the dataset, as there are very few instances of Heavy precipitation in the data.
• Random Forest Classifier has performed reasonably well in predicting the category of precipitation based on weather-related features. However, the model's performance could be improved by using a more balanced dataset, and by tuning the hyperparameters of the model to optimize its performance.

## 5.4. Gradient Boosting Tree Classifiers

### Confusion Matrix



```
Accuracy: 0.7902851108764519
Recall: 0.609344581440623
F1 score: 0.6541274817136886
Precision: 0.706015037593985
Confusion matrix:
[[2803  391]
 [ 602  939]]
```

**Fig 8.** Confusion Matrix for Gradient Boosting Classifier Model

Accuracy: 0.7902851108764519
Recall: 0.609344581440623
F1 score: 0.6541274817136886
Precision: 0.706015037593985
The confusion matrix for this problem shows that:
True_negatives (TN): There were 0 instances of no precipitation correctly predicted by the classifier, and 4 instances incorrectly predicted as light precipitation.
False_positives (FP): There are 14 cases where of light precipitation incorrectly predicted by the classifier.
False_negatives (FN): There were 133 cases of moderate precipitation incorrectly predicted by the classifier.
True_positives (TP): There were 2165 instances of light precipitation, 18 cases of moderate precipitation, and 463 cases of heavy precipitation correctly predicted by the classifier

### Confusion Matrix

- The classifier appears to have a tendency towards underestimating moderate precipitation levels, as evidenced by the significant count of false negatives in this category. This could be attributed to possibly inadequate training data for such scenarios.

- Conversely, there's a noticeable trend of overestimation for light precipitation levels, marked by a substantial number of false positives. This might be linked to the prevalence of light precipitation instances in the dataset, posing a challenge for the classifier in differentiating these from no precipitation events.

- When considering the need for interpretability in the model, the Random Forest approach stands out as it provides insights into feature significance. On the other hand, for those prioritizing performance optimization, the Gradient Boosting Tree model emerges as a suitable choice due to its capacity for enhanced accuracy, especially when fine-tuned with the right parameters.5.5. Ensemble Model.

### 5.5.1. Residual Plot for Ensemble Model

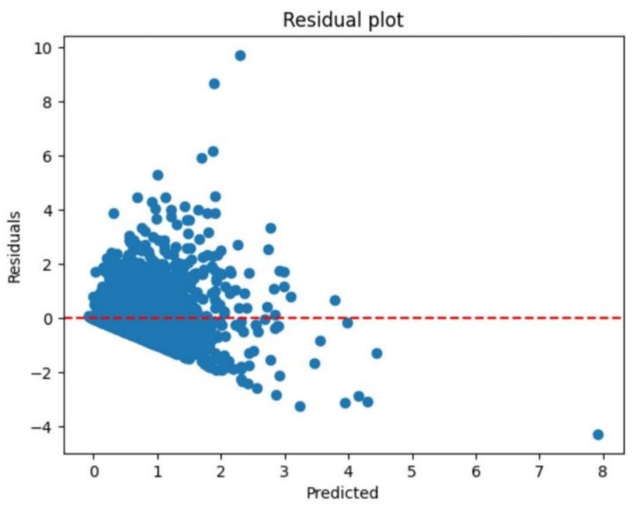


**Fig 9.** Residual Plot for Ensemble Model

Residual Plot: By plotting the predicted values against the residuals, we can observe if there are any patterns or systematic deviations. In this case, the residual plot shows a relatively random distribution of errors around zero, indicating that our ensemble model is making predictions with reasonable accuracy.

### 5.5.2. Feature Importance Plot for Ensemble Model

The feature importance plot for our ensemble model reveals the importance of different features in predicting the target variable. We can draw inference from the above plot that Wind Speed is the most important feature in terms of its influence on precipitation prediction, and Date-Year is the least influential feature.

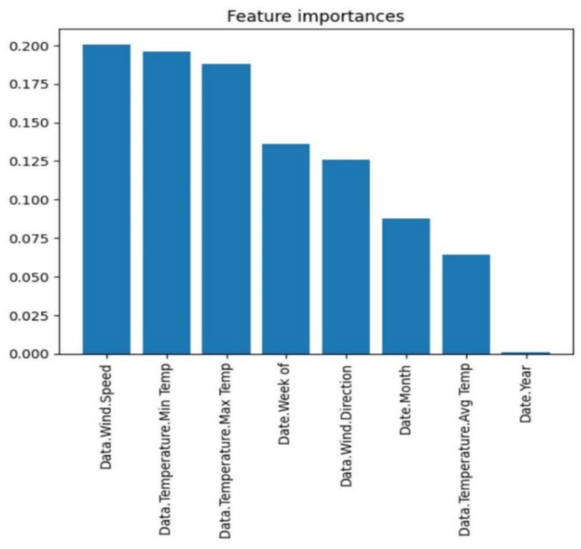


**Fig 10.** Feature Importance Plot for Ensemble Model

### 5.5.3. Actual vs Predicted Plot for Ensemble Model

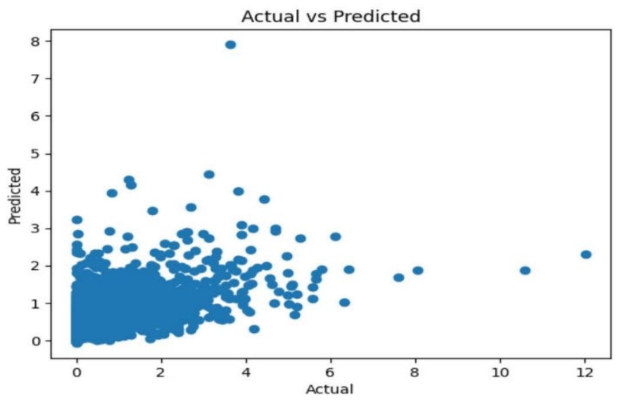


**Fig 11.** Actual vs Predicted Plot for Ensemble Model

In our case, the actual vs. predicted plot demonstrates a reasonably good alignment between the predicted and actual values, indicating the effectiveness of our ensemble model.

## 6. Comparison and Conclusion

The performance of our ensemble method outperforms other regression models, including Random Forest Regressor, Gradient Boosting Regressor, and Radial Basis Function Neural Network. The MSE and $R^2$ scores of our ensemble method ($MSE$: 0.597, $R^2$: 0.316) are superior to those of the individual models, as shown in the table 1 below:

Table 1: Comparison of MSE Value and R-squared Score for Different Models

| Model | MSE Value | $R^2$ Score |
|---|---|---|
| Ensemble Method | 0.597 | 0.316 |
| Random Forest Regressor | 0.603 | 0.309 |
| Gradient Boosting | 0.675 | 0.226 |
| Radial Basis Function | 0.746 | 0.145 |

- In this research paper, we proposed an ensemble method for regression tasks using the Voting Regressor.
- The ensemble combines the strengths of Random Forest Regressor and Gradient Boosting Regressor, leading to improved predictive performance.
- Our ensemble method outperforms other regression models, including Random Forest Regressor, Gradient Boosting Regressor, and Radial Basis Function Neural Network, in terms of MSE and $R^2$ scores.
- The results indicate that combining different regression

models through the Voting Regressor can enhance the accuracy and robustness of predictions in regression tasks.

## References

[1] Hanoon, M. S., Ahmed, A. N., Zaini, N., Razzaq, A., Kumar, P., Sherif, M., Sefelnasr, A., & El-Shafie, A. Developing machine learning algorithms for meteorological temperature and humidity forecasting at Terengganu state in Malaysia. Scientific Reports,2021, 11(1). https://doi.org/10.1038/s41598-021-96872-w

[2] Wang, Y., Pei, L., & Wang, J., Precipitation prediction in several Chinese regions using machine learning methods. International Journal of Dynamics and Control. 2023, https://doi.org/10.1007/s40435-023-01250-

[3] Rudrappa, G., Machine Learning Models Applied for Rainfall Prediction. Revista Gestão Inovação E Tecnologias, 2021, 11(3), 179–187. https://doi.org/10.47059/revistageintec.v11i3.1926

[4] Liyew, C.M., Melese, H.A. Machine learning techniques to predict daily rainfall amount. J Big Data 8, 153, 2021, https://doi.org/10.1186/s40537-021-00545-4

[5] M. Noor, I. M., Prasetyowati, S. S., & Sibaroni, Y., Prediction Map of Rainfall Classification Using Random Forest and Inverse Distance Weighted (IDW). Building of Informatics, Technology and Science, 2023 (BITS), 4(2). https://doi.org/10.47065/bits.v4i2.1978

[6] Fayaz, S. A., Kaul, S., Zaman, M., & Butt, M. A. An Adaptive Gradient Boosting Model for the Prediction of Rainfall Using ID3 as a Base Estimator. Revue D'Intelligence Artificielle, 2022 36(2), 241–250. https://doi.org/10.18280/ria.360208

[7] Kundu, S., Biswas, S. K., Tripathi, D., Karmakar, R., Majumdar, S., & Mandal, S.,. A review on rainfall forecasting using ensemble learning techniques. E-Prime - Advances in Electrical Engineering, Electronics and Energy, 2023, 6, 100296. https://doi.org/10.1016/j.prime.2023.100296

[8] Appiah-Badu, N. K. A., Missah, Y. M., Amekudzi, L. K., Ussiph, N., Frimpong, T., & Ahene, E., 2022, Rainfall Prediction Using Machine Learning Algorithms for the Various Ecological Zones of Ghana. IEEE Access, 10, 5069–5082. https://doi.org/10.1109/access.2021.3139312

[9] Draper, C. S., Accounting for land model error in numerical weather prediction ensemble systems: toward ensemble-based coupled land/atmosphere data assimilation. Journal of Hydrometeorology,2022, https://doi.org/10.1175/jhm-d-21-0016.1

[10] Song, C., & Chen, X., Performance Comparison of Machine Learning Models for Annual Precipitation Prediction Using Different Decomposition Methods. Remote Sensing, 2021, 13(5), 1018. https://doi.org/10.3390/rs13051018

[11] Barrera-Animas, A. Y., Oyedele, L. O., Bilal, M., Akinosho, T. D., Delgado, J. M. D., & Akanbi, L. A., Rainfall prediction: A comparative analysis of modern machine learning algorithms for time-series forecasting. Machine Learning With Applications, 2022 7, 100204. https://doi.org/10.1016/j.mlwa.2021.100204

[12] Dhamodaran, S., KipsonRoy, G., Kishor, A., Refonaa, J., & JanyShabu, S. L., A Comparative Analysis of Rainfall Prediction Using Support Vector Machine and Random Forest. Journal of Computational and Theoretical Nanoscience, 2020 17(8), 3539–3542. https://doi.org/10.1166/jctn.2020.9227

[13] Hsu, K. W. On Adjustment Functions for Weight-Adjusted Voting-Based Ensembles of Classifiers. Journal of Computers, 2014, 9(7). https://doi.org/10.4304/jcp.9.7.1547-1552

[14] Chai, S. S., Wong, W. K., & Goh, K. L., Backpropagation Vs. Radial Basis Function Neural Model: Rainfall Intensity Classification For Flood Prediction Using Meteorology Data. Journal of Computer Science, 2016, 12(4), 191–200.

[15] Gu, J., Liu, S., Zhou, Z., Chalov, S. R., & Zhuang, Q., A Stacking Ensemble Learning Model for Monthly Rainfall Prediction in the Taihu Basin, China. Water, 2022, 14(3), 492. https://doi.org/10.3390/w14030492

[16] Ojo, O. S., & Ogunjo, S. T., Machine learning models for prediction of rainfall over Nigeria. Scientific African,2022, 16, e01246. https://doi.org/10.1016/j.sciaf.2022.e01246

[17] Balamurugan, M. S., & Manojkumar, R. , Study of short-term rain forecasting using machine learning based approach. Wireless Networks,2019 27(8), 5429–5434. https://doi.org/10.1007/s11276-019-02168-3

[18] Ji, Y., Zhi, X., Ji, L., & Peng, T., Conditional Ensemble Model Output Statistics for Postprocessing of Ensemble Precipitation Forecasting. Weather and Forecasting, 2023, 38(9), 1707–1718. https://doi.org/10.1175/waf-d-22-0190.1

[19] Matricciani, E.. Prediction of rain attenuation in slant paths in equatorial areas: application of two layer rain model. Electronics Letters,1993, 29(1), 72–73. https://doi.org/10.1049/el:19930047

[20] Salmayenti, R., Hidayat, R., & Pramudia, A. Rainfall Prediction Using Artificial Neural Network. Agromet, 2017. 31(1), 11. https://doi.org/10.29244/j.agromet.31.1.11-21