**RAINFALL PREDICTION SYSTEM USING**

**MACHINE LEARNING AND DEEP LEARNING**

### A Major Project Report Submitted

**In partial fulfillment of the requirements for the award of the degree of**

**Bachelor of Technology In**

**Computer Science and Engineering**

**By**

## K.SULTAN SALAUDDIN 20N31A05B2

**K.GAYATRI 20N31A0598**

**G.SAI DEEP 20N31A0577**

Under the esteemed guidance of

**Dr.N.SATHISH KUMAR**

**Associate Professor**



### Department of Computer Science and Engineering

**Malla Reddy College of Engineering and Technology**

### (Autonomous Institution - UGC, Govt. of India)

(Affiliated to JNTUH, Hyderabad, Approved by AICTE, NBA & NAAC with ‘A’ Grade) Maisammaguda, Kompally, Dhulapally, Secunderabad – 500100

Website:[www.mrcet.ac.in](http://www.mrcet.ac.in/)

### 2020 - 2024



**Malla Reddy College of Engineering and Technology**

### (Autonomous Institution - UGC, Govt. of India)

(Affiliated to JNTUH, Hyderabad, Approved by AICTE, NBA & NAAC with ‘A’ Grade) Maisammaguda, Kompally, Dhulapally, Secunderabad – 500100

Website:[www.mrcet.ac.in](http://www.mrcet.ac.in/)

# CERTIFICATE

This is to certify that this is the Bonafide record of the project entitled **“RAINFALL PREDICTION SYSTEM USING MACHINE LEARNING AND DEEP LEARNING”**, submitted by **K.SULTAN SALAUDDIN (20N31A05B2) , K.GAYATRI (20N31A0598) and G.SAI DEEP (20N31A0577)** of B.Tech in the partial fulfillment of the requirements for the degree of Bachelor of Technology in Computer Science and Engineering, Department of CSE, during the year 2023-2024. The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree or diploma.

**Internal Guide Head of the Department Dr.N.Sathish Kumar Dr. S.Shanthi**

**Associate Professor Professor**

**External Examiner**

**DECLARATION**

## We hereby declare that the project titled “RAINFALL PREDICTION SYSTEM USING MACHINE LEARNING AND DEEP LEARNING” submitted to Malla Reddy College of Engineering and Technology (UGC Autonomous), affiliated to Jawaharlal Nehru Technological University Hyderabad (JNTUH) for the award of the degree of Bachelor of Technology in Computer Science and Engineering is a result of original research carried-out in this thesis. It is further declared that the project report or any part thereof has not been previously submitted to any University or Institute for the award of degree or diploma.

## K.SULTAN SALAUDDIN – 20N31A05B2

**K.GAYATRI – 20N31A0598**

**G.SAI DEEP – 20N31A0577**

# ACKNOWLEDGMENT

We feel honored to place our warm salutation to our college Malla Reddy College of Engineering and Technology (UGC-Autonomous) for all owing us to do this Project as part of our B. Tech Program. We are ever grateful to our Director **Dr. V. S. K Reddy** and Principal **Dr. S. Srinivasa Rao** who enabled us to have experience in engineering and gain profound technical knowledge.

We express our heartiest thanks to our HOD, **Dr.S.Shanthi** for encouraging us in every aspect of our course and helping us realize our full potential.

We would like to thank our Project Guide **Dr.N.Sathish Kumar** for the regular guidance, suggestions, constant encouragement, continuous monitoring, and unflinching cooperation throughout project work.

We would like to thank our Project Coordinator **Ms.D.Sai Eswari** who in spite of being busy with their duties took time to guide and keep us on the correct path.

We would like to thank our Class Incharge **Mr.M.Sandeep** who despite being busy with the academic duties took time to guide and keep us on the correct path.

We would also like to thank all the faculty members and supporting staff of the Department of CSE and all other departments who have helped directly or indirectly in making our project a success.

We are extremely grateful to our parents for their blessings and prayers for the completion of our project which gave us the strength to do our project.

With regards and gratitude

## K.SULTAN SALAUDDIN – 20N31A05B2

**K.GAYATRI – 20N31A0598**

**G.SAI DEEP – 20N31A0577**

# ABSTRACT

Forecasting the amount of rain that will fall each day increases agricultural productivity and guarantees a steady supply of food and water to keep populations healthy. Numerous experiments have been conducted in different countries to forecast rainfall using machine learning and data mining techniques.The agriculture, which is the foundation of the nation's economy, is impacted by the country's uneven rainfall distribution. The nation must plan for and carry out the prudent use of rainfall water in order to mitigate the problems of flooding and drought.This study's primary goal is to determine the pertinent atmospheric factors that contribute to precipitation and utilize artificial intelligence to forecast the daily amount of rain that will fall. The machine learning model's input variables were chosen using the Pearson correlation technique to be pertinent environmental factors. The root mean squared error and mean absolute error methodologies are used to assess the performance of the machine learning model. The study's findings show that the Extreme Gradient Boosting machine learning technique performs better than its rivals.. We accomplish this using the machine learning methods of Extreme Gradient Boost, Random Forest, and Multivariate Linear Regression.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **TITLE** | **PG.NO** |
| **1** | **INTRODUCTION** | **1** |
|  | 1.1 PURPOSE AND OBJECTIVES | 2 |
|  | 1.2 EXISTING AND PROPOSED SYSTEM | 3 |
|  | 1.3 SCOPE OF PROJECT | 5 |
| **2** | **LITERATURE SURVEY** | **7** |
| **3** | **SYSTEM ANALYSIS** | **10** |
|  | 3.1 HARDWARE AND SOFTWARE REQUIREMENTS | 10 |
|  | 3.2 SOFTWARE REQUIREMENTS SPECIFICATION | 11 |
| **4** | **SYSTEM DESIGN** | **15** |
|  | 4.1 DESCRIPTION | 15 |
|  | 4.2 ARCHITECTURE | 19 |
|  | 4.3 UMLDIAGRAMS | 21 |
| **5** | **METHODOLOGY** | **26** |
|  | 5.1 TECHNOLOGIES | 26 |
|  | 5.2 MODULES | 33 |
|  | 5.3 PROCESS/ALGORITHMS | 35 |
| **6** | **IMPLEMENTATION** | **41** |
|  | 6.1 SAMPLE CODE | 41 |
|  | 6.2 OUTPUT SCREENS | 58 |

**7 CONCLUSION 66**

**8 BIBLIOGRAPHY 68**

|  |  |  |
| --- | --- | --- |
|  | **CHAPTER - 1**  **1.INTRODUCTION** |  |

Rainfall has a major impact on how the plants and fauna of the natural world evolve.Its significance must be taken into account by all living things, including humans, animals, and plants. Unquestionably one of the most valuable natural resources in the world, water is essential to farming and agriculture.The planet Earth and people are finding it more challenging to experience the necessary quantity of rainfall required to meet human needs and the continuous usage of that rainfall in day-to-day living as a result of shifting climate patterns and increasing greenhouse gas emissions.Even though rain is good for agriculture, heavy showers can damage crops.Flooding, which poses a threat to human life and causes infrastructure and building damage.

Landslides represent a threat to human life and impede transportation and communications. An abundance of precipitation can have detrimental effects on human activities, the environment, business and industry, and agriculture. Predicting the amount of rain is one of the more challenging parts of weather forecasting.Due to the significant climate changes, it is now harder than ever to estimate rainfall accurately.

The ability to predict rainfall will aid not just in understanding the shifting patterns of precipitation but also planning for emergency preparedness and catastrophe management. Precipitation forecasts could prove useful in formulating strategies and policies to combat the escalating worldwide issue of ozone depletion. Temperature variations and variations in precipitation patterns are linked to global warming, which is the increase in Earth's temperature caused by increased emissions of chlorofluorocarbons from commonplace objects like refrigerators, air conditioners, deodorants, printers, etc. Predictions of rains assist individuals in coping with the hot, humid weather. The space for innovation and revolution has increased as a result of technical advancement in the modern world.

The study will be important for flood management agencies as well since a more exact and accurate forecast for high monsoon rains will keep the agencies ready and focused for an impending catastrophe, the destruction of which might be reduced by adopting the preventative steps. Water is a limited resource that must be conserved for the benefit of humans since it is a rare resource. The rainfall prediction will significantly aid in addressing this growing problem. Additionally, it will assist people in managing and scheduling their social activities appropriately.

## 1.1 PURPOSE AND OBJECTIVES

Rainfall Prediction System is a web based project which is used to predict the conditions of the atmosphere for a given location and time. Rainfall Prediction System is crucial since it helps to determine future climate changes. With the use of latitude, we can determine the probability of snow and hail reaching the surface. We are able to identify the thermal energy from the sun that is exposed to a region. The Rainfall Prediction System will provide users with real-time weather information, forecasts, and other weather-related data, which can help them make better decisions about their day-to-day activities.

People can get accurate weather information is the main aim of the proposed system. The important issue faced in our country is climatic changes and that can be resolved by our proposed system “RAINFALL PREDICTION SYSTEM USING MACHINE LEARNING AND DEEP LEARNING”.

This application uses a neural network to predict precipitation based on historical weather data such as temperature, humidity, wind speed, and pressure. Models are trained on large datasets of weather information to learn the complex relationships between weather variables and precipitation.Rainfall Prediction Model has a main objective in prediction of the amount of rain in a specific well or division in advance by using various regression technique and find out which one is best for rainfall prediction.

Rainfall Prediction System using Machine Learning and Deep Learning has the potential to benefit several industries, including agriculture, flood prediction and management, water resource management, and weather forecasting. Rainfall Prediction Model has a main objective in prediction of the amount of rain in a specific well or division in advance by using various regression technique and find out which one is best for rainfall prediction.

**1.2 EXISTING SYSTEM AND PROPOSED SYSTEM**

**Existing System :**

* **Logistic Regression :**

A classification process known as logistic regression is employed to calculate the likelihood of an event occurring based on logistic function. When the dependent variable has only two possible values, such as 0 and 1 or True and False, it is described as a binary or dichotomous dependent variable. For each independent variable, the logistic regression model generates a coefficient that estimates the relative influence of the independent variable on the dependent variable.

* **K -Nearest Neighbor :**

Instance-based learning, often known as lazy learning, is the category in which the K-Nearest Neighbors (K-NN) method belongs. Although It is workable for regression, classification is where it is most frequently utilized. Many measures, consisting of Euclidean distance, Manhattan distance, Minkowski distance, etc., can be employed to establish the distance between the test and training data. The K-NN technique has a number of advantages, including being straightforward, simple to use, and easy to comprehend. It also offers good accuracy for smaller datasets. Yet it can be expensive to compute, especially for larger datasets., and it might not work well if the dataset is unbalanced.

* **Support Vector Machine (SVM) :**

Machine learning uses the support vector machine (SVM) method as a classification tool. It is a binary linear classifier that distinguishes between the various classes of data by identifying the ideal border, or hyperplane. The SVM's goal is to maximize the distance between the nearest datapoints in each class and the hyperplane, which separates the two classes' data points. SVM is a potent algorithm that is frequently employed when high accuracy classification in complex datasets is required. The SVM technique has a number of benefits, including the capacity to handle big datasets, low-dimensional data, and non-linear data by applying kernel approaches.

**Proposed System :**

* **Extreme Gradient Boosting :**

XGBoost is a supervised machine learning algorithm that belongs to the boosting algorithm family. It is an optimized version of the gradient boosting algorithm, whose goal is to make the model faster and more accurate by shortening the time required to calculate the gradient. Due to its prowess in managing big datasets, high accuracy, and handling missing values, XGBoost has become quite well-liked in the data science field. Decision trees, which are weak learners, are systematically added to the model as part of the process, with each new tree being trained to rectify the mistakes produced by its predecessors. To put it another way, XGBoost creates a collection of decision trees, where each tree aims to identify the patterns in the data that were missed by the previous trees.

XGBoost stands for Extreme Gradient Boosting, which applies a Gradient Boosting technique based on decision trees. It constructs short, basic decision trees iteratively. Each tree is termed as a “weak learner” because of its high bias. XGBoost begins by building the first basic tree that has a poor performance.

* **Random Forest :**

An approach for ensemble learning called RF is utilized to solve classification and regression issues. A final prediction is made by combining the results of many decision trees created by this supervised learning technique.

A forest of decision trees is created using Random Forest, and each tree is trained using a random subset of the training data and the features. This lessens overfitting and enhances generalization ability. The user can alter two hyperparameters: the size of the subgroups and the number of trees in the forest.

Each decision tree in the forest makes a separate prediction about the class or value of the new data point during prediction. The final forecast is then made by averaging or taking the majority vote (for classification) from all of the decision trees' outputs.

**1.3 SCOPE OF PROJECT**

Rainfall Prediction Model has a main objective in prediction of the amount of rain in a specific well or division in advance by using various techniques.This model also helps the farmer for agriculture to decide the crop, helping the watershed department for water storage and also helps to analyze the ground water level.

Rainfall Prediction Model is made by collecting as much data as possible about the current state of the atmosphere (particularly the temperature, humidity and wind) and using understanding of atmospheric processes (through meteorology) to determine how the atmosphere evolves in the future.

However, the chaotic nature of the atmosphere and incomplete understanding of the processes mean that forecasts become less accurate as the range of the forecast increases. The proposed system for forecasting the weather involving wind speed, cloud cover, rain or snow in order to nurture the needs of people all around the globe.

Rainfall Prediction Model has a main objective in prediction of the amount of rain in a specific well or division in advance by using various regression technique and find out which one is best for rainfall prediction. It suggests that neural networks are effective tools for rainfall prediction. They are capable of capturing complex and non-linear relationships between meteorological variables and rainfall, and they have been shown to outperform other machine learning techniques in many studies

The main scope is to forecast the Rainfall Prediction using different Machine Learning and Deep Learning Algorithms,It also uses different technologies which will used to predict the Rainfall Prediction with best accurate results.

**CHAPTER - 2**

**2.LITERATURE SURVEY**

The goal of Kunverji et al. was to create a working, flood-decisive prototype. The forecast model was created by the authors using DTs, RFs, and gradient boost algorithms, three supervised ML techniques. The Indian Water Portal for Bihar and Orissa provided the data set that was utilized in their research. With a 94.4% accuracy rate, it was found that DTs outperformed all other algorithms used. Temperature and rainfall intensity parameters have been incorporated in ML algorithms for flood prediction. KNNs, naïve Bayes, and SVMs—supervised learning techniques—were contrasted with deep learning models.

Therefore, to attain improved accuracy, a fresh data collection is necessary. The India Water Portal provided the set of information used in this study. Deep neural networks were found to perform better than the other algorithms, obtaining 91.18% accuracy. In their study, the scientists utilized both an SVM and a convolutional neural network (CNN), and they discovered that the CNN performed significantly better in terms of spatial resolution imaging while the SVM can make predictions based on linear data.

A robust flood map is also produced when an SVM and CNN are combined. The analysts confirmed that a probabilistic approach to examining the possibility of coastal flooding in the future would successfully aid in the creation of decisions for a coordinated beachside zone of the board . To forecast floods, they used a variety of machine learning algorithms on a set of flood data from southern Korea. The official website of the Korean Government served as the source of the data set for this research work. KNNs outperformed the other algorithms, according to the results, obtaining 94.6% accuracy.

The goal of Rani et al. was to create a reliable technique that can identify local floods and warn residents. Mean absolute error (MAE) and standard deviation are used to assess the efficacy of various flood detection techniques, including neural networks, SVMs, linear regression (LR), logistic regression, and linear regression (LR). The Indian Meteorological Department provided the data set that the researchers used. In the results, it can be shown that neural networks outperformed the other machine learning methods, reaching an MAE of 21.809 as opposed to the SVM's 90.606 and logistic regression's 40.246

In a quicker region-based CNN-based multilayer perceptron (R-CNN) is suggested for detecting coastal rubbish. If you want to get a generally faster R-CNN performance, it is challenging to synchronize a number of settings. To find small items, high-resolution features from a low-resolution image could be combined with high-dimensional ones. Automated coastal garbage identification could perform better if region of interest (RoI) align was used in place of RoI pooling to address position offset. Investigated for river-flood

prediction accuracy were models using the radial basis function-firefly algorithm (RBF-FA) and support vector machine-firefly algorithm (SVM-FA) . RBF-FA and SVM-FA models were developed by combining an FA with an RBF and an SVM. The statistical measures used to analyze the error of the approaches show reduced root-mean-square error and higher R2 compared to normal SVM and RBF models that take into account all nodes. The assessment's findings also showed that the SVM-FA and SVM models fared better in predicting river floods than the RBF-FA and RBF models.

Finding the ideal supervised learning model configuration was the aim . According to the investigation, water levels at both nearby stations and in the control station's past are the most important prediction factors. It has been proven that rainfall amounts are a poor indicator of when floods may occur. This article offered a challenge because there was a lack of experimental data and unknown significant variables.

Using imperviousness maps and data from social media, emergency warning response management can identify crucial locations during pluvial flooding disasters. Eleven different flood model combinations were used along with the top model. The findings allow for the following inferences: In comparison to the other models, the LR model has a higher prediction rate (area under the receiver operating characteristic (AUROC) of 86.8%). Researchers used KNNs and extreme gradient boosting (XGB) supervised learning models to study flash flood-prediction mapping . The ROC area under the curve accuracy for the XGB and KNN algorithms was 90.2 and 80.7%, respectively, with XGBoost's capacity to offer more outputs permitting higher accuracy.

It is hoped that employing different optimization strategies may improve the model's performance in next studies. In addition, it was discovered that the topographic wetness index (TWI) conditioning factors, slope, topography, and distance from the stream network were the parameters that had the most influence on the modelling processes.

According to study by Tehrany et al. adding condition-based factors to the parameters obtained from lidar does not necessarily increase the accuracy of the results. Some academicians claim that height is a significant effect in flooding. This creates additional chances for enhancing and fortifying the Malaysian flood monitoring model. The computational components were efficiently combined to create the model. The device can automatically send out a warning message if the water level rises above a certain threshold. The tool can also be used to keep an eye on flash floods. But the system hasn't yet been put to the test in a real-world scenario in a flood-affected area. Future work should consider the following elements to be more effective: 1) Data transmission (such as wireless communication), 2) multipurpose messaging, and 3) the Android app. Given Malaysia's enormous developments in information and communication technology, the development of these three is highly anticipated. A technique that will outperform the probability mapping system and produce flood susceptibility was put forth by Mosavi et al. By utilizing CNN's social media data, emergency warning response management can employ imperviousness maps to pinpoint crucial regions during pluvial flooding calamities.

Working with high-dimensional images and items of interest results in a strong spatial structure and aids in other research' conclusions, which is advantageous. Nonlinear data can also be predicted using an SVM. These two distinct network designs when combined will produce a flood map that is both accurate and dependable.

Machine Learning algorithms are mostly useful in predicting rainfall. Some of the major Machine 13 Learning algorithms are ARIMA Model (Auto-Regressive Integrate d Moving Average), Artificial Neural Network, Logistic Regression, Support Vector Machine and Self Organizing Map.

**CHAPTER - 3**

**3.SYSTEM ANALYSIS**

**3.1 HARDWARE AND SOFTWARE REQUIREMENTS**

**3.1.1 Hardware Requirements :**

* System : Intel Core
* Hard Disk : 500 GB.
* Floppy Drive : 1.44 MB.
* Monitor : 14 Colour Monitor.
* Mouse : Optical Mouse.
* RAM : 2 GB

**3.1.2 Software Requirements :**

* Operating System : Windows 11
* System Coding : Python
* Tool : Jupiter Notebook
* Libraries : Pandas,Numpy,Matplotlib,Scikit-Learn,etc,…

## 3.2 SOFTWARE REQUIREMENTS SPECIFICATION

#### 3.2.1 FUNCTIONAL REQUIREMENTS :

* **Data Collection and Preprocessing :**

Retrieve historical weather data from various sources such as meteorological stations, satellites, and weather models.Preprocess the data to clean outliers, handle missing values, and standardize formats for model input.

* **Feature Engineering :**

Extract relevant meteorological, geographical, and environmental features such as temperature, humidity, wind speed, topography, and land cover.Perform feature selection or dimensionality reduction techniques to identify the most informative features for rainfall prediction.

* **Model Development :**

Design and train machine learning and deep learning models capable of learning the complex relationships between input features and rainfall patterns.Experiment with different model architectures and optimization algorithms to find the best-performing configuration.Validate models using cross-validation techniques to assess their generalization performance on unseen data.

* **Model Evaluation :**

Evaluate the performance of the rainfall prediction models using appropriate metrics such as accuracy, precision, recall, F1-score, and area under the ROC curve.Conduct sensitivity analysis to assess the robustness of the models to variations in input data and model parameters.

* **Real-time Prediction :**

Implement mechanisms for real-time or near real-time prediction of rainfall events based on updated weather data streams.Integrate the prediction models with data ingestion pipelines and streaming of platforms for the seamless deployment in the operational environments.

* **Visualization and Interpretation :**

Develop interactive visualization tools to display predicted rainfall patterns, uncertainty estimates, and model confidence levels.Provide interpretable explanations of model predictions to facilitate understanding and decision-making by end-users.

* **Integration with Decision Support Systems :**

Integrate the rainfall prediction system with decision support systems used by government agencies, emergency responders, agriculture stakeholders, and other end-users.Enable automated alerts and notifications to notify stakeholders about potential rainfall events and associated risks.

* **Model Maintenance and Monitoring :**

Establish procedures for monitoring model performance over time and detecting drift or degradation in prediction accuracy.Implement retraining pipelines to periodically update the models using fresh data and adapt to changing environmental conditions.

* **Security and Privacy :**

Ensure the security and privacy of sensitive data used in the prediction system, following best practices for data encryption, access control, and compliance with regulations such as GDPR.

* **Documentation and Reporting :**

Document the system architecture, model configurations, data processing workflows, and evaluation results.Generate regular reports summarizing the performance of the prediction system and insights gained from rainfall predictions.

#### 3.2.2 NON FUNCTIONAL REQUIREMENTS

* **Performance :**

The system should be capable of delivering timely rainfall predictions, with low latency, to support real-time decision-making.It should be able to handle large volumes of data efficiently, ensuring fast processing and response times.Models should be computationally efficient, enabling rapid training and inference without excessive resource consumption.

* **Accuracy and Reliability :**

The system should produce accurate and reliable rainfall predictions, with high precision and recall rates, to instill confidence among users and stakeholders.Models should be robust against noise, uncertainties, and variations in input data, ensuring consistent performance under different conditions.

* **Scalabilty :**

The system should be scalable to accommodate increasing data volumes, user loads, and computational requirements as the user base grows or as additional functionalities are added.It should support horizontal and vertical scaling strategies to distribute workloads across multiple nodes and handle spikes in demand effectively.

* **Availabilty :**

The system should be highly available, with minimal downtime and disruptions, to ensure continuous access to rainfall predictions.It should incorporate redundancy, fault tolerance, and disaster recovery mechanisms to mitigate the impact of hardware failures, software errors, or other system faults.

* **Security :**

The system should adhere to industry standards and best practices for data security, encryption, authentication, and access control to protect sensitive information.It should implement measures to prevent unauthorized access, data breaches, and other security threats, safeguarding both the system and user data.

* **Usabillty :**

The system should have a user-friendly interface, with intuitive navigation, clear visualizations, and informative feedback, to enhance user experience.It should support customization and personalization options to cater to the diverse needs and preferences of different user groups.

* **Interoperability :**

The system should be interoperable with existing weather monitoring systems, databases, APIs, and external applications, facilitating seamless data exchange and integration.It should adhere to open standards and protocols to promote compatibility and interoperability across different platforms and environments.

* **Maintainability :**

The system should be easy to maintain, with modular architecture, well-documented codebase, and automated testing procedures, to facilitate troubleshooting, debugging, and software updates.It should support version control, configuration management, and continuous integration/deployment practices to streamline development workflows and ensure code quality.

* **Ethical and Legal Considerations :**

The system should comply with ethical principles, regulations, and legal requirements governing data privacy, consent, fairness, transparency, and accountability.It should incorporate mechanisms for bias detection and mitigation, ensuring fairness and equity in decision-making processes.

* **Performance Monitoring and Optimization :**

The system should include tools and processes for monitoring performance metrics, the

identifying bottle necks, and optimizing system efficiency.It should enable to continuous

improvement through feedback loops, experimentation, and iterative development practices.

# CHAPTER - 4

**4.SYSTEM DESIGN**

## 4.1 DESCRIPTION

## System design is the process of creating a system's architecture, parts ,and interfaces to ensure that it satisfies the needs of its users.Thus, in order to examine the design of this project, we first go through the specifics of establishing the concept of drone detection through a few fundamental modules that would clearly describe the workings of the system that would come from the development.

#### MACHINE LEARNING :

#### A subset of artificial intelligence known as machine learning focuses primarily on the creation of algorithms that enable a computer to independently learn from data and previous experiences. Without being explicitly programmed, machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things.

Machine learning algorithms create a mathematical model that, without being explicitly programmed, aids in making predictions or decisions with the assistance of sample historical data, or training data. For the purpose of developing predictive models, machine learning brings together statistics and computer science. Algorithms that learn from historical data are either constructed or utilized in machine learning. The performance will rise in proportion to the quantity of information we provide.

**A machine can learn if it can gain more data to improve its performance.** A machine learning system builds prediction models, learns from previous data, and predicts the output of new data whenever it receives it. The amount of data helps to build a better model that accurately predicts the output, which in turn affects the accuracy of the predicted output. It is used mainly for the prediction the model by training and testing.

## Classification of Machine Learning

At a broad level, machine learning can be classified into three types:

**1.Supervised Learning**

**2.Unsupervised Learning**

**3.Reinforcement Learning**

1. **Supervised Learning :**

In supervised learning, sample labeled data are provided to the machine learning system for training, and the system then predicts the output based on the training data. The system uses labeled data to build a model that understands the datasets and learns about each one. After the training and processing are done, we test the model with sample data to see if it can accurately predict the output. Supervised learning can be grouped further in two categories of algorithms:

* Classification
* Regression

**2.Unsupervised Learning :**

The training is provided to the machine with the set of data that has not been labeled, classified, or categorized, and the algorithm needs to act on that data without any supervision. The goal of unsupervised learning is to restructure the input data into new features or a group of objects with similar patterns.  It can be further classifieds into two categories of algorithms:

* Clustering
* Association

3. **Reinforcement Learning :**

Reinforcement learning is a feedback-based learning method, in which a learning agent gets a reward for each right action and gets a penalty for each wrong action. The agent learns automatically with these feedbacks and improves its performance. In reinforcement learning, the agent interacts with the environment and explores it. The goal of an agent is to get the most reward points, and hence, it improves its performance.

**DEEP LEARNING :**

Deep learning is based on the branch of machine learning, which is a subset of artificial intelligence. Since neural networks imitate the human brain and so deep learning will do. In deep learning, nothing is programmed explicitly. Basically, it is a machine learning class that makes use of numerous nonlinear processing units so as to perform feature extraction as well as transformation. The output from each preceding layer is taken as input by each one of the successive layers.

Deep learning models are capable enough to focus on the accurate features themselves by requiring a little guidance from the programmer and are very helpful in solving out the problem of dimensionality. Deep learning algorithms are used, especially when we have a huge no of inputs and outputs.

Since deep learning has been evolved by the machine learning, which itself is a subset of artificial intelligence and as the idea behind the artificial intelligence is to mimic the human behavior, so same is "the idea of deep learning to build such algorithm that can mimic the brain".

Deep learning is implemented with the help of Neural Networks, and the idea behind the motivation of Neural Network is the biological neurons, which is nothing but a brain cell.

Deep learning is a collection of statistical techniques of machine learning for learning feature hierarchies that are actually based on artificial neural networks.

It mainly focuses on three important types of neural networks that form the basis for most pre-trained models in deep learning:

1. Artificial Neural Networks (ANN)

2. Convolution Neural Networks (CNN)

3. Recurrent Neural Networks (RNN)

**1.Artificial Neural Networks (ANN) :**

Artificial Neural Networks (ANN) are algorithms based on brain function and are used to model complicated patterns and forecast issues. The Artificial Neural Network (ANN) is a deep learning method that arose from the concept of the human brain Biological Neural Networks. The development of ANN was the result of an attempt to replicate the workings of the human brain. The workings of ANN are extremely similar to those of biological neural networks, although they are not identical. ANN algorithm accepts only numeric and structured data.

**2.Convolution Neural Networks (CNN) :**

Deep Learning has proved to be a very powerful tool because of its ability to handle large amounts of data. The interest to use hidden layers has surpassed traditional techniques, especially in pattern recognition. One of the most popular deep neural networks is Convolutional Neural Networks (also known as CNN or ConvNet) in deep learning, especially when it comes to Computer Vision applications.

**3. Recurrent Neural Networks (RNN) :**

A Deep Learning approach for modelling sequential data is Recurrent Neural Networks (RNN). RNNs were the standard suggestion for working with sequential data before the advent of attention models. Specific parameters for each element of the sequence may be required by a deep feedforward model. It may also be unable to generalize to variable-length sequences. Recurrent Neural Networks use the same weights for each element of the sequence, decreasing the number of parameters and allowing the model to generalize to sequences of varying lengths.

## 4.2 ARCHITECTURE

## Architecture1.PNG

## Fig 4.2 : Architecture of Rainfall Prediction

## Firstly,we will take the Rainfall Dataset that will contain all the data regarding rainfall data.It will show the data such as temperature,humidity,wind and others.It is used to identify the Rainfall Patterns.It is used to train the different Machine Learning and Deep Learning Models.It will use the different types of algorithms and modules to train the model.After the completion of training the model,then it will test the model using different types of technologies.It will check the Performance Evaluation of Machine Learning and Deep Learning Models.Lastly,It will forecast the Rainfall Prediction.

Gather historical rainfall data from various sources such as weather stations, satellites, and weather forecasting agencies. This data should include relevant features like temperature, humidity, wind speed, air pressure, geographical information, etc. Clean the collected data by

## handling missing values, outliers, and inconsistencies. Normalize or standardize numerical features to ensure that they are on a similar scale. Choose appropriate Machine Learning and Deep Learning algorithms for rainfall prediction.

## Machine Learning algorithms like Random Forest, Support Vector Machines (SVM), Gradient Boosting Machines (GBM), or DL models like Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), or Long Short-Term Memory (LSTM) networks can be considered.

## Split the preprocessed data into training, validation, and test sets.Train the selected Machine Learnong and Deep Learning models using the training data. For Deep Learning models, it's essential to define the architecture (number of layers, activation functions, etc.) and optimize hyperparameters like learning rate, batch size, and regularization techniques.Monitor the model's performance on the validation set to prevent overfitting.We will evaluate the performance of the model. Once a satisfactory model is trained and evaluated, deploy it into a production environment where it can make real-time predictions.

## 

## Set up a pipeline to continuously update the model with new data and retrain it periodically to maintain its accuracy. Monitor the deployed model's performance over time and retrain it as necessary to adapt to changing patterns and conditions.Regularly update the system with the latest data and incorporate any improvements or advancements in Machine Learning/Deep Learning techniques.

## It uses a neural network to predict precipitation based on historical weather data such as temperature, humidity, wind speed, and pressure. Models are trained on large datasets of weather information to learn the complex relationships between weather variables and precipitation.

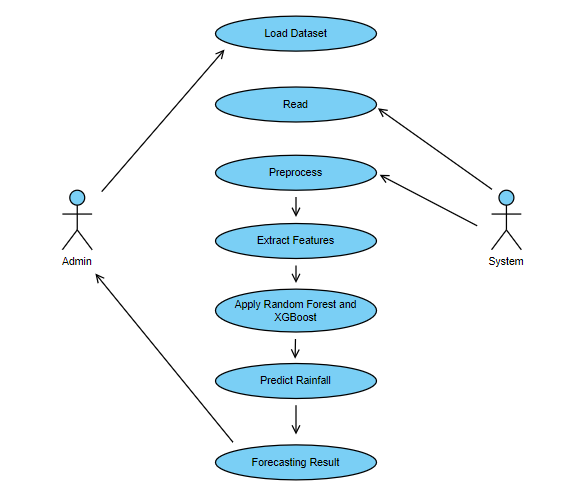
## It uses a neural network to predict precipitation based on historical weather data such as temperature, humidity, wind speed, and pressure. Models are trained on large datasets of weather information to learn the complex relationships between weather variables and precipitation.

## The isohyetal method is used to estimate the mean precipitation across an area by drawing lines of equal depth rainfall (called isohyets) and calculating the areas enclosed either between the isohyets or between isohyets and the catchment boundary.

**4.3 UML DIAGRAMS**

**4.3.1 USE CASE DIAGRAM**

A Use Case Diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses.

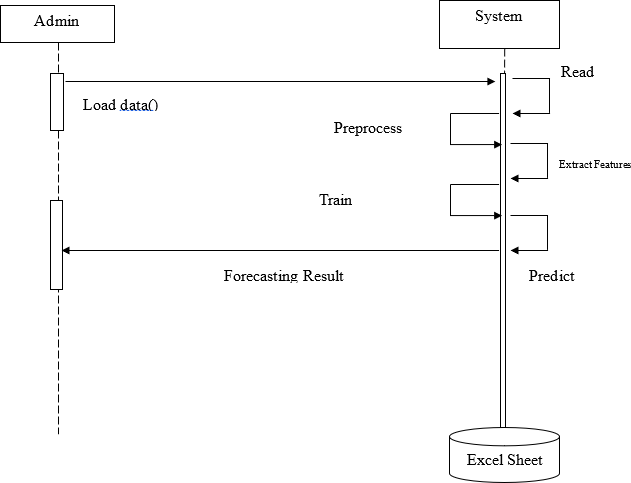


**Fig : 4.3.1 Use Case Diagram**

In these Use Case Diagram, we can see two actors and seven use cases. Use Case are self explanatory and they represent main functions of Rainfall Prediction System. Firstly, We have to load the dataset, then, we have to read, preprocess and extract the dataset. After that, We have to apply the Random Forest and Extreme Gradient Boosting Algorithms. So,It will help us to predict the rainfall and then generate the final result.

**4.3.2 SEQUENCE DIAGRAM**

A Sequence Diagram is a Unified Modeling Language (UML) diagram that illustrates the sequence of messages between objects in an interaction. A sequence diagram consists of a group of objects that are represented by lifelines, and the messages that they exchange over time during the interaction. A sequence diagram shows the sequence of messages passed between objects. Sequence diagrams can also show the control structures between objects.

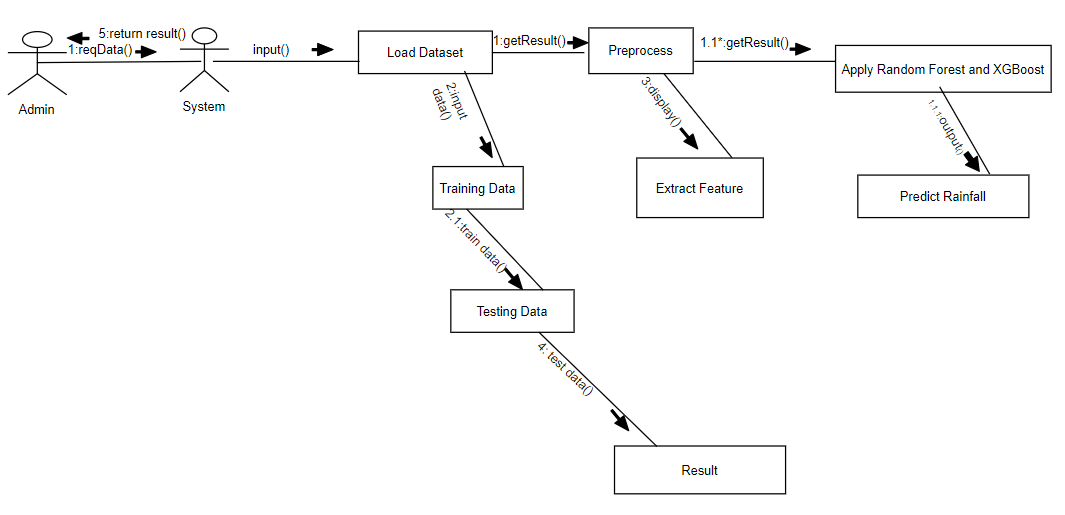


**Fig : 4.3.2 Sequence Diagram**

In these Sequence Diagram, we can see that there are two objects which are interacting with each other, namely Admin and System. Firstly, We have to load the dataset, then, we have to read, preprocess and extract the dataset. After that, We have to train the dataset by applying the Random Forest and Extreme Gradient Boosting Algorithms. So, It will help us to predict the rainfall and then generate the final result.

**4.3.3 COLLABORATION DIAGRAM**

The Collaboration Diagram is used to show the relationship between the objects in a system. Both the sequence and the collaboration diagrams represent the same information but differently. Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming. An object consists of several features. Multiple objects present in the system are connected to each other. The collaboration diagram, which is also known as a communication diagram, is used to portray the object's architecture in the system.

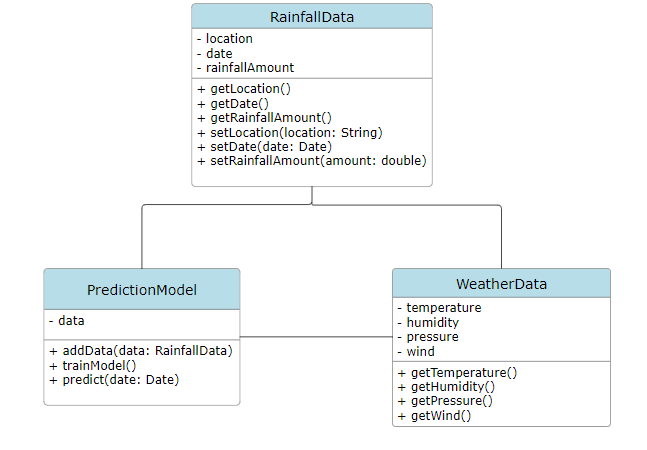


**Fig : 4.3.3 Collaboration Diagram**

In these Collaboration Diagram, it is used to show the relationship between the objects in a system. Both the sequence and the collaboration diagrams represent the same information but differently. Firstly, We have to load the dataset, then, we have to read, preprocess and extract the dataset. After that, We have to train and test the data by using the Random Forest and Extreme Gradient Boosting Algorithms. So, It will help us to predict the rainfall and then generate the final result.

**4.3.4 CLASS DIAGRAM**

A Class Diagram in the [Unified Modeling Language](https://en.wikipedia.org/wiki/Unified_Modeling_Language) (UML) is a type of static structure diagram that describes the structure of a system by showing the system's [classes](https://en.wikipedia.org/wiki/Class_(computer_science)), their attributes, operations (or methods), and the relationships among objects. Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of objectoriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

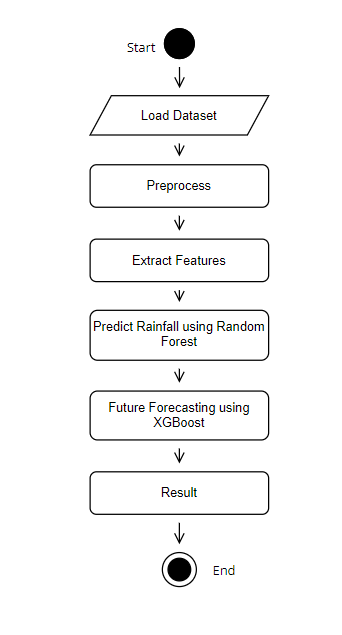


**Fig : 4.3.4 Class Diagram**

In these Class Diagram, describes the structure of a system by showing the system's [classes](https://en.wikipedia.org/wiki/Class_(computer_science)), their attributes, operations (or methods), and the relationships among objects. Firstly, We have to load the dataset, then, we have to read, preprocess and extract the dataset. After that, We have to train and test the data by using the Random Forest and Extreme Gradient Boosting Algorithms. So, It will help us to predict the rainfall and then generate the final result.We have to perform the operations for the given dataset.

**4.3.5 ACTIVITY DIAGRAM**

An Activity Diagram is a type of Unified Modeling Language (UML) flowchart that shows the flow from one activity to another in a system or process. It's used to describe the different dynamic aspects of a system and is referred to as a 'behavior diagram' because it describes what should happen in the modeled system.

****

**Fig : 4.3.5 Activity Diagram**

In these Activity Diagram, it shows the flow from one activity to another in a system or process. Firstly, We have to load the dataset, then, we have to preprocess the dataset and extract the dataset. After that, We have predict the rainfall using the Random Forest and Future Forecasting using the Extreme Gradient Boosting . Finally,it will generate the result.

**CHAPTER - 5**

**5.METHODOLOGY**

**5.1 TECHNOLOGIES**

**Python**

Python is a [high-level](https://en.wikipedia.org/wiki/High-level_programming_language), [general-purpose programming language](https://en.wikipedia.org/wiki/General-purpose_programming_language). Its design philosophy emphasizes [code readability](https://en.wikipedia.org/wiki/Code_readability) with the use of [significant indentation](https://en.wikipedia.org/wiki/Off-side_rule). Python is [dynamically typed](https://en.wikipedia.org/wiki/Type_system#DYNAMIC) and [garbage-collected](https://en.wikipedia.org/wiki/Garbage_collection_(computer_science)). It supports multiple [programming paradigms](https://en.wikipedia.org/wiki/Programming_paradigm), including [structured](https://en.wikipedia.org/wiki/Structured_programming) (particularly [procedural](https://en.wikipedia.org/wiki/Procedural_programming)), [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) and [functional programming](https://en.wikipedia.org/wiki/Functional_programming). It is often described as a "batteries included" language due to its comprehensive [standard library](https://en.wikipedia.org/wiki/Standard_library).

[Guido van Rossum](https://en.wikipedia.org/wiki/Guido_van_Rossum) began working on Python in the late 1980s as a successor to the [ABC programming language](https://en.wikipedia.org/wiki/ABC_(programming_language)) and first released it in 1991 as Python 0.9.0. Python 2.0 was released in 2000. Python 3.0, released in 2008, was a major revision not completely [backward-compatible](https://en.wikipedia.org/wiki/Backward_compatibility) with earlier versions. Python 2.7.18, released in 2020, was the last release of Python 2.

Python consistently ranks as one of the most popular programming languages, and has gained widespread use in the [machine learning](https://en.wikipedia.org/wiki/Machine_learning) community. Python is a [multi-paradigm programming language](https://en.wikipedia.org/wiki/Multi-paradigm_programming_language). [Object-oriented programming](https://en.wikipedia.org/wiki/Object-oriented_programming) and [structured programming](https://en.wikipedia.org/wiki/Structured_programming) are fully supported, and many of their features support functional programming and [aspect-oriented programming](https://en.wikipedia.org/wiki/Aspect-oriented_programming) (including [metaprogramming](https://en.wikipedia.org/wiki/Metaprogramming) and [metaobjects](https://en.wikipedia.org/wiki/Metaobject)). Many other paradigms are supported via extensions, including [design by contract](https://en.wikipedia.org/wiki/Design_by_contract) and [logic programming](https://en.wikipedia.org/wiki/Logic_programming).

Python uses [dynamic typing](https://en.wikipedia.org/wiki/Dynamic_typing) and a combination of [reference counting](https://en.wikipedia.org/wiki/Reference_counting) and a cycle-detecting garbage collector for [memory management](https://en.wikipedia.org/wiki/Memory_management). It uses dynamic [name resolution](https://en.wikipedia.org/wiki/Name_resolution_(programming_languages)) ([late binding](https://en.wikipedia.org/wiki/Late_binding)), which binds method and variable names during program execution.Its design offers some support for functional programming in the [Lisp](https://en.wikipedia.org/wiki/Lisp_(programming_language)) tradition. It has filter, map and reduce functions; [list comprehensions](https://en.wikipedia.org/wiki/List_comprehension), [dictionaries](https://en.wikipedia.org/wiki/Associative_array), sets, and [generator](https://en.wikipedia.org/wiki/Generator_(computer_programming)) expressions. The standard library has two modules (itertools and functools) that implement functional tools borrowed from [Haskell](https://en.wikipedia.org/wiki/Haskell) and [Standard ML](https://en.wikipedia.org/wiki/Standard_ML).

Its core philosophy is summarized in the [Zen of Python](https://en.wikipedia.org/wiki/Zen_of_Python) (PEP 20), which includes [aphorisms](https://en.wikipedia.org/wiki/Aphorism) such as:

* Beautiful is better than ugly.
* Explicit is better than implicit.
* Simple is better than complex.
* Complex is better than complicated.
* Readability counts.

Rather than building all of its functionality into its core, Python was designed to be highly [extensible](https://en.wikipedia.org/wiki/Extensibility) via modules. This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications. Van Rossum's vision of a small core language with a large standard library and easily extensible interpreter stemmed from his frustrations with [ABC](https://en.wikipedia.org/wiki/ABC_(programming_language)), which espoused the opposite approach.

Python strives for a simpler, less-cluttered syntax and grammar while giving developers a choice in their coding methodology. In contrast to [Perl](https://en.wikipedia.org/wiki/Perl)'s "[there is more than one way to do it](https://en.wikipedia.org/wiki/There_is_more_than_one_way_to_do_it)" motto, Python embraces a "there should be one—and preferably only one—obvious way to do it" philosophy. [Alex Martelli](https://en.wikipedia.org/wiki/Alex_Martelli), a [Fellow](https://en.wikipedia.org/wiki/Fellow) at the [Python Software Foundation](https://en.wikipedia.org/wiki/Python_Software_Foundation) and Python book author, wrote: "To describe something as 'clever' is *not* considered a compliment in the Python culture." One benefit of this approach is greater consistency across the Python user communities and shared understanding of the coding principles.

Python's developers usually strive to avoid [premature optimization](https://en.wikipedia.org/wiki/Premature_optimization) and reject patches to non-critical parts of the [CPython](https://en.wikipedia.org/wiki/CPython) reference implementation that would offer marginal increases in speed at the cost of clarity. Execution speed can be improved by moving speed-critical functions to extension modules written in languages such as C, or by using a [just-in-time compiler](https://en.wikipedia.org/wiki/Just-in-time_compilation) like [PyPy](https://en.wikipedia.org/wiki/PyPy). It is also possible to [cross-compile to other languages](https://en.wikipedia.org/wiki/Python_(programming_language)#Cross-compilers_to_other_languages), but it either doesn't provide the full speed-up that might be expected, since Python is a very dynamic language, or a restricted subset of Python is compiled, and possibly semantics are slightly changed. Python's developers aim for it to be fun to use.

This is reflected in its name a tribute to the British comedy group [Monty Python](https://en.wikipedia.org/wiki/Monty_Python) and in occasionally playful approaches to tutorials and reference materials, such as the use of the terms "spam" and "eggs" (a reference to [a Monty Python sketch](https://en.wikipedia.org/wiki/Spam_(Monty_Python))) in examples, instead of the often-used ["foo" and "bar"](https://en.wikipedia.org/wiki/Foobar).A common [neologism](https://en.wikipedia.org/wiki/Neologism) in the Python community is *pythonic*, which has a wide range of meanings related to program style. "Pythonic" code may use Python [idioms](https://en.wikipedia.org/wiki/Programming_idiom) well, be natural or show fluency in the language, or conform with Python's minimalist philosophy and emphasis on readability. Code that is difficult to understand or reads like a rough transcription from another programming language is called unpythonic.

**Anaconda Navigator**

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda® Distribution that allows you to launch applications and manage conda packages, environments, and channels without using command line interface (CLI) commands. Navigator can search for packages on Anaconda.org or in a local Anaconda Repository. It is available for Windows, macOS, and Linux.

Anaconda is a [distribution](https://en.wikipedia.org/wiki/Software_distribution) of the [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) and [R](https://en.wikipedia.org/wiki/R_(programming_language)) [programming languages](https://en.wikipedia.org/wiki/Programming_language) for [scientific computing](https://en.wikipedia.org/wiki/Scientific_computing) ([data science](https://en.wikipedia.org/wiki/Data_science), [machine learning](https://en.wikipedia.org/wiki/Machine_learning) applications, large-scale [data processing](https://en.wikipedia.org/wiki/Data_processing), [predictive analytics](https://en.wikipedia.org/wiki/Predictive_analytics), etc.), that aims to simplify [package management](https://en.wikipedia.org/wiki/Package_management) and [deployment](https://en.wikipedia.org/wiki/Deployment_environment). The distribution includes data-science packages suitable for [Windows](https://en.wikipedia.org/wiki/Microsoft_Windows), [Linux](https://en.wikipedia.org/wiki/Linux), and [macOS](https://en.wikipedia.org/wiki/MacOS). It is developed and maintained by Anaconda, Inc., which was founded by Peter Wang and [Travis Oliphant](https://en.wikipedia.org/wiki/Travis_Oliphant) in 2012.As an Anaconda, Inc. product, it is also known as Anaconda Distribution or Anaconda Individual Edition, while other products from the company are Anaconda Team Edition and Anaconda Enterprise Edition, neither of which are free.

Package versions in Anaconda are managed by the package management system [conda](https://en.wikipedia.org/wiki/Conda_(package_manager)).This package manager was spun out as a separate [open-source](https://en.wikipedia.org/wiki/Open_source) package as it ended up being useful on its own and for things other than Python.There is also a small, [bootstrap](https://en.wikipedia.org/wiki/Bootstrapping) version of Anaconda called Miniconda, which includes only conda, Python, the packages they depend on, and a small number of other packages.

**Why use Navigator?**

In order to run, many scientific packages depend on specific versions of other packages. Data scientists often use multiple versions of many packages and use multiple environments to separate these different versions.

The CLI program conda is both a package manager and an environment manager. This helps data scientists ensure that each version of each package has all the dependencies it requires and works correctly.

Navigator is a graphical interface that enables you work with packages and environments without needing to type conda commands in a terminal window. You can use it to find the packages you want, install them in an environment, run the packages, and update them – all inside Navigator.

**Pandas**

Pandas is a [software library](https://en.wikipedia.org/wiki/Software_library) written for the [Python programming language](https://en.wikipedia.org/wiki/Python_(programming_language)) for data manipulation and [analysis](https://en.wikipedia.org/wiki/Data_analysis). In particular, it offers [data structures](https://en.wikipedia.org/wiki/Data_structure) and operations for manipulating numerical tables and [time series](https://en.wikipedia.org/wiki/Time_series). It is [free software](https://en.wikipedia.org/wiki/Free_software) released under the [three-clause BSD license](https://en.wikipedia.org/wiki/3-clause_BSD_license).The name is derived from the term "[panel data](https://en.wikipedia.org/wiki/Panel_data)", an [econometrics](https://en.wikipedia.org/wiki/Econometrics) term for [data sets](https://en.wikipedia.org/wiki/Data_set) that include observations over multiple time periods for the same individuals, as well as a play on the phrase "Python data analysis".[Wes McKinney](https://en.wikipedia.org/wiki/Wes_McKinney) started building what would become Pandas at [AQR Capital](https://en.wikipedia.org/wiki/AQR_Capital) while he was a researcher there from 2007 to 2010.

The development of Pandas introduced into Python many comparable features of working with DataFrames that were established in the [R programming language](https://en.wikipedia.org/wiki/R_(programming_language)). The library is built upon another library, [NumPy](https://en.wikipedia.org/wiki/NumPy).

Pandas – This library helps to load the data frame in a 2D array format and has multiple functions to perform analysis tasks in one go.

Pandas is a powerful Python library primarily used for data manipulation and analysis. While Pandas itself may not directly contribute to rainfall prediction, it plays a crucial role in handling, preprocessing, and analyzing the data used in rainfall prediction systems.It is also used for data loading,data preprocessing,data analysis,model evaluation and validation.

Pandas is a versatile library that plays a crucial role in handling, preprocessing, analyzing, and transforming data in a rainfall prediction system, ultimately contributing to the development of accurate and robust prediction models.

**NumPy**

NumPy is a [library](https://en.wikipedia.org/wiki/Library_(computing)) for the [Python programming language](https://en.wikipedia.org/wiki/Python_(programming_language)), adding support for large, multi-dimensional [arrays](https://en.wikipedia.org/wiki/Array_data_structure) and [matrices](https://en.wikipedia.org/wiki/Matrix_(mathematics)), along with a large collection of [high-level](https://en.wikipedia.org/wiki/High-level_programming_language) [mathematical](https://en.wikipedia.org/wiki/Mathematics) [functions](https://en.wikipedia.org/wiki/Function_(mathematics)) to operate on these arrays. The predecessor of NumPy, Numeric, was originally created by [Jim Hugunin](https://en.wikipedia.org/wiki/Jim_Hugunin) with contributions from several other developers. In 2005, [Travis Oliphant](https://en.wikipedia.org/wiki/Travis_Oliphant) created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is [open-source software](https://en.wikipedia.org/wiki/Open-source_software) and has many contributors.

The core functionality of NumPy is its "ndarray", for *n*-dimensional array, [data structure](https://en.wikipedia.org/wiki/Data_structure). These arrays are [strided](https://en.wikipedia.org/wiki/Stride_of_an_array) views on memory.In contrast to Python's built-in list data structure, these arrays are homogeneously typed: all elements of a single array must be of the same type.

Such arrays can also be views into memory buffers allocated by [C](https://en.wikipedia.org/wiki/C_Programming_Language)/[C++](https://en.wikipedia.org/wiki/C%2B%2B), [Python](https://en.wikipedia.org/wiki/Python_(programming_language)), and [Fortran](https://en.wikipedia.org/wiki/Fortran) extensions to the CPython interpreter without the need to copy data around, giving a degree of compatibility with existing numerical libraries. This functionality is exploited by the SciPy package, which wraps a number of such libraries (notably BLAS and LAPACK). NumPy has built-in support for [memory-mapped](https://en.wikipedia.org/wiki/Memory-mapped_file) ndarrays.

Numpy – Numpy arrays are very fast and can perform large computations in a very short time.

NumPy is another fundamental library in the Python ecosystem, specifically designed for numerical computing. In a rainfall prediction system, NumPy can be incredibly useful for several reasons as data structures and array operations.The data might be stored in multi-dimensional arrays, and NumPy allows for easy manipulation and extraction of relevant subsets of data.NumPy is a critical component of a rainfall prediction system, providing essential tools for data representation, manipulation, mathematical computation, and integration with machine learning libraries. Its efficiency and versatility make it a valuable asset for developing accurate and efficient rainfall prediction models.

**Sklearn**

scikit-learn (formerly scikits.learn and also known as sklearn) is a [free software](https://en.wikipedia.org/wiki/Free_software) [machine learning](https://en.wikipedia.org/wiki/Machine_learning) [library](https://en.wikipedia.org/wiki/Library_(computing)) for the [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) [programming language](https://en.wikipedia.org/wiki/Programming_language).It features various [classification](https://en.wikipedia.org/wiki/Statistical_classification), [regression](https://en.wikipedia.org/wiki/Regression_analysis) and [clustering](https://en.wikipedia.org/wiki/Cluster_analysis) algorithms including [support-vector machines](https://en.wikipedia.org/wiki/Support_vector_machine), [random forests](https://en.wikipedia.org/wiki/Random_forests), [gradient boosting](https://en.wikipedia.org/wiki/Gradient_boosting), [*k*-means](https://en.wikipedia.org/wiki/K-means_clustering) and [DBSCAN](https://en.wikipedia.org/wiki/DBSCAN), and is designed to interoperate with the Python numerical and scientific libraries [NumPy](https://en.wikipedia.org/wiki/NumPy) and [SciPy](https://en.wikipedia.org/wiki/SciPy). Scikit-learn is a [NumFOCUS](https://en.wikipedia.org/w/index.php?title=NumFOCUS&action=edit&redlink=1) fiscally sponsored project.

scikit-learn is largely written in Python, and uses [NumPy](https://en.wikipedia.org/wiki/NumPy) extensively for high-performance linear algebra and array operations. Furthermore, some core algorithms are written in [Cython](https://en.wikipedia.org/wiki/Cython) to improve performance. Support vector machines are implemented by a Cython wrapper around [LIBSVM](https://en.wikipedia.org/wiki/LIBSVM); logistic regression and linear support vector machines by a similar wrapper around [LIBLINEAR](https://en.wikipedia.org/wiki/LIBLINEAR). In such cases, extending these methods with Python may not be possible.

scikit-learn integrates well with many other Python libraries, such as [Matplotlib](https://en.wikipedia.org/wiki/Matplotlib) and [plotly](https://en.wikipedia.org/wiki/Plotly) for plotting, [NumPy](https://en.wikipedia.org/wiki/NumPy) for array vectorization, [Pandas](https://en.wikipedia.org/wiki/Pandas_(software)) dataframes, [SciPy](https://en.wikipedia.org/wiki/SciPy), and many more.

We will use Scikit-learn's linear regression model to train our dataset. Once the model is trained, we can give our own inputs for the various columns such as temperature, dew point, pressure, etc. to predict the weather based on these attributes.

Scikit-learn (sklearn) is a widely used machine learning library in Python that provides a diverse range of tools for building predictive models. In a rainfall prediction system, scikit-learn can be leveraged in several ways as model selection,model training,model evaluation and model validation. Overall, scikit-learn is a versatile library that provides essential tools and algorithms for building, training, and evaluating machine learning models in rainfall prediction systems.

**Pickle**

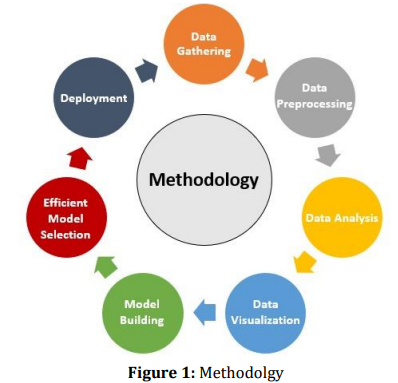
The [pickle](https://docs.python.org/3/library/pickle.html#module-pickle) module implements binary protocols for serializing and de-serializing a Python object structure. “Pickling” is the process whereby a Python object hierarchy is converted into a byte stream, and “unpickling” is the inverse operation, whereby a byte stream (from a [binary file](https://docs.python.org/3/glossary.html#term-binary-file) or [bytes-like object](https://docs.python.org/3/glossary.html#term-bytes-like-object)) is converted back into an object hierarchy. Pickling (and unpickling) is alternatively known as “serialization”, “marshalling,”  or “flattening”; however, to avoid confusion, the terms used here are “pickling” and “unpickling”.

Generally you can pickle any object if you can pickle every attribute of that object. Classes, functions, and methods cannot be pickled -- if you pickle an object, the object's class is not pickled, just a string that identifies what class it belongs to. This works fine for most pickles (but note the discussion about long-term storage of pickles).

With pickle protocol v1, you *cannot* pickle open file objects, network connections, or database connections. When you think about it, it makes sense -- pickle cannot will the connection for file object to exist when you unpickle your object, and the process of creating that connection goes beyond what pickle can automatically do for you. The pickle module can be used for serializing and deserializing Python objects. Here's how pickle could be useful in such a system as model persistence and data persistence.

However, it's essential to note that while pickle is convenient for saving and loading Python objects, it has limitations. Pickled files are specific to Python and may not be compatible across different Python versions or with other programming languages. Additionally, unpickling untrusted data can potentially lead to security vulnerabilities.The above are the different technologies used for the predict the correct accuracy of rainfall prediction.It will predict the data with best and correct accuracy.

**5.2 MODULES**

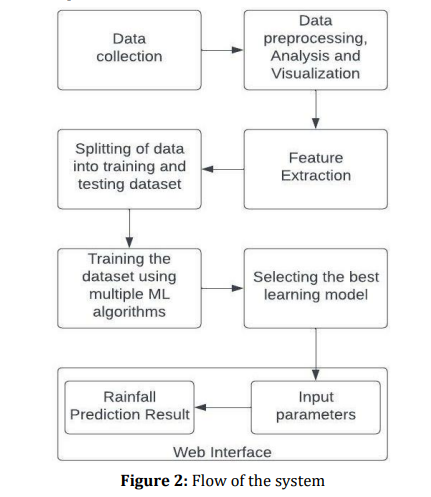


**Fig : 5.2.1 Modules**

The first and most important step in the process is data collection. We have collected our data from the official website of the Indian Government from the year 1951 to 2015 from all the districts and for all months. Next step is Converting the data into the correct format to conduct experiments i.e. doing preprocessing tasks on our dataset like replacing missing and null values with the mean of the column and detecting and removing outliersand visualizing using boxplots. Next step involves analysis of the data and observing variations in the patterns of rainfall to derive conclusions and also to determine the correlation between the different parameters.

After that, we visualize our analyzed data using bar graphs, heat maps, histograms, etc. for a better and easy understanding of the information obtained. After that, we try to predict the average rainfall by separating data into training and testing datasets. We apply various statistical and machine learning approaches like SVR, KNN, logistic regression, etc. in prediction and make analysis over various approaches. At last, we intend to create a simple, user-friendly, and interactive webpage so that users can input different parameters and can get predictions accordingly. Figure 1 depicts the methodology.

Figure 2 explains the basic data flow of the entire methodology and explains each of the individual components of our approach. The different components are data collection, data cleaning and analysis, data visualization, splitting into training and testing set, implementing all the mentioned ML Models, input state, month and other values through a web interface to predict the rainfall.

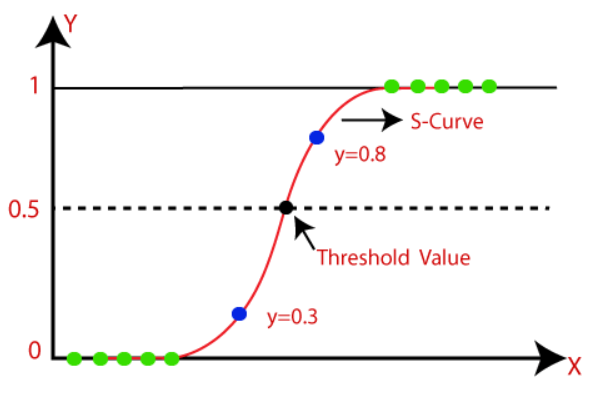


**Fig : 5.2.2 Flow of the System**

**5.3 ALGORITHMS**

**5.3.1 Logistic Regression**

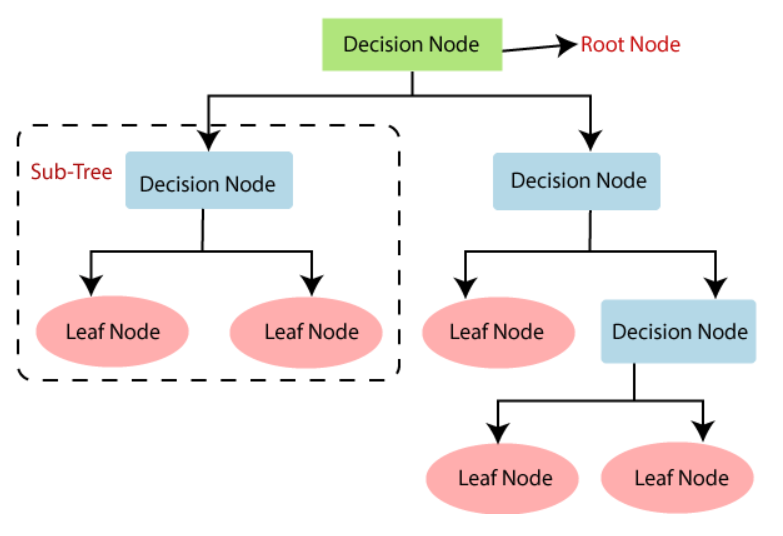
Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems.In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. The below image is showing the logistic function:

****

**Fig : 5.3.1 Logistic Regression**

**5.3.2 Decision Tree**

Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.The decisions or the test are performed on the basis of features of the given dataset.It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions*.*It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm.A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.



**Fig : 5.3.2 Decision Tree**

**Root Node:** Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.

**Leaf Node:** Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.

**Splitting:** Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.

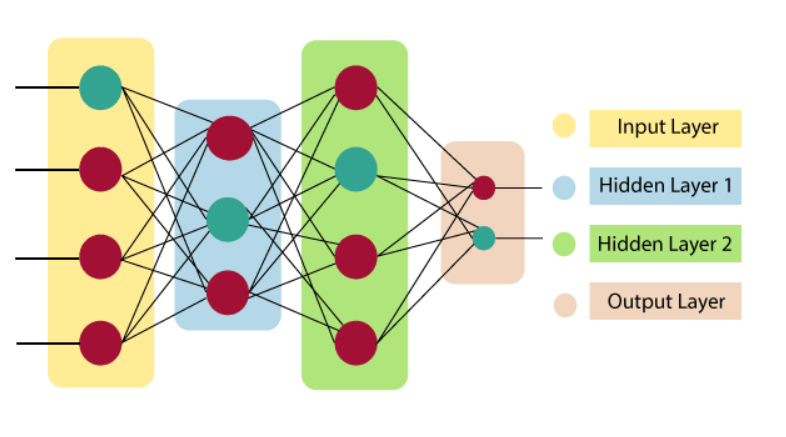
**Branch/Sub Tree:** A tree formed by splitting the tree.

**Pruning:** Pruning is the process of removing the unwanted branches from the tree.

**Parent/Child node:** The root node of the tree is called the parent node, and other nodes are called the child nodes.

**5.3.3 Artificial Neural Network**

The Artificial Neural Network Tutorial covers both fundamental and sophisticated ANN principles. We have designed our Artificial Neural Network tutorial with both novices and experts in mind."Artificial neural network" describes a branch of artificial intelligence that draws inspiration from biology and is structured like the human brain. Generally speaking, an artificial neural network is a computational network that is modeled after the biological neural networks that give the human brain its structure. Artificial neural networks feature neurons that are connected to one another at different layers of the network, just as neurons in a real brain.These neurons are known as nodes.Artificial neural network tutorial covers all the aspects related to the artificial neural network. Building blocks, unsupervised learning, genetic algorithms, Kohonen self-organizing maps, adaptive resonance theory, artificial neural networks, etc.will all be covered in this tutorial.



**Fig : 5.3.3 Artificial Neural Network**

**Input Layer**

As the name suggests, it accepts inputs in several different formats provided by the programmer.

**Hidden Layer**

The hidden layer presents in-between input and output layers. It performs all the calculations to find hidden features and patterns.

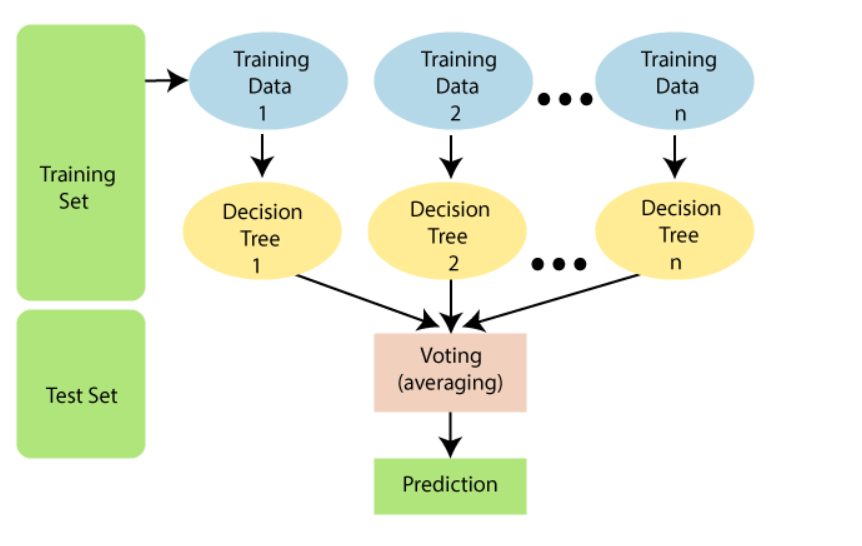
**Output Layer**

The input goes through a series of transformations using the hidden layer, which finally results in output that is conveyed using this layer.

The artificial neural network takes input and computes the weighted sum of the inputs and includes a bias. This computation is represented in the form of a transfer function.

**5.3.4 Random Forest**

Among the supervised learning methods is the well-known machine learning algorithm Random Forest. It can be used for machine learning problems that involve regression and classification. Its basis is the concept of ensemble learning, which is the act of combining multiple classifiers to improve the functionality of the model and solve a difficult problem."Random Forest is a classifier that contains multiple decision trees on different subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset," the name of the algorithm states dataset." Rather than depending on a single decision tree, the random forest forecasts the outcome based on the majority vote of projections from each tree.Because there are more trees in the forest, accuracy is higher and overfitting is avoided.



**Fig : 5.3.4 Random Forest**

An approach for ensemble learning called RF is utilized to solve classification and regression issues. A final prediction is made by combining the results of many decision trees created by this supervised learning technique. A forest of decision trees is created using Random Forest, and each tree is trained using a random subset of the training data and the features. This lessens overfitting and enhances generalization ability. The user can alter two hyperparameters: the size of the subgroups and the number of trees in the forest. Each decision tree in the forest makes a separate prediction about the class or value of the new data point during prediction. The final forecast is then made by averaging or taking the majority vote (for classification) from all of the decision trees' outputs. (in the case of regression).

**5.3.5 XG Boost**

Gradient boosted decision trees are implemented by the XGBoost library of Python, intended for speed and execution, which is the most important aspect of ML (machine learning).

**XgBoost** : XgBoost (Extreme Gradient Boosting) library of Python was introduced at the University of Washington by scholars. It is a module of Python written in C++, which helps ML model algorithms by the training for Gradient Boosting.

**Gradient boosting :** This is an AI method utilized in classification and regression assignments, among others. It gives an expectation model as a troupe of feeble forecast models, commonly called decision trees.

XGBoost is a supervised machine learning algorithm that belongs to the boosting algorithm family. It is an optimized version of the gradient boosting algorithm, whose goal is to make the model faster and more accurate by shortening the time required to calculate the gradient. Due to its prowess in managing big datasets, high accuracy, and handling missing values, XGBoost has become quite well-liked in the data science field. Decision trees, which are weak learners, are systematically added to the model as part of the process, with each new tree being trained to rectify the mistakes produced by its predecessors. To put it another way, XGBoost creates a collection of decision trees, where each tree aims to identify the patterns in the data that were missed by the previous trees.

**CHAPTER - 6**

**6.IMPLEMENTATION**

**6.1 SAMPLE CODE**

**#Importing the Libraries**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import pickle

import time

import matplotlib.gridspec as gridspec

import itertools

from sklearn.utils import resample

import warnings

warnings.filterwarnings("ignore")

from sklearn.experimental import enable\_iterative\_imputer

from sklearn.impute import IterativeImputer

from sklearn.preprocessing import LabelEncoder

from sklearn import preprocessing

from sklearn.feature\_selection import SelectKBest, chi2

from sklearn.feature\_selection import SelectFromModel

from sklearn.metrics import confusion\_matrix

from sklearn.metrics import accuracy\_score, roc\_auc\_score, cohen\_kappa\_score, confusion\_matrix, roc\_curve, classification\_report

from sklearn.preprocessing import StandardScaler

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LogisticRegression

from sklearn.ensemble import RandomForestClassifier

from sklearn.neural\_network import MLPClassifier

from sklearn.tree import DecisionTreeClassifier

import xgboost as xgb

from mlxtend.classifier import EnsembleVoteClassifier

from mlxtend.plotting import plot\_decision\_regions

**#Dataset**

df = pd.read\_csv('rainfall.csv')

df.head()

df.shape

df.info()

df['RainToday'].replace({'No': 0, 'Yes': 1},inplace = True)

df['RainTomorrow'].replace({'No': 0, 'Yes': 1},inplace = True)

fig = plt.figure(figsize = (8,5))

df.RainTomorrow.value\_counts(normalize = True).plot(kind='bar', color= ['skyblue','navy'], alpha = 0.9, rot=0)

plt.title('RainTomorrow Indicator No(0) and Yes(1) in the Imbalanced Dataset')

plt.show()

**#Handling class imbalance for Rainfall Prediction**

no = df[df.RainTomorrow == 0]

yes = df[df.RainTomorrow == 1]

yes\_oversampled = resample(yes, replace=True, n\_samples=len(no), random\_state=123)

oversampled = pd.concat([no, yes\_oversampled])

fig = plt.figure(figsize = (8,5))

oversampled.RainTomorrow.value\_counts(normalize = True).plot(kind='bar', color= ['skyblue','navy'], alpha = 0.9, rot=0)

plt.title('RainTomorrow Indicator No(0) and Yes(1) after Oversampling (Balanced Dataset)')

plt.show()

**# Missing Data**

sns.heatmap(oversampled.isnull(), cbar=False, cmap='PuBu')

plt.show()

total = oversampled.isnull().sum().sort\_values(ascending=False)

percent = (oversampled.isnull().sum()/oversampled.isnull().count()).sort\_values(ascending=False)

missing = pd.concat([total, percent], axis=1, keys=['Total', 'Percent'])

missing.head(4)

**#Imputation and Transformation**

oversampled.select\_dtypes(include=['object']).columns

**# Impute categorical var with Mode**

oversampled['Date'] = oversampled['Date'].fillna(oversampled['Date'].mode()[0])

oversampled['Location'] = oversampled['Location'].fillna(oversampled['Location'].mode()[0])

oversampled['WindGustDir']=oversampled['WindGustDir'].fillna(oversampled['WindGustDir'].mode()[0])

oversampled['WindDir9am']=oversampled['WindDir9am'].fillna(oversampled['WindDir9am'].mode()[0])

oversampled['WindDir3pm']=oversampled['WindDir3pm'].fillna(oversampled['WindDir3pm'].mode()[0])

df2 = oversampled[['Location','WindGustDir', 'WindDir9am' ,'WindDir3pm']]

**# Convert categorical features to continuous features with Label Encoding**

lencoders = {}

for col in oversampled.select\_dtypes(include=['object']).columns:

lencoders[col] = LabelEncoder()

oversampled[col] = lencoders[col].fit\_transform(oversampled[col])

oversampled.head()

**# Multiple Imputation by Chained Equations**

MiceImputed = oversampled.copy(deep=True)

mice\_imputer = IterativeImputer()

MiceImputed.iloc[:, :] = mice\_imputer.fit\_transform(oversampled)

**# Detecting outliers with IQR**

Q1 = MiceImputed.quantile(0.25)

Q3 = MiceImputed.quantile(0.75)

IQR = Q3 - Q1

print(IQR)

**# Removing outliers from the Dataset**

MiceImputed = MiceImputed[~((MiceImputed < (Q1 - 1.5 \* IQR)) |(MiceImputed > (Q3 + 1.5 \* IQR))).any(axis=1)]

MiceImputed.shape

**# Correlation Heatmap**

corr = MiceImputed.corr()

mask = np.triu(np.ones\_like(corr, dtype=np.bool\_))

f, ax = plt.subplots(figsize=(20, 20))

cmap = sns.diverging\_palette(250, 25, as\_cmap=True)

sns.heatmap(corr, mask=mask, cmap=cmap, vmax=None, center=0,square=True, annot=True, linewidths=.5, cbar\_kws={"shrink": .9})

plt.show()

sns.pairplot( data=MiceImputed, vars=('MaxTemp','MinTemp','Pressure9am','Pressure3pm', 'Temp9am', 'Temp3pm', 'Evaporation'), hue='RainTomorrow' )

plt.show()

**#Feature Selection for Rainfall Prediction**

**#Filter Method (Chi-Square)**

**# Standardizing Data**

r\_scaler = preprocessing.MinMaxScaler()

r\_scaler.fit(MiceImputed)

modified\_data = pd.DataFrame(r\_scaler.transform(MiceImputed), index=MiceImputed.index, columns=MiceImputed.columns)

**# Feature Importance using Filter Method (Chi-Square)**

X = modified\_data.loc[:,modified\_data.columns!='RainTomorrow']

y = modified\_data[['RainTomorrow']]

selector = SelectKBest(chi2, k=10)

selector.fit(X, y)

X\_new = selector.transform(X)

print(X.columns[selector.get\_support(indices=True)])

**#Wrapping Method (Random Forest)**

X = MiceImputed.drop('RainTomorrow', axis=1)

y = MiceImputed['RainTomorrow']

selector = SelectFromModel(RandomForestClassifier(n\_estimators=100, random\_state=0))

selector.fit(X, y)

support = selector.get\_support()

features = X.loc[:,support].columns.tolist()

print(features)

print(RandomForestClassifier(n\_estimators=100, random\_state=0).fit(X,y).feature\_importances\_)

**#Training Rainfall Prediction Model with Different Models**

features = MiceImputed[['Location', 'MinTemp', 'MaxTemp', 'Rainfall', 'Evaporation', 'Sunshine', 'WindGustDir', 'WindGustSpeed', 'WindDir9am', 'WindDir3pm', 'WindSpeed9am', 'WindSpeed3pm', 'Humidity9am', 'Humidity3pm', 'Pressure9am', 'Pressure3pm', 'Cloud9am', 'Cloud3pm', 'Temp9am', 'Temp3pm', 'RainToday']]

target = MiceImputed['RainTomorrow']

**# Split into Test and Train**

X\_train, X\_test, y\_train, y\_test = train\_test\_split(features, target, test\_size=0.25, random\_state=12345)

**# Normalize Features**

scaler = StandardScaler()

X\_train = scaler.fit\_transform(X\_train)

X\_test = scaler.fit\_transform(X\_test)

def plot\_roc\_cur(fper, tper):

plt.plot(fper, tper, color='orange', label='ROC')

plt.plot([0, 1], [0, 1], color='darkblue', linestyle='--')

plt.xlabel('False Positive Rate')

plt.ylabel('True Positive Rate')

plt.title('Receiver Operating Characteristic (ROC) Curve')

plt.legend()

plt.show()

def run\_model(model, X\_train, y\_train, X\_test, y\_test, verbose=True):

t0 = time.time()

if verbose == False:

model.fit(X\_train,y\_train, verbose=0)

else:

model.fit(X\_train,y\_train)

y\_pred = model.predict(X\_test)

accuracy = accuracy\_score(y\_test, y\_pred)

roc\_auc = roc\_auc\_score(y\_test, y\_pred)

coh\_kap = cohen\_kappa\_score(y\_test, y\_pred)

time\_taken = time.time()-t0

print("Accuracy = {}".format(accuracy))

print("ROC Area under Curve = {}".format(roc\_auc))

print("Cohen's Kappa = {}".format(coh\_kap))

print("Time taken = {}".format(time\_taken))

print(classification\_report(y\_test,y\_pred,digits=5))

probs = model.predict\_proba(X\_test)

probs = probs[:, 1]

fper, tper, thresholds = roc\_curve(y\_test, probs)

plot\_roc\_cur(fper, tper)

plot\_confusion\_matrix(model, X\_test, y\_test, cmap=plt.cm.Blues, normalize = 'all')

return model, accuracy, roc\_auc, coh\_kap, time\_taken

**# Logistic Regression**

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)

**# Decision Tree**

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)

**# Neural Network**

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)

**# XGBoost**

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)

**# Random Forest**

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)

value = 1.80

width = 0.90

clf1 = LogisticRegression(random\_state=12345)

clf2 = DecisionTreeClassifier(random\_state=12345)

clf3 = MLPClassifier(random\_state=12345, verbose = 0)

clf4 = RandomForestClassifier(random\_state=12345)

clf5 = xgb.XGBClassifier(random\_state=12345)

eclf = EnsembleVoteClassifier(clfs=[clf4, clf5], weights=[1, 1], voting='soft')

X\_list = MiceImputed[["Sunshine", "Humidity9am", "Cloud3pm"]] #took only really important features

X = np.asarray(X\_list, dtype=np.float32)

y\_list = MiceImputed["RainTomorrow"]

y = np.asarray(y\_list, dtype=np.int32)

**# Plotting Decision Regions**

gs = gridspec.GridSpec(3,3)

fig = plt.figure(figsize=(18, 14))

labels = ['Logistic Regression', 'Decision Tree', 'Neural Network','Random Forest','XGBoost','Ensemble']

for clf, lab, grd in zip([clf1, clf2, clf3, clf4, clf5, eclf],labels,itertools.product([0, 1, 2],repeat=2)):

clf.fit(X, y)

ax = plt.subplot(gs[grd[0], grd[1]])

fig = plot\_decision\_regions(X=X, y=y, clf=clf, filler\_feature\_values={2: value}, filler\_feature\_ranges={2: width}, legend=2)

plt.title(lab)

plt.show()

**# Rainfall Prediction Model Comparison**

accuracy\_scores = [accuracy\_lr, accuracy\_dt, accuracy\_nn, accuracy\_rf, accuracy\_xgb]

roc\_auc\_scores = [roc\_auc\_lr, roc\_auc\_dt, roc\_auc\_nn, roc\_auc\_rf, roc\_auc\_xgb]

coh\_kap\_scores = [coh\_kap\_lr, coh\_kap\_dt, coh\_kap\_nn, coh\_kap\_rf, coh\_kap\_xgb]

tt = [tt\_lr, tt\_dt, tt\_nn, tt\_rf, tt\_xgb]

model\_data = {'Model': ['Logistic Regression','Decision Tree','Neural Network','Random Forest','XGBoost'],

'Accuracy': accuracy\_scores, 'ROC\_AUC': roc\_auc\_scores, 'Cohen\_Kappa': coh\_kap\_scores, 'Time taken': tt}

data = pd.DataFrame(model\_data)

fig, ax1 = plt.subplots(figsize=(12,10))

ax1.set\_title('Model Comparison: Accuracy and Time taken for execution', fontsize=13)

color = 'tab:green'

ax1.set\_xlabel('Model', fontsize=13)

ax1.set\_ylabel('Time taken', fontsize=13, color=color)

ax2 = sns.barplot(x='Model', y='Time taken', data = data, palette='summer')

ax1.tick\_params(axis='y')

ax2 = ax1.twinx()

color = 'tab:red'

ax2.set\_ylabel('Accuracy', fontsize=13, color=color)

ax2 = sns.lineplot(x='Model', y='Accuracy', data = data, sort=False, color=color)

ax2.tick\_params(axis='y', color=color)

fig, ax3 = plt.subplots(figsize=(12,10))

ax3.set\_title('Model Comparison: Area under ROC and Cohens Kappa', fontsize=13)

color = 'tab:blue'

ax3.set\_xlabel('Model', fontsize=13)

ax3.set\_ylabel('ROC\_AUC', fontsize=13, color=color)

ax4 = sns.barplot(x='Model', y='ROC\_AUC', data = data, palette='winter')

ax3.tick\_params(axis='y')

ax4 = ax3.twinx()

color = 'tab:red'

ax4.set\_ylabel('Cohen\_Kappa', fontsize=13, color=color)

ax4 = sns.lineplot(x='Model', y='Cohen\_Kappa', data = data, sort=False, color=color)

ax4.tick\_params(axis='y', color=color)

plt.show()

**#Loading the Model**

pickle.dump(model\_rf, open('model\_rb.pkl', 'wb'))

pickle.dump(model\_xgb , open('model\_xgb.pkl', 'wb'))

model = pickle.load(open('model\_xgb.pkl', 'rb'))

input1 = [[12,4.4,12.8,0,2.2,6.1,8,22,8,8,6,7,77,50,1022.5,1019.5,7,4,7.1,12.4,0]]

prediction1 = model.predict(input1)

pred = int(prediction1[0])

if pred == 0:

print("Tomorrow will be no Rain fall")

else:

print("Tomorrow will be Rain fall")

**#Code for the creation of GUI**

import tkinter as tk

from tkinter import ttk, messagebox

from PIL import Image, ImageTk

import pickle

class RainfallPredictionApp:

def \_\_init\_\_(self, root):

self.root = root

self.root.title("Rainfall Prediction App")

# Load the background image

self.background\_image = Image.open("C:/Users/sulta/Downloads/MP1/background.jpg")

self.background\_image = self.background\_image.resize((1600, 1000)) # Resize the background image

self.background\_photo = ImageTk.PhotoImage(self.background\_image)

self.background\_label = tk.Label(self.root, image=self.background\_photo)

self.background\_label.place(x=0, y=0, relwidth=1, relheight=1)

# Frame for input fields

self.frame = ttk.Frame(self.root)

self.frame.place(relx=0.5, rely=0.5, anchor="center")

# Labels and Entry fields for input data

self.label\_location = ttk.Label(self.frame, text="Location:")

self.label\_location.grid(row=0, column=0, padx=5, pady=5, sticky="e")

self.location\_entry = ttk.Entry(self.frame)

self.location\_entry.grid(row=0, column=1, padx=5, pady=5)

self.label\_date = ttk.Label(self.frame, text="Date:")

self.label\_date.grid(row=1, column=0, padx=5, pady=5, sticky="e")

self.date\_entry = ttk.Entry(self.frame)

self.date\_entry.grid(row=1, column=1, padx=5, pady=5)

# Predict button

self.predict\_button = ttk.Button(self.frame, text="Predict", command=self.predict)

self.predict\_button.grid(row=2, columnspan=2, padx=5, pady=5)

def predict(self):

try:

# Get input data

location = self.location\_entry.get()

date = self.date\_entry.get()

# Check if both location and date are provided

if not location or not date:

messagebox.showwarning("Warning", "Please provide both location and date.")

return

# Load trained models

model\_rf = pickle.load(open('model\_rf.pkl', 'rb'))

model\_xgb = pickle.load(open('model\_xgb.pkl', 'rb'))

print("Models loaded successfully")

# Perform prediction for rainfall condition

#input\_data = [12,4.4,12.8,0,2.2,6.1,8,22,8,8,6,7,77,50,1022.5,1019.5,7,4,7.1,12.4,0]

#input\_data= [2,15.9,21.7,2.2,5.6,10.0,13,31.0,3,7,15.0,13.0,89.0,91.0,1010.5,1004.2,8.0,8.0,15.9,17.0,1]

input\_data = [25, 80, 1015, 10, 10.0, 6.0, 20, 20, 30, 20, 12, 14, 90, 60, 1010.0, 1005.0, 15, 10, 8.0, 15.0, 1]

prediction\_rf = model\_rf.predict([input\_data])[0]

prediction\_xgb = model\_xgb.predict([input\_data])[0]

print("Prediction RF:", prediction\_rf)

print("Prediction XGB:", prediction\_xgb)

# Display the prediction using images

if prediction\_rf == 0:

result\_image\_path = "C:/Users/sulta/Downloads/MP1/norainfall.jpg"

else:

result\_image\_path = "C:/Users/sulta/Downloads/MP1/rainfall.jpg"

# Create a new window to display the result image

result\_window = tk.Toplevel(self.root)

result\_window.title("Prediction Result")

# Load and display the result image

result\_image = Image.open(result\_image\_path)

result\_image = result\_image.resize((1600, 900)) # Resize the result image

result\_photo = ImageTk.PhotoImage(result\_image)

result\_label = tk.Label(result\_window, image=result\_photo)

result\_label.image = result\_photo

result\_label.pack()

except Exception as e:

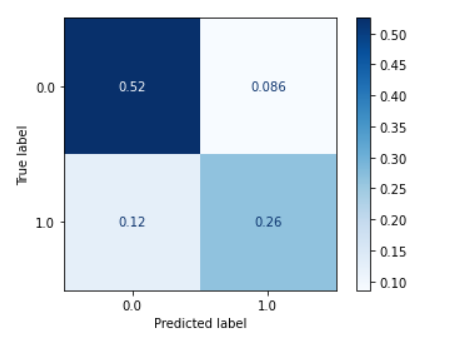
messagebox.showerror("Error", f"An error occurred: {str(e)}")

root = tk.Tk()

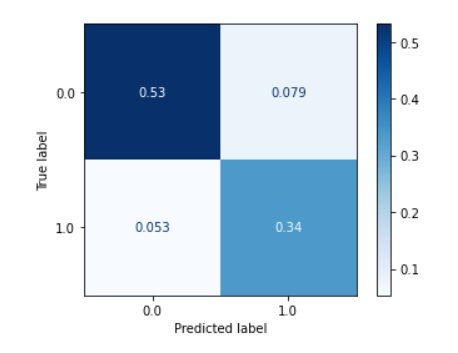
app = RainfallPredictionApp(root)

root.mainloop()

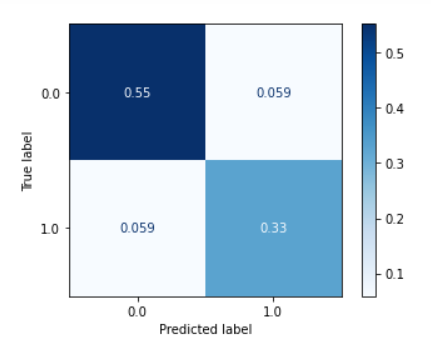
**6.2 OUTPUT SCREENS**



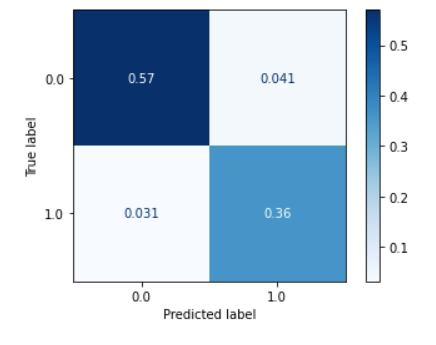
**Fig 6.2.1 : Confusion Matrix for Logistic Regression**

****

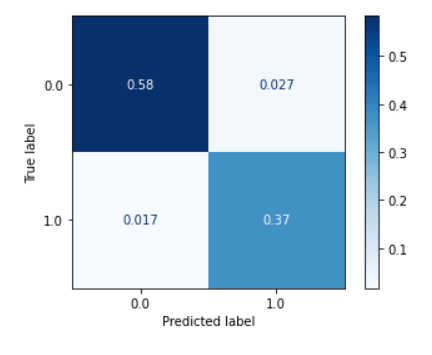
**Fig 6.2.2 : Confusion Matrix for Decision Tree**

****

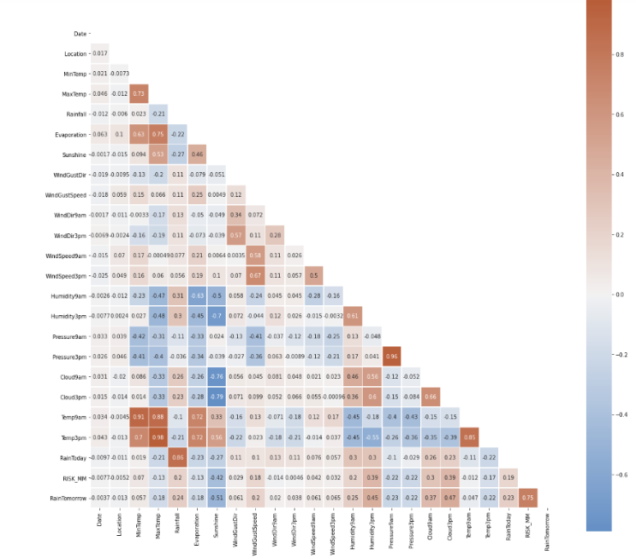
**Fig 6.2.3 : Confusion Matrix for Neural Network**

****

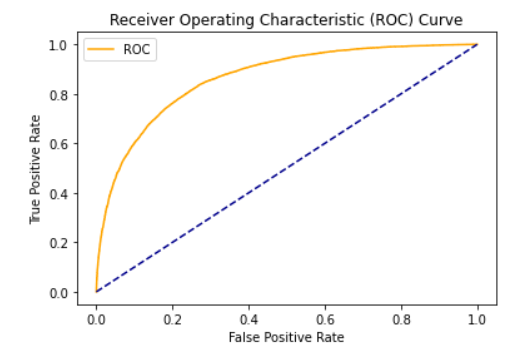
**Fig 6.2.4 : Confusion Matrix for Random Forest**

****

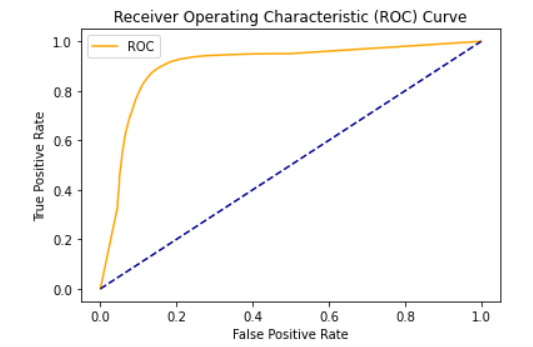
**Fig 6.2.5 : Confusion Matrix for XG Boost**



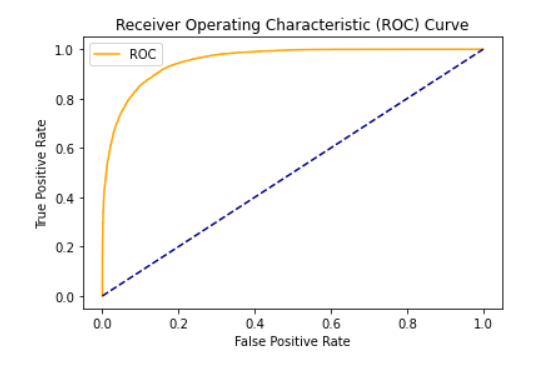
**Fig 6.2.6 : Corelation Heat Map**



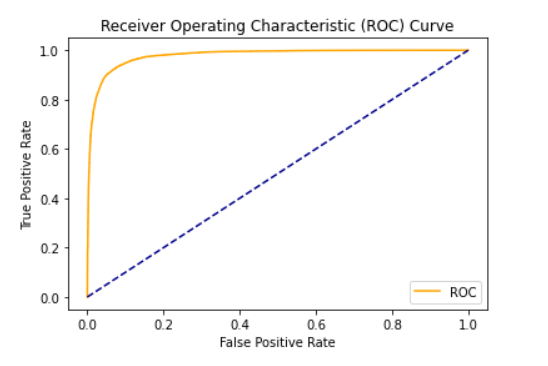
**Fig 6.2.7 : ROC of Logistic Regression**



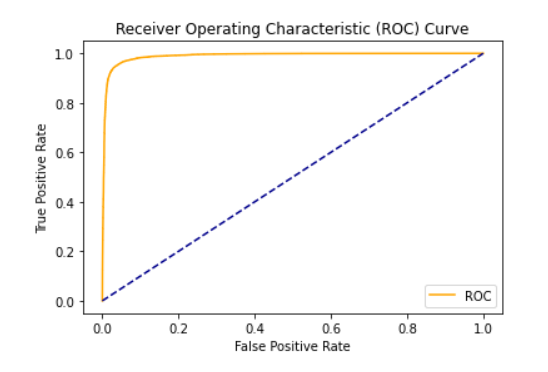
**Fig 6.2.8 : ROC of Decision Tree**



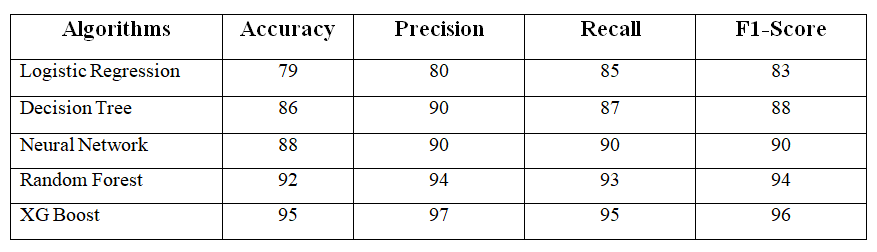
**Fig 6.2.9 : ROC of Neural Network**



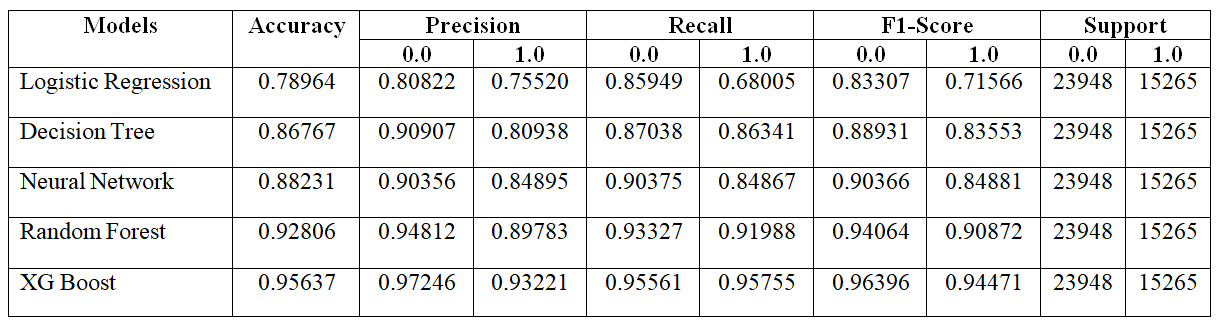
**Fig 6.2.10 : ROC of Random Forest**



**Fig 6.2.11 : ROC of XG Boost**



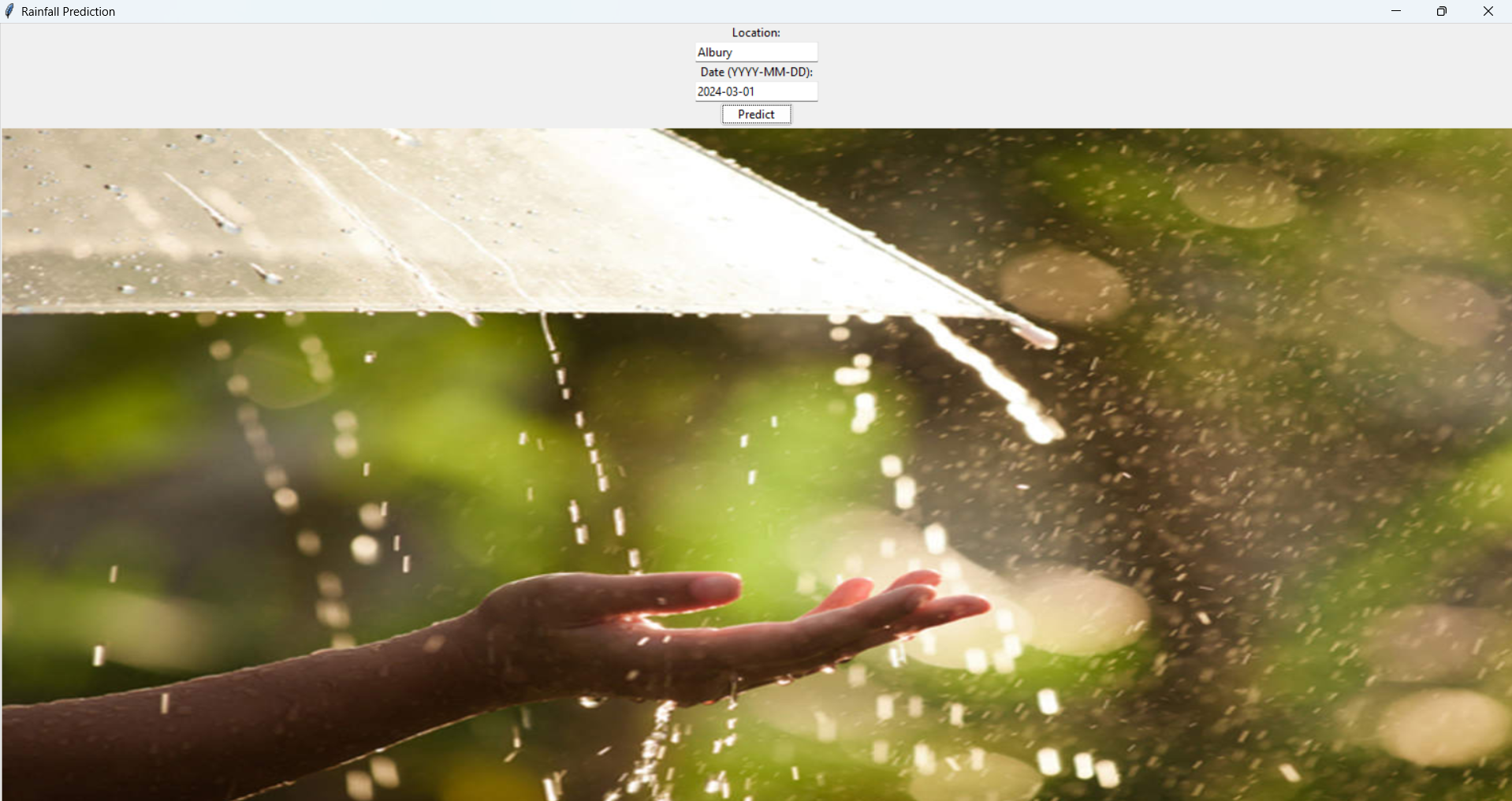
**Fig 6.2.12 : Evaluation Metrics**



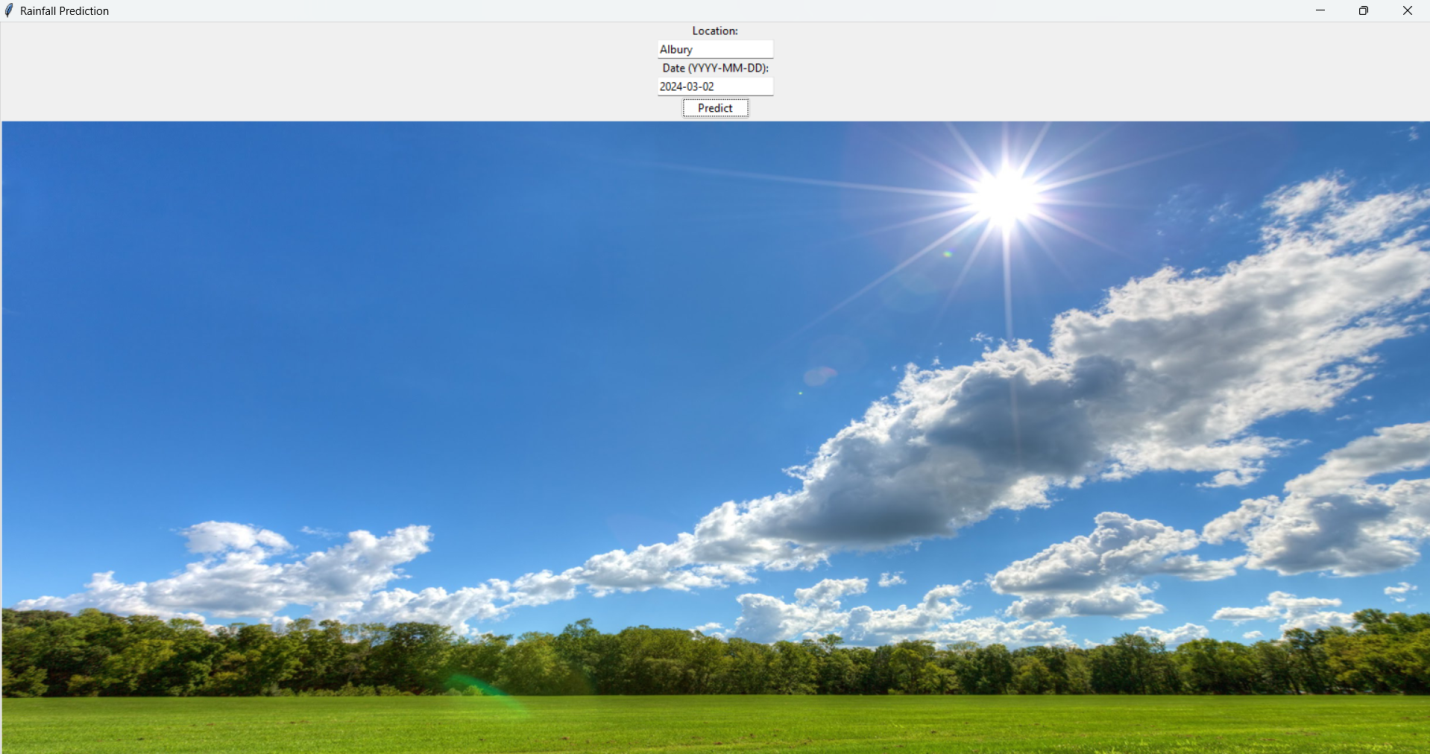
**Fig 6.2.13 : Model Evaluation**

****

**Fig 6.2.14 : Home Screen**

****

**Fig 6.2.15 : Rainy Day**

****

**Fig 6.2.16 : Sunny Day**

**CHAPTER - 7**

**7.CONCLUSION**

Machine learning and data science are employed in the application area of rainfall prediction to foretell atmospheric conditions. Predicting rainfall intensity is crucial for efficient water usage, crop production, and the reduction of rain-related disease and food-related death. This article examined different machine learning algorithms for predicting rainfall. Using the data gathered from the meteorological station in Bahir Dar City, Ethiopia, two machine learning algorithms, FR and XGBoost, were presented and tested.

The input variables for the machine learning model utilized in this paper were the chosen features. The XGBoost algorithm was discovered to be a more effective machine learning method for daily rainfall amount prediction utilizing specified environmental parameters when results from the two algorithms (RF and XGBoost) were compared. If the sensor Data is employed in the study, The precision of the rainfall amount prediction may improve. However, this study did not take into account the sensor data.

Using sensor and meteorological datasets with extra different environmental parameters, the accuracy of the rainfall prediction can be increased. Therefore, if sensor and meteorological information are combined to predict daily rainfall amounts, big data analysis can be employed for rainfall prediction in future studies.

**FUTURE SCOPE**

In the future, this project can be enhanced by

• Improving the model’s accuracy by hyper parameter tuning.

• By deploying into end-to-end application.

• Predicting with live dataset.

More changes & ideas can be adapted such as integrating the system with a flood prediction and alert model pertaining to the rivers. The system can also include another prediction model to predict whether it will rain tomorrow or not based on weather data for that particular day in major cities. Additional Features such as realtime updates & major alerts around the country can be displayed on the website for precautionary measures.

**CHAPTER - 8**

**8.BIBLIOGRAPHY**

[1] Ehsan MA. Seasonal predictability of Ethiopian Kiremt rainfall and forecast skill of ECMWF’s SEAS5 model. Climate Dynamics. 2021; 1–17.

[2] Kusiak A, Verma AP, Roz E. Modeling and prediction of rainfall using radar refectivity data: a data-mining approach. IEEE Trans Geosci Remote Sens. 2013;51:2337–42.

[3] Namitha K, Jayapriya A, SanthoshKumar G. Rainfall prediction using artifcial neural network on map-reduce framework. ACM. 2015. https://doi.org/10.1145/2791405.2791468.

[4] Tharun VP, Prakash R, Devi SR. Prediction of Rainfall Using Data Mining Techniques. In 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT). IEEE Xplore. 2018; pp. 1507–1512.

[5] Zainudin S, Jasim DS, Bakar AA. Comparative analysis of data mining techniques for malaysian rainfall prediction. Int J Adv Sci Eng Inform Technol. 2016;6(6):1148–53.

[6] Manandhar S, Dev S, Lee YH, Meng YS, Winkler S. A data-driven approach for accurate rainfall prediction. IEEE Trans Geosci Remote Sens. 2019;5(11):9323–31.

[7] Arnav G, Kanchipuram Tamil Nadu. Rainfall prediction using machine learning. Int J Innovative Sci Res Technol. 2019. 56–58.

[8] Aswin S, Geetha P, Vinayakumar R. Deep learning models for the prediction of rainfall.. IEEE: New York. 2018; pp. 0657–0661.

[9] Zeelan BCMAK, Bhavana N, Bhavya P, Sowmya V. Proceedings of the International Conference on Electronics and Sustainable Communication Systems (ICESC 2020). Middlesex University: IEEE Xplore. 2020; pp. 92–97.

[10] Aftab S., Ahmad M., Hameed N., Salman M., Ali I., Nawaz Z. Rainfall Prediction in Lahore City using Data Mining Techniques. Int. J. Adv. Comput. Sci. Appl. 2018;9:254–260. doi: 10.14569/IJACSA.2018.090439.

[11] Aftab S., Ahmad M., Hameed N., Salman M., Ali I., Nawaz Z. Rainfall Prediction using Data Mining Techniques: A Systematic Literature Review. Int. J. Adv. Comput. Sci. Appl. 2018;9:143–150. doi: 10.14569/IJACSA.2018.090518.

[12] Nayak M.A., Ghosh S. Prediction of extreme rainfall event using weather pattern recognition and support vector machine classifier. Arch. Meteorol. Geophys. Bioclimatol. Ser. B. 2013;114:583–603. doi: 10.1007/s00704-013-0867-3.