IJCRT.ORG ISSN : 2320-2882
INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

Rainfall Prediction Using Machine Learning Techniques

Dipanjan Mondal[1] , Arpan Sen[2] , Dhairya Mohan Jha[3]

Javed Ansari[4] , Ranjana Ray[5]

[1, 2,3,4,5]Department of Electronics and Communication Engineering (ECE)

JIS College of Engineering , Kalyani , West Bengal , India

Abstract: Rainfall is of utmost importance in agriculture, but predicting it accurately has become a significant challenge in recent times. Accurate forecasts of precipitation provide valuable insights in advance, enabling individuals to take necessary precautions and devise better crop strategies. The impact of global warming on nature and humanity has accelerated changes in climatic conditions, resulting in adverse effects such as flooding and drought. As a result, accurately predicting rainfall has become one of the best techniques to understand the climate. However, forecasting precipitation is a demanding and uncertain task that significantly impacts human society. Precise and timely predictions can proactively mitigate potential human and financial losses. To address this issue, a series of experiments were conducted using prevalent machine learning methods to construct models that anticipate whether it will rain the following day based on weather data for that day in major Australian cities. The outcomes offer a comparison of diverse evaluation metrics of these machine learning approaches and their capability to predict precipitation by scrutinizing the weather data. The study aims to provide accurate climate descriptions to clients from various domains such as agriculture, research, power generation, etc., to comprehend the need for climate transformation and its parameters, such as temperature, humidity, precipitation, and wind speed, which ultimately lead to rainfall predictions. It is essential to note that predicting rainfall is a challenging task that also depends on geographic location.

Keywords: rainfall prediction, machine learning, dataset, visualization, classifiers, model evaluation

I. Introduction

The prediction of rainfall remains a pressing concern and has garnered the attention of various entities, including governments, industries, risk management organizations, and the scientific community. Rainfall is a crucial climatic factor that significantly impacts numerous human activities, such as agricultural production, construction, power generation, forestry, and tourism. Therefore, accurate rainfall prediction is very important as it has the highest correlation with adverse natural events like landslides, flooding, mass-movements, avalanches, and other hazards which have always troubled humans for ages. Having an appropriate approach for rainfall prediction enables the implementation of preventive and mitigation measures for these natural phenomena. To address this uncertainty, we employed various machine learning techniques and models to make precise and timely predictions. This study aims to implement machine learning models and evaluate their performance. Data pre-processing includes handling missing values and feature selection. We then employed different machine learning models on this data, using evaluation metrics like Accuracy, Root Mean Squared error, and Mean Absolute error.

II. Data Overview and Methodology

For our project, we trained the classifiers using Australian weather data gathered from different Australian weather stations. The target variable is 'RainTomorrow,' which represents whether rain will fall the next day as a simple Yes or No. The dataset has been taken from: https://www.kaggle.com/jsphyg/weather-dataset-rattle-package, and the definitions were taken from: http://www.bom.gov.au/climate/dwo/IDCJDW0000.shtml. The dataset has 23 features and 142k instances in total.

Here, the methodology includes five major components as shown in fig. below-

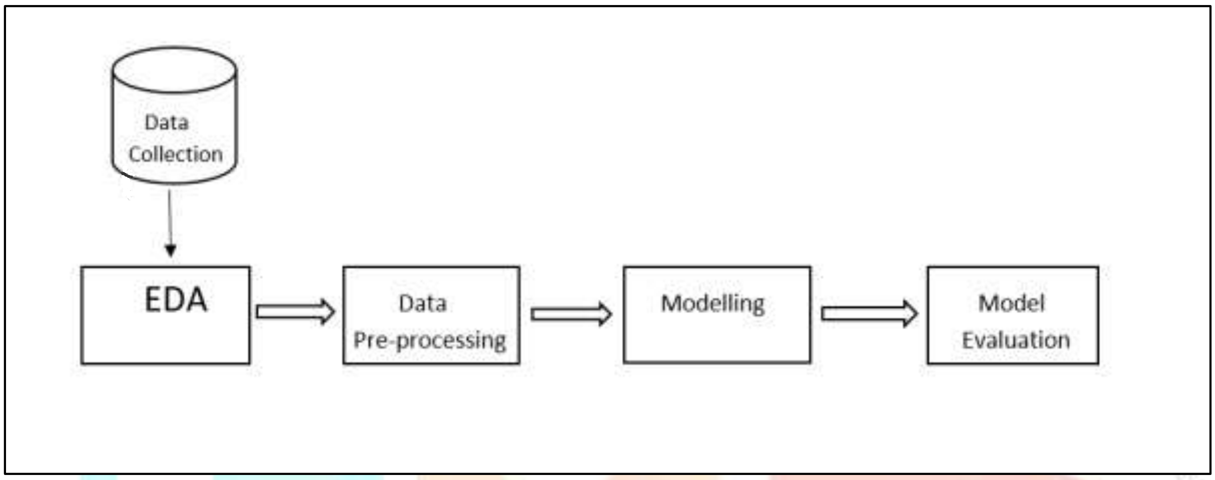

Fig. 1. Complete Architecture

1. Data Exploration and Analysis

Exploratory Data Analysis (EDA) is a powerful data analysis methodology that relies on visual techniques to explore data. The main objective of EDA is to uncover trends, patterns, and validate assumptions using statistical summaries and graphical representations. Through the use of EDA, data analysts can gain a deeper understanding of their data, enabling them to make informed decisions based on the insights revealed. In particular, EDA can prove highly beneficial in machine learning applications, as it helps to boost confidence in the accuracy, validity, and applicability of future findings to the business contexts of interest. However, it is important to note that this level of certainty can only be achieved by thoroughly validating raw data, checking for anomalies, and ensuring that the data set has been collected without errors. Furthermore, EDA can also help to identify insights that were previously unknown or overlooked by both researchers and business stakeholders, making it a valuable tool for data analysis in various contexts.

Here, EDA was performed using two methods - Univariate Visualization which provides the univariate distribution of data i.e., data distribution of a particular variable against the density distribution in the raw data set and Pairwise Correlation Matrix which is used to understand interactions/relations between different fields in the data-set.

Below a table of total number of non-null data in every variable(feature) has been displayed.

```
 #  Column          Non-Null Count    Dtype
---  ---------       --------------------  -------
 0   Date            145460 non-null   object
 1   Location        145460 non-null   object
 2   MinTemp         143975 non-null   float64
 3   MaxTemp         144199 non-null    float64
 4   Rainfall        142199 non-null   float64
 5   Evaporation     82670 non-null    float64
 6   Sunshine        75625 non-null    float64
 7   WindGustDir     135134 non-null   object
 8   WindGustSpeed   135197 non-null   float64
 9   WindDir9am      134894 non-null   object
 10  WindDir3pm      141232 non-null   object
 11  WindSpeed9am    143693 non-null   float64
 12  WindSpeed3pm    142398 non-null   float64
 13  Humidity9am     142806 non-null   float64
 14  Humidity3pm     140953 non-null   float64
 15  Pressure9am     130395 non-null   float64
 16  Pressure3pm     130432 non-null   float64
 17  Cloud9am        89572 non-null    float64
 18  Cloud3pm        86102 non-null    float64
 19  Temp9am         143693 non-null   float64
 20  Temp3pm         141851 non-null   float64
 21  RainToday       142199 non-null   object
 22  RainTomorrow    142193 non-null   object
```

From the above table the following features are irrelevant-

- Evaporation
- Sunshine
- Cloud9am
- Cloud3pm

Statistical tests are conducted to identify and select only those features that exhibit the most robust correlation with the output variable. Here, distplot() function was used from the seaborn library which accepts the data variables(instances) as arguments and returns the plot with the density distribution for each.

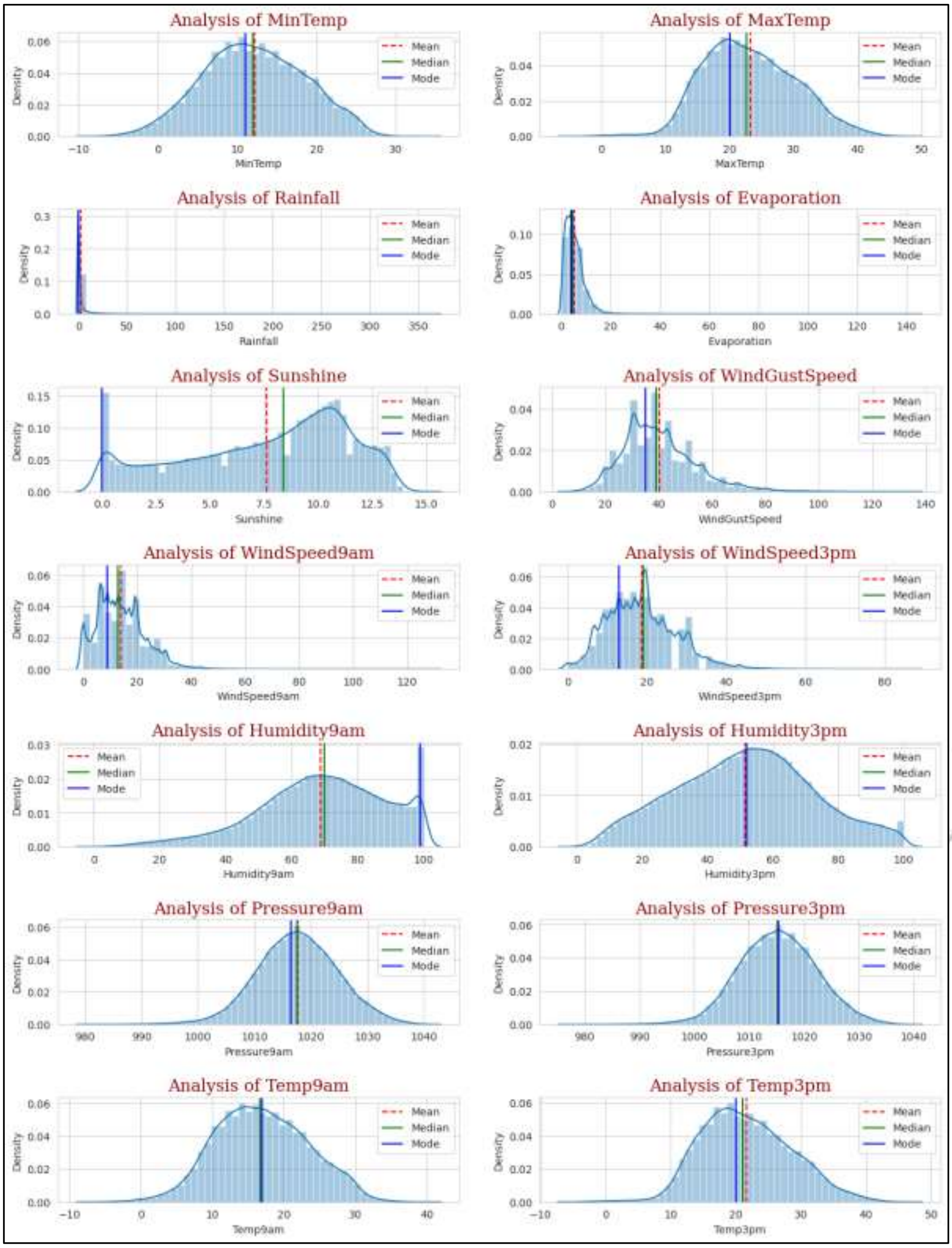

Fig. 2. Univariate Visualization of continuous features

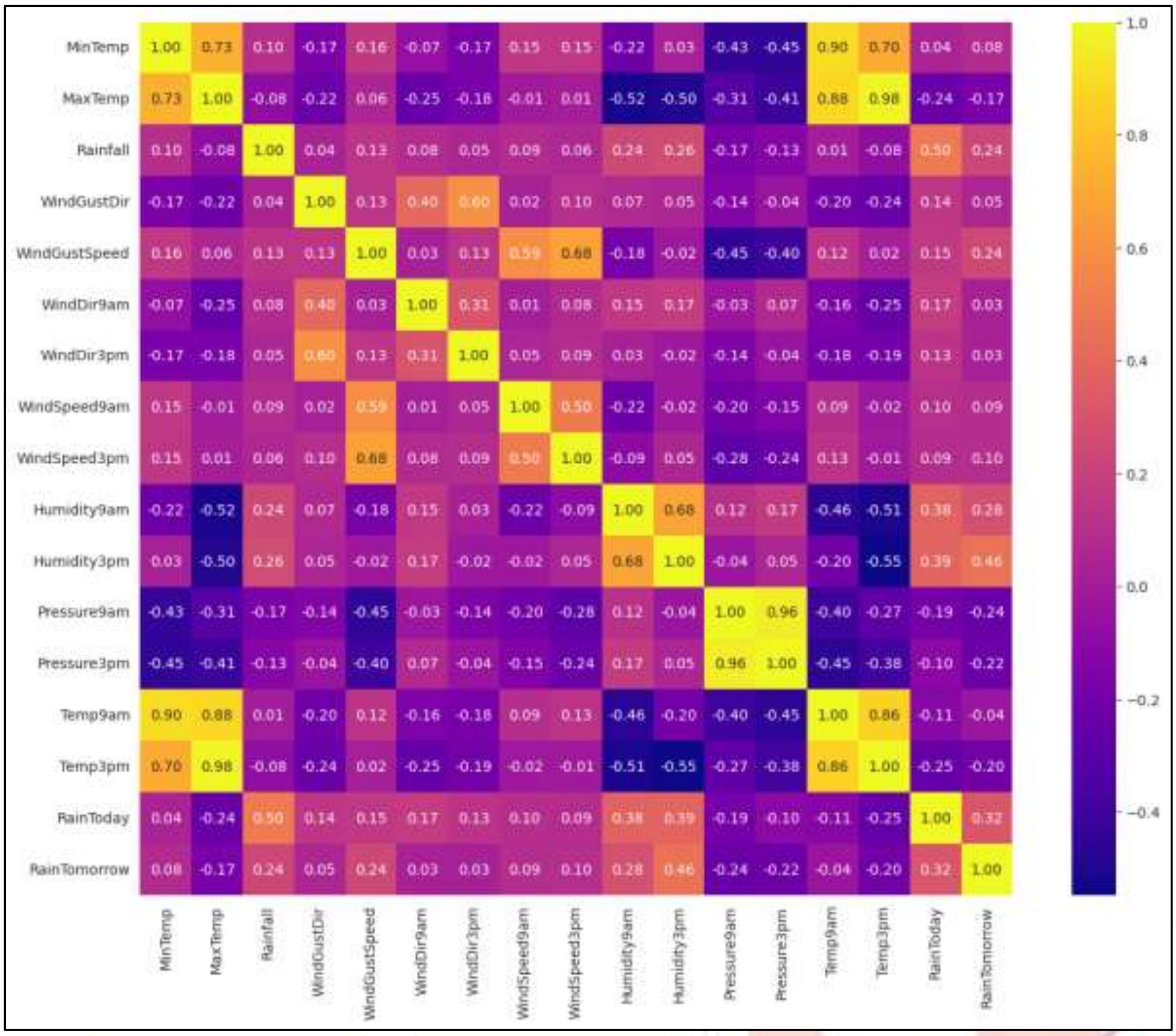

Fig. 3. Heatmap

Correlation reveals how the features are related to each other or the target variable. Correlation can be either positive (increase in one value of the feature increases the value of the target variable) or negative (increase in one value of the feature decreases the value of the target variable). Heatmaps help identify features that are most related to the target variable, here the heatmap of correlated features was plotted using the seaborn library. From the above correlation matrix, it can be seen that the features- MaxTemp, Pressure9am, Pressure3pm, Temp9am and Temp3pm are all negatively correlated with the target variable.

2. Data Pre-processing

Data pre-processing is a crucial technique in data mining that involves converting raw data into a format that is easy to understand and analyze. Real-world data is often incomplete, inconsistent, or lacking in particular patterns, making it difficult to work with. Additionally, data can contain errors that may affect the accuracy of the analysis. To address these issues, pre-processing is carried out to ensure that the data is in a suitable format for analysis, enabling us to extract meaningful insights and make informed decisions based on the provided data.

In the earlier EDA step, we learned that we have quiet a few instances with null values. Hence, we need to handle the missing data. It is done by getting rid of the instances containing highest proportions of missing values. We are aware of the fact that this is not the most efficient technique to handle missing values, but this process saves a lot of time and effort.

The next step is Feature Selection where we select those features which contribute the most to our prediction variable or output variable. Incorporating irrelevant features in the data can substantially influence the accuracy of machine learning models and cause the model to learn based on irrelevant or insignificant features.

3. Modelling

The pre-processed data was segregated into 80% training and 20% for testing.

We chose different classifiers each belonging to a different machine learning model family. All the classifiers used in this project were implemented using scikit-learn. Below are the classification algorithms that have been used to build our prediction models:

Logistic Regression is a classification algorithm used in supervised learning that aims to forecast the likelihood of a particular target variable. This model is particularly helpful in predicting binary outcomes, such as Yes/No or True/False, based on a series of independent variables. Logistic regression can also be viewed as a variant of linear regression in which the outcome variable is categorical, and the log of the odds is utilized as the dependent variable. In essence, this model forecasts the probability of an event occurring by adjusting the data to a logit function, making it well-suited for binary classification problems like ours.

Decision Tree algorithm is a versatile classification tool that can effectively handle both categorical and numerical data. It operates by creating a tree-like structure, which presents data in a graphical form that is easy to analyze. By utilizing the most important indicators, this algorithm splits the data into two or more related sets. The process begins with the calculation of the entropy of each attribute. Next, the data is divided, with the predictors having the maximum information gain or minimum entropy. The results obtained from this approach are easy to read and interpret. Compared to other algorithms, the Decision Tree algorithm offers higher accuracy, thanks to its ability to analyze datasets in a tree-like graph. This characteristic makes it a suitable choice for our problem, given that our target variable is a binary categorical variable.

Random Forest algorithm is a type of supervised machine learning model that uses an ensemble approach. This approach combines multiple weaker learners to form a single strong predictor. The Random Forest model is composed of a collection of decision trees, which is known as a forest. When a new object is classified based on its attributes, each decision tree in the forest produces a classification. Each tree's classification is then considered a vote for that particular class. The final prediction for the object is determined by the forest, which selects the class that receives the most votes across all of the trees in the forest.

K-Nearest Neighbor (KNN) algorithm is both a lazy algorithm and an non-parametric learning method. As a non-parametric method, it does not make assumptions about the underlying data distribution and instead determines the model structure from the dataset. KNN generally performs better with a lower number of features, as increasing the number of features can result in the need for more data and lead to overfitting. To mitigate this issue, we have performed feature selection to reduce dimensionality, making KNN a strong candidate for our problem. In our model we experimented with values of n ranging from 3 to 30 and it was found that the model performs best with n as 27.

Support Vector Machine (SVM) is a machine learning algorithm that falls under the category of supervised learning. It can be used for both regression and classification tasks, but it is most effective in classification. The primary objective of SVM is to identify a hyperplane in an N-dimensional space that accurately separates the data points into different classes. The dimensions of the hyperplane is proportional directly to the number of features/instances present in the given data. SVM algorithms are recognized for their exceptional ability to detect the optimal separating hyperplane between distinct classes within the target feature. This hyperplane is influenced by data points known as support vectors, which are positioned closer to the hyperplane. using these support vectors one can maximize the margin of the classifier. However, deleting these support vectors would alter the position of the hyperplane. Thus, these data points play a crucial role in building our SVM model.

Naïve Bayes is a simple method to build classifiers, which are models that assign labels to problem instances represented as feature vectors, where the labels belong to a finite set. While there is no singular algorithm for training classifiers, there exists a set of algorithms that follow a shared principle. Naïve Bayes classifiers operate under the assumption that the value of a specific feature is independent of the value of any other feature, provided the class variable is constant. The name "Naïve Bayes" comes from the fact that it is based on two concepts: Naïve, because it assumes that the occurrence of a feature is independent of the occurrence of other features. And Bayes, because it relies on the Bayes' Theorem principle.

Stochastic Gradient Descent (SGD) is a highly effective way of fitting linear classifiers and also regressors whenever one deals with convex loss functions like (linear) SVMs and Logistic Regressions in machine learning. In each iteration of SGD, only a single sample, or a batch size of one, is utilized for training. The chosen sample is randomly selected and shuffled for each iteration. The Stochastic Gradient Descent algorithm calculates the gradient and updates the parameters using only one randomly selected training example at each iteration. SGD has demonstrated successful implementation in large-scale and sparse machine learning problems, often encountered in natural language processing and text classification. Due to sparsity of data, classifiers used in this module are very easily scalable to problems that containing over 10^5 training examples(data) and 10^5 features/instances. It should be noted that SGD is just an optimization technique and does not relate to a specific category of machine learning. It serves only as a method for training a model.

Gradient Boosting is an ensemble learning method that trains models in a sequential manner. The underlying principle of this approach is based on the idea that the most efficient model for the subsequent stage, when merged with the previous models, would lead to the least amount of prediction error overall. The key principle of this approach is to set the target outcomes for the next model such that the error is minimized. By using the Gradient Descent method, each new model progressively reduces the loss function (y = ax + b + e, where e represents the error term) of the entire system. This iterative process gradually improves the estimation accuracy of the response variable by fitting new models consecutively. The fundamental concept behind this algorithm is to generate fresh base learners that possess optimal correlation with the adverse gradient of the loss function, which is pertinent to the complete ensemble. In our model it was found that the model performs best with learning rate=0.1.

Extreme Gradient Boosting (XGBoost) is a popular implementation of gradient-boosting decision trees. Its popularity is primarily due to its execution speed and model performance. Moreover, XGBoost does not require parameter optimization or tuning, thus saving time on implementation. XGBoost offers the benefit of regularization, which can effectively tackle the issue of overfitting. This is achieved by applying L1/L2 penalties to the biases and weights of each tree. This feature sets it apart from other implementations of gradient boosting. It efficiently utilizes the weighted quantile sketch algorithm to manage non-zero entries in the feature matrix. This algorithm retains the same computational complexity as other algorithms like stochastic gradient descent. Scalability is not an issue with XGBoost. XGBoost ultimately provides the capability of out-of-core computing during the computation phase, utilizing disk-based data structures in place of in-memory ones. XGBoost has no limitations on the dataset size, and it outperforms other algorithms like random forest (RF), gradient boosting machines (GBM), and gradient boosting decision trees (GBDT) in both execution speed and model performance.

4. Model Evaluation

To evaluate our classifiers, we have used the following evaluation metrics:

Accuracy is defined as the ratio of correct predictions to the total number of input samples, it assumes an equal number of samples for each class. Therefore, it is important to consider other metrics in addition to accuracy. In cases where data is imbalanced, accuracy alone may not provide an accurate representation of the model's performance.

Root Mean Square Error (RMSE) is the Standard Deviation(SD) of the residuals or prediction errors. Root Mean Square Error (RMSE) is a commonly used metric to measure the accuracy of a model's predictions in regression analysis. It denotes the square root of the average of squared differences between predicted values and actual values. RMSE is the measure of average magnitude of errors made by the model in predicting the target variable. A lower RMSE indicates better accuracy of the model's predictions.

Mean Absolute Error (MAE) in statistics refers to the results of measuring the difference between two continuous variables. Mean Absolute Error (MAE) is a commonly used metric to measure the accuracy of a model's predictions in regression analysis. It represents the average of the absolute differences between the predicted values and the actual values. The model's ability to predict the target variable is measured by MAE, which calculates the average size of the errors made. It represents the average absolute difference between the predicted and actual values. A lower MAE indicates better accuracy of the model's predictions. MAE is widely used in various fields, including machine learning, data science, and engineering, to evaluate the performance of regression models. Compared to Root Mean Square Error (RMSE), MAE is less sensitive to outliers, as it takes the absolute difference between the predicted and actual values. However, it does not penalize large errors as heavily as RMSE does, since it does not involve squaring the errors.

III. Result and Analysis

During experimentation with all the different classifiers, Python 3 and Kaggle's in-built Jupyter Notebook were used. Libraries used include sckit-learn, Matplotlib, Seaborn, Pandas, Numpy.

The dataset was segregated into training and testing data and then those models were trained and the accuracy score and other evaluation metrics were noted in each case. A comparison of the performance of each algorithm is represented in the table below-

Methods	Classification Accuracy	Root Mean Square Error (RMSE)	Mean Absolute Error (MAE)
Logistic Regression	84.51	0.3934	0.1548
Decision Tree	79.30	0.4549	0.2069
Random Forest	85.76	0.3772	0.1423
K-Nearest Neighbor (KNN) [n=27]	85.13	0.3855	0.1486
Support Vector Machine (SVM)	84.31	0.3960	0.1568
Naïve Bayes	81.02	0.4355	0.1897
Stochastic Gradient Descent (SGD)	84.32	0.3959	0.1567
Gradient Boosting [learning rate=0.1]	85.43	0.3816	0.1456
Extreme Gradient Boosting (XGBoost)	85.55	0.3801	0.1444

IV. Conclusion

The objective of this research paper is to develop a model and evaluate the effectiveness of multiple Machine Learning algorithms to identify the most precise algorithm for predicting rainfall. The primary aim is to elucidate various ML techniques that can be applied to forecast rainfall. The ultimate goal of this research is to devise a model that is both accurate and efficient by utilizing a minimal number of attributes and tests. The initial step involves pre-processing the data, which is then utilized in the model.

Accuracy wise Random Forest (85.76), followed by XGBoost (85.55) performed the best.

And, Decision Tree (79.30) performed worst.

Based on our analysis, it can be inferred that the weather in Australia is highly unpredictable, and there is no apparent correlation between rainfall and specific regions or timeframes.

The research can further be extended to cover other Machine Learning techniques such as clustering, time series, and association rules and many other ensemble techniques. Considering the limitations of this study, it is necessary to develop more intricate and hybrid models to achieve a higher level of accuracy for rainfall prediction. Moreover, future studies could be designed with more comprehensive monitoring of specific areas and develop models for massive datasets, thus increasing the calculation rate and improving the precision and accuracy of the model.

References

[1] Kumar Abhishek. Abhay Kumar, Rajeev Ranjan, Sarthak Kumar," A Rainfall Prediction Model using Artificial Neural Network", 2012 IEEE Control and System Graduate Research Colloquium (ICSGRC2012), pp. 82-87, 2012.

[2] V. Veeralakshmi and D. Ramyachitra, Ripple Down Rule learner (RIDOR) Classifier for IRIS Dataset. Issues, vol 1, p. 79-85.

[3] World Health Organization: Climate Change and Human Health: Risks and Responses. World Health Organization, January 2003.

[4] G. Geetha and R. S. Selvaraj, "Prediction of monthly rainfall in Chennai using Back Propagation Neural Network model," Int. J. of Eng. Sci. and Technology, vol. 3, no. 1, pp. 211 213, 2011.

[5] Zahoor Jan, Muhammad Abrar, Shariq Bashir and Anwar M Mirza, "Seasonal to interannual climate prediction using data mining KNN technique", International Multi-Topic Conference, pp. 40-51, 2008.

[6] Alcntara-Ayala, I.: Geomorphology, natural hazards, vulnerability and prevention of natural disasters in developing countries. Geomorphology 47(24), 107124 (2002).

[7] [Online] Sckit-Learn Documentation Available: https://scikit-learn.org/stable/user_guide.html

[8] Elia Georgiana Petre, "A decision tree for weather prediction", Seria Matematica - Informatica] – Fizic, no. 1, pp. 77-82, 2009.

[9] Gupta D, Ghose U. A Comparative Study of Classification Algorithms for Forecasting Rainfall. IEEE. 2015.

[10] Wang J, Su X. An improved K-Means clustering algorithm. IEEE. 2014.

[11] Nicholls, N.: Atmospheric and climatic hazards: Improved monitoring and prediction for disaster mitigation. Natural Hazards 23(23), 137155 (2001).

[12] Rajeevan, M., Pai, D. S., Anil Kumar, R. & Lal, B. New statistical models for long-range forecasting of southwest monsoon rainfall over India. Clim. Dyn. 28, 813–828 (2007).

[13] Mishra, V., Smoliak, B. V., Lettenmaier, D. P. & Wallace, J. M. A prominent pattern of year-to-year variability in Indian Summer Monsoon Rainfall. Proc. Natl Acad. Sci. USA 109, 7213–7217 (2012).

[14] Thirumalai, C., Harsha, K. S., Deepak, M. L., & Krishna, K. C. (2017). Heuristic prediction of rainfall using machine learning techniques. 2017 International Conference on Trends in Electronics and Informatics (ICEI).