RAINFALL PREDICTION USING MACHINE LEARNING

Dr.S.V.N.Sreenivasu
Professor
Computer Science and Engineering
Narasaraopet Engineering College
narasaraopet,India
drsvnsrinivasu@gmail.com

K.V.N.D.Varshitha
Student
Computer Science and Engineering
Narasaraopet Engineering College
Vijayawada,India
kurakulavarshitha396@gmail.com

P.Mani Meghana
Student
Computer Science and Engineering
Narasaraopet Engineering College
Chilakaluripet,India
manipodili00@gmail.com

G.Anuhya
Student
Computer Science and Engineering
Narasaraopet Engineering College
Ponnuru,India
anuhyagovathoti@gmail.com

***Abstract*— The title is "Rainfall Prediction Using Machine Learning". The initiative's dataset is written in Python and stored in Microsoft Excel. A wide range of machine learning algorithms are used to discover which strategy generates the best accurate predictions. In many sections of the country, rainfall forecasting is critical for avoiding major natural disasters. This forecast was created using a variety of machine learning approaches, including catboost, xgboost, decision tree, random forest, logistic regression, neural network, and light gbm. It incorporates several components. The Weather Dataset was utilized. The primary goal of the research is to evaluate a variety of algorithms and determine which one performs best. Farmers may greatly profit from growing the appropriate crops based on the amount of water they require.**

Keywords— Machine learning algorithms include the catboost, xgboost, decision tree, random forest, logistic regression, neural network, and light gbm.

I. INTRODUCTION

Predicting rainfall efficiency is critical in reducing the negative effects of atypical weather patterns on agriculture, property, and livelihood. Traditional systems that rely heavily on meteorological parameters like temperature, humidity, and pressure may fail to provide reliable forecasts. Machine learning techniques provide a possible approach by examining previous rainfall data to predict future patterns. Tailored models can be constructed using methodologies such as classification and regression to produce more accurate predictions, allowing farmers to receive early warnings and make better use of water supplies. Selecting the best algorithms to meet specific needs is critical for improving forecast accuracy and reducing errors between actual observations and projections, Moulana Mohammed [2].

This paper examines the technique used in the research, including data retrieval, pre-processing, and the use of various machine learning algorithms. It highlights the importance of data analysis. It forecasts rainfall using machine learning techniques such as random forest, decision tree, logistic regression, and neural networks. It underlines the need of precisely projecting rainfall patterns in the face of shifting weather trends [1].

It describes the dataset utilized in the project, which was saved in Microsoft Excel format. The dataset comprises a variety of properties, with the key prediction attribute being RAINTOMORROW,which indicates whether or not it will rain tomorrow.[3] It explains the project's approach, with an emphasis on the several machine learning algorithms used to predict rainfall: Random Forest Classifier, Logistic Regression, Decision Tree, and Catboost. Each algorithm is briefly explained, emphasizing its importance in the prediction process,[3].

II. LITERATURE SURVEY

Akash [1] focuses on applying machine learning algorithms to forecast rainfall, stressing its significance in agriculture and emergency management. It studies numerous strategies for accurately forecasting rainfall, including random forest, logistic regression, decision tree, and neural networks. The study also underlines the importance of accurate rainfall forecasting in dealing with macro-level issues like floods and agricultural challenges, as well as micro-level concerns like public health. The analysis primarily underlines the need of technological advancements in improving rainfall prediction systems, with the ultimate goal of producing more precise and accessible forecasting approaches.

Moulana [2] looks into many methods and techniques for predicting rainfall, including linear regression, decision trees, artificial neural networks, support vector regression, and fuzzy logic. Researchers used these algorithms on datasets from multiple geographies and historical periods to improve rainfall forecasting accuracy. The efficacy of these solutions was evaluated using metrics like mean absolute error and R2 scores. Overall, the study underlines the need of accurate rainfall forecasting in avoiding natural disasters and boosting agricultural practices.

Shah [3] examines numerous machine learning and forecasting approaches used to predict rainfall, highlighting the significance of accurate forecasting in India's agricultural and economic sectors. ARIMA, SVM, decision trees, and neural networks have all been used to predict meteorological variables such as precipitation. Temperature, humidity, and wind speed have all been demonstrated in studies to have an important role in rainfall prediction, with machine learning methods such as random forest exhibiting promising classification accuracy.

Le,Vuong Minh, et al. [4] created a Nonlinear Autoregressive Neural Model to predict daily rainfall, gathered required data for eight years. Networks with external variables (NARX). Metrics used for evaluation include correlation coefficient, MAE, RSME, error mean, median, and standard deviation.Deepak Kumar [5] predicted rainfall using a hybrid approach that included machine learning techniques.The F-score, precision, accuracy, and recall criteria were used to assess performance on North Carolina rainfall data between 2007 and 2017. After analyzing eight hybrid models, Gradient boosting-Ada boost was found to be the most effective and produced positive outcomes.

III. METHODOLOGY

A. Data set

The Fig.2 dataset contains 23 attributes in total, and our main goal is to forecast whether or not it will rain tomorrow. The major attribute used for this prediction is RAINTOMORROW. Our dataset has no null values in any attribute. We will use data cleaing approach for preparing data. During the data preprocessing phase, categorical variables are encoded using label encoding to turn them into numerical representation, making them easier to use in machine learning models.Missing data is handled by multiple imputation by chained equations, a technique that iteratively predicts missing values based on observable data, increasing the dataset's completeness. Outliers are also found and deleted using the Interquartile Range (IQR), a statistical tool that detects data points that are considerably different from the central trend, ensuring the dataset's resilience and dependability for subsequent analysis and modeling.

	Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	...	Humidity9am	Humidity3pm
0	2008-12-01	Albury	13.4	22.9	0.6	NaN	NaN	W	44.0	W	...	71.0	22.0
1	2008-12-02	Albury	7.4	25.1	0.0	NaN	NaN	WNW	44.0	NNW	...	44.0	25.0
2	2008-12-03	Albury	12.9	25.7	0.0	NaN	NaN	WSW	46.0	W	...	38.0	30.0
3	2008-12-04	Albury	9.2	28.0	0.0	NaN	NaN	NE	24.0	SE	...	45.0	16.0
4	2008-12-05	Albury	17.5	32.3	1.0	NaN	NaN	W	41.0	ENE	...	82.0	33.0

5 rows × 23 columns

Fig. 1. describing the "weatherAUS" dataset

```
 #   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
dtypes: float64(16), object(7)
memory usage: 25.5+ MB
```

Fig. 2. Dataset attributes and data types

B. Data Preprocessing

Data preparation is the process of transforming raw data into a tidy dataset. For example, raw data is just a set of null values. The machine learning system is unable to understand null values; our goal is to eliminate them. To remove null values, we shall apply the data cleaning aspect of the pretreatment procedures. The dataset can be prepared before being used in our technique. Similarly, the Random Forest Algorithm cannot do analysis if the dataset contains null values. Data preparation can also be used to structure our dataset in a specific way. The data preparation procedure consists of some steps.After completing these seven processes, we refer to the dataset as clean.This dataset can now be utilized in our required machine learning methods.

1) Missing Data Imputation

Missing values in continuous features are handled with Multiple Imputation by Chained Equations (MICE). MICE iteratively imputes missing data, modeling each feature based on the others to improve accuracy. It uses mode imputation to fill missing values with the most frequent value while maintaining data integrity. It generates multiple possible values for each missing entry, allowing for ambiguity and producing more robust results.

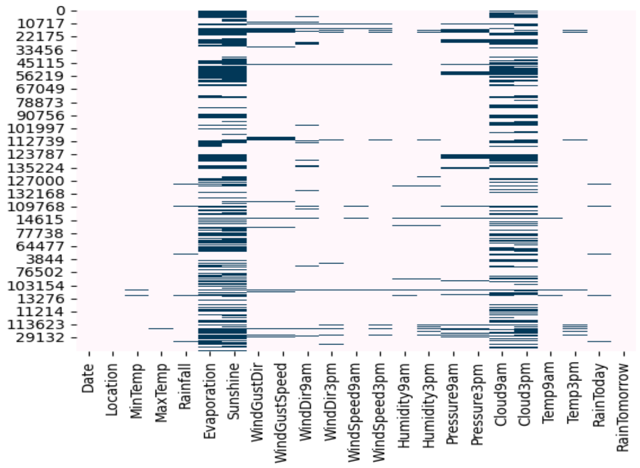

Fig. 3. Missing Data indicates that dark blue cells represent existing data and light blue cells represent missing data.

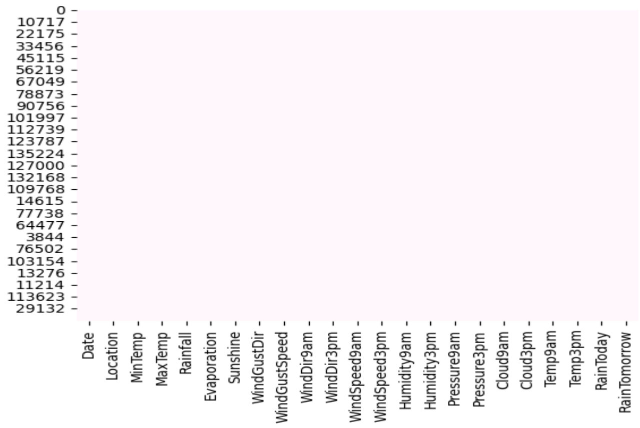

Fig. 4. Indicates non-missing values

2) Handling Imbalanced data

To remedy the dataset's class imbalance, the minority class is oversampled using scikit-learn's resample function. It shows the disproportionate distribution of incidences predicting rain (Yes) vs those that do not (No). Fig. 5. depicts how the data is divided into two groups depending on rain prediction (yes or no).The group containing Yes Rain occurrences (which is likely smaller) is replicated to match the size of the No Rain group.

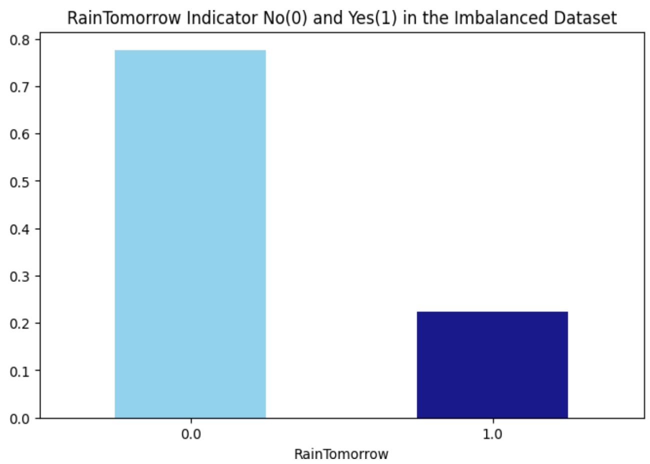

Fig. 4. Imbalanced Data using bar chart

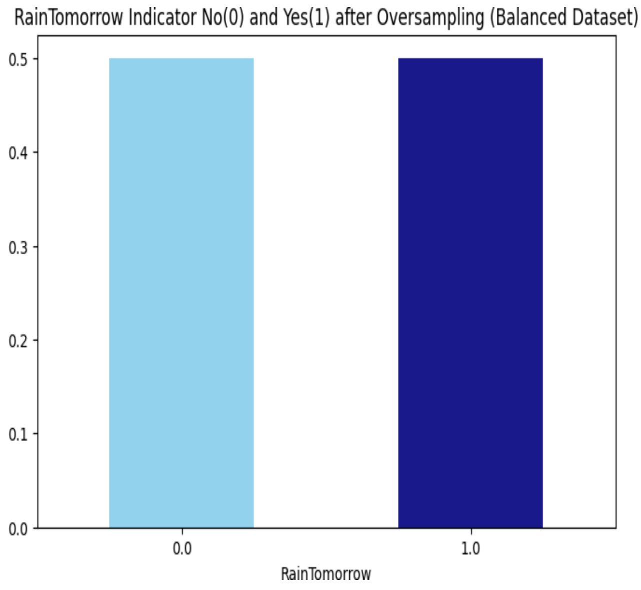

Fig. 5. Class Balancing using bar chart

3) Outlier Detection and Removal

The IQR (Interquartile Range) approach detects outliers in a dataset after missing values have been imputed. The IQR is calculated by subtracting Q1 from Q3, which represents the central half of the data distribution. Outliers are then identified using these quartiles,Any data value that falls below Q1 minus 1.5 times the IQR or exceeds Q3 plus 1.5 times the IQR is considered an outlier.Data rows that have outlier values in any of their columns are subsequently removed from the dataset entirely. This technique ensures the study's integrity by eliminating outliers that could skew the results. In Fig. 6 and 7, we employ boxplots to find and remove outliers within the interquartile range.

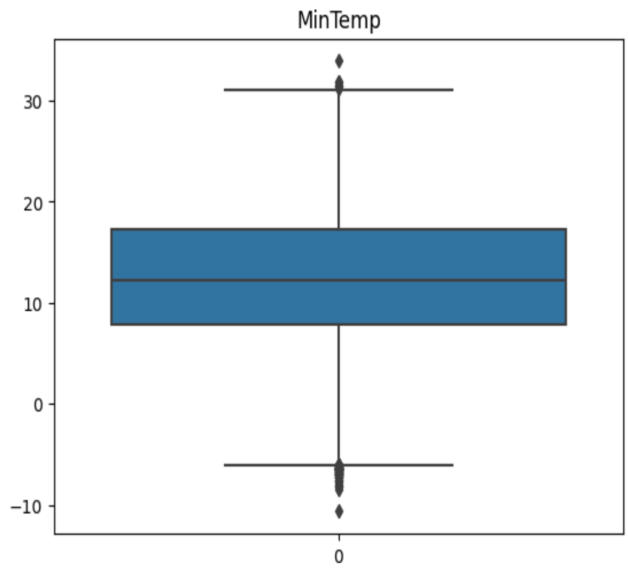

Fig. 6. Detecting outliers using boxplot

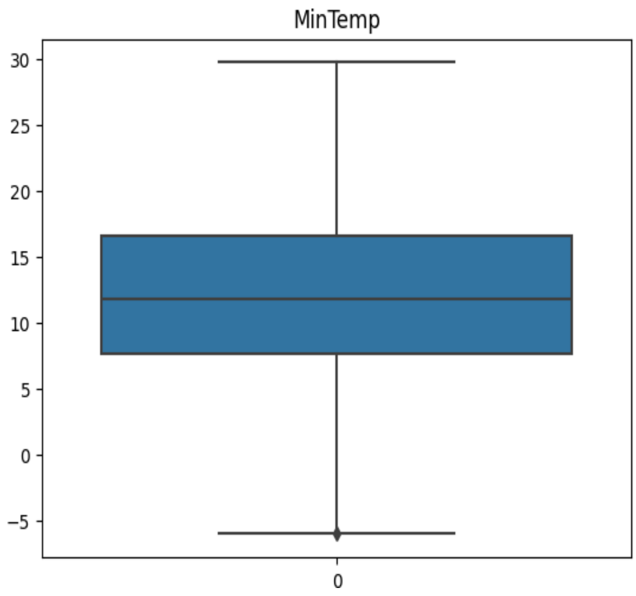

Fig. 7. Removing outliers using boxplot

4) Encoding Categorical Features

Scikit-learn LabelEncoder() is used to convert categorical features to continuous ones. This method simplifies the conversion of non-numerical data into numerical format that machine learning algorithms can understand. When we iterate through the dataset's category columns, we construct a separate LabelEncoder object for each feature to guarantee that categorical values are properly encoded. Categorical variables are transformed into numerical equivalents by fitting and converting each column using the appropriate LabelEncoder. This transformation allows the model to better read and assess categorical data, resulting in higher accuracy and performance for machine learning models trained on the dataset.

```
Date: ['2008-12-01' '2008-12-02' '2008-12-03' ...
'2007-12-15' '2008-01-30''2007-12-27']
Location: ['Albury' 'BadgerysCreek' 'Cobar' 'CoffsHarbour'
'Moree' 'Newcastle' 'NorahHead' 'NorfolkIsland' 'Penrith'
 'Richmond' 'Sydney' 'SydneyAirport' 'WaggaWagga'
 'Williamtown' 'Wollongong' 'Canberra' 'Tuggeranong'
 'MountGinini' 'Ballarat' 'Bendigo' 'Sale' 'MelbourneAirport'
 'Melbourne' 'Mildura' 'Nhil' 'Portland' 'Watsonia' 'Dartmoor'
 'Brisbane' 'Cairns' 'GoldCoast' 'Townsville' 'Adelaide'
 'MountGambier' 'Nuriootpa' 'Woomera' 'Albany' 'Witchcliffe'
 'PearceRAAF' 'PerthAirport' 'Perth' 'SalmonGums'
 'Walpole' 'Hobart' 'Launceston' 'AliceSprings' 'Darwin'
 'Katherine' 'Uluru']
WindGustDir: ['W' 'WNW' 'WSW' 'NE' 'SW' 'SSE' 'S' 'N'
 'NNW' 'NW' 'SE' 'NNE' 'ESE' 'E' 'SSW' 'ENE']
WindDir9am: ['W' 'NNW' 'SE' 'ENE' 'SW' 'SSE' 'S' 'N'
'WSW' 'NE' 'ESE' 'E' 'WNW' 'NNE' 'NW' 'SSW']
WindDir3pm: ['WNW' 'WSW' 'E' 'NW' 'W' 'SSE' 'SSW'
'SW' 'NNW' 'SE' 'N' 'S' 'NNE' 'ESE' 'ENE' 'NE']
```

Fig. 8. Categorical features before label encoding

```
Unique values of Date after label encoding:
['2007-11-01' '2007-11-02' '2007-11-03' ... '2017-06-23' '2017-06-24'
 '2017-06-25']
Unique values of Location after label encoding:
['Adelaide' 'Albany' 'Albury' 'AliceSprings' 'BadgerysCreek' 'Ballarat'
 'Bendigo' 'Brisbane' 'Cairns' 'Canberra' 'Cobar' 'CoffsHarbour'
 'Dartmoor' 'Darwin' 'GoldCoast' 'Hobart' 'Katherine' 'Launceston'
 'Melbourne' 'MelbourneAirport' 'Mildura' 'Moree' 'MountGambier'
 'MountGinini' 'Newcastle' 'Nhil' 'NorahHead' 'NorfolkIsland' 'Nuriootpa'
 'PearceRAAF' 'Penrith' 'Perth' 'PerthAirport' 'Portland' 'Richmond'
 'Sale' 'SalmonGums' 'Sydney' 'SydneyAirport' 'Townsville' 'Tuggeranong'
 'Uluru' 'WaggaWagga' 'Walpole' 'Watsonia' 'Williamtown' 'Witchcliffe'
 'Wollongong' 'Woomera']
Unique values of WindGustDir after label encoding:
['E' 'ENE' 'ESE' 'N' 'NE' 'NNE' 'NNW' 'NW' 'S' 'SE' 'SSE' 'SSW' 'SW' 'W'
 'WNW' 'WSW']
Unique values of WindDir9am after label encoding:
['E' 'ENE' 'ESE' 'N' 'NE' 'NNE' 'NNW' 'NW' 'S' 'SE' 'SSE' 'SSW' 'SW' 'W'
 'WNW' 'WSW']
Unique values of WindDir3pm after label encoding:
['E' 'ENE' 'ESE' 'N' 'NE' 'NNE' 'NNW' 'NW' 'S' 'SE' 'SSE' 'SSW' 'SW' 'W'
 'WNW' 'WSW']
```

Fig. 9. Label Encoding transforms categorical features into continuous features.

5) Standardization

The data included in the MiceImputed DataFrame is normalized using the Scikit-Learn module's MinMaxScaler. This preprocessing phase guarantees that all features are translated to a common scale, commonly 0 to 1, making them comparable and boosting the performance of machine learning algorithms. The revised data is subsequently placed in a new DataFrame named modified_data, which retains the original index and column names. This standardized data is now ready for further analysis and modeling, which will yield more accurate predictions or classifications in machine learning tasks.

6) Feature Selection

Scikit-learn offers two different ways for feature selection. First, Select KBest is used with the chi-squared statistical test to identify the top k attributes most related to the target variable, RainTomorrow. This filtering algorithm selects the top ten features based on statistical significance. SelectFromModel then assigns importance scores to each feature using a RandomForestClassifier, allowing important features to be selected based on their relevance. Fitting the RandomForestClassifier and extracting support for each feature yields a list of selected features. These techniques help to uncover and maintain the most useful features, hence improving the prediction ability of machine learning models trained on the dataset.

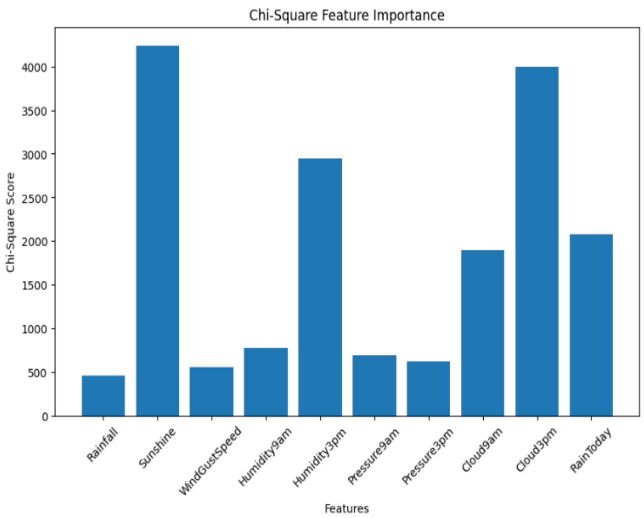

Fig. 10. Uses filter methods for feature selection. The most relevant attributes are identified using the chi-squared test.

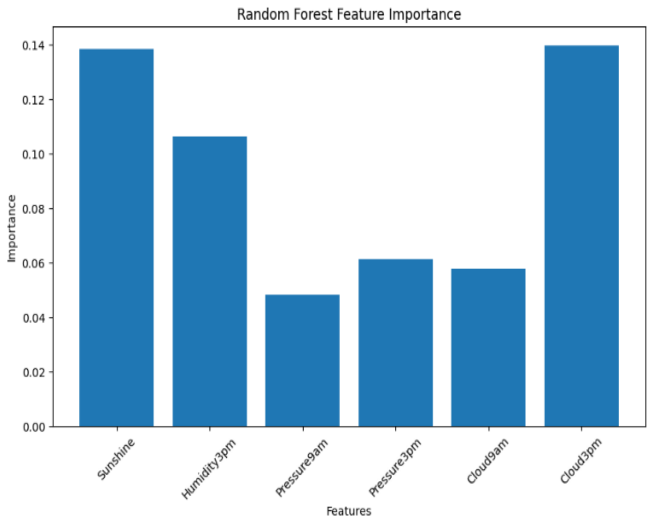

Fig. 11. Use random forest feature importance,shows importance of different features according to chi square scores

7) Feature Scaling

The data standardization approach uses scikit-learn's MinMaxScaler() function to scale the features from 0 to 1. Fitting the MinMaxScaler to the MiceImputed dataset identifies each feature's range, and the data is translated accordingly. The resulting modified_data dataframe contains the standardized features, which ensures scale consistency across all variables. This scaling method is particularly helpful for increasing the interpretability and performance of machine learning.

8) Model evaluation

The effectiveness of multiple categorization models is assessed using performance indicators such as accuracy, ROC-AUC, and Cohen's Kappa. ROC curves reflect the trade-off between sensitivity and specificity, as well as information on the model's discrimination capacity.

Confusion matrices provide a detailed study of the model's classification performance across multiple classes. Comparing these metrics across different models, such as Logistic Regression, Decision Tree, Neural Network, Random Forest, LightGBM, XGBoost, and CatBoost, provides a comprehensive understanding of their relative strengths and weaknesses, allowing for more informed model selection for the classification task.

a) Logistic Regression

A linear regression model developed for binary classification applications. It computes the likelihood that a given input belongs to a specified class by applying a logistic function to the observed data.

```
#LOGISTIC REGRESSION

from sklearn.linear_model import LogisticRegression

params_lr = {'penalty': 'l1', 'solver':'liblinear'}

model_lr = LogisticRegression(**params_lr)
model_lr, accuracy_lr, roc_auc_lr,coh_kap_lr,
tt_lr = run_model(model_lr, X_train, y_train, X_test, y_test)
```

```
Accuracy = 0.7952798350051561
ROC Area under Curve = 0.7895421146723749
Cohen's Kappa = 0.5823246558119864
Time taken = 3.410825252532959
              precision    recall  f1-score   support

         0.0    0.80459   0.83764   0.82078     23879
         1.0    0.78229   0.74144   0.76132     18789

    accuracy                        0.79528     42668
   macro avg    0.79344   0.78954   0.79105     42668
weighted avg    0.79477   0.79528   0.79460     42668
```

Fig. 12.Logistic Regression Accuracy

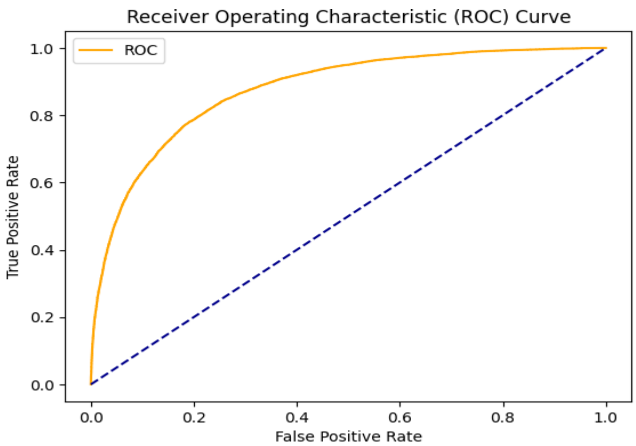

Fig. 13. Logistic regression ROC curve

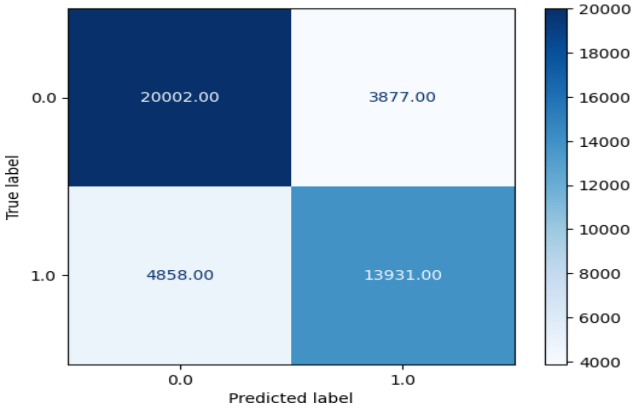

Fig. 14. Logistic Regression Confusion matrix

b) Decision Tree

The decision tree divides the feature space into regions, each representing a separate decision. It is a non-parametric, supervised learning technique for classification and regression applications.

```
#DECISION TREE
from sklearn.tree import DecisionTreeClassifier

params_dt = {'max_depth': 16,
             'max_features': "sqrt"}

model_dt = DecisionTreeClassifier(**params_dt)
model_dt, accuracy_dt, roc_auc_dt, coh_kap_dt,
tt_dt = run_model(model_dt, X_train, y_train, X_test, y_test)
```

```
Accuracy = 0.8661057466954157
ROC Area under Curve = 0.86640490489276
Cohen's Kappa = 0.7295529464666346
Time taken = 0.8912515640258789
              precision    recall  f1-score   support

         0.0    0.89334   0.86390   0.87837     23879
         1.0    0.83398   0.86891   0.85109     18789

    accuracy                        0.86611     42668
   macro avg    0.86366   0.86640   0.86473     42668
weighted avg    0.86720   0.86611   0.86636     42668
```

Fig. 15. Decision Tree Accuracy

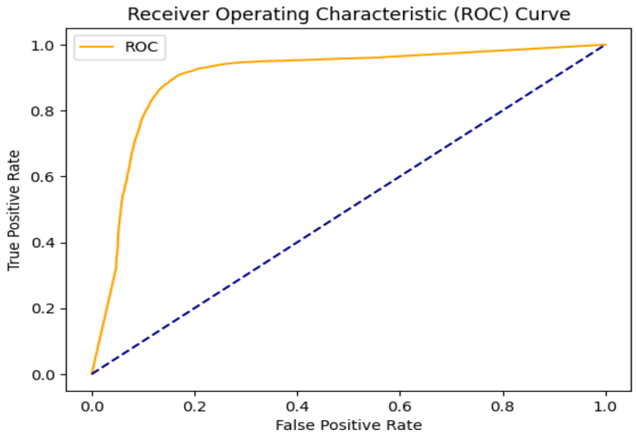

Fig. 16. Decision Tree ROC curve

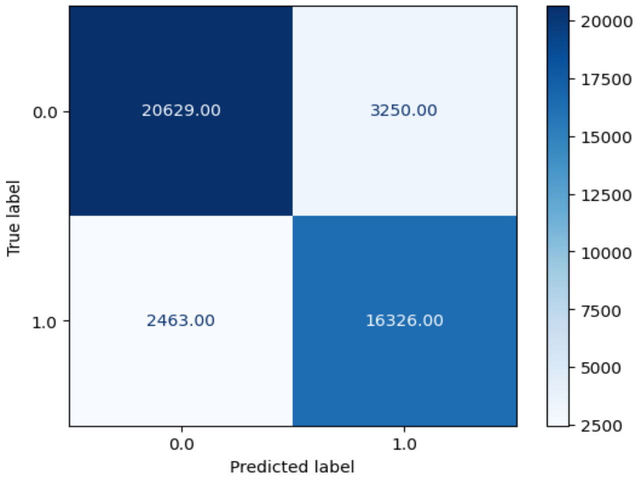

Fig. 17. Decision Tree Confusion matrix

c) Neural Network

The neural network was based on the structure of the human brain. It is made up of interconnected nodes organized into layers, with each performing a simple computation. Neural networks are extremely adaptable and capable of understanding complicated patterns in data, making them ideal for a variety of tasks, including categorization.

```
from sklearn.neural_network import MLPClassifier

params_nn = {'hidden_layer_sizes': (30,30,30),
             'activation': 'logistic',
             'solver': 'lbfgs',

             'max_iter': 500}

model_nn = MLPClassifier(**params_nn)
model_nn, accuracy_nn, roc_auc_nn, coh_kap_nn,
tt_nn = run_model(model_nn, X_train, y_train, X_test, y_test)
```

```
Accuracy = 0.8822068060373114
ROC Area under Curve = 0.8807161879983243
Cohen's Kappa = 0.7611128922397232
Time taken = 1025.734163761139
              precision    recall  f1-score   support

         0.0    0.89599   0.89321   0.89460     23879
         1.0    0.86481   0.86822   0.86651     18789

    accuracy                        0.88221     42668
   macro avg    0.88040   0.88072   0.88056     42668
weighted avg    0.88226   0.88221   0.88223     42668
```

Fig. 18.Neural network Accuracy

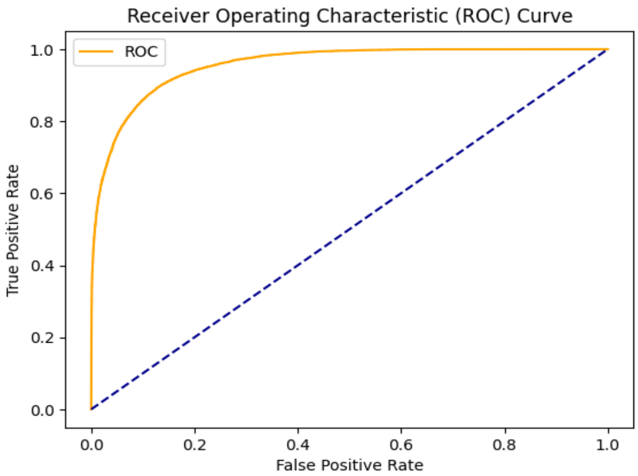

Fig. 19. Neural Network ROC curve

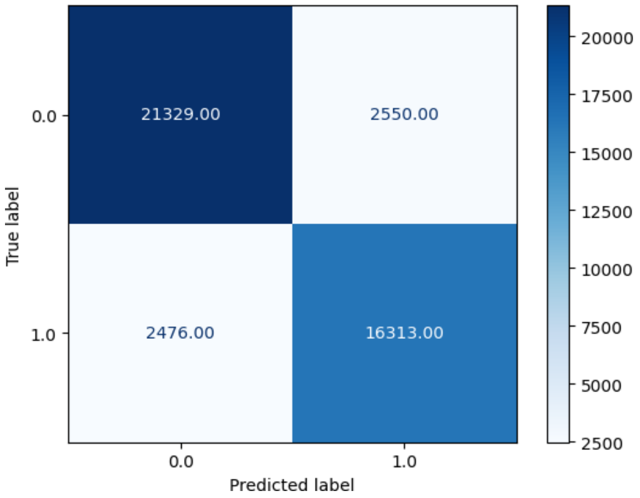

Fig. 20. Neural network confusion matrix

d) Random Forest

Random Forest, which generates a huge number of decision trees during training and returns the mode of classification or regression. Random forests beat single decision trees for prediction accuracy and overfitting.

e) LightGBM

LightGBM is a gradient boosting framework . It can process vast volumes of data quickly because to a new technique known as gradient-based one-side sampling. It outperforms previous gradient-boosting implementations in terms of efficiency, memory use, and training time.

```
import lightgbm as lgb
params_lgb ={'colsample_bytree': 0.95,
         'max_depth': 16,
         'min_split_gain': 0.1,
         'n_estimators': 200,
         'num_leaves': 50,
         'reg_alpha': 1.2,
         'reg_lambda': 1.2,
         'subsample': 0.95,
         'subsample_freq': 20}

model_lgb = lgb.LGBMClassifier(**params_lgb)model_lgb,accuracy_lgb,
roc_auc_lgb, coh_kap_lgb,
tt_lgb = run_model(model_lgb, X_train, y_train, X_test, y_test)
```

```
[LightGBM] [Info] Number of positive: 56230, number of negative: 71771
[LightGBM] [Info] Auto-choosing row-wise multi-threading, the overhead of testing was 0.004145 sec
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
[LightGBM] [Info] Total Bins 4324
[LightGBM] [Info] Number of data points in the train set: 128001, number of used features: 21
[LightGBM] [Info] [binary:BoostFromScore]: pavg=0.439293 -> initscore=-0.244030
[LightGBM] [Info] Start training from score -0.244030
Accuracy = 0.8662698040686229
ROC Area under Curve = 0.8634883731799748
Cohen's Kappa = 0.7282159692817018
Time taken = 2.9818294048309326
              precision    recall  f1-score   support

         0.0    0.87580   0.88680   0.88127     23879
         1.0    0.85380   0.84017   0.84693     18789

    accuracy                        0.86627     42668
   macro avg    0.86480   0.86349   0.86410     42668
weighted avg    0.86612   0.86627   0.86615     42668
```

Fig. 21. LightGBM Accuracy

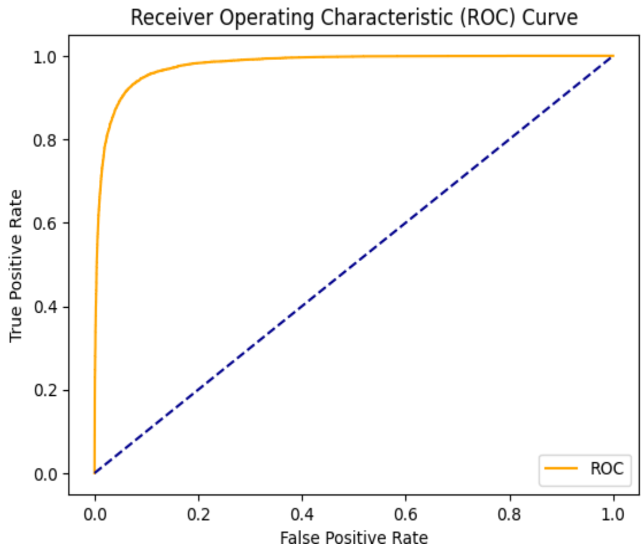

Fig. 22. LightGBM ROC curve

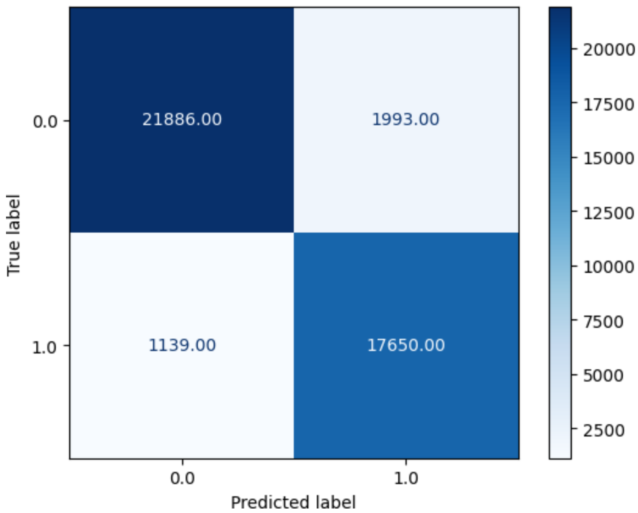

Fig. 23. LightGBM confusion matrix

f) XG Boost

XGBoost is another gradient-boosting technology known for its speed and performance. It enhances the model's performance by employing boosting, which sequentially adds models to an ensemble, with each model fixing errors made by its predecessors.

g) Cat Boost

CatBoost is a program for gradient boosting. It is specifically built to handle category attributes in an efficient and automated manner, with no need for pre-processing. CatBoost offers a number of ways for reducing overfitting and enhancing model generalization.

```python
import catboost as cb
params_cb ={'iterations': 50,
            'max_depth': 16}

model_cb = cb.CatBoostClassifier(**params_cb)
model_cb, accuracy_cb, roc_auc_cb, coh_kap_cb,
tt_cb = run_model(model_cb, X_train, y_train, X_test, y_test, verbose=False)
```

```
Accuracy = 0.9392050248429736
ROC Area under Curve = 0.9424115092864753
Cohen's Kappa = 0.8776540964755455
Time taken = 695.8269679546356
              precision    recall  f1-score   support

         0.0    0.97429   0.91553   0.94400     23879
         1.0    0.90029   0.96929   0.93352     18789

    accuracy                        0.93921     42668
   macro avg    0.93729   0.94241   0.93876     42668
weighted avg    0.94170   0.93921   0.93938     42668
```

Fig. 24. CatBoost Accuracy

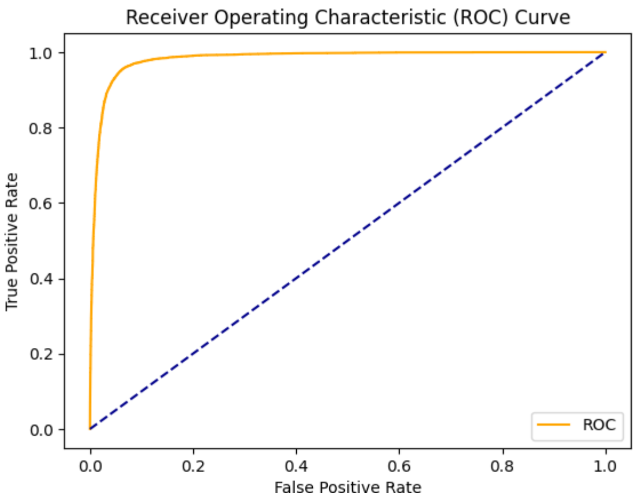

Fig. 25. CatBoost ROC curve

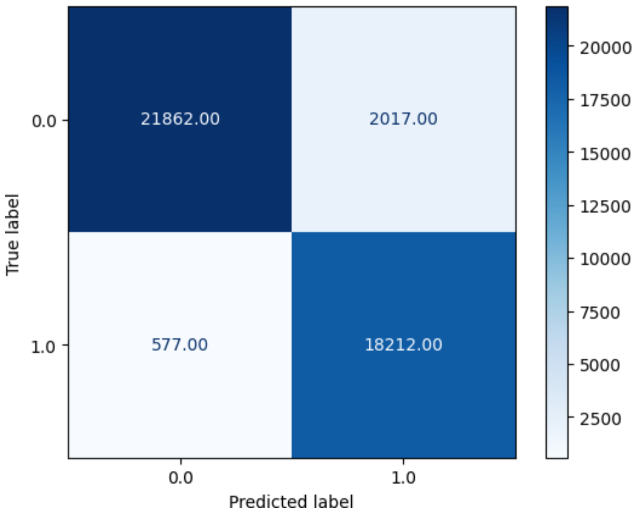

Fig. 26. CatBoost confusion matrix

C. Result

The graphs in Fig. 12 and 15 demonstrate the models accuracy and these models are performed better compared to remaining models. Using this models , we were able to forecast rainfall with reasonable accuracy.

```python
from sklearn.ensemble import RandomForestClassifier

params_rf = {'max_depth': 16,
             'min_samples_leaf': 1,
             'min_samples_split': 2,
             'n_estimators': 100,
             'random_state': 12345}

model_rf = RandomForestClassifier(**params_rf)
model_rf, accuracy_rf,
roc_auc_rf, coh_kap_rf,
tt_rf = run_model(model_rf, X_train, y_train, X_test, y_test)
```

```
Accuracy = 0.9265960438736289
ROC Area under Curve = 0.9279584837896794
Cohen's Kappa = 0.8517906871231153
Time taken = 96.86882638931274
              precision    recall  f1-score   support

         0.0    0.95053   0.91654   0.93323     23879
         1.0    0.89854   0.93938   0.91851     18789

    accuracy                        0.92660     42668
   macro avg    0.92454   0.92796   0.92587     42668
weighted avg    0.92764   0.92660   0.92674     42668
```

Fig. 27. Random Forest Accuracy

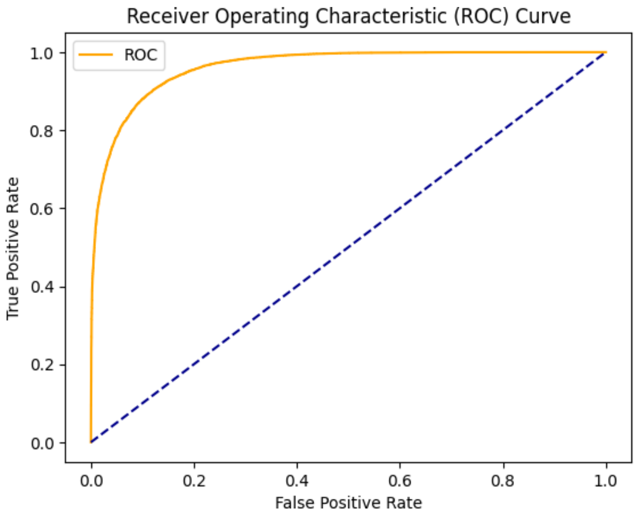

Fig. 28. Random Forest ROC curve

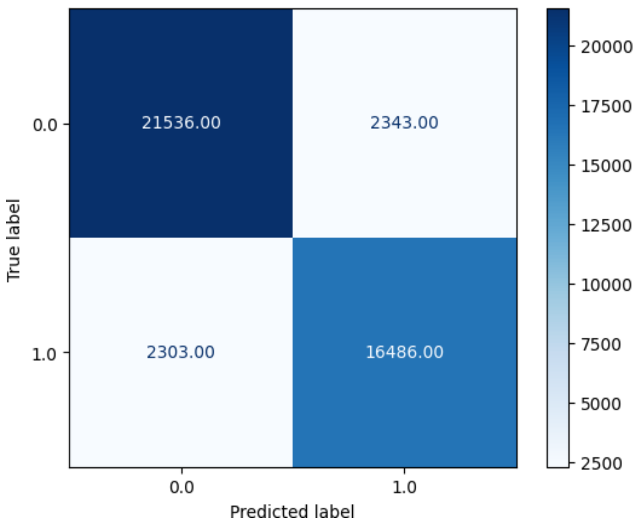

Fig. 29. Random Forest Confusion matrix

```python
import xgboost as xgb
params_xgb ={'n_estimators': 500,
            'max_depth': 16}

model_xgb = xgb.XGBClassifier(**params_xgb)
model_xgb, accuracy_xgb,
roc_auc_xgb, coh_kap_xgb,
tt_xgb = run_model(model_xgb, X_train, y_train, X_test, y_test)
```

```
Accuracy = 0.9336973844567358
ROC Area under Curve = 0.938222609233219
Cohen's Kappa = 0.8669394202789557
Time taken = 29.161067008972168
              precision    recall  f1-score   support

         0.0    0.97959   0.90029   0.93827     23879
         1.0    0.88510   0.97616   0.92840     18789

    accuracy                        0.93370     42668
   macro avg    0.93234   0.93822   0.93333     42668
weighted avg    0.93798   0.93370   0.93392     42668
```

Fig. 30. XGBoost accuracy

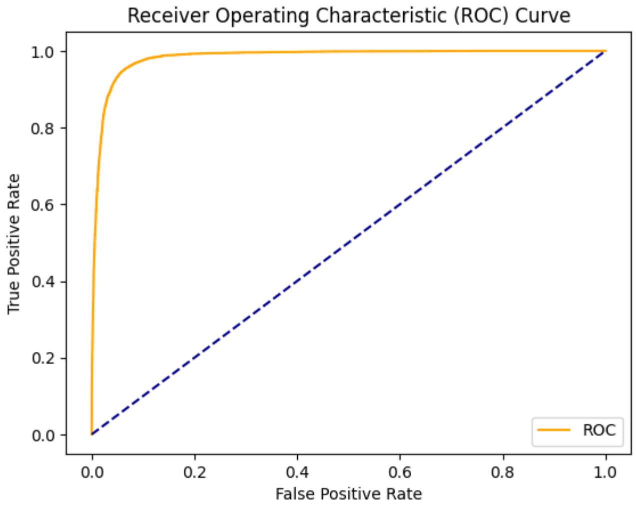

Fig. 31. XGBoost ROC curve

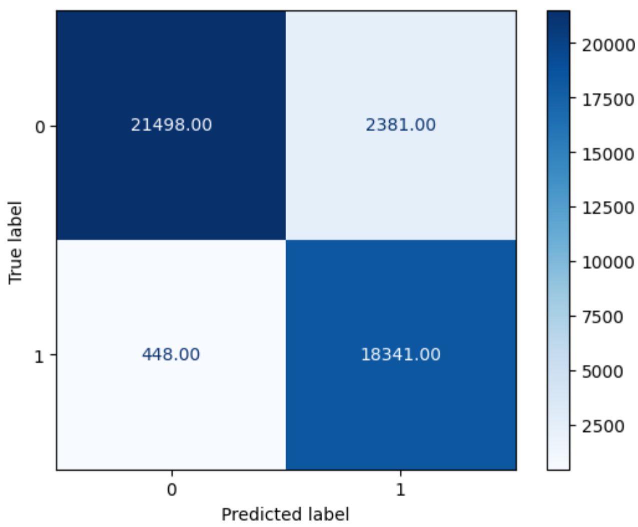

Fig. 32. XGBoost Confusion Matrix

IV. Result of Analysis

In the final phase, we will use a variety of machine learning methods to assess the weather dataset's accuracy. Table.1 shows the accuracy of the reference base paper results obtained using four different algorithms.

S.No	Algorithm	Accuracy
1	Logistic Regression[1]	78%
2	Decision Tree[1]	83%
3	Neural Network[1]	88%
4	Random Forest[1]	93%
5	Logistic Regression	79%
6	Decision Tree	85%
7	Neural Network	89%
8	Random Forest	92%
9	LightGBM	86%
10	XG Boost	93%
11	Cat Boost	93%

Table. 1. Demonstrates the accuracy of algorithms used in existing and proposed work.

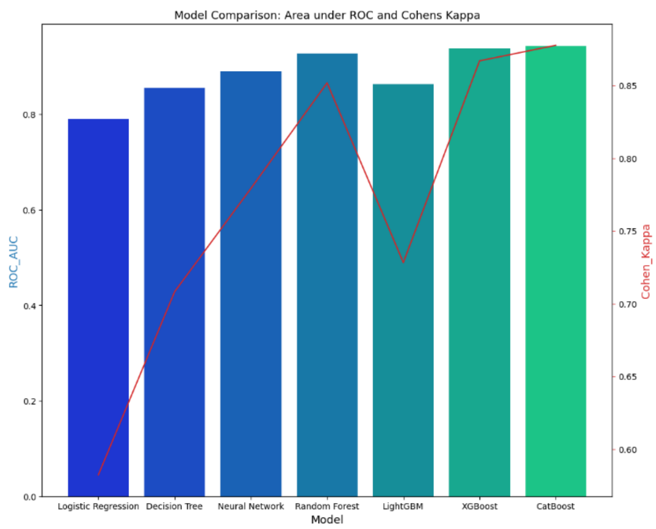

Fig. 33. the Random Forest model has the highest ROC_AUC, indicating that it is the most effective at distinguishing between wet and non-rainy days. However, the LightGBM model has the highest Cohen's Kappa, indicating that it has the best agreement between anticipated and observed rainfall.

V. CONCLUSION

The weatherAUS dataset was used in this work to investigate machine learning algorithms for rainfall prediction. Following data preprocessing, which included dealing with missing values, encoding categorical variables, and removing outliers, a variety of approaches were trained and tested. The results showed that XGBoost was the most accurate algorithm. However, data quality and processing capacity have a role in algorithm selection. Overall, our findings help meteorologists and other practitioners better understand machine learning methods for rainfall prediction, providing valuable insights.

REFERENCES

[1] Akash Gupta, Hitesh Kumar Mall and Janarthanan.S, Rainfall Prediction Using Machine Learning,Ieee,Doi: 10.1109, 2022

[2] Moulana Mohammed, Roshitha Kolapalli, Niharika Golla and Siva Sai Maturi, Prediction Of Rainfall Using Machine Learning Techniques, International Journal Of Scientific & Technology Research Volume 9, Issue:1, 01, January 2020

[3] U. Shah, S. Garg, N. Sisodiya, N. Dube and S. Sharma, Rainfall Prediction: Accuracy Enhancement Using Machine Learning and Forecasting Techniques, 2018 Fifth International Conference on Parallel, Distributed and Grid Computing (PDGC),Solan,India,pp.776-782, doi: 10.1109/PDGC.2018.8745763, 2018.

[4] Lê Vương,Binh Thai Pham,Tien-Thinh Le,Lu Minh Le and Hai-Bang Ly,Daily Rainfall Prediction Using Nonlinear Autoregressive Neural Network,springer,DOI:10.1007/978-981-15-2329-8_22,2019.

[5] Singh, Gurpreet, and Deepak Kumar. "Hybrid Prediction Models for Rainfall Forecasting." 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence). IEEE, 2019.

[6] S. Manandhar, S. Dev, Y. H. Lee, Y. S. Meng and S. Winkler, A Data-Driven Approach for Accurate Rainfall Prediction, *in* IEEE Transactions on Geoscience and Remote Sensing, vol. 57, no. 11, pp. 9323-9331, doi: 10.1109/TGRS.2019.2926110, Nov. 2019

[7] D.A. Sachindra a, K. Ahmed b, Md. Mamunur Rashid c, S. Shahid d and B.J.C. Perera a, Statistical downscaling of precipitation using machine learning techniques,science direct,V olume 212, 2018.

[8] A. Haidar and B. Verma,Monthly Rainfall Forecasting Using One-Dimensional Deep Convolutional Neural Network, in IEEE Access, vol. 6, pp. 69053-69063,doi: 10.1109/ACCESS.2018.2880044,2018.

[9] Chalachew Muluken Liyew & Haileyesus Amsaya Melese ,Machine learning techniques to predict daily rainfall amount,springer,Journal Of Big Data,Volume 8,2021.

[10] Lingling Ni a, Dong Wang a, Vijay P. Singh b, Jianfeng Wu a, Yuankun Wang a, Yuwei Tao a, Jianyun Zhang c,Streamflow and rainfall forecasting by two long short-term memory-based models,Journal of Hydrology Volume 583, 124296,April 2020.

[11] Q. Zhao, Y. Yao, W. Yao, and Z. Li, “Real-time precise point positioningbased zenith tropospheric delay for precipitation forecasting,” Sci. Rep., vol. 8, no. 7939, 2018, doi:10.1038/s41598-018-26299 3, April 2020.

[12] Refonaa, J., Lakshmi, M. ,; Dhamodaran, S., Teja, Surya , Pradeep, T. N. M. 1 ;Machine Learning Techniques for Rainfall Prediction Using Neural Network Journal of Computational Nanoscience, Volume 16, 2019.

[13] Vijayan R, Mareeswari V, Mohankumar P, Gunasekaran G and Srikar K, (JUNE,. Estimating rainfall prediction using machine learning techniques on a dataset. Int J Sci Technol Res.;9(06):440–5,2020.

[14] Balan MS, Selvan JP, Bisht HR, Gadgil YA, Khaladkar IR, Lomte VM. Rainfall prediction using deep learning on highly non-linear data. Int J Res Eng Sci Manage,Volume-2, Issue-3,2019.

[15] Pengcheng Zhang, Yangyang Jia, Jerry Gao, Wei Song and Hareton Leung,Short-Term Rainfall Forecasting Using Multi-Layer Perceptron,IEEE,Volume: 6,Issue:1,DOI: 10.1109/TBDATA.2018.2871151,2018.

[16] Yajnaseni Dash, Saroj K. Mishra,Bijaya K. Panigrahi,Rainfall prediction for the Kerala state of India using artificial intelligence approaches,ScienceDirect,Volume 70, Pages 66-73,2018.

[17] R. Kingsy Grace; B. Suganya,Machine Learning based Rainfall Prediction,IEEE,DOI:10.1109/ICACCS48705.2020.9074233, 2020.