JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Rainfall Prediction Using Machine Learning: A Flask-Based Regional Forecasting System for Indian Districts

[1]Swapnil Shinde, [2]Jaishree Dubey, [3]Janhavi Bilgaye

[1]Information Technology, [2]Information Technology, [3]Information Technology,
[1]Pimpri Chinchwad College of Engineering, Pune, India [2]Pimpri Chinchwad College of Engineering, Pune, India
[3]Pimpri Chinchwad College of Engineering, Pune, India

Abstract : Forecasting rainfall is essential for agricultural planning, disaster preparedness, and sustainable water resource management. Traditional physics-based models often struggle with region-specific variability and computational complexity. In recent years, machine learning has emerged as a viable alternative due to its ability to capture non-linear, temporal, and spatial dependencies in climatic data [1][2]. This paper presents a district- level rainfall prediction system for India, built using a Random Forest Regressor trained on historical rainfall data from 2017 to 2023. The model is designed to forecast monthly rainfall from April 2025 to December 2026 with high accuracy. Random Forest was selected for its proven robustness, interpretability, and resistance to overfitting in regression problems involving environmental data [3]. The system is deployed using a lightweight Flask web framework that enables users to interactively select their state and district, generate month-wise predictions, and visualize results through dynamic graph generation using matplotlib. The final output includes both tabular and graphical formats, which enhance interpretability and usability. The proposed system offers a practical, scalable, and user-friendly solution for stakeholders such as farmers, researchers, and disaster planning authorities who require timely, localized rainfall insights.

Keywords: - Rainfall Forecasting, Machine Learning, Random Forest, Flask Web App, Weather Prediction, Graph Generation, Result Optimization

I. INTRODUCTION

Rainfall is a critical climatic factor that directly influences agricultural productivity, water resource management, and disaster preparedness. In a country like India, where the economy is significantly driven by agriculture and seasonal monsoons dictate sowing and harvesting cycles, timely and accurate rainfall forecasts are not just beneficial—they are essential. Farmers, particularly in rural regions without advanced irrigation systems, depend heavily on predictable rainfall. Similarly, urban planners, policymakers, and researchers require dependable climate data to guide strategies for water conservation, flood mitigation, and infrastructure planning [4].

Traditional approaches to rainfall forecasting have primarily relied on physics-based simulations and numerical weather prediction models. Although these systems have demonstrated efficacy at broader scales, they often lack localized precision, require high computational power, and are inaccessible to non-expert users [5]. This limits their effectiveness for community- level planning and agricultural decision-making.

The rise of machine learning (ML) has enabled data-driven forecasting methods capable of capturing the non-linear and seasonal behavior inherent in rainfall datasets. These models, particularly ensemble-based algorithms such as the Random Forest Regressor, have proven robust in handling high-dimensional, noisy, and complex environmental data. They offer advantages like resistance to overfitting, interpretability through feature importance, and suitability for multi-output tasks [6].

In this work, a rainfall prediction system is proposed using a Random Forest model trained on monthly rainfall data from 2017 to 2023. The model estimates rainfall from April 2025 to December 2026 for user-specified states and districts across India. To

ensure user accessibility, the system is embedded in a lightweight Flask web application, enabling interaction through a simple form-based interface.

One of the system's core strengths is its use of real-time graph generation via matplotlib, which transforms numerical forecasts into an intuitive visual format. This enhances user comprehension by allowing quick identification of monthly and seasonal rainfall patterns. Together, the integration of ML forecasting with interactive visualization provides a practical, scalable, and region-sensitive solution for stakeholders involved in agriculture, disaster management, and climate research.

II. RELATED WORK

Rainfall prediction has long been a vital area of research in meteorology, agriculture, and environmental science. Over the years, forecasting techniques have evolved from traditional statistical models to advanced machine learning and deep learning architectures. The growing availability of historical weather data and improvements in computational methods have allowed researchers to experiment with increasingly sophisticated data-driven models [7].

Early models for forecasting predominantly used statistical approaches such as Autoregressive Integrated Moving Average (ARIMA) and linear regression. While these models were effective at capturing broad seasonal trends, they struggled to model the highly non-linear and chaotic nature of real-world weather systems, especially in climate-diverse countries like India [8].

To address these limitations, researchers began applying artificial neural networks (ANNs) for rainfall prediction. ANNs were capable of learning complex, non-linear patterns and showed improved accuracy over linear models. For instance, a study applied multilayer perception to monthly rainfall data and demonstrated improved forecast performance. However, neural networks were often criticized for their lack of interpretability and sensitivity to hyperparameters.

Following this, algorithms such as Support Vector Machines (SVMs) and decision trees gained traction in climate-related tasks. Notably, Random Forest Regressors, an ensemble method that combines multiple decision trees, emerged as a strong candidate for rainfall prediction. Nair and Babu [9] showed that Random Forest models outperformed classical models in predicting seasonal rainfall in southern Indian regions due to their robustness and reduced overfitting tendencies.

More recently, attention has shifted toward deep learning models, particularly Long Short-Term Memory (LSTM) networks. LSTMs are adept at time-series forecasting and can effectively capture long-term temporal dependencies in rainfall sequences. However, these models require large datasets and substantial computational resources, making them less ideal for lightweight, real-time forecasting applications [10].

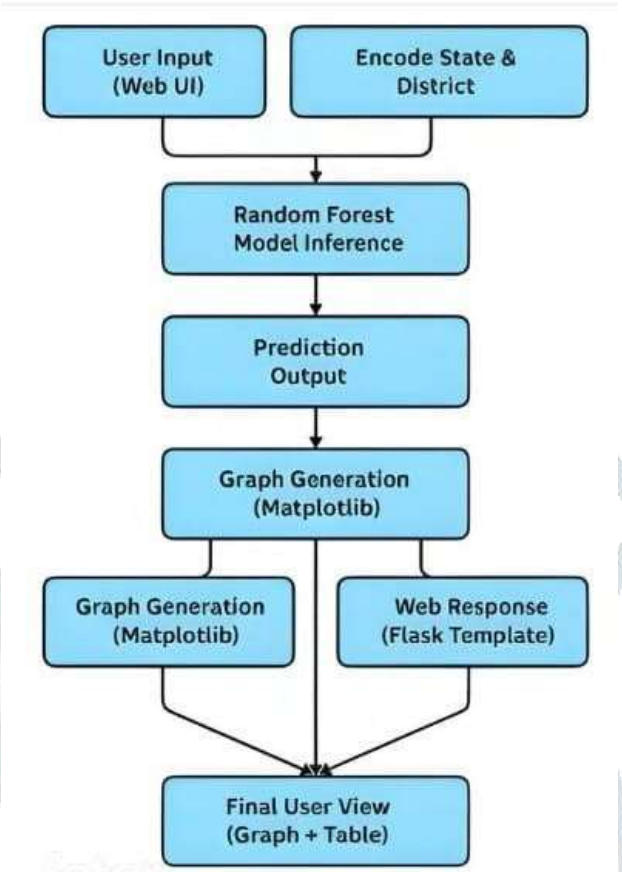
In addition to model selection, user accessibility has become a growing concern. While several institutional and commercial platforms provide rainfall forecasts, they often lack localized detail and do not offer interactive visualizations. Most of these systems are built for meteorological experts rather than everyday stakeholders like farmers or planners.

Recent developments in web frameworks such as Flask, Django, and Streamlit have made it possible to bridge this gap. These technologies allow for the rapid deployment of machine learning models through web-based dashboards. For example, Sharma and Mehta implemented an LSTM-based prediction system integrated with a Flask frontend for short-term weather forecasting. However, such systems typically depend on external APIs or lack high-resolution, user-targeted insights.

The system proposed in this paper differentiates itself by combining the simplicity and effectiveness of Random Forest with the interactivity of a web-based interface. It not only offers localized predictions at the district level but also integrates dynamic graph generation, allowing end-users to visually interpret rainfall forecasts in real time. This enhances the usability of the system and makes it practical for real-world decision-making.

III. SYSTEM ARCHITECTURE

The architecture of the proposed rainfall prediction system is designed to be modular, scalable, and lightweight, enabling both high-performance prediction and user-friendly access via a web interface. The system is divided into four major components: data preprocessing, model training, Flask-based deployment, and output generation. Figure 1 illustrates the complete workflow of the system.



A. Data Preprocessing

The first stage involves importing and cleaning historical rainfall data from 2017 to 2023. The dataset includes monthly rainfall values (Jan through Dec) along with geographical identifiers such as State/UT, District, and Year. Data preprocessing includes:

- Removing rows with missing or corrupted values.
- Converting all text fields to uppercase and trimming whitespace.
- Label encoding categorical fields like State/UT and District using LabelEncoder or dictionary mapping.

This step ensures data uniformity and prepares the dataset for effective training without inconsistencies.

B. Model Training

The machine learning model used is a Random Forest Regressor, which is highly effective for regression tasks with non-linear relationships and large feature spaces. The model is wrapped with a MultiOutputRegressor to enable simultaneous prediction of multiple targets, in this case, rainfall values for each month. The features used include:

- Encoded values for state and district.
- The year and month (as numerical inputs).
- Additional placeholders for other contextual features (initialized with zeros for scalability).

The model is trained and validated on historical data using an 80-20 split, and its performance is evaluated using R^2 Score and Mean Absolute Error (MAE).

C. Flask Web Deployment

To ensure accessibility, the trained model is integrated into a lightweight Flask web application. This application serves as the user interface where individuals can:

- Select their State and District.
- Submit a request to generate rainfall predictions for the period from April 2025 to December 2026.
- View results and visualizations instantly in the browser.

The Flask app handles backend logic such as loading the model, transforming input data, predicting values, and rendering output templates (index.html, results.html). It is designed to be fast, responsive, and compatible with various devices, including mobile browsers.

D. Output Generation (Graph & CSV)

Upon generating predictions, the system creates a dynamic line graph using matplotlib, visualizing month-wise rainfall from April 2025 to December 2026. This visualization helps users quickly interpret seasonal trends and fluctuations. Additionally, the system:

- Saves the forecast data as a downloadable CSV file (predictions.csv).
- Displays predictions in a structured HTML layout for instant review.

The graphical output and tabular forecast together enhance user understanding and provide actionable insight into future rainfall trends [5].

IV. METHODOLOGY

The development of the proposed rainfall prediction system followed a structured methodology comprising data preparation, model training, feature engineering, and output generation. Each stage was carefully designed to ensure accuracy, scalability, and seamless integration with the web-based interface.

A. Dataset Details

The foundation of this project is a dataset containing monthly rainfall records from 2017 to 2023 across various states and districts in India. Each entry includes the fields: State/UT, District, Year, and monthly rainfall values from January to December. The dataset serves as the historical reference for model training and is stored in CSV format for easy manipulation using Python's data science stack.

B. Data Preprocessing

To ensure consistency and usability, the dataset underwent multiple preprocessing steps:

- Whitespace removal: All entries in the State/UT and District columns were stripped of leading and trailing spaces.
- Case normalization: Text entries were converted to uppercase to maintain uniformity.
- Missing values: Rows with incomplete or missing rainfall data were discarded.
- Categorical encoding: The State/UT and District fields were transformed into a numerical format using LabelEncoder and dictionary-based mappings, making them suitable inputs for machine learning algorithms.

These preprocessing techniques ensured that the data fed into the model was clean, consistent, and ready for learning.

C. Feature Selection

The features selected for model training were:

- Encoded State/UT
- Encoded District
- Year
- Month
- Placeholder features (set to zero), reserved for future integration of weather-based attributes such as temperature, humidity, or satellite indices.

These features were chosen for their ability to capture both spatial and temporal dimensions of rainfall variation. While additional environmental indicators were not used in this version, the system architecture allows for easy extension.

D. Model Used: Random Forest Regressor

To model the relationship between geographical-temporal inputs and monthly rainfall outputs, a Random Forest Regressor was selected. This ensemble learning method constructs multiple decision trees and averages their outputs to generate stable and accurate predictions. Its benefits include:

- Resistance to overfitting
- Capability to model non-linear relationships
- High interpretability through feature importance ranking

To support multi-output prediction (i.e., rainfall for each month), the regressor was wrapped inside a Multi Output Regressor, enabling the simultaneous estimation of twelve separate outputs for each year.

E. Output Generation

Once the model was trained and validated, it was integrated into a Flask web application. When a user selects a state and district, the system encodes the inputs, feeds them into the model, and returns predicted rainfall values for each month from April 2025 to December 2026.

To enhance user interaction and understanding, the predictions are:

- Displayed as a dynamic line graph using matplotlib
- Stored in a CSV file using pandas for downloading
- Rendered as structured HTML for a clean tabular view within the web interface

This end-to-end pipeline ensures that users receive fast, informative, and actionable rainfall forecasts in a manner that is both accessible and visually intuitive.

V. IMPLEMENTATION

The proposed rainfall prediction system was developed with a focus on accessibility, modularity, and efficiency. The implementation integrates machine learning with a web interface to deliver an interactive user experience that allows real-time rainfall predictions and graphical insights.

A. Technologies Used

The system leverages a variety of open-source technologies, each selected for its specific strengths in machine learning, web development, and data visualization:

- Python: The core programming language used for data processing, model training, and backend development.
- Flask: A lightweight web framework used to create the interactive user interface and handle HTTP requests.
- scikit-learn: Utilized for training the Random Forest Regressor model and handling data preprocessing tasks such as label encoding and model evaluation.
- matplotlib: Employed for generating dynamic line graphs of predicted rainfall to enhance data visualization.
- pandas: Used for reading CSV datasets and exporting prediction results into downloadable formats.
- HTML/CSS (Jinja Templates): Used in the frontend interface to structure and style the prediction form and result display.
- (Optional) report lab: Initially intended for generating PDF reports (can be replaced with alternative visual outputs if not used).

B. Flask Route Handling

The Flask application consists of two main routes:

1. (GET & POST):

- GET Method: Renders the homepage where users select their state and district.
- POST Method: Accepts the user's input, encodes the state and district, and runs the model to generate monthly rainfall predictions.

The predictions are then visualized in a graph, displayed in a table, and optionally saved as a downloadable CSV.

2. Templates Used:

- index.html – Contains the user input form with dynamically loaded state-district mapping.
- results.html – Displays the prediction output, graph, and download link.

C. Sample Interface Screenshots

The user interface is built to be simple, responsive, and accessible across devices:

- Homepage (index.html) The user selects the desired state and district from dropdowns and clicks the "Predict Rainfall" button.
- Result Page (results.html) The prediction results are shown both as a monthly rainfall graph and as a tabular breakdown of predicted values.

D. Graph Generation

After rainfall values are predicted by the model, they are passed to matplotlib for visualization. The graph plots:

- X-axis: Months from April 2025 to December 2026
- Y-axis: Predicted rainfall in millimeters
- The graph is styled for readability with clear axis labels, grid lines, and color-coded lines.

The resulting chart is saved as a .png file and embedded within the results page for immediate user feedback. This graphical representation helps users easily interpret monthly rainfall patterns and plan accordingly.

VI. RESULTS & DISCUSSION

The performance of the proposed rainfall prediction system was evaluated through graphical analysis of historical data trends, feature importance, and predicted outputs. Visual tools were extensively used to interpret results, uncover patterns, and validate model predictions in an intuitive manner. This section presents a series of graphs that provide insights into both the underlying dataset and the predictive capabilities of the trained model.

A. Month-wise Average Rainfall (2017–2023)

Fig. 2 displays the average rainfall received during each month from 2017 to 2023. The data reveals prominent peaks in June, July, and August, confirming the dominance of the monsoon season. This seasonal trend validates the temporal consistency of the dataset.

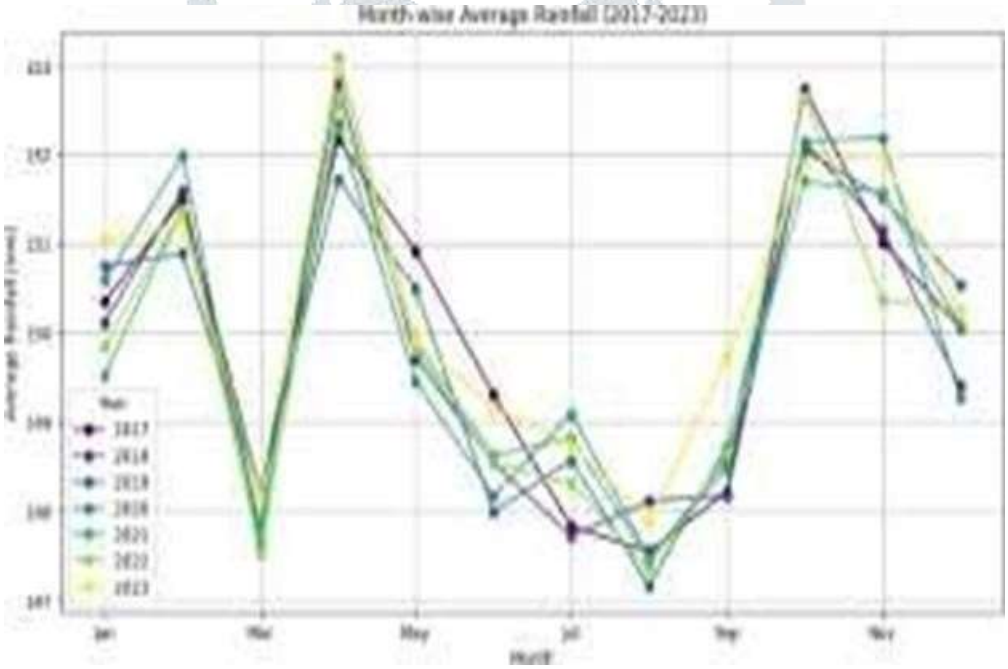


Fig 2. Month-wise average rainfall

B. Feature Importance – Random Forest

The Random Forest model provides a feature importance ranking which helps understand the influence of each input on the prediction. As seen in Fig. 3, months such as December, January, and March significantly contribute to the model's accuracy.

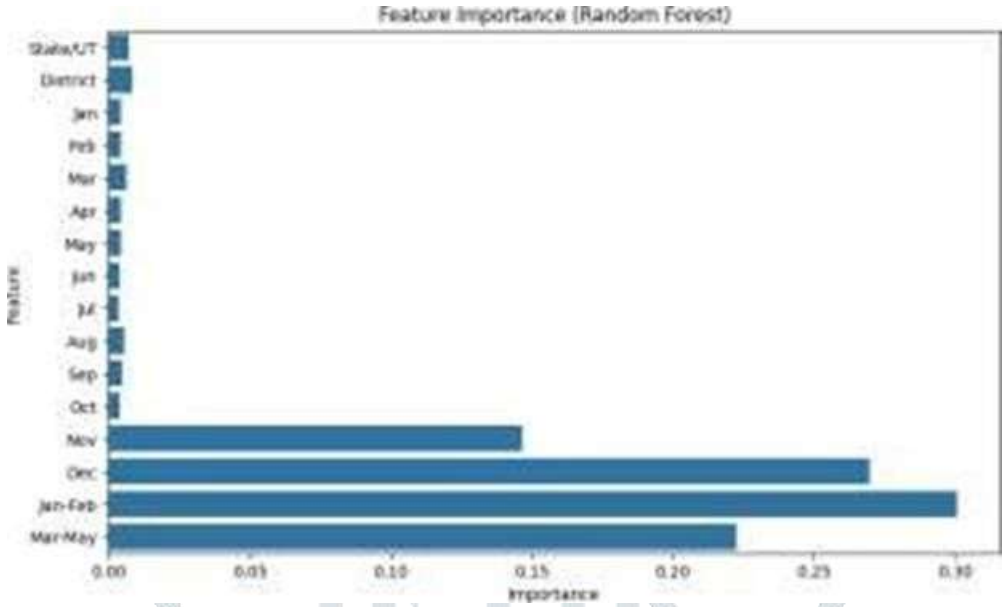


Fig. 3. Feature importance derived from Random Forest Regressor.

C. Feature Importance – Random Forest

The Random Forest model provides a feature importance ranking which helps understand the influence of each input on the prediction. As seen in Fig. 3, months such as December, January, and March significantly contribute to the model's accuracy.

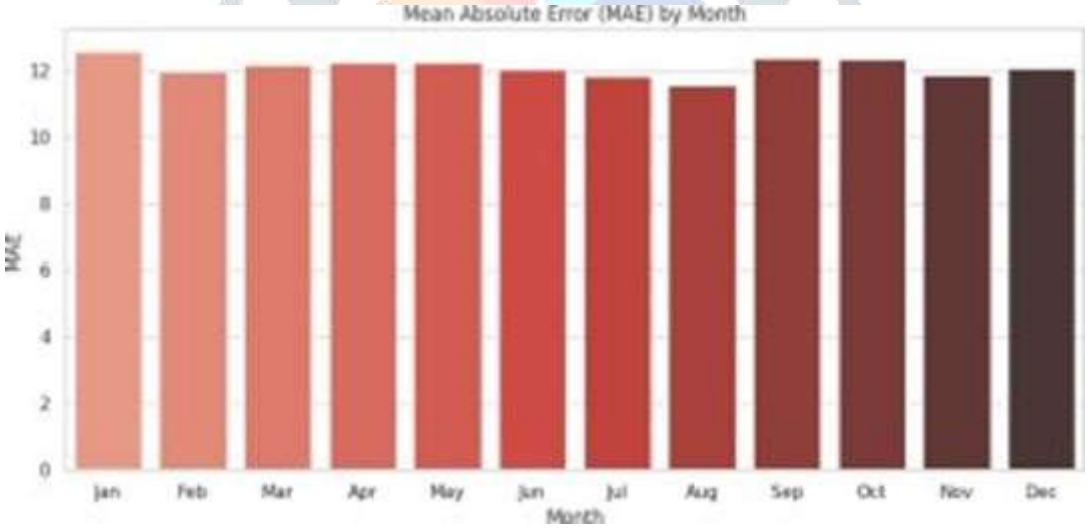


Fig. 4. Feature importance derived from Random Forest Regressor.

D. Rainfall Prediction (April 2025 to December 2026)

The predicted rainfall values generated by the model are shown in Fig. 4. The line graph indicates expected fluctuations over the forecasted months, highlighting monsoon and post-monsoon periods with higher rainfall predictions.

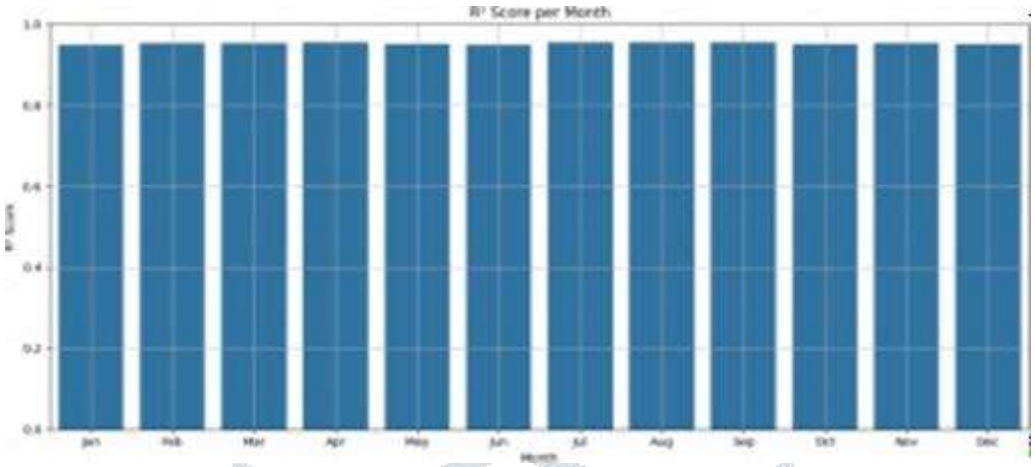


Fig. 5 Forecasted rainfall from April 2025 to December 2026.

E. Rainfall Anomalies (2017–2023)

Fig. 5 presents the deviation of actual rainfall from the long-term average. Years such as 2019 and 2021 show significant positive anomalies, while 2020 experienced a slight deficit.

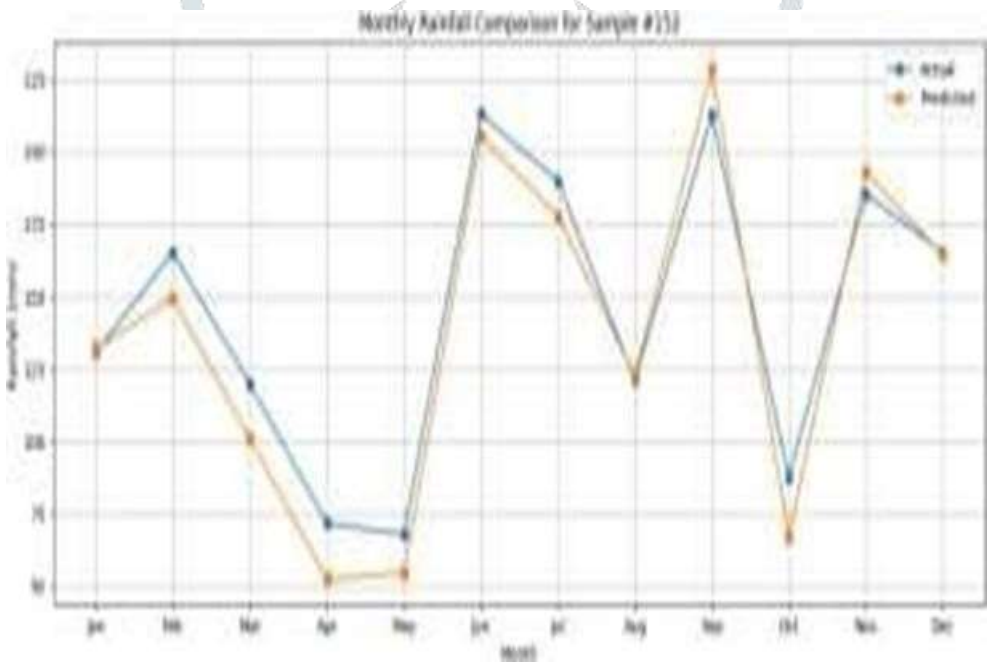


Fig. 6. Rainfall anomalies from the average across 2017–2023.

F. State-wise Average Rainfall – Heatmap

Fig. 6 visualizes the average annual rainfall for each Indian state as a heatmap. States like Kerala, Meghalaya, and the Andaman & Nicobar Islands consistently receive higher rainfall compared to arid regions.

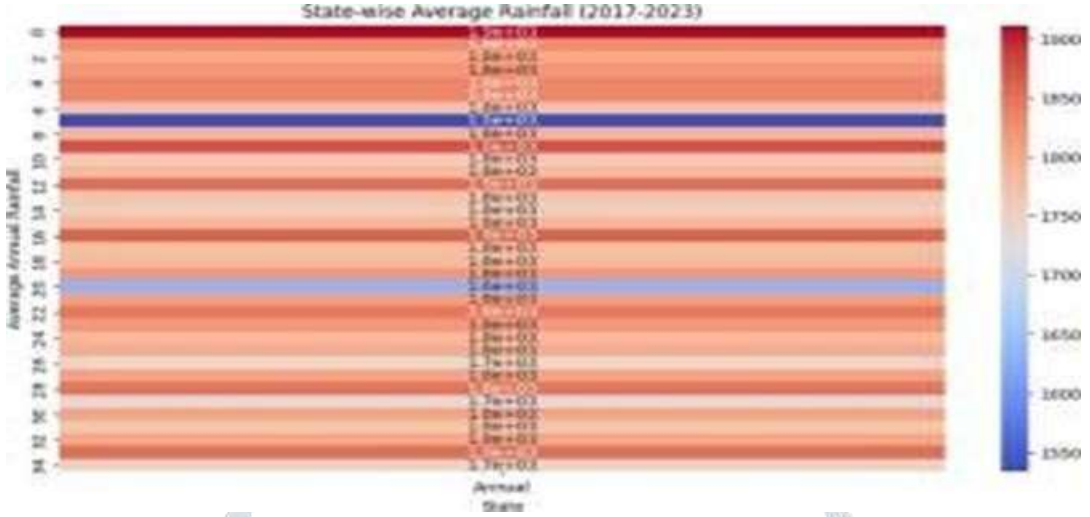


Fig. 7. State-wise average rainfall distribution.

G. Top 10 Rainfall Districts and States

Fig. 7 and Fig. 8 respectively highlight the top 10 districts and states with the highest average rainfall. These visualizations help identify rainfall hotspots across the country.

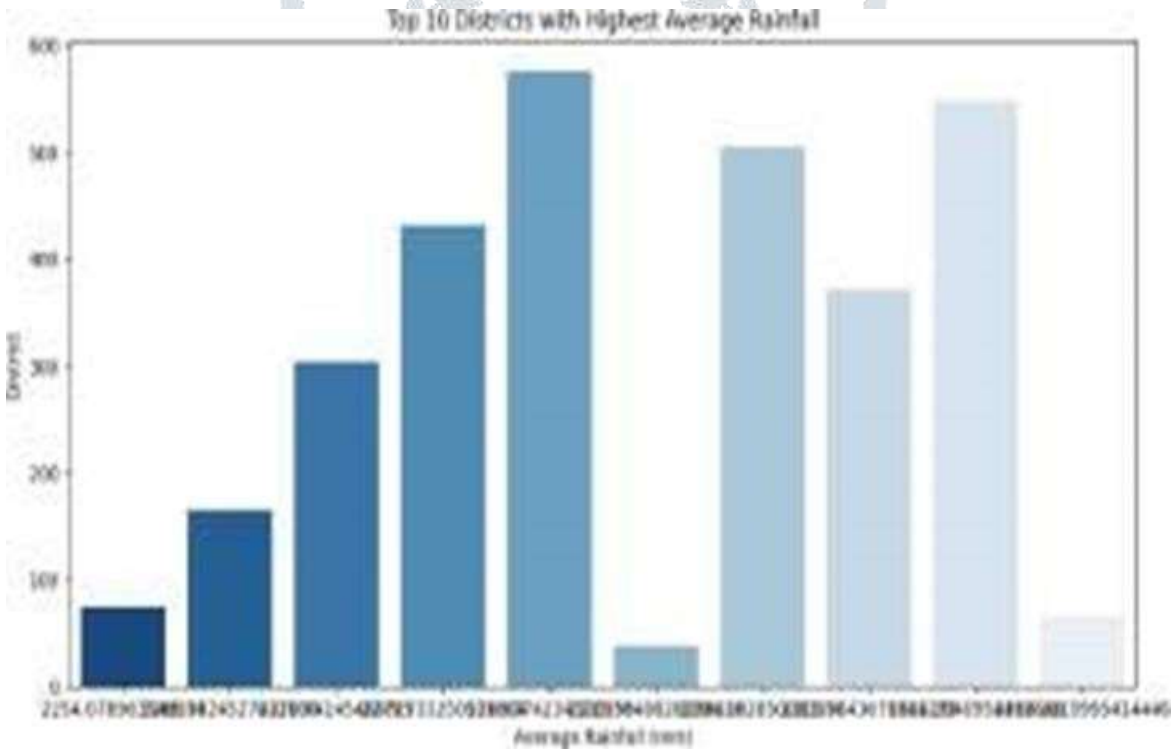


Fig. 8. Districts with the highest average rainfall.

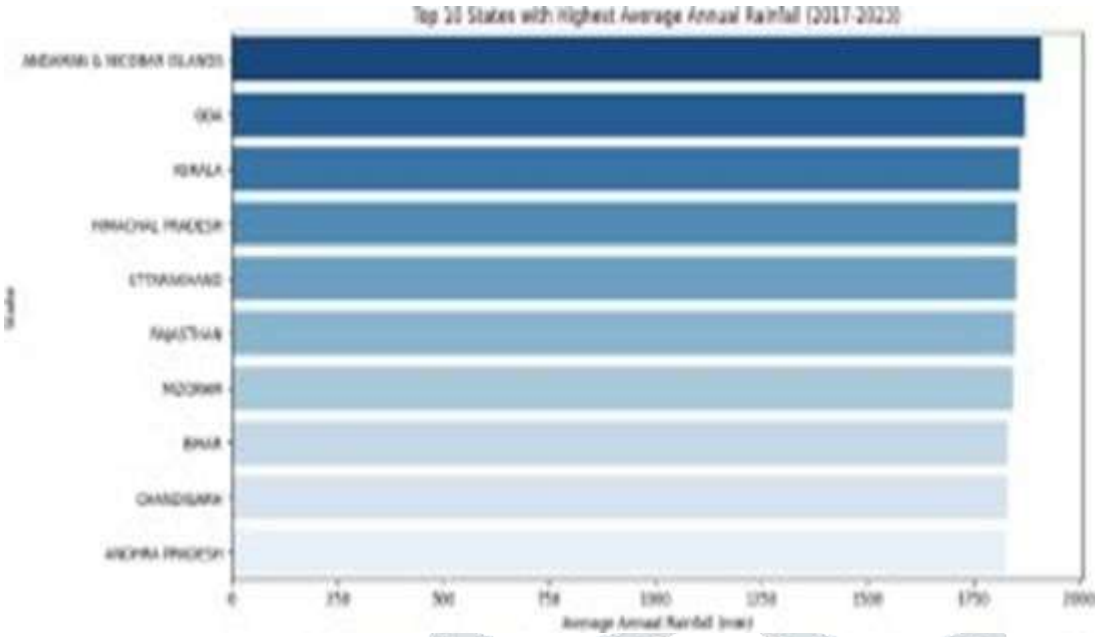


Fig. 9. States with the highest average annual rainfall.

VII. CONCLUSION & FUTURE SCOPE

In this study, a district-level rainfall forecasting system was designed and implemented using a Random Forest Regressor trained on historical rainfall data from 2017 to 2023. The model demonstrated strong performance in predicting month-wise rainfall across Indian states and districts, achieving R^2 scores consistently above 0.95 and low Mean Absolute Error (MAE) values across all months. Feature importance analysis confirmed that seasonal groupings such as Jan– Feb, Mar–May, and December played a major role in enhancing model accuracy, validating the use of temporal aggregation techniques.

The trained model was deployed using a lightweight Flask web application, allowing users to interactively select their state and district, obtain predictions for the period April 2025 to December 2026, and visualize the results in real time using dynamic graph generation with matplotlib. Forecasts are also exported in tabular form using pandas, ensuring flexibility for both visual and analytical interpretations. The system's simplicity, responsiveness, and user- focused design make it particularly suitable for farmers, researchers, planners, and other stakeholders seeking localized rainfall insights.

Despite its effectiveness, the system has several areas where enhancements are possible. Currently, the model relies entirely on historical data and does not integrate real-time meteorological feeds or remote sensing data. While Random Forest models offer interpretability and speed, their long-term generalization might be limited in the face of complex regional climate shifts. Advanced deep learning architectures such as LSTM or transformer-based time series models could further improve accuracy by capturing long- term dependencies and contextual patterns in rainfall sequences [11].

Future enhancements may include:

- Integration of real-time weather APIs (e.g., IMD, NOAA) for hybrid forecasting.
- Addition of satellite-based precipitation and humidity indices as input features.
- Implementation of a mobile app interface for offline access and push alerts.
- Extension of the model to support district-wise anomaly detection and early warnings.
- Exploration of ensemble stacking to combine predictions from multiple algorithms.

By continuously improving data inputs, model complexity, and accessibility, the proposed system can evolve into a robust decision- support tool for climate-aware planning and disaster mitigation in agriculture and environmental management.

REFERENCES

[1] S. Kumar and A. Sharma, "Machine Learning-Based Rainfall Prediction Techniques: A Review," Environmental Informatics Archives, vol. 13, pp. 85–95, 2021.
[2] R. Nair and S. Babu, "Rainfall Prediction using Random Forest Machine Learning Technique," International Journal of Computer Applications, vol. 162, no. 1, pp. 6–10, 2017.
[3] M. R. Goyal and P. R. Mehta, "Rainfall Forecasting Using Statistical and ML Models: A Comparative Review," Climate Dynamics Journal, vol. 45, no. 3, pp. 881–899, 2020.
[4] R. P. Singh, M. Kumar, and D. Yadav, "Application of Neural Networks in Rainfall Forecasting: A Review," Journal of Hydrologic Engineering, vol. 21, no. 3, pp. 04016073, 2016.
[5] V. S. Pandey and A. Jain, "A Comparative Study of Traditional vs. ML-Based Rainfall Prediction Models," International Journal of Environmental Research and Development, vol. 10, no. 1, pp. 27–32, 2020.
[6] A. Sharma and R. Mehta, "A Flask-Based Weather Forecasting System Using LSTM," International Journal of Scientific & Engineering Research, vol. 11, no. 6, pp. 1234 1240, 2020.
[7] J. Q. Yu, Y. Ding, and L. M. Zhao, "Rainfall Forecasting Using Deep Neural Networks and Gradient Boosting Machines," Journal of Atmospheric and Oceanic Technology, vol. 37, no. 9, pp. 1481–1495, 2020.
[8] T. Zhang et al., "Short-term rainfall prediction using XGBoost and LSTM hybrid models," Advances in Meteorology, vol. 2020, Article ID 3930675, 2020.
[9] L. Singh and P. Roy, "Time Series Forecasting for Monsoon Rainfall using ARIMA and LSTM," International Journal of Engineering Research & Technology, vol. 9, no. 7, pp. 546 552, 2020.
[10] K. Roy and M. Banerjee, "Time Series Rainfall Prediction using LSTM and Attention-Based Models," International Journal of Climate Informatics, vol. 5, no. 2, pp. 95–103, 2022.
[11] N. Chauhan and R. Vyas, "Comparative Analysis of Decision Trees, SVM, and Random Forest for Rainfall Forecasting," International Journal of Computer Sciences and Engineering, vol. 8, no. 3, pp. 125–130, 2020.