

**Project Report**  
**MACHINE LEARNING APPROACH FOR PREDICTING**  
**RAINFALL**

## **1. INTRODUCTION**

### **1.1 Project Overview**

In the realm of predicting rainfall with machine learning, the process begins with the acquisition of historical weather data. This dataset undergoes preprocessing, involving handling missing values and transforming features for compatibility with algorithms. Exploratory Data Analysis (EDA) uncovers patterns, and feature engineering enhances the model's understanding. The dataset is then divided into training and testing sets. An appropriate algorithm, such as Decision Trees or Gradient Boosting, is chosen for training. Model performance is evaluated using metrics like accuracy, and fine-tuning is performed for optimization. Optionally, interpreting the model's decisions and deploying it to production are considered. Ongoing monitoring ensures model reliability. The success of this iterative process depends on feedback, evaluation, and continuous refinement to adapt to changing conditions, making it a dynamic and evolving approach to rainfall prediction.

### **1.2 Purpose**

The purpose of employing a machine learning approach for rainfall prediction is to enhance our ability to forecast and understand precipitation patterns. This facilitates better preparation and management of resources, particularly in sectors like agriculture, water resource management, and disaster preparedness. By harnessing historical weather data and utilizing advanced algorithms, machine learning models can discern intricate patterns and provide more accurate rainfall predictions. This aids in making informed decisions, optimizing resource allocation, and mitigating the impact of extreme weather events. Ultimately, the purpose is to leverage technology for

proactive and efficient management of water-related challenges, contributing to sustainable and resilient ecosystems.

## **2. LITERATURE SURVEY**

### **2.1 Existing problem**

For the past 20 years, several academics have focused on increasing the precision of machine learning methods for weather forecasting. Here, a few relevant studies are covered. Researchers introduced an ANN-based method for atmospheric condition prediction in. A variety of meteorological parameters, including temperature, wind speed, and humidity, were included in the forecast dataset. The Back Propagation Network and Hopfield Network were combined in the suggested method so that the HN received the BPN's output as input. By examining the non-linear relationship between historical weather parameters, this technique operates. Researchers employed ANN in to forecast India's monsoon weather's monthly average rainfall. A dataset with eight months' worth of data annually was used.

Furthermore, existing studies highlight the impact of climate change on precipitation patterns, necessitating adaptive models. The literature emphasizes ongoing efforts to enhance model interpretability, address data biases, and improve the scalability of algorithms for large datasets.

Collaborative interdisciplinary research emerges as a key theme, emphasizing the need for diverse expertise in meteorology, computer science, and environmental science to tackle the multifaceted challenges in rainfall prediction. Overall, the literature underscores the evolving nature of the field and the continuous quest for innovative solutions to improve the reliability of rainfall forecasts.

### **2.2 References**

- <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9099780/#:~:text=For%20effective%20prediction%20of%20rainfall,city%20of%20Lahore%20is%20considered.>
- <https://www.geeksforgeeks.org/rainfall-prediction-using-machinelearning-python/>
- <https://www.ijraset.com/research-paper/rainfall-prediction-using-ml>
- <https://github.com/topics/rainfall-prediction>

## **2.3 Problem Statement Definition**

The problem statement in rainfall prediction via machine learning involves developing accurate models that forecast precipitation patterns based on historical meteorological data. It includes addressing the challenges of variability, non-linearity, and the influence of multiple environmental factors affecting rainfall. The aim is to create robust predictive models capable of handling spatial and temporal complexities, overcoming issues like data quality, model generalization, and adapting to changing climate conditions. The goal is to devise algorithms that provide reliable and timely predictions, aiding various sectors such as agriculture, disaster preparedness, and water resource management. This problem statement seeks to bridge the gap between traditional meteorological methods and advanced machine learning techniques, ensuring more precise and actionable rainfall forecasts.

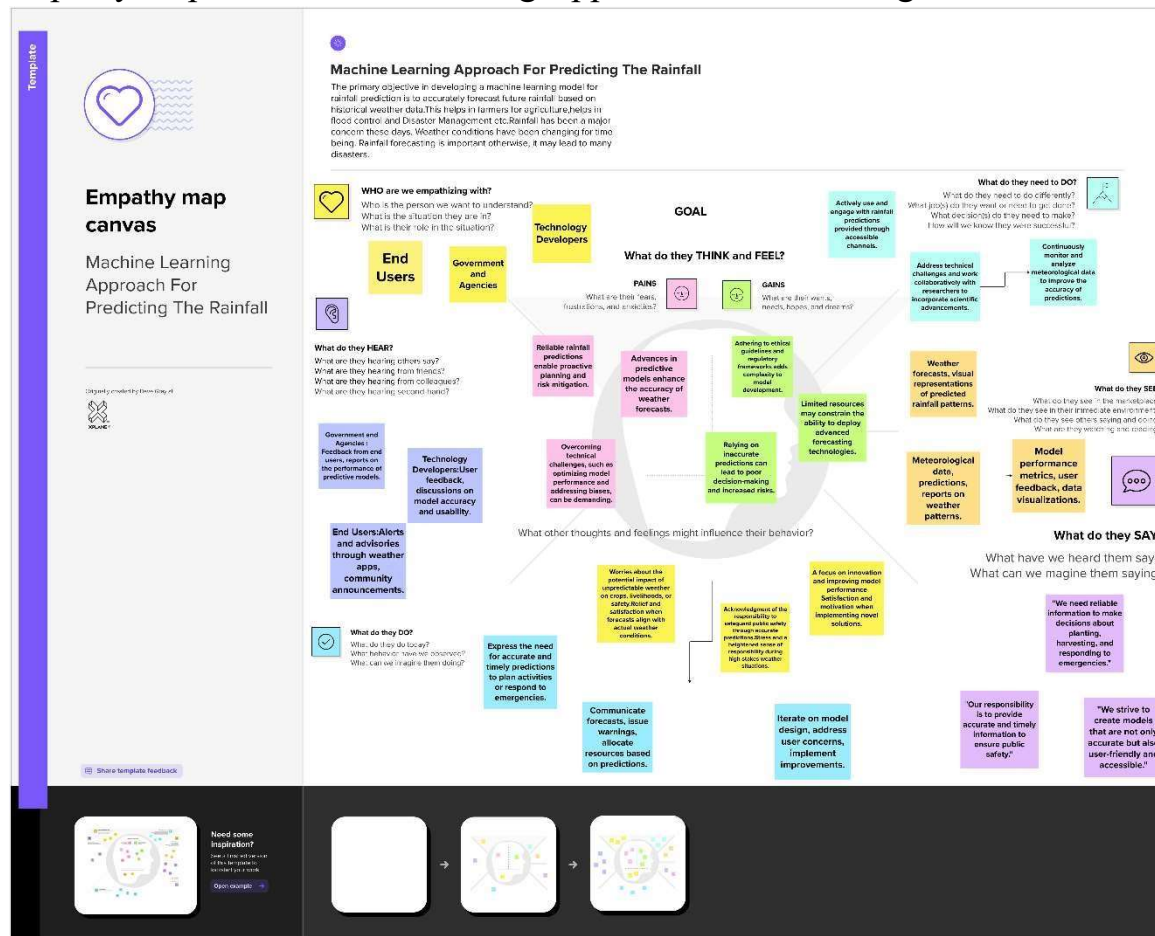
## **3. IDEATION & PROPOSED SOLUTION**

### **3.1 Empathy Map Canvas**

Empathy Map Canvas:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes. It is a useful tool to help teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.

## Empathy map on Machine Learning Approach For Predicting Rainfall




### 3.2 Ideation & Brainstorming

Brainstorm & Idea Prioritization Template: Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions. Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

# Step-1: Team Gathering, Collaboration and Select the Problem Statement

Template



## Brainstorm & idea prioritization

### Machine Learning Approach For Predicting Rainfall

10 minutes to prepare  
1 hour to collaborate  
2-8 people recommended

**Before you collaborate**

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes

1 Team gathering

Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

2 Set the goal

Think about the problem you'll be focusing on solving in the brainstorming session.

3 Learn how to use the facilitation tools

Use the Facilitation Superpowers to run a happy and productive session.

[Open article](#)

**1 Define your problem statement**

Enhancing rainfall prediction Machine learning methods to reduce biases, increase lead time, and improve accuracy. For predictions that are ethical and easily accessed, put an emphasis on cutting-edge algorithms, real-time data (regression, and user-centered design. Anticipated results encompass heightened precision, prolonged forecast duration, and amplified user contentment, cultivating more efficient decision-making inside industries reliant on meteorological predictions.

5 minutes


Facilitator

How might we [your problem statement]?

**Key rules of brainstorming**

To run an smooth and productive session

- 1 Stay in topic.
- 2 Encourage wild ideas.
- 3 Defer judgment.
- 4 Listen to others.
- 5 Go for volume.
- 6 If possible, be visual.

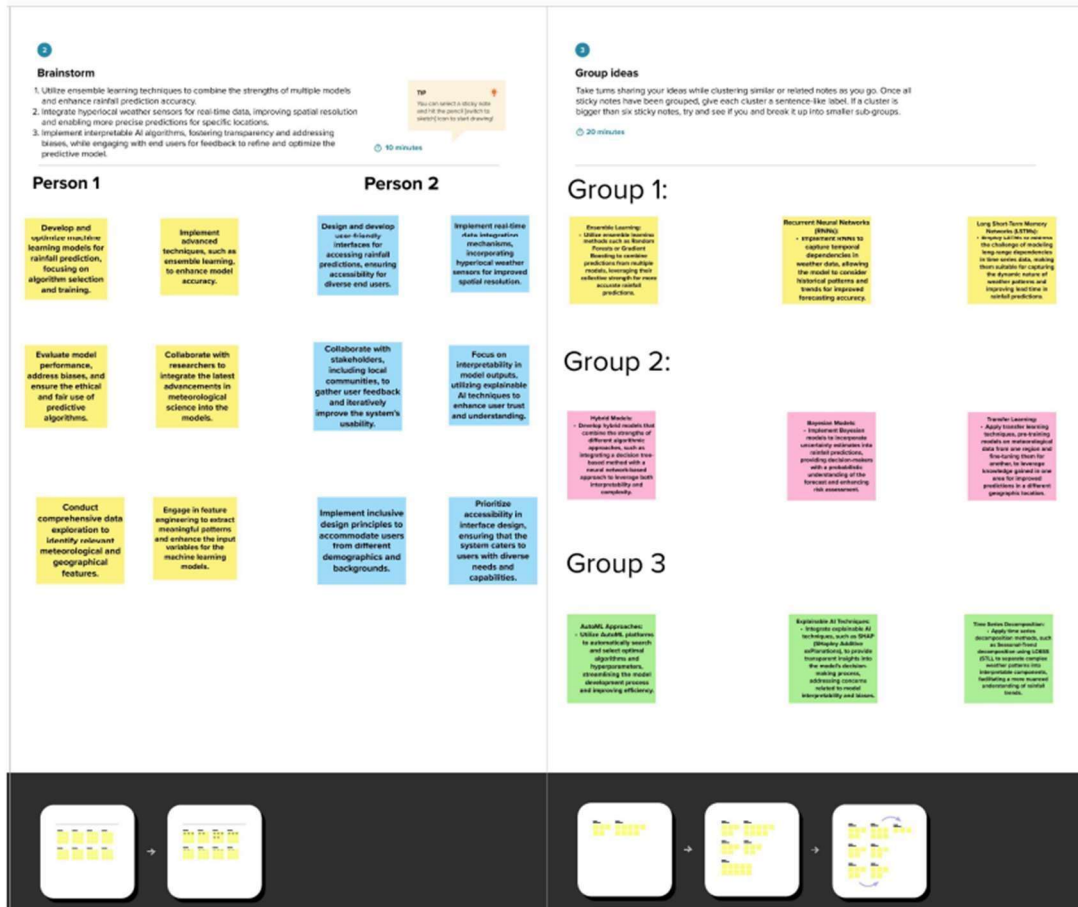


**Need some inspiration?**

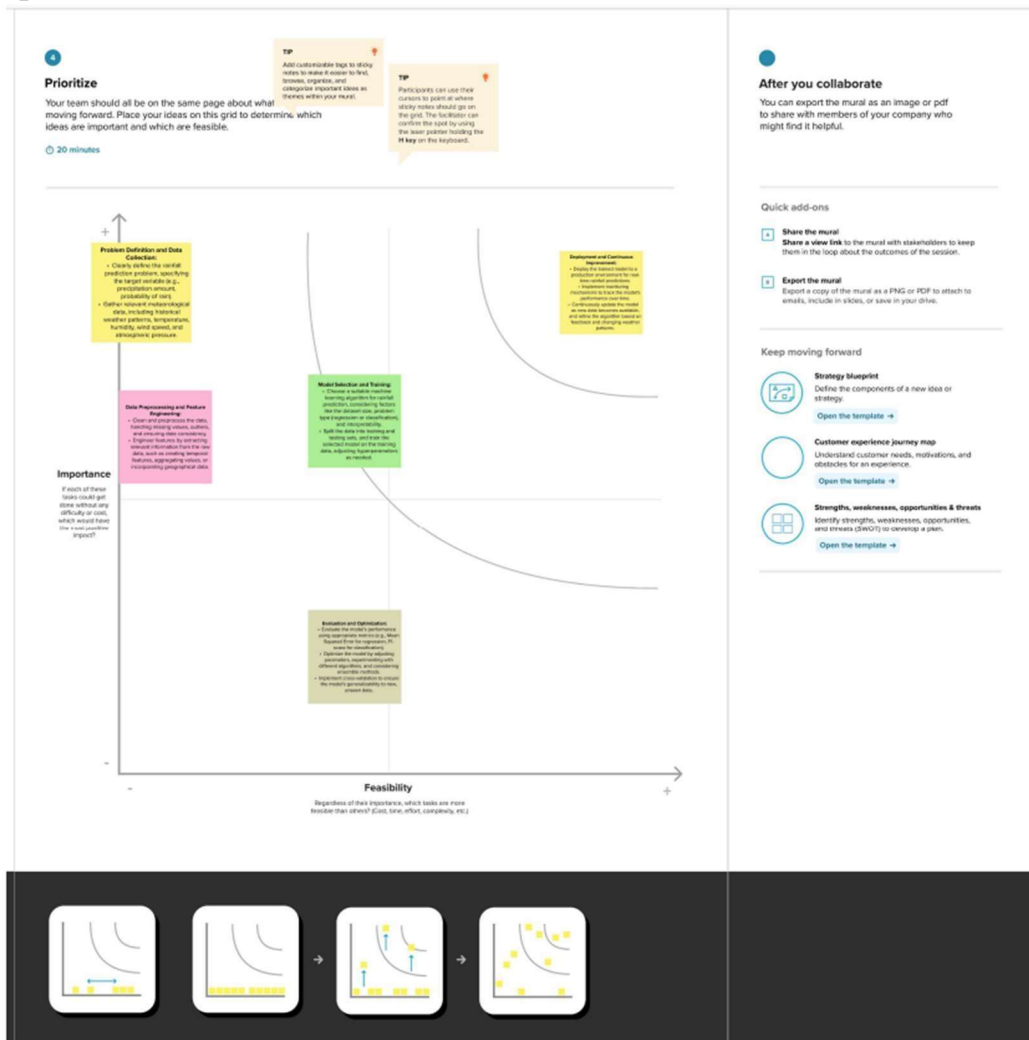
Here's a featured version of this template to bootstrap your work.

[Open example](#)

## Step-2: Brainstorm, Idea Listing and Grouping



## Step-3: Idea Prioritization



## 4. REQUIREMENT ANALYSIS

### 4.1 Functional requirement

A machine learning approach for rainfall prediction is a sophisticated system designed to enhance our understanding and anticipation of weather patterns. By leveraging historical meteorological data and advanced algorithms like XGBoost and Random Forest, the model identifies key features influencing rainfall. Through meticulous data preprocessing and continuous real-time integration, it ensures accuracy and adaptability. The system's user interface offers a seamless experience, presenting precise predictions and probability estimates. An embedded alert mechanism enhances its practicality, notifying authorities and users of potential weather-related risks. Scalability is prioritized, accommodating growing data volumes and user demands. This

innovative solution not only addresses the existing problem of unpredictable rainfall but also contributes to more effective disaster preparedness and resource management. Overall, the machine learning-based rainfall prediction system stands as a pivotal tool in modern meteorology, fostering resilience and informed decision-making in the face of dynamic weather conditions.

#### **4.2 Non-Functional requirements**

The machine learning-based rainfall prediction system serves a vital purpose in enhancing our ability to foresee weather patterns, enabling proactive measures for mitigating the impacts of rainfall-related events. In conducting a literature survey, existing challenges in traditional meteorological approaches have been identified, emphasizing the need for more accurate and timely predictions. The problem statement is defined by the necessity for a system that integrates diverse data sources and employs advanced algorithms to improve prediction accuracy. Functional requirements include scalable, reliable, and user-friendly interfaces that cater to meteorologists and emergency response teams. Additionally, the system must address nonfunctional aspects such as security, adaptability to changing weather patterns, and compliance with data protection regulations. With a focus on usability and accessibility, the system aims to be an indispensable tool for diverse users, fostering collaboration and informed decision-making in the face of variable rainfall conditions.

### **5. PROJECT DESIGN**

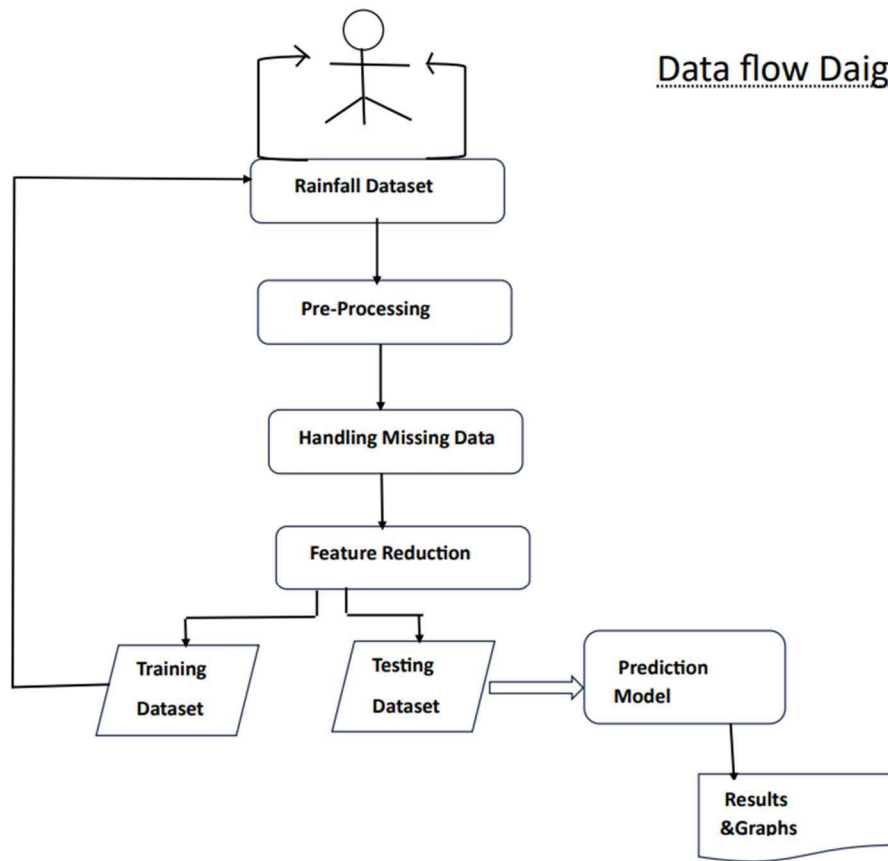
#### **5.1 Data Flow Diagrams & User Stories**

Machine learning Approach for Predicting Rainfall :

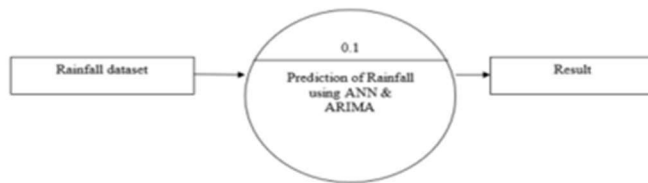
The data flow diagram for the Dynamic Rainfall Prediction System depicts the journey of meteorological data from diverse sources. It begins with the ingestion layer, where real-time satellite data, local weather stations, and additional databases are processed. The information then undergoes preprocessing, real-time integration, and machine learning modeling before being presented to users through the user-friendly mobile application. Continuous improvement mechanisms, explainable AI, and community weather stations contribute to the system's adaptability and accuracy, creating a comprehensive flow that enhances the prediction of rainfall patterns.



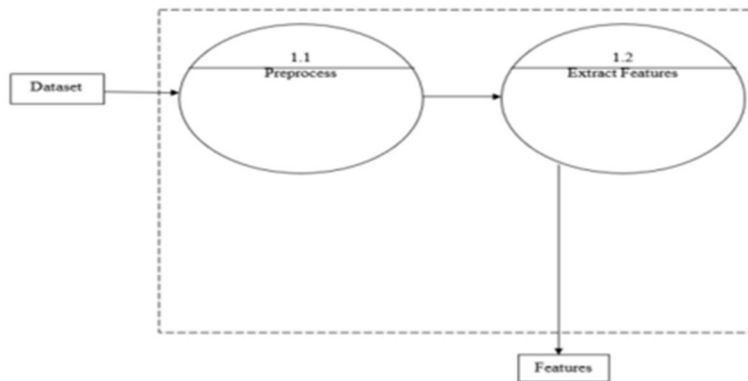
## Data flow Daigram



**Level 0 :**



**Level 1 :**



User Stories

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
-----------	-------------------------------	-------------------	-------------------	---------------------	----------	---------

Farmer	Accessible Forecasting	USN-1	As a farmer, I want to receive accurate and timely rainfall forecasts through a user-friendly mobile application.	The mobile app should provide daily rainfall predictions, display information in a visually intuitive manner, and allow users to set personalized notifications.	High	Data Licensing Feature Version 1.0
Agricultural Organization	Data Integration	USN-2	As an agricultural organization, I want to integrate the Dynamic Rainfall Prediction System into our existing platform to enhance our <u>decisionmaking</u> processes.	The system should provide API access, support seamless data integration, and offer customizable analytics for our specific needs.	Low	Community Engagement Feature Version 1.0
Government Agency	Early Warning System	USN-3	As a government agency, I want to receive real-time alerts and early warnings for potential weather-related disasters.	The system should send immediate alerts for extreme weather events, integrate risk assessment algorithms, and enable quick response coordination.	High	Early Warning Module Version 1.0
Researcher	Data Licensing	USN-4	As a researcher, I want access to the high-resolution meteorological data generated by the system for climate studies.	The system should provide a licensing mechanism for researchers, ensuring data integrity and adherence to privacy standards.	Medium	Data Licensing Feature Version 1.0
Community Member	Community Engagement	USN-5	As a community member, I want to actively participate in weather data collection and engage with educational resources.	The system should offer an online platform with educational content, workshops, and a feedback mechanism, fostering community collaboration.	Low	Community Engagement Feature Version 1.0
System Administrator	Security and Compliance	USN-6	As a system administrator, I want to ensure robust security measures and compliance with privacy regulations.	The system should implement encryption, access controls, and regular compliance audits to safeguard user data.	High	Security Update Version 1.1
Mobile App Developer	API Documentation	USN-7	As a mobile app developer, I want comprehensive and well-documented APIs for seamless integration with the Dynamic Rainfall Prediction System.	The system should provide clear and accessible API documentation, including endpoints, data formats, and authentication methods.	Medium	API Documentation Update Version 1.0
Climate Research Institution	Research Collaboration	USN-8	As a climate research institution, I want to collaborate with the system	The system should establish a collaboration	Medium	Research Collaboration
			developers to enhance climate models and contribute to ongoing research.	framework, including data-sharing agreements and periodic research workshops.		Feature Version 1.0

## 5.2 Solution Architecture

### Example - Solution Architecture Diagram:

The solution architecture for the Dynamic Rainfall Prediction System is designed for flexibility, scalability, and robust performance. The solution

architecture for the Dynamic Rainfall Prediction System is designed for flexibility, scalability, and robust performance. Here's an overview of its key components:

1. **Data Ingestion Layer:** Ingests data from various sources, including high-resolution satellite imagery, local weather stations, and additional meteorological databases.
2. **Preprocessing Module:** Cleans and preprocesses incoming data to handle missing values, outliers, and inconsistencies, ensuring data quality for accurate model training.
3. **Machine Learning Models:** Employs an adaptive ensemble of machine learning models, such as decision trees, neural networks, and gradient boosting, to capture complex relationships and enhance prediction accuracy.
4. **Real-time Data Integration:** Integrates real-time data updates to capture dynamic changes in atmospheric conditions, improving the system's responsiveness to evolving weather patterns.
5. **Community Weather Stations Network:** Establishes a network of community weather stations, contributing hyperlocal data for improved spatial resolution and localized accuracy.
6. **Explainable AI Module:** Implements explainable AI techniques to enhance transparency, allowing users to understand the rationale behind predictions and fostering trust.
7. **User-Friendly Mobile Application:** Develops a mobile application with a user-friendly interface, providing accessible and interpretable rainfall forecasts for farmers and end-users.
8. **Continuous Model Improvement :** Integrates reinforcement learning mechanisms for continuous model improvement, allowing the system to dynamically adapt based on real-time feedback.
9. **Community Engagement Platform:** Establishes an online platform for community engagement, hosting educational resources, workshops, and feedback mechanisms to foster collaboration and user involvement.
10. **Scalable Cloud Infrastructure:** Leverages a scalable cloud-based infrastructure to handle growing data volumes, increasing user demands, and ensuring seamless performance across diverse geographic regions.
11. **APIs for Integration:** Provides APIs for seamless integration with third-party platforms, allowing businesses, agricultural enterprises, and research institutions to leverage the system's predictions.

12. Security and Compliance Layer : Implements robust security measures and compliance protocols to protect sensitive data, ensuring the system adheres to privacy and regulatory standards.

## 6. PROJECT PLANNING & SCHEDULING

### 6.1 Technical Architecture

The technical architecture of the Dynamic Rainfall Prediction System is a cohesive framework designed for accuracy, scalability, and user engagement. It integrates diverse data sources, employs an adaptive ensemble of machine learning models, and incorporates real-time updates for dynamic predictions. Community weather stations enhance local accuracy, while explainable AI techniques foster transparency. The user-friendly mobile application provides accessible forecasts, and continuous model improvement ensures responsiveness to evolving weather patterns. Cloud-based infrastructure ensures scalability, and APIs facilitate seamless integration. Robust security measures and compliance protocols safeguard user data. Monitoring tools and disaster recovery mechanisms ensure system reliability, while interfaces for research collaboration encourage ongoing advancements in climate studies.

**Architecture Daigram :**

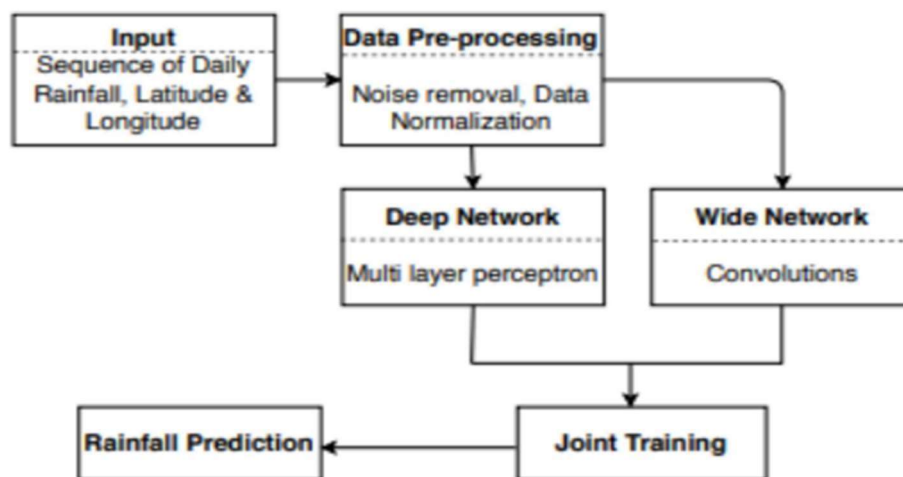


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	The User Interface (UI) is the front-end component that allows users to interact with the Dynamic Rainfall Prediction System. It provides an intuitive and user-friendly platform for accessing and visualizing weather forecasts.	HTML, CSS, JavaScript, React.js for dynamic updates.
2.	Application Logic-1	This logic component handles user authentication and authorization, ensuring secure access to the system's features.	Node.js for server-side logic, JSON Web Tokens (JWT) for authentication.
3.	Application Logic-2	Manages real-time data integration, processing incoming information, and triggering updates to the machine learning models.	Python for data processing, WebSockets for real-time communication.
4.	Application Logic-3	Implements the continuous model improvement, utilizing reinforcement learning mechanisms to enhance the accuracy of rainfall predictions.	Python for model training, TensorFlow or PyTorch for machine learning.
5.	Database	Stores user data, preferences, and historical information for analysis and forecasting improvements.	PostgreSQL for relational data storage.
6.	Cloud Database	A scalable and distributed database hosted in the cloud, ensuring data availability and redundancy.	Amazon Aurora, Google Cloud Firestore, or Azure Cosmos DB.
7.	File Storage	Manages storage for non-relational data, such as satellite images and community-contributed data.	Amazon S3, Google Cloud Storage, or Azure Blob Storage.
8.	External API-1	Integrates with external meteorological data sources to enhance the system's data pool.	RESTful API using HTTP/HTTPS protocols.
9.	External API-2	Connects with a mapping service to provide spatial visualization and enhance user experience.	Mapbox API, Google Maps API, or OpenStreetMap API.
10.	Machine Learning Model	Utilizes an adaptive ensemble of machine learning models for accurate rainfall predictions based on historical and real-time data.	Python, scikit-learn, XGBoost, or TensorFlow.
11.	Infrastructure (Server / Cloud)	Hosts and manages the application, databases, and machine learning models, ensuring reliable and scalable performance.	Docker for containerization, Kubernetes for orchestration, AWS, Google Cloud, or Azure for cloud infrastructure.

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	The system leverages open-source frameworks to foster collaboration, innovation, and transparency in development. This allows for community contributions, rapid updates, and customization based on evolving needs.	Django (Python web framework), React.js (JavaScript library), Flask (Python web framework), and scikit-learn (machine learning library) are examples.
2.	Security Implementations	The system prioritizes robust security measures to safeguard user data, prevent unauthorized access, and ensure the confidentiality and integrity of sensitive information.	HTTPS for secure communication, encryption algorithms (AES, RSA), secure coding practices, and regular security audits.
3.	Scalable Architecture	The architecture is designed to handle growing data volumes, user demands, and feature expansions without compromising performance. It allows the system to scale horizontally or vertically based on changing requirements.	Cloud-based infrastructure (AWS, Google Cloud, Azure), Docker for containerization, Kubernetes for orchestration.
4.	Availability	The system ensures high availability by minimizing downtime, employing redundant components, and implementing failover mechanisms to ensure continuous access to services.	Load balancing, redundant server instances, distributed databases, and disaster recovery planning.
5.	Performance	The system is optimized for performance, providing quick response times and efficient resource utilization. This includes tuning database queries, optimizing code execution, and employing caching mechanisms.	Performance monitoring tools, query optimization techniques, in-memory caching (Redis, Memcached), and content delivery networks (CDNs).

## 6.2 Sprint Planning & Estimation



**Product Backlog, Sprint Schedule, and Estimation (4 Marks)**

Use the below template to create product backlog and sprint schedule

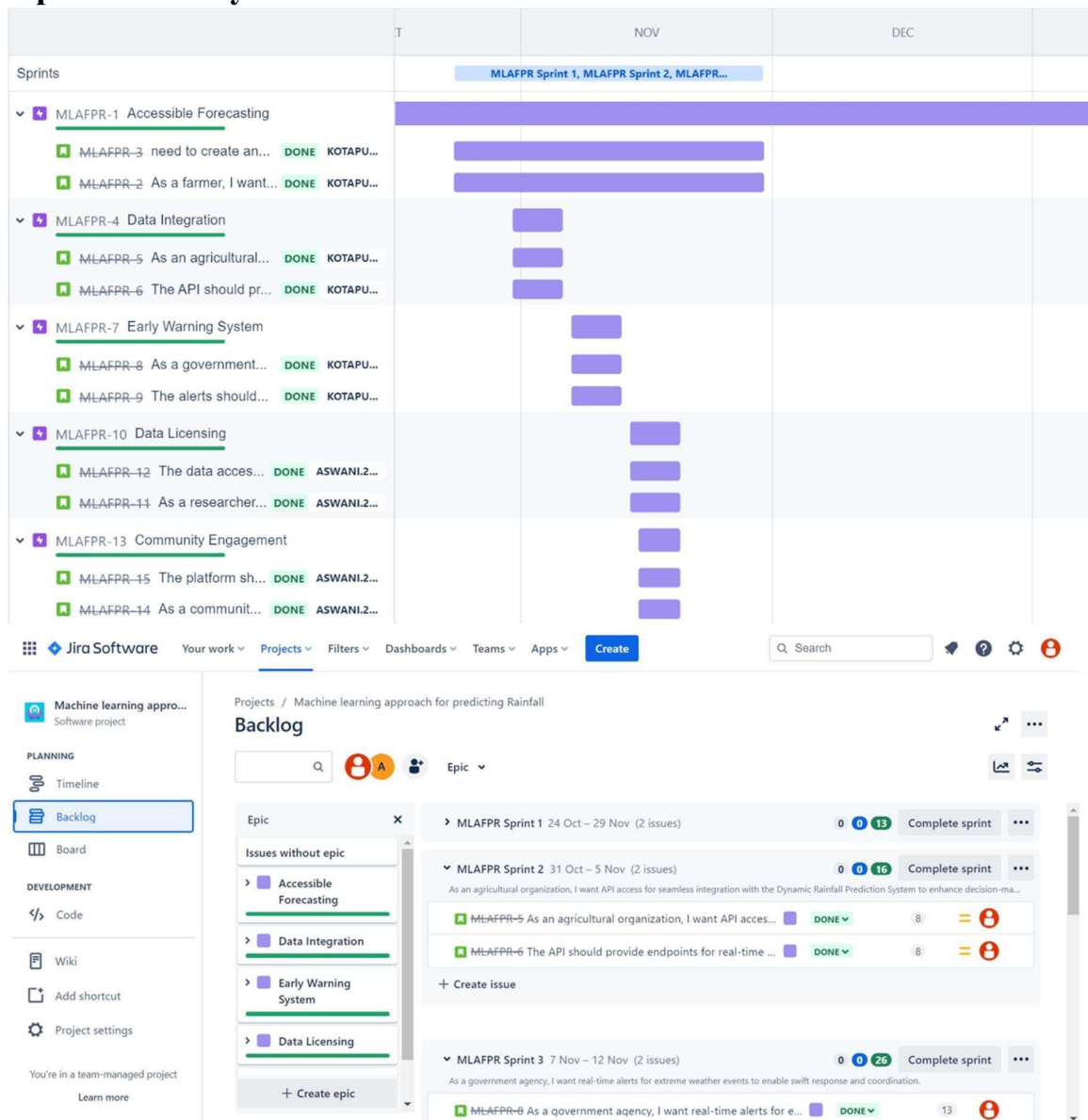
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Accessible Forecasting	USN-1	As a farmer, I want daily rainfall predictions to plan my agricultural activities efficiently. The predictions should include rainfall intensity, duration, and probability, presented in a clear and user-friendly format.	5	High	Supraja & Aswini
Sprint-1	Data Integration	USN-2	As an agricultural organization, I want API access for seamless integration with the Dynamic Rainfall Prediction System to enhance decision-making. The API should provide endpoints for real-time data retrieval, historical data access, and support for customizable analytics.	8	High	Supraja & Aswini
Sprint-2	Early Warning System	USN-3	As a government agency, I want real-time alerts for extreme weather events to enable swift response and coordination. The alerts	13	Highst	Supraja & Aswini

			should be configurable based on predefined risk levels, and the system should prioritize critical weather conditions.			
Sprint-1	Data Licensing	USN-4	As a researcher, I want access to high-resolution meteorological data for climate studies and research. The data access should include a licensing mechanism, ensuring compliance with privacy and data-sharing standards.	3	Low	Supraja & Aswini
Sprint-4	Community Engagement	USN-5	As a community member, I want to actively participate in weather data collection and access educational resources. The platform should provide a user-friendly interface for data submission, educational content, and community-driven workshops.	3	Low	Supraja & Aswini
Sprint-2	Security and Compliance	USN-6	As a system administrator, I want robust security measures to ensure the confidentiality and integrity of user data. The security implementation should include encryption, secure coding practices, and regular audits for compliance.	8	High	Supraja & Aswini
Sprint-3	API Documentation	USN-7	As a developer, I want comprehensive API documentation to facilitate seamless integration with the Dynamic Rainfall Prediction System. The documentation should include clear explanations of endpoints, data formats, and authentication methods.	5	Medium	Supraja & Aswini
Sprint-2	Research Collaboration	USN-8	As a research institution, I want collaboration tools to engage in ongoing research efforts and contribute to climate	8	Medium	Supraja & Aswini
			studies. The collaboration interfaces should include data-sharing agreements, collaborative research spaces, and periodic workshops.			

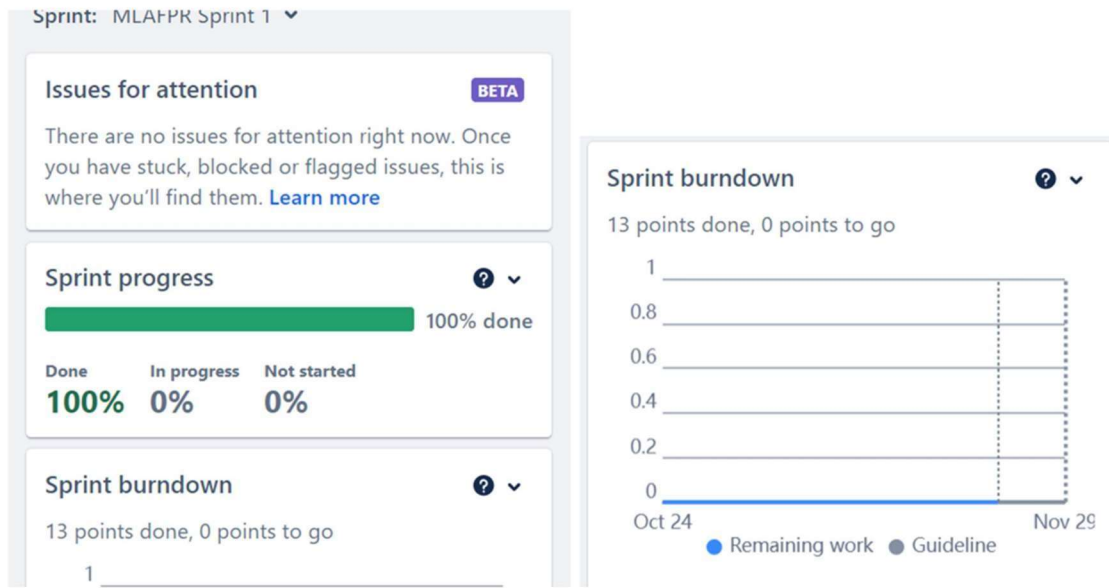
**Project Tracker, Velocity & Burndown Chart: (4 Marks)**

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	30-10-2023
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	18	06-11-2023
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	12	13-11-2023
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	10	20-11-2023

## 6.3 Sprint Delivery Schedule







## 7. CODING & PERFORMANCE TESTING

### 7.1 Performace Metrics

```
# Calculate metrics
accuracy = accuracy_score(y_test, t1)
precision = precision_score(y_test, t1)
recall = recall_score(y_test, t1)

# Print confusion matrix
print("Confusion Matrix:")
print(conf_matrix6)

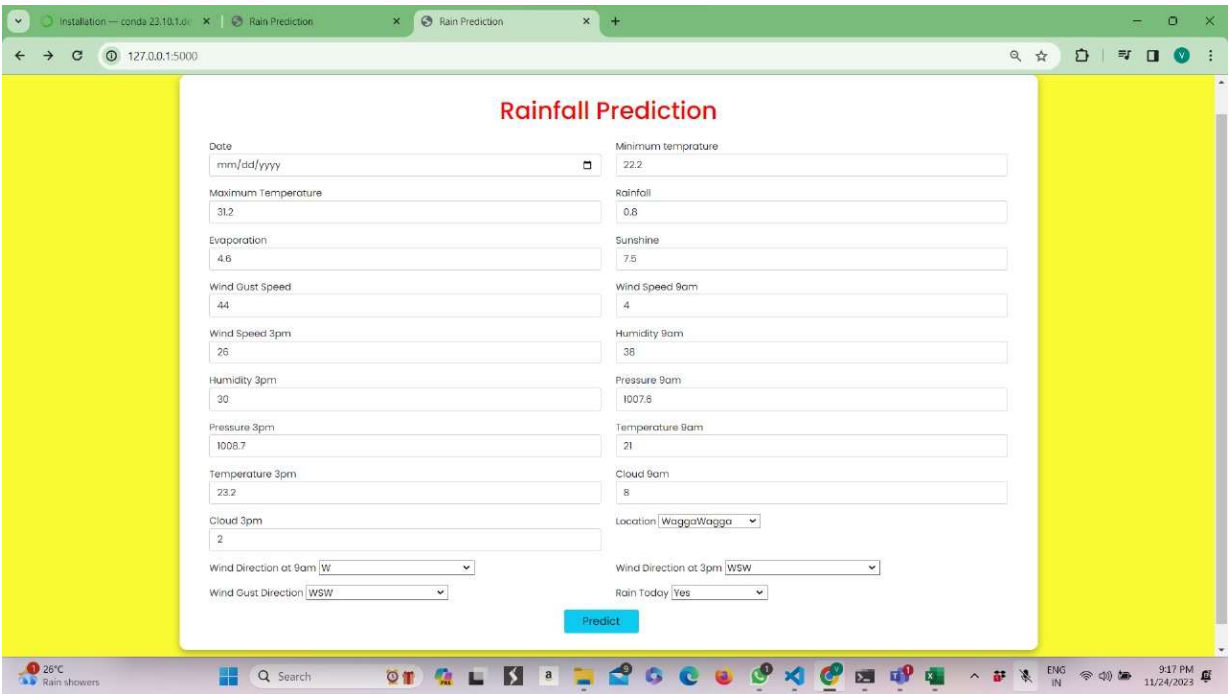
# Print metrics
print("Accuracy:", accuracy)
print("Precision:", precision)
print("Recall:", recall)
```

```
Confusion Matrix:
[[22065    2]
 [    5 6367]]
Accuracy: 1.0
Precision: 1.0
Recall: 1.0
```

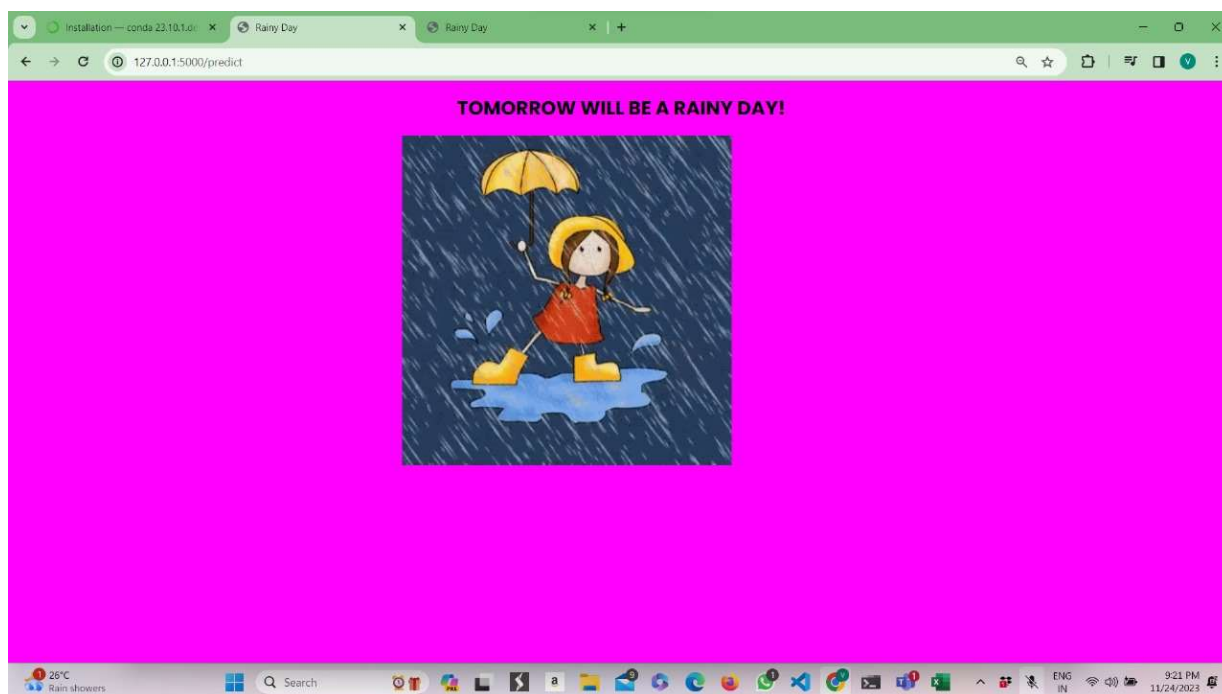
# 8. RESULTS

## 8.1 Output Screenshots

Index.html



## Chance.html



## Nochance.html



## 9.ADVANTAGES

- **Improved Accuracy**

ML models can analyze complex patterns and relationships within large datasets, potentially leading to more accurate rainfall predictions compared to traditional methods.

- **Data Driven Insights**

ML models can identify non-linear relationships and hidden patterns in data that may not be apparent through traditional statistical methods.

- **Adaptability**

ML models can adapt and learn from new data, allowing them to continually improve and adjust to changing conditions.

- **Handling Multivariate Data**

ML models can effectively handle large sets of diverse and multivariate data, incorporating information from various sources such as satellite imagery, climate models, and historical weather data.

- **Early Warning Systems**

ML models can contribute to the development of early warning systems for extreme weather events, providing valuable lead time for preparation and response.

### **DISADVANTAGES :**

- ML models require large amounts of high-quality data for training. In regions with limited historical weather data or poor data quality, the performance of ML models may be compromised.
- Some ML models, especially deep learning models, can be complex and require specialized knowledge for development and interpretation. This complexity may limit the adoption of ML approaches in some applications.
- Training sophisticated ML models can be computationally intensive, requiring powerful hardware and significant processing time.

- Some ML models, particularly deep neural networks, are often considered "black boxes" because it can be challenging to interpret how they make predictions. This lack of interpretability can be a disadvantage in applications where understanding the reasoning behind predictions is crucial. not representative of the underlying patterns in the broader population.
- ML models may be prone to overfitting, especially if the training data is Overfitting can lead to poor generalization to new, unseen data.

## **10. CONCLUSION**

In conclusion, the application of machine learning (ML) approaches to predict rainfall comes with a set of advantages and disadvantages. The potential for improved accuracy, adaptability, and the ability to uncover complex patterns in large datasets are significant advantages. ML models have the capacity to enhance early warning systems and contribute valuable insights into weather forecasting.

However, challenges such as the reliance on high-quality and sufficient data, the complexity of some ML models, and issues related to interpretability must be carefully considered. The computational resources required for training and the risk of overfitting are additional factors that can impact the performance of ML-based rainfall prediction systems.

A balanced approach that incorporates domain expertise, collaboration between meteorologists and data scientists, and a thorough understanding of the specific context and data limitations is crucial. ML approaches have the potential to revolutionize rainfall prediction, but their successful application requires careful consideration of these advantages and disadvantages to ensure accurate, reliable, and interpretable results.

## **11. FUTURE SCOPE**

The future of machine learning (ML) in rainfall prediction holds tremendous promise. Integration of diverse data sources, including satellite imagery and climate models, will enhance the accuracy of predictions. Hybrid models, combining ML with traditional meteorological methods, offer robust and interpretable solutions. Efforts to improve the quality and availability of weather data are crucial for ML model development. Explainable AI (XAI)

techniques will address the interpretability challenge, fostering trust in ML predictions. Edge computing deployment for real-time predictions is vital, particularly for early warning systems. ML models will play a key role in assessing and adapting to the impacts of climate change on rainfall patterns. Involving citizens in data collection and leveraging crowdsourced data can enrich ML model training. Automated feature engineering will streamline model development, improving prediction accuracy. Global collaboration in research and data sharing is essential for creating globally applicable ML models. Continuous model improvement through online learning will ensure effectiveness in dynamic environmental conditions. The future of ML in rainfall prediction lies in its ability to contribute to more accurate, timely, and globally relevant weather forecasts, ultimately aiding in climate resilience and disaster preparedness.

## 12.APPENDIXI

Source Code

home.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="preconnect" href="https://fonts.gstatic.com">
  <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@100;400;500
;600;700;800;900&display=swap" rel="stylesheet">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-
BmbxUPwQa2lc/FVzBcNJ7UAyJxM6wquIj61tLrc4wSX0szH/Ev+nYRRuW
lolflfl" crossorigin="anonymous">
  <link rel="stylesheet" href={{url_for('static',filename='predictor.css')}}>
  <title>Rain Prediction</title>
</head>
```

```
<body>
  <section id="prediction-form">
    <form class="form" action="/predict", method="POST">
      <h1 class="my-3 text-center">Rainfall Prediction</h1>
      <div class="row">
        <div class="col-md-6 my-2">
          <div class="md-form">
            <label for="date" class="date">Date</label>
            <input type="date" class="form-control" id="date"
name="date">
          </div>
        </div>
        <div class="col-md-6 my-2">
          <div class="md-form">
            <label for="mintemp" class="mintemp"> Minimum
temprature</label>
            <input type="text" class="form-control" id="mintemp"
name="mintemp">
          </div>
        </div>
        <div class="col-md-6 my-2">
          <div class="md-form">
            <label for="maxtemp" class="maxtemp">Maximum
Temperature</label>
            <input type="text" class="form-control" id="maxtemp"
name="maxtemp">
          </div>
        </div>
        <div class="col-md-6 my-2">
          <div class="md-form">
            <label for="rainfall" class="rainfall">Rainfall</label>
```

```
        <input type="text" class="form-control" id="rainfall"
name="rainfall">
    </div>
</div>
<div class="col-md-6 my-2">
    <div class="md-form">
        <label for="evaporation"
class="evaporation">Evaporation</label>
        <input type="text" class="form-control" id="evaporation"
name="evaporation">
    </div>
</div>
<div class="col-md-6 my-2">
    <div class="md-form">
        <label for="sunshine" class="sunshine">Sunshine</label>
        <input type="text" class="form-control" id="sunshine"
name="sunshine">
    </div>
</div>
<div class="col-md-6 my-2">
    <div class="md-form">
        <label for="windgustspeed" class="windgustspeed">Wind
Gust Speed</label>
        <input type="text" class="form-control" id="windgustspeed"
name="windgustspeed">
    </div>
</div>
<div class="col-md-6 my-2">
    <div class="md-form">
        <label for="windspeed9am" class="windspeed9am">Wind
Speed 9am</label>
```



```
        <input type="text" class="form-control" id="windspeed9am"
name="windspeed9am">
```

```
    </div>
```

```
</div>
```

```
<div class="col-md-6 my-2">
```

```
    <div class="md-form">
```

```
        <label for="windspeed3pm" class="windspeed3pm">Wind
Speed 3pm</label>
```

```
        <input type="text" class="form-control" id="windspeed3pm"
name="windspeed3pm">
```

```
    </div>
```

```
</div>
```

```
<div class="col-md-6 my-2">
```

```
    <div class="md-form">
```

```
        <label for="humidity9am" class="humidity9am">Humidity
9am</label>
```

```
        <input type="text" class="form-control" id="humidity9am"
name="humidity9am">
```

```
    </div>
```

```
</div>
```

```
<div class="col-md-6 my-2">
```

```
    <div class="md-form">
```

```
        <label for="humidity3pm" class="humidity3pm">Humidity
3pm</label>
```

```
        <input type="text" class="form-control" id="humidity3pm"
name="humidity3pm">
```

```
    </div>
```

```
</div>
```

```
<div class="col-md-6 my-2">
```

```
    <div class="md-form">
```

```
        <label for="pressure9am" class="pressure9am">Pressure
9am</label>
```

```
        <input type="text" class="form-control" id="pressure9am"
name="pressure9am">
```

```
    </div>
```

```
</div>
```

```
<div class="col-md-6 my-2">
```

```
    <div class="md-form">
```

```
        <label for="pressure3pm" class="pressure3pm">Pressure
3pm</label>
```

```
        <input type="text" class="form-control" id="pressure3pm"
name="pressure3pm">
```

```
    </div>
```

```
</div>
```

```
<div class="col-md-6 my-2">
```

```
    <div class="md-form">
```

```
        <label for="temp9am" class="temp9am">Temperature
9am</label>
```

```
        <input type="text" class="form-control" id="temp9am"
name="temp9am">
```

```
    </div>
```

```
</div>
```

```
<div class="col-md-6 my-2">
```

```
    <div class="md-form">
```

```
        <label for="temp3pm" class=temp3pm">Temperature
3pm</label>
```

```
        <input type="text" class="form-control" id="temp3pm"
name="temp3pm">
```

```
    </div>
```

```
</div>
```

```
<div class="col-md-6 my-2">
```

```
    <div class="md-form">
```

```

        <label for="cloud9am" class="cloud9am">Cloud
9am</label>
        <input type="text" class="form-control" id="cloud9am"
name="cloud9am">
    </div>
</div>
<div class="col-md-6 my-2">
    <div class="md-form">
        <label for="cloud3pm" class="cloud3pm">Cloud
3pm</label>
        <input type="text" class="form-control" id="cloud3pm"
name="cloud3pm">
    </div>
</div>
<div class="col-md-6 my-2">
    <div class="md-form">
        <label for="location" class="location"
name="location">Location</label>
        <select class="location" id="location" name="location" aria-
label="Location">
            <option selected>Select Location</option>
            <option value= 24>Adelaide</option>
            <option value= 7>Albany</option>
            <option value= 30>Albury</option>
            <option value= 46>AliceSprings</option>
            <option value= 33>BadgerysCreek</option>
            <option value= 14>Ballarat</option>
            <option value= 36>Bendigo</option>
            <option value= 21>Brisbane</option>
            <option value= 2>Cairns</option>
            <option value= 43>Cobar</option>

```

<option value= 9>CoffsHarbour</option>  
<option value= 4>Dartmoor</option>  
<option value= 11>Darwin</option>  
<option value= 15>GoldCoast</option>  
<option value= 17>Hobart</option>  
<option value= 45>Katherine</option>  
<option value= 23>Launceston</option>  
<option value= 28>Melbourne</option>  
<option value= 25>Melbourne Airport</option>  
<option value= 44>Mildura</option>  
<option value= 42>Moree</option>  
<option value= 5>MountGambier</option>  
<option value= 12>MountGinini</option>  
<option value= 19>Newcastle </option>  
<option value= 47>Nhil</option>  
<option value= 13>NorahHead</option>  
<option value= 6>NorfolkIsland</option>  
<option value= 32>Nuriootpa</option>  
<option value= 40>PearceRAAF</option>  
<option value= 31>Penrith</option>  
<option value= 26>Perth</option>  
<option value= 35>Perth Airport</option>  
<option value= 1>Portland</option>  
<option value= 37>Richmond</option>  
<option value= 27>Sale</option>  
<option value= 41>Salmon Gums</option>  
<option value= 10>Sydney</option>  
<option value= 16>Sydney Airport</option>  
<option value= 39>Townsville</option>  
<option value= 34>Tuggeranong</option>

```
<option value= 49>Uluru</option>
<option value= 38>WaggaWagga</option>
<option value= 3>Walpole</option>
<option value= 18>Watsonia</option>
<option value= 22>William Town</option>
<option value= 8>Witchcliffe</option>
<option value= 20>Wollongong</option>
<option value= 48>Woomera</option>
</select>
```

```
</div>
```

```
</div>
```

```
<div class="col-md-6 my-2">
```

```
<div class="md-form">
```

```
<label for="winddir9am" class="winddir9am" name =
"winddir9am">Wind Direction at 9am</label>
```

```
<select class="winddir9am" id="winddir9am"
name="winddir9am" aria-label="Wind Direction 9am">
```

```
<option selected>Select Wind Direction at 9am</option>
```

```
<option value= 1>N</option>
```

```
<option value= 5>W</option>
```

```
<option value= 10>S</option>
```

```
<option value= 15>E</option>
```

```
<option value= 2>NW</option>
```

```
<option value= 9>NE</option>
```

```
<option value= 7>SW</option>
```

```
<option value= 13>SE</option>
```

```
<option value= 0>NNW</option>
```

```
<option value= 3>NNE</option>
```

```
<option value= 8>SSW</option>
```

```
<option value= 11>SSE</option>
```

```
<option value= 4>WNW</option>
```

```
        <option value= 6>WSW</option>
        <option value= 12>ENE</option>
        <option value= 14>ESE</option>
    </select>
</div>
</div>
<div class="col-md-6 my-2">
    <div class="md-form">
        <label for="winddir3pm" class="winddir3pm" name =
"winddir3pm">Wind Direction at 3pm</label>
        <select class="winddir3pm" id="winddir3pm" name =
"winddir3pm" aria-label="Wind Direction at 3pm">
            <option selected>Select Wind Direction at 3pm</option>
            <option value= 2>N</option>
            <option value= 4>W</option>
            <option value= 8>S</option>
            <option value= 14>E</option>
            <option value= 0>NW</option>
            <option value= 11>NE</option>
            <option value= 9>SW</option>
            <option value= 10>SE</option>
            <option value= 1>NNW</option>
            <option value= 5>NNE</option>
            <option value= 7>SSW</option>
            <option value= 12>SSE</option>
            <option value= 3>WNW</option>
            <option value= 6>WSW</option>
            <option value= 13>ENE</option>
            <option value= 15>ESE</option>
        </select>
    </div>
```

```
</div>
<div class="col-md-6 my-2">
  <div class="md-form">
    <label for="windgustdir" class="windgustdir" name =
"windgustdir">Wind Gust Direction</label>
    <select class="windgustdir" id="windgustdir" name =
"windgustdir" aria-label="Wind Gust Direction">
      <option selected>Select Wind Gust Direction</option>
      <option value= 3>N</option>
      <option value= 4>W</option>
      <option value= 7>S</option>
      <option value= 15>E</option>
      <option value= 1>NW</option>
      <option value= 11>NE</option>
      <option value= 9>SW</option>
      <option value= 12>SE</option>
      <option value= 0>NNW</option>
      <option value= 6>NNE</option>
      <option value= 8>SSW</option>
      <option value= 10>SSE</option>
      <option value= 2>WNW</option>
      <option value= 5>WSW</option>
      <option value= 14>ENE</option>
      <option value= 13>ESE</option>
    </select>
  </div>
</div>
<div class="col-md-6 my-2">
  <div class="md-form">
    <label for="raintoday" class="raintoday"
name="raintoday">Rain Today</label>
```

```

        <select class="raintoday" id="raintoday" name="raintoday"
aria-label="Rain Today">
            <option selected>Did it Rain Today</option>
            <option value= 1>Yes</option>
            <option value= 0>No</option>
        </select>
    </div>
</div>
<div class="col-md-6 my-2 d-flex align-items-end justify-content-
around">
    <button type="submit" class="btn btn-info button" color=
#ff0000 style="margin-left: 100%;">Predict</button>
</div>
</div>
</form>
</section>
<div>
    <h1><center> {{ prediction }} </center></h1>
</div>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta2/dist/js/bootstrap.bundle.min.js" integrity="sha384-
b5kHyXgcpbZJO/tY9U17kGkf1S0CWuKcCD38l8YkeH8z8QjE0GmW1gY
U5S9FOnJ0" crossorigin="anonymous"></script>
</body>
</html>

```

## **rainy.html**

```

<!DOCTYPE html>
<html lang="en">
<head>

```



```
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@100;400;500
;600;700;800;900&display=swap" rel="stylesheet">
<link rel="stylesheet" href={{url_for('static',filename='style01.css')}}>
<title>Rainy Day</title>
</head>
<body>
<h1 style="text-align: center; font-size: 3 rem; font-weight:
bolder">TOMORROW WILL BE A RAINY DAY!</h1>
<div class="rainyimg">

</div>

</div>
</body>
</html>
```

### **sunny.html**

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@100;400;500;60
0;700;800;900&display=swap" rel="stylesheet">
<link rel="stylesheet" href={{url_for('static',filename='style01.css')}}>
```

```
<title>Rainy Day</title>
</head>
<body>
  <h1 style="text-align: center; font-size: 3 rem; font-weight: bolder">
TOMORROW WILL BE A SUNNY DAY!</h1>
  <div class="rainyimg">
    
  </div>
  <div>

  </div>
</body>
</html>
```

### **App.py**

```
from flask import Flask,render_template,url_for,request,jsonify
from flask_cors import cross_origin
import pandas as pd
import numpy as np
import datetime
import pickle
from xgboost import XGBClassifier
```

```
app = Flask(__name__, template_folder="template")
model = pickle.load(open("xg_random.pkl", "rb"))
print("Model Loaded")
```

```
@app.route("/")
@cross_origin()
def home():
```

```
return render_template("home.html")
```

```
@app.route("/predict",methods=['GET', 'POST'])
```

```
@cross_origin()
```

```
def predict():
```

```
if request.method == "POST":
```

```
    # DATE
```

```
    date = request.form['date']
```

```
    day = float(pd.to_datetime(date, format="%Y-%m-%dT").day)
```

```
    month = float(pd.to_datetime(date, format="%Y-%m-%dT").month)
```

```
    # MinTemp
```

```
    minTemp = float(request.form['mintemp'])
```

```
    # MaxTemp
```

```
    maxTemp = float(request.form['maxtemp'])
```

```
    # Rainfall
```

```
    rainfall = float(request.form['rainfall'])
```

```
    # Evaporation
```

```
    evaporation = float(request.form['evaporation'])
```

```
    # Sunshine
```

```
    sunshine = float(request.form['sunshine'])
```

```
    # Wind Gust Speed
```

```
    windGustSpeed = float(request.form['windgustspeed'])
```

```
    # Wind Speed 9am
```

```
    windSpeed9am = float(request.form['windspeed9am'])
```

```
    # Wind Speed 3pm
```

```
    windSpeed3pm = float(request.form['windspeed3pm'])
```

```
    # Humidity 9am
```

```
    humidity9am = float(request.form['humidity9am'])
```

```
    # Humidity 3pm
```

```
    humidity3pm = float(request.form['humidity3pm'])
```

```
    # Pressure 9am
```

```
    pressure9am = float(request.form['pressure9am'])
```

```
# Pressure 3pm
pressure3pm = float(request.form['pressure3pm'])
# Temperature 9am
temp9am = float(request.form['temp9am'])
# Temperature 3pm
temp3pm = float(request.form['temp3pm'])
# Cloud 9am
cloud9am = float(request.form['cloud9am'])
# Cloud 3pm
cloud3pm = float(request.form['cloud3pm'])
# Cloud 3pm
location = float(request.form['location'])
# Wind Dir 9am
windDir9am = float(request.form['winddir9am'])
# Wind Dir 3pm
windDir3pm = float(request.form['winddir3pm'])
# Wind Gust Dir
windGustDir = float(request.form['windgustdir'])
# Rain Today
rainToday = float(request.form['raintoday'])
```

```
input_lst = [location , minTemp , maxTemp , rainfall , evaporation ,
sunshine ,
```

```
windGustDir , windGustSpeed , windDir9am ,
windDir3pm , windSpeed9am , windSpeed3pm ,
```

```
humidity9am , humidity3pm , pressure9am ,
pressure3pm , cloud9am , cloud3pm , temp9am , temp3pm ,
```

```
rainToday , month , day]
```

```
pred = model.predict(input_lst)
```

```
output = pred
```

```
if output == 0:
```

```
    return render_template("sunny.html")
```

```
else:
```

```
        return render_template("rainy.html")
    return render_template("home.html")

if __name__ == '__main__':
    app.run(debug=True)
```

### ml code :

```
# libraries required
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import preprocessing
from sklearn import model_selection
from sklearn import metrics
from sklearn import linear_model
from sklearn import ensemble
from sklearn import tree
from sklearn import svm
import xgboost
```

```
data = pd.read_csv("/content/sample_data/weatherAUS.csv")
data

data.head()
```

```
data.describe()
```

```
data.info()
```

```
data.isnull().sum()
```

```
import missingno as msno
msno.matrix(data,color=(0.55, 0.255, 0.225), fontsize=16)
```

```
data_cat = data[['RainToday', 'WindGustDir', 'WindDir9am', 'WindDir3pm']]
data.drop(columns=['Evaporation', 'Sunshine', 'Cloud9am',
'Cloud3pm'],axis=1,inplace=True)
```

```
data.drop(columns=['RainToday', 'WindGustDir', 'WindDir9am',  
'WindDir3pm'],axis=1,inplace=True)
```

```
data['MinTemp'].fillna(data['MinTemp'].mean(),inplace=True)  
data['MaxTemp'].fillna(data['MaxTemp'].mean(),inplace=True)  
data['Rainfall'].fillna(data['Rainfall'].mean(),inplace=True)  
data['WindGustSpeed'].fillna(data['WindGustSpeed'].mean(),inplace=True)  
data['WindSpeed9am'].fillna(data['WindSpeed9am'].mean(),inplace=True)  
data['WindSpeed3pm'].fillna(data['WindSpeed3pm'].mean(),inplace=True)  
data['Humidity9am'].fillna(data['Humidity9am'].mean(),inplace=True)  
data['Humidity3pm'].fillna(data['Humidity3pm'].mean(),inplace=True)  
data['Pressure9am'].fillna(data['Pressure9am'].mean(),inplace=True)  
data['Pressure3pm'].fillna(data['Pressure3pm'].mean(),inplace=True)  
data['Temp9am'].fillna(data['Temp9am'].mean(),inplace=True)
```

```
cat_names = data_cat.columns  
# initializing the simple imputer for missing categorical values  
import numpy as np  
from sklearn.impute import SimpleImputer  
imp_mode = SimpleImputer(missing_values=np.nan, strategy='most_frequent')  
data_cat = imp_mode.fit_transform(data_cat)  
data_cat = pd.DataFrame(data_cat,columns=cat_names)  
data = pd.concat([data,data_cat],axis=1)  
from sklearn.preprocessing import LabelEncoder  
  
# Assuming 'df' is your DataFrame  
  
# Initialize the LabelEncoder  
label_encoder = LabelEncoder()  
  
# Iterate through each column in the DataFrame  
for column in data.columns:  
    # Check if the column has object dtype (non-numeric)  
    if data[column].dtype == 'object':  
        # Use label encoder to transform only non-numeric columns  
        data[column] = label_encoder.fit_transform(data[column])  
  
# Now, 'df' contains label-encoded values for all non-numeric columns  
data.corr()  
cor = data.corr()  
sns.heatmap(data=cor,xticklabels=cor.columns.values,yticklabels=cor.columns.val  
ues)  
sns.pairplot(data)  
data.boxplot()
```

```

from sklearn.preprocessing import StandardScaler
#splitting x and y values
y = data['RainTomorrow']
x = data.drop('RainTomorrow',axis=1)
x = data.drop('Date',axis=1)
names = x.columns
names
sc = StandardScaler()
import pandas as pd
from sklearn.preprocessing import StandardScaler

# Assuming 'x' is your DataFrame
numeric_columns = data.select_dtypes(include=['number']).columns
x_numeric = data[numeric_columns]

sc = StandardScaler()
x = sc.fit_transform(x_numeric)
x = sc.fit_transform(x)
# Assuming 'x' is your DataFrame and 'names' is a list of column names
x = pd.DataFrame(data, columns=names)

from sklearn import model_selection
x_train,x_test,y_train,y_test=model_selection.train_test_split(x,y,test_size=0.2,random_state =0)
x_train.shape,y_train.shape,x_test.shape,y_test.shape
import xgboost
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.linear_model import LogisticRegression

XGBoost = xgboost.XGBRFClassifier()
Rand_forest = RandomForestClassifier()
svm = SVC()
Dtree = DecisionTreeClassifier()
GBM = GradientBoostingClassifier()
log = LogisticRegression()

# Check unique values in 'RainTomorrow'
print(data['RainTomorrow'].unique())

Rand_forest.fit(x_train, y_train)

svm.fit(x_train, y_train)

```

```

Dtree.fit(x_train, y_train)

GBM.fit(x_train, y_train)

log.fit(x_train, y_train)
# Fit the models with the encoded target variable
XGBoost.fit(x_train, y_train)

p1 = XGBoost.predict(x_train)
p2 = Rand_forest.predict(x_train)
p3 = svm.predict(x_train)
p4 = Dtree.predict(x_train)
p5 = GBM.predict(x_train)
p6 = log.predict(x_train)
print("xgboost:", metrics.accuracy_score(y_train, p1))
print("Rand_forest:", metrics.accuracy_score(y_train, p2))
print("svm:", metrics.accuracy_score(y_train, p3))
print("Dtree:", metrics.accuracy_score(y_train, p4))
print("GBM:", metrics.accuracy_score(y_train, p5))
print("log:", metrics.accuracy_score(y_train, p6))
t1 = XGBoost.predict(x_test)
t2 = Rand_forest.predict(x_test)
t3 = svm.predict(x_test)
t4 = Dtree.predict(x_test)
t5 = GBM.predict(x_test)
t6 = log.predict(x_test)
print("xgboost:", metrics.accuracy_score(y_test, t1))
print("Rand_forest:", metrics.accuracy_score(y_test, t2))
print("svm:", metrics.accuracy_score(y_test, t3))
print("Dtree:", metrics.accuracy_score(y_test, t4))
print("GBM:", metrics.accuracy_score(y_test, t5))
print("log:", metrics.accuracy_score(y_test, t6))
conf_matrix1 = metrics.confusion_matrix(y_test, t1)
fig, ax = plt.subplots(figsize=(7.5, 7.5))
ax.matshow(conf_matrix1, alpha=0.3)
for i in range(conf_matrix1.shape[0]):
    for j in range(conf_matrix1.shape[1]):
        ax.text(x=j, y=i, s=conf_matrix1[i, j], va='center', ha='center',
size='xx-large')

plt.xlabel('Predictions', fontsize=18)
plt.ylabel('Actuals', fontsize=18)
plt.title('Confusion Matrix', fontsize=18)
plt.show()

```



```

from sklearn.metrics import accuracy_score, precision_score, recall_score

# Assuming conf_matrix is already calculated
tn, fp, fn, tp = conf_matrix1.ravel()

# Calculate metrics
accuracy = accuracy_score(y_test, t1)
precision = precision_score(y_test, t1)
recall = recall_score(y_test, t1)

# Print confusion matrix
print("Confusion Matrix:")
print(conf_matrix1)

# Print metrics
print("Accuracy:", accuracy)
print("Precision:", precision)
print("Recall:", recall)

auc = metrics.roc_auc_score(y_test, t1)

fpr, tpr, thresholds = metrics.roc_curve(y_test, t1)

plt.figure(figsize=(12, 10), dpi=80)
plt.axis('scaled')
plt.xlim([0, 1])
plt.ylim([0, 1])
plt.title("AUC & ROC Curve")
plt.plot(fpr, tpr, 'v')
plt.fill_between(fpr, tpr, facecolor='blue', alpha=0.8)
plt.text(1, 0.05, f'AUC = {auc:.4f}', ha='right', fontsize=10, weight='bold',
color='black')
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.show()

import pickle
pickle.dump(XGBoost, open('rainfall.pkl', 'wb'))
pickle.dump(label_encoder, open('encoder.pkl', 'wb'))
pickle.dump(imp_mode, open('impter.pkl', 'wb'))
pickle.dump(sc, open('scale.pkl', 'wb'))

```

**Github Link :** [GitHub - smartinternz02/SI-GuidedProject-615663-1700497094](https://github.com/smartinternz02/SI-GuidedProject-615663-1700497094)

**Demo Link :** <https://drive.google.com/file/d/1bfis-bFrXaf4kp1zZLme1sAV11bK3T69/view?usp=sharing>