Rainfall Analysis of India

Major Project

by

Manasa Swamireddy
(CS20B1022)

ಭಾರತೀಯ ಮಾಹಿತಿ ತಂತ್ರಜ್ಞಾನ ಸಂಸ್ಥೆ ರಾಯಚೂರು
भारतीय सूचना प्रौद्योगिकी संस्थान रायचूर
Indian Institute of Information Technology Raichur

to

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF INFORMATION TECHNOLOGY
RAICHUR-584135, INDIA

Apr 2023

DECLARATION

I, **Manasa Swamireddy** (**Roll No: CS20B1022**), hereby declare that, this report entitled **"Rainfall Analysis of India"** submitted to Indian Institute of Information Technology Raichur towards partial requirement of **Bachelor of Technology** in **Computer Science and Engineering** is an original work carried out by me under the supervision of **Dr. Priodyuti Pradhan** and has not formed the basis for the award of any degree or diploma, in this or any other institution or university. I have sincerely tried to uphold the academic ethics and honesty. Whenever an external information or statement or result is used then, that have been duly acknowledged and cited.

Raichur-584135	**Manasa Swamireddy**
Apr 2024	**CS20B1022**

CERTIFICATE

This is to certify that the work contained in this project report entitled **"Rainfall Analysis of India"** submitted by **Manasa Swamireddy** (**Roll No: CS20B1022**) to the Indian Institute of Information Technology Raichur towards partial requirement of **Bachelor of Technology** in **Indian Institute of Information Technology, Raichur** has been carried out by her under my supervision and that it has not been submitted elsewhere for the award of any degree.

Raichur-584135
Apr 2024

Dr. Priodyuti Pradhan
Project Supervisor

ABSTRACT

The main aim of the project **Rainfall Analysis of India** is to understand diverse rainfall patterns in India. The study of rainfall patterns is essential for understanding the climate and planning water resources. This project takes a comprehensive approach to analyzing rainfall in India, considering trends, and district correlations, and forecasting it. The study begins by scrutinizing the trend of rainfall in each state of India from 2009 - 2023. We have successfully validated trends for 25 states in India through Mann-Kendall (MK), and Sens's Slope methods. Leveraging Principal Component Analysis (PCA), the research further investigates district-level features within each state, identifying pivotal districts that significantly influence rainfall within a particular state. Understanding the influence of these key districts helps to provide valuable insights for water resource management, agriculture planning, and disaster preparedness efforts. Then by utilizing a dataset spanning 115 years, a Long Short-Term Memory (LSTM) model is employed for forecasting rainfall values. The integration of LSTM offers a framework for capturing the intricate temporal dependencies inherent in rainfall data, thereby facilitating accurate predictions. Through the amalgamation of trend analysis, district-level scrutiny, and advanced forecasting techniques, this study provides valuable insights into the complex dynamics of rainfall variability in India. The study's findings can be used to understand the climate of India and plan for future water resources.

Keywords: Rainfall analysis, Trends, District correlations, Forecasting, Principal Component Analysis (PCA), Long Short-Term Memory (LSTM).

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Rainfall patterns in India exhibit considerable variability across regions and time periods, profoundly impacting the nation's climate, agriculture, water resources, and overall socio-economic landscape. Understanding these patterns is paramount for effective resource management, disaster preparedness, and sustainable development initiatives. The significance of studying rainfall patterns in India cannot be overstated. With its diverse geography, ranging from arid deserts to humid tropical regions, India experiences a wide spectrum of rainfall regimes. Rainfall patterns influence the frequency and intensity of natural hazards such as floods, droughts, and landslides. By analyzing historical rainfall data it helps the authorities to take proactive measures to mitigate their impacts and ensure community safety. The combination of trend analysis, district-level correlations, and advanced forecasting techniques, we obtain valuable insights into the intricate dynamics of India's rainfall patterns. The findings of this study hold immense potential for enhancing our understanding of the Indian climate. This report provides an overview of various rainfall patterns in India, explores the potential benefits of the proposed solution, outlines the architecture and features, and discusses the implications and prospects of implementing such a solution.

1.1 Rainfall and Monsoon Patterns in India

Rainfall is the lifeblood of India's agriculture and a critical factor in its socio-economic well-being. India's climate is heavily influenced by the southwest and northeast monsoons, which bring the majority of the country's annual rainfall. Figure 1.1 depicts the rainfall of India over the past 15 years, i.e. from 2009 to 2013. The southwest monsoon, typically occurring from June to September, is the primary source of rainfall for most of India. This monsoon is characterized by moist air masses from the Indian Ocean, which interact with the Indian landmass, resulting in widespread rainfall. The northeast monsoon, also known as the retreating or winter monsoon, occurs from October to December. It brings rainfall to the southeastern coast of India, including states such as Tamil Nadu, Andhra Pradesh, and parts of Karnataka and Kerala.

Based on diverse geographical locations in India and monsoon periods in India, we could classify the states of India into six groups which are listed below.

- **Group 1 (May to October)**: This monsoon period comprises the northeastern states of India, which includes Assam, Meghalaya, Nagaland, Sikkim, Mizoram, Tripura, Arunachal Pradesh, West Bengal, and Manipur. They primarily receive rainfall from both the southwest

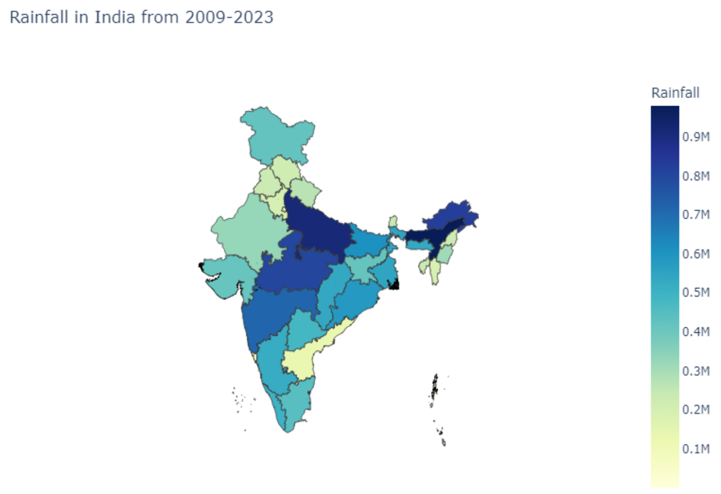


Figure 1.1: India's Rainfall from 2009-2023

and northeast monsoons. Figure 1.2 shows the distribution of rainfall among these states.

- **Group 2 (June to October):** The states for this group comprise Andhra Pradesh, Bihar, Gujarat, Maharashtra, Telangana, and Uttar Pradesh. They receive rainfall, particularly from the southwest monsoon. Figure 1.3 depicts the amount of rainfall received by these states.

- **Group 3 (June to September)**: The North Indian states namely, Jammu and Kashmir, Himachal Pradesh, Haryana, Punjab, Jharkhand, Rajasthan, Uttarakhand, and Chhattisgarh comprise this group. These also receive rainfall from the southwest monsoon but for a short

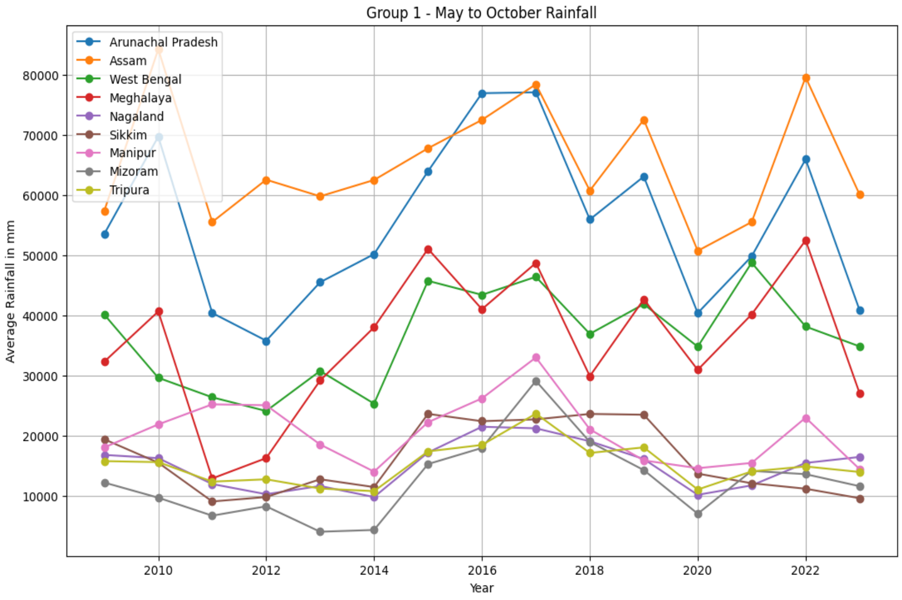


Figure 1.2: Rainfall in Group 1 states

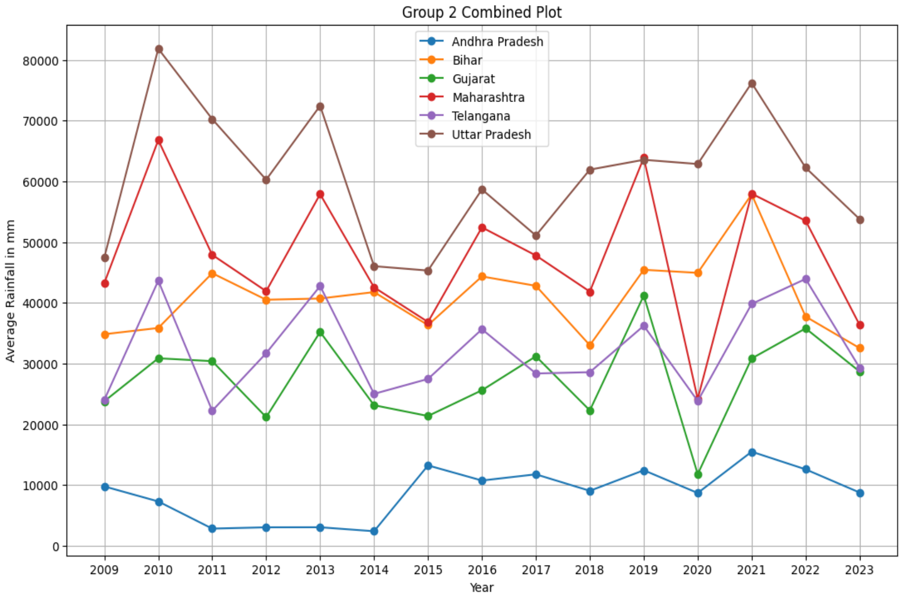


Figure 1.3: Rainfall in Group 2 states

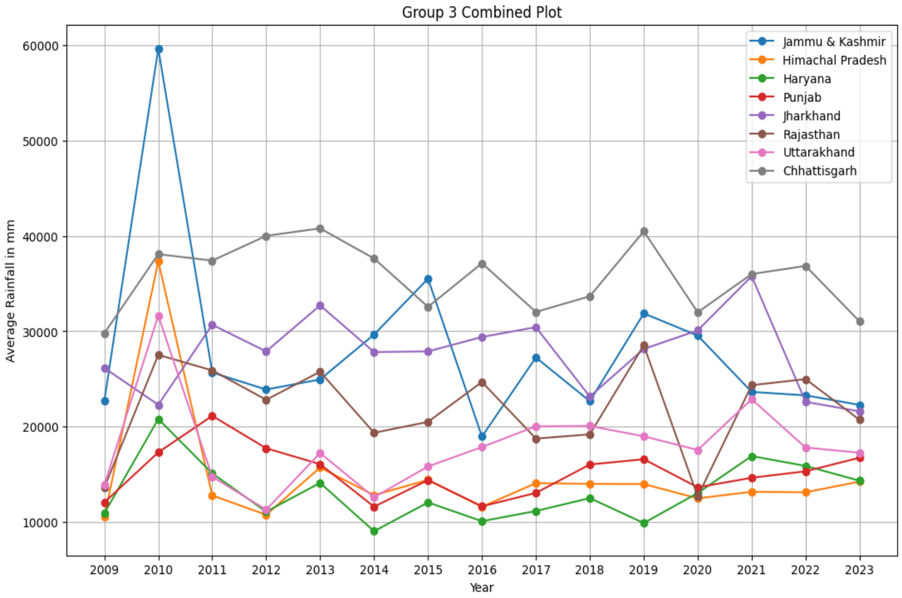


Figure 1.4: Rainfall in Group 3 states

period, i.e. from June to September. Figure 1.4 shows the combined plot of these states.

- **Group 4 (June to November):** This monsoon period comprises the states, of Karnataka, Madhya Pradesh, and Odisha. Their primary source of rainfall is the southwest monsoon from June to September, and also receive rainfall from the retreating monsoon from October to November. Figure 1.5 shows the distribution of rainfall among these states.

- **Group 5 (June to December):** This group consists of Puducherry and Tamil Nadu, which receive rainfall from the southwest monsoon in the months of June to September. However, unlike other parts of

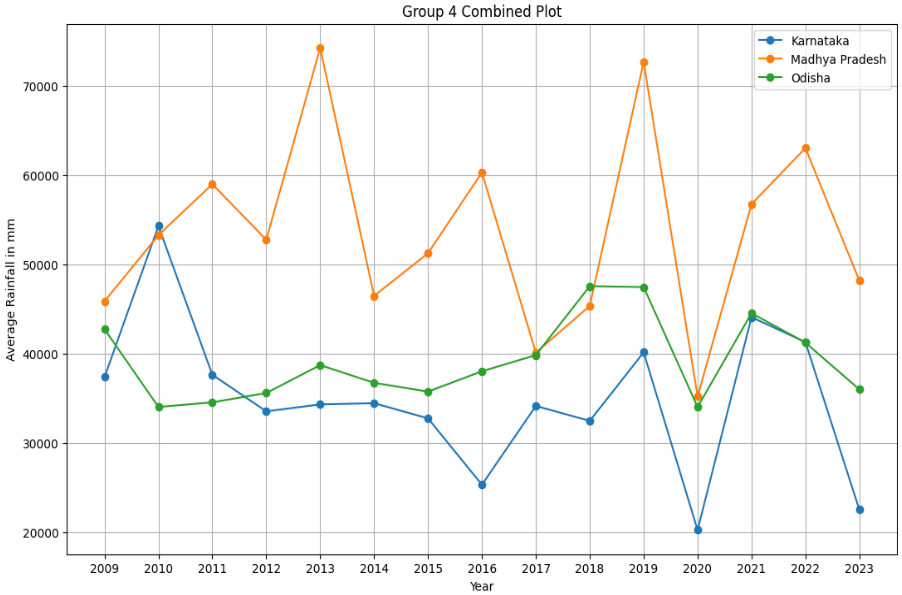


Figure 1.5: Rainfall of Group 4 states

India, Tamil Nadu and Puducherry experience a phenomenon known as the ”break monsoon” or ”Northwest Monsoon,” which brings rainfall to the state during the initial onset of the Southwest Monsoon. This usually happens in June and July. Also, they receive rainfall from the northwest monsoon also known as the retreating monsoon from October to December. Figure 1.6 shows the rainfall received by this group.

- **Group 6 (June to August, October to November):** This group comprises the state of Kerala which has two monsoon periods. The Southwest Monsoon is the primary rainy season for Kerala, typically occurring from June to September. It also receives rainfall from the Northeast Monsoon, which occurs from October to December The state’s geography, with its Western Ghats in the east and its long coast-

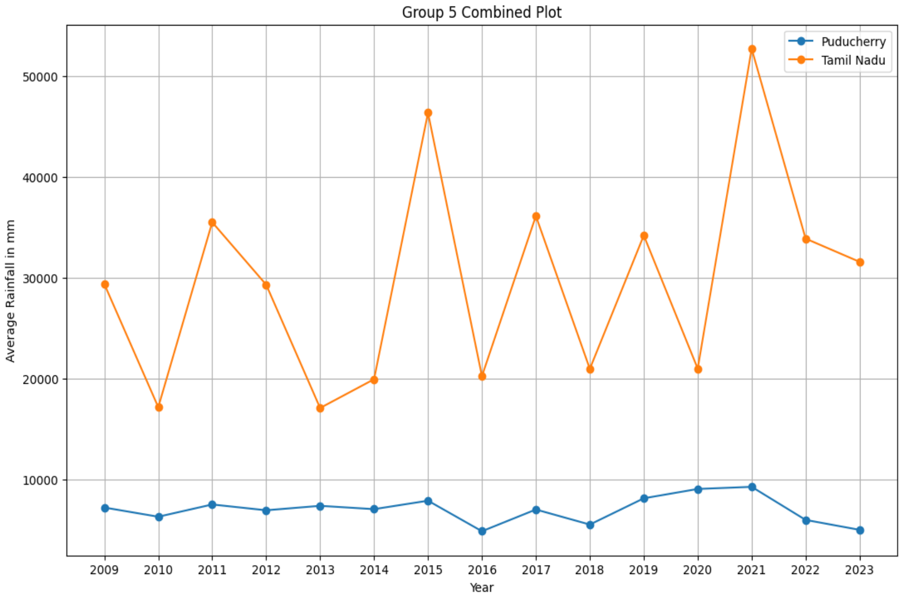


Figure 1.6: Rainfall of Group 5 states

line along the Arabian Sea in the west, contributes to its unique monsoon patterns and rainfall distribution. Figure 1.7 shows the rainfall distribution in Kerala.

Thus, we grouped all the states of India in the above-mentioned monsoon periods.

1.2 Existing Challenges

The rainfall patterns in India play a vital role in shaping the country's agriculture, water resources, and overall economy. However, analyzing rainfall data in India poses several significant challenges that require careful consideration. Here, we elaborate on these challenges in more detail:

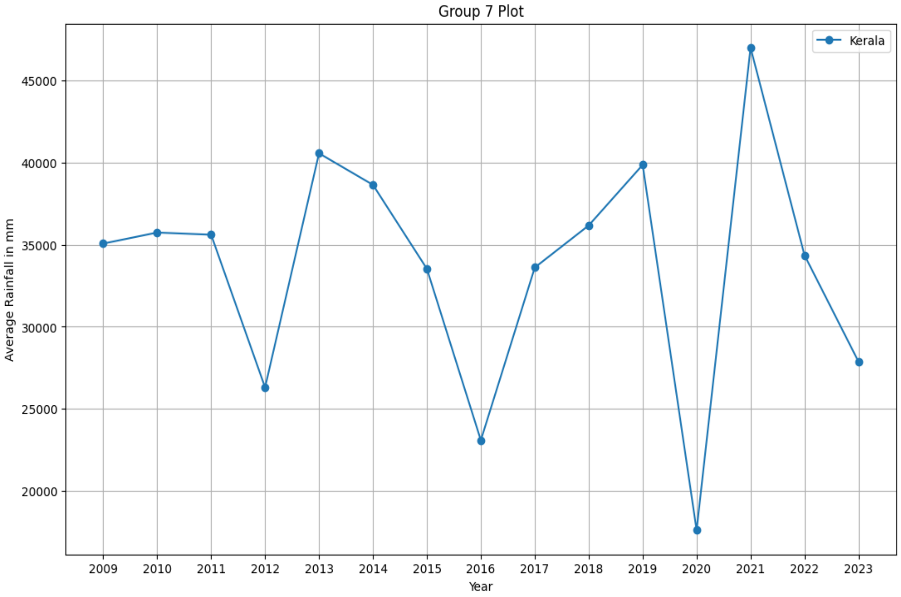


Figure 1.7: Rainfall of Group 6 states

1.2.1 Lack of District-Level Analysis in Trend Detection

In trend analysis of rainfall data, a common practice is to aggregate the data at the state or regional level to derive average trends. However, this approach overlooks the spatial heterogeneity of rainfall patterns within states and fails to capture local-scale variations, particularly at the district level. For example, in the state of Maharashtra, India, annual rainfall trends from 1901 to 2018 show a significant decline in the western part of the state, while the eastern part has experienced an increase. This contrast is attributed to the influence of the Arabian Sea on the western coast and the Bay of Bengal on the eastern coast. Thus we could not capture the correct trend of the state as a whole by taking average rainfall values into account. We need to analyze the trend of each district in a state and calculate the trend of that particular state.

1.2.2 Modelling Data without Dimensionality Reduction

When dealing with data at a fine geographic scale like districts for forecasting rainfall, considering all districts individually without any dimensionality reduction can lead to several challenges. Each district represents the rainfall value in the dataset, leading to a high-dimensional dataset. With potentially hundreds or thousands of districts, the dataset can become extremely large and computationally intensive to analyze. For a state, districts within close proximity or with similar characteristics may exhibit redundancy or high correlation in their data. Including all districts individually to forecast the rainfall can hinder model performance, generalization to new data, and the ability to extract meaningful insights from the results in redundant information, leading to inefficiencies and potentially biased estimates. Also, analyzing high-dimensional datasets with numerous districts requires significant computational resources, including memory, processing power, and time. This can pose challenges for data processing, model training, and inference tasks. Thus we need to address these challenges by transforming the original dataset into a lower-dimensional space while preserving most of the important information.

1.2.3 Rainfall Forecasting

Traditional rainfall forecasting models like ARIMA are linear models and may struggle to capture the complex nonlinear relationships present in rainfall data. ARIMA models are typically designed for short-term forecasting and may not effectively capture long-term dependencies in rainfall time series data. Feedforward neural networks, including Multilayer Perceptrons (MLPs), lack explicit memory mechanisms to capture temporal dependencies over time. Similar to feedforward neural networks, CNNs lack explicit memory mechanisms to retain information over long time horizons. This limitation may affect their ability to model long-term trends and seasonal patterns in rainfall data effectively. In summary, while other models may have their strengths in certain applications, they may lack the ability to effectively capture the complex temporal dynamics, nonlinear relationships, and long-term dependencies present in rainfall data. Thus we need a model that excels in capturing complex temporal dynamics, nonlinear relationships, and long-term dependencies, making it highly suitable for rainfall data analysis, i.e. LSTM (Long Short-Term Memory).

1.3 Structure

This thesis is organized as follows. Chapter 2 describes the Related Works. Chapter 3 discusses the Methodology, which includes the problem statement, architecture, and approach to solving the problem. Chapter 4 explains the Results and Inference, and finally, Chapter 5 discusses the Conclusion and the future prospects of this thesis.

Chapter 2

Related Works

In the analysis of rainfall in India, numerous research efforts and notable works have significantly contributed to the advancement of this field. Here are some key related works in this area:

2.1 Comparison of long-term and short-term trends in India

This study analyzes long-term and short-term rainfall trends across India's subdivisions using four trend tests, Mann-Kendall test, Spearman's Rank Order Correlation test, Wald-Wolfowitz Run tests on data, and Wald-Wolfowitz Run tests on successive differences of data and four change point tests, Pettitt's test, Von Neumann Ratio test, Buishand's Range test, and the graphical test of Cumulative Departures from Mean. Statistically significant trends were found in seven subdivisions, with five also showing a change point. Decreasing trends were observed in four subdivisions while increasing trends were seen in three. The mean was unstable in nineteen subdivisions, including all nine with identified change points. Short-term trends differed from long-term trends in several subdivisions, emphasizing the need for short-term trend analysis to assess possible climate change effects. These methods detected the trends but did not detect for other nineteen subdivisions. Moreover, the average values of the rainfall of a subdivision is taken into account which results in misinterpretation, rather than analyzing each district's trend.

2.2 Study of Long Short-Term Memory for Rainfall Prediction in India

In this study, four LSTM models were applied to predict average monthly rainfall in India, and their performances are compared with a benchmark model found in the literature. Average monthly rainfall data for All-India from 1871 to 2016 was employed for training and testing the four LSTM models. The models are compiled using the MSE loss function, and the Adam optimization technique was employed as the optimizer. The performance of the four LSTM models was estimated using statistical metrics such as MAE and RMSE. This study found that more numbers of neurons and stacking the LSTM layers can improve the LSTM model performance. The LSTM Model-4 achieved an RMSE of 245.30, whereas the existing benchmark model achieved an RMSE of 251.63. Although implemented, this model is not accurate enough as the RMSE values are high. So there is a need to improve the accuracy of the LSTM model for more accurate results.

2.3 Annual Rainfall Prediction Using Time Series Forecasting

This study reviews univariate forecasting techniques for rainfall prediction. The purpose is to determine the most accurate and statistically compelling technique. The techniques considered are regression analysis, clustering, autoregressive integrated moving average (ARIMA), error, trend, seasonality (ETS), and artificial neural network (ANN). The paper provides a comparison of the performance of these techniques based on accuracy measures such as mean square error (MSE), root mean square error (RMSE), mean absolute percentage error (MAPE), and mean absolute error (MAE). The results indicate that ARIMA performs well on the given data. However, this study only considered only a few regions of India. When taking the large data which comprises all the districts of India, the ARIMA model fails to correctly forecast annual rainfall. Thus, there is a need to reduce the dimensionality of the features, i.e. the districts, and retain the important data when forecasting for a state in India.

Chapter 3

Methodology

This section focuses on the problem statement, the approach, and methodology used for trend detection, district dimensionality reduction, forecasting rainfall, the detailed explanation of the design of architecture, and the diagram of the proposed architecture.

3.1 Problem Statement

The research work done in the field of Rainfall Analysis in India with existing methodologies have several challenges. In trend detection of rainfall, the complete analysis of all the states of India has not been done. Traditional methods utilize average rainfall data at the state or regional level, instead of analyzing the trend at the district level of each state. Consequently, the average rainfall values do not accurately represent the state's overall trend. For forecasting the rainfall we require a less dimensional dataset for accurate prediction. High-dimensional datasets from districts lead to computational challenges and hinder model performance. To address this, we need dimensionality reduction techniques while preserving important information. Another significant challenge is that the existing rainfall forecasting models have limitations, such as a lack of memory mechanisms to capture complex nonlinear relationships and long-term dependencies in rainfall data. Therefore, a model with the ability to capture complex temporal dynamics, nonlinear relationships, and long-term dependencies, such as LSTM, is needed for accurate rainfall data analysis.

3.2 Approach

This section is divided into three subsections, one for trend detection of all states of India, the second for dimensionality reduction of features,i.e. the districts for each state, and the third for forecasting rainfall.

3.2.1 Rainfall Trend Detection

In the process of rainfall trend detection, we recursively find the trend for each district in a state, and then by combining the results of all the districts we identify the trend of the rainfall in that particular state whether it is increasing, decreasing or no change observed for the period 2009-2023. The steps involved in the detection of trends are briefly described below.

1. **Time Series Graph Analysis:** Plot the time series graph as shown in Figure 3.1 for each district in the state with the date as the x-axis and rainfall values as the y-axis. We then compute the slope for each straight line between 2 points. If the slope is positive, the trend is

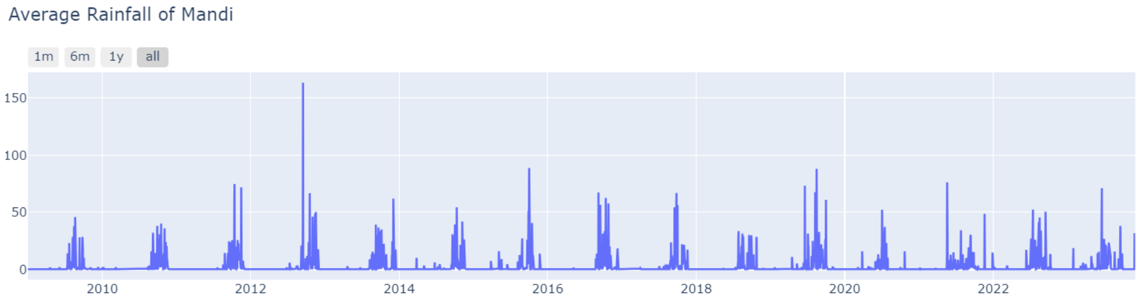


Figure 3.1: Time Series Graph for a district in India

increasing otherwise, it is negative. After computing all the slopes, we store them for further analysis to detect the trend.

2. **Calculating Parameters:** After we obtain slopes of all points in the time series graph, three parameters should be calculated. One is the average - the mean of all the slopes, second is the number of positive(increasing) slopes, and third is the number of negative(decreasing) slopes. Then we need to sum up whether the trend is increasing or decreasing based on certain conditions.

3. **Decision Trigger:** Now, we have the number of positive slopes (p), negative slopes (n), and the average of all the slopes. First, we check if the average value of slopes is positive or negative. If it is positive, there are two possibilities - p is greater than n or p is less than n. If p is greater than n then the trend could be concluded as increasing, else if p is less than n, we need to consider a threshold value of 10 to determine the trend. If the difference between negative slopes and positive slopes is less than 10, then the trend is increasing, else the trend is decreasing. Similarly, we formulate the decision trigger for

the negative average slope value. When the average is negative, two situations arise - **p** is greater than **n** or **p** is less than **n**. The same threshold value of 10 is taken into account. If the difference between positive slopes and negative slopes is less than 10, then the trend is increasing, else the trend is decreasing.

4. **Trend Detection:** We follow the above steps in detecting the trend of each district of a particular state. If there are more positive or increasing trends than negative or decreasing trends, the overall trend is considered INCREASING, else there are more negative trends than positive trends, the overall trend for the state is considered DECREASING. If the overall average slope equals or nearly equals zero and the number of increasing trends is equal to the number of decreasing trends, the overall trend is considered NEUTRAL.

Thus, through the above-mentioned steps we calculate the trends for all the states of India. Figure 3.2 gives a detailed understanding of the above-proposed trend detection methodology.

3.2.2 Forecasting Rainfall using LSTM

Rainfall forecasting is a crucial task for various applications such as agriculture, water resource management, and disaster preparedness. Traditional methods of rainfall prediction often rely on statistical models that may not capture the complex temporal dependencies present in meteorological data. Long Short-Term Memory (LSTM) networks, a type of recurrent neural network (RNN), have shown promising results in time series forecasting tasks

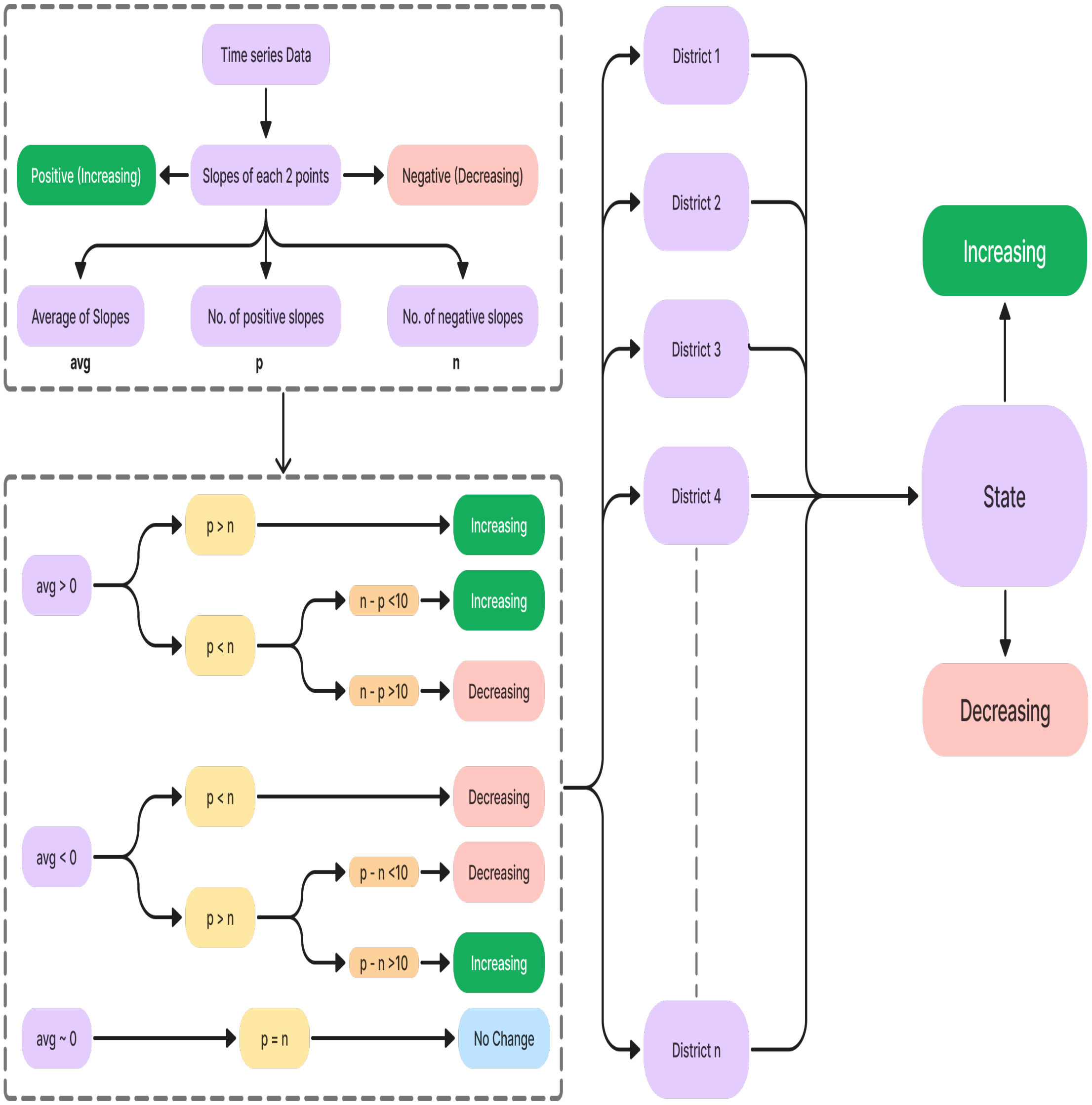


Figure 3.2: Architecture of Trend Detection

due to their ability to capture long-term dependencies and remember information over extended periods. The steps involved in implementing the model for forecasting rainfall is described below.

1. **Data Preprocessing:** Before training an LSTM model for rainfall forecasting, the rainfall data needs to be preprocessed to ensure that the model can learn effectively from the data. Rainfall data can exhibit a wide range of values, making it difficult for the LSTM model to learn. We need to normalize the data as it scales the data to a specific range, typically between 0 and 1. This normalization is done using MinMaxScaler from the sklearn library. This helps to ensure that the model is not biased towards extreme values and can learn from the entire dataset.

2. **Splitting the train and test data:** The preprocessed rainfall data is then split into two sets: a training set and a testing set. The training set is used to train the LSTM model, while the testing set is used to evaluate the performance of the trained model. The training set comprises of 80 percent of the total data while the testing set comprises of 20 percent of data. To feed the data to the model, we split the data using sliding window approach, where the data is divided as sequences. Each sequence serves as an input, while the subsequent data point represents the corresponding output value. This process constructs input-output pairs necessary for training machine learning models transforming raw time series data into structured input-output pairs, facilitating the development and evaluation of forecasting models.

3. **LSTM Layer:** LSTM is a type of recurrent neural network (RNN) architecture designed to overcome the vanishing gradient problem in traditional RNNs. It is well-suited for processing and making predictions on sequential data. We have defined two LSTM layers once with 64 momory units and the other with 32 momory units, activation function ReLU. It plays a crucial role in capturing temporal dependencies and extracting meaningful features from the input sequences, making it suitable for accurate time series prediction.

4. **ReLU Activation function:** ReLU activation is applied layer-wise to introduce non-linearity into the model. Non-linearity is essential for deep learning models to learn complex patterns. Without non-linearity, the model would be linear and would only be able to learn linear relationships between the input and output data. ReLU activation helps to prevent vanishing gradients by ensuring that the gradients are always positive.

5. **Dropout:** Dropout is a regularization technique that prevents overfitting by randomly dropping a certain percentage of neurons during training. This forces the network to learn redundant representations, improving generalization. Dropout works by randomly setting a certain percentage of the network's neurons to zero during training. This forces the network to learn to rely on multiple paths to make predictions, which makes it more robust to noise and outliers.

6. **Adam Optimizer:** The Adam optimizer is a versatile and efficient optimization algorithm widely used in training LSTM models. It com-

bines ideas from two other popular optimization algorithms, Adaptive Gradient Algorithm (AdaGrad) and RMSProp (Root Mean Square Propagation), to provide an efficient and effective approach to gradient descent. Adam adapts the learning rates for each parameter during training. It computes individual learning rates for each parameter by considering the magnitudes of recent gradients and the magnitudes of recent squared gradients. It can converge faster compared to traditional gradient descent algorithms, especially for complex and high-dimensional optimization problems.

7. **Root Mean Squared Error:** For this model, we have used Root Mean Square Error to evaluate the performance of the model. It measures the average magnitude of the errors or residuals between predicted values and actual values. It represents the standard deviation of the residuals or prediction errors. Lower RMSE values indicate better model performance, as they reflect smaller deviations between predicted and actual values. Its intuitive interpretation and sensitivity to large errors make it a widely adopted metric in various domains, aiding in the assessment and improvement of predictive accuracy.

Thus, the above-mentioned steps are crucial in implementing an LSTM model for forecasting rainfall. Figure 3.3 depicts the architecture of an LSTM model.

3.2.3 Dimensionality Reduction with PCA

The steps involved in implementing Principal Component analysis are explained below:

1. **Data Preprocessing:** Before applying PCA, it's important to preprocess the data. This includes standardizing or normalizing the features to ensure they have a similar scale. This step helps prevent certain features from dominating the analysis due to their large values. Standardization involves subtracting the mean and dividing by the standard deviation of each feature, resulting in a mean of 0 and a standard deviation of 1. Missing values in the dataset can introduce bias and affect the accuracy of the PCA results. Therefore, it's important to handle missing values appropriately before applying PCA.

2. **Covariance Matrix Computation:** PCA begins by computing the covariance matrix of the standardized or normalized data. The covariance matrix is a square matrix that contains the variances of the individual features on the diagonal and the covariances between the features off the diagonal. It summarizes the relationships between different features in the dataset.

3. **Eigenvalue Decomposition:** Next, the covariance matrix is decomposed into its eigenvectors and eigenvalues. Eigenvectors represent the directions of maximum variance in the data, while eigenvalues represent the magnitude of variance along each eigenvector.

4. **Selection of Principal Components:** The eigenvectors are sorted in descending order based on their corresponding eigenvalues. The

eigenvectors with the highest eigenvalues capture the most variance in the data and are retained as principal components. The number of principal components to retain is determined based on the desired level of variance retention or the cumulative explained variance ratio.

5. **Projection onto Principal Components:** The original data is projected onto the selected principal components to obtain the reduced-dimensional representation of the dataset. This is achieved by multiplying the standardized or normalized data matrix by the matrix of selected principal components.

6. **Interpretation and Visualization:** The principal components are analyzed to interpret the underlying structure or patterns in the data. Each principal component represents a linear combination of the original features. The reduced-dimensional dataset can be visualized using scatter plots or other visualization techniques to explore relationships between data points and identify clusters or patterns.

In summary, implementing PCA for dimensionality reduction involves preprocessing the data, computing the covariance matrix, performing eigenvalue decomposition, selecting principal components, projecting the data onto the selected components, interpreting the results, and potentially reconstructing the original dataset. PCA is a powerful tool for reducing the dimensionality of high-dimensional data while preserving important information and facilitating subsequent analysis tasks.

Chapter 4

Implementation and Results

This section delves into the practical implementation and evaluation of the devised methodology for trend detection, forecasting rainfall and reduction of district dimensionality within a specific state. It delves into the various aspects of the methodology's application and provides a detailed analysis of its effectiveness.

4.1 Trend Detection

By the method proposed in section 3.2.1, trends were identified across all states of India. These trends plays a vital role in decision-making processes across various sectors, ranging from agriculture and water management to climate change adaptation, infrastructure planning, and public health. By understanding and monitoring rainfall trends, stakeholders can better prepare for and mitigate the impacts of climate variability and change, ultimately contributing to sustainable development and resilience in the face of environmental challenges. The obtained results are verified by using Mann Kendall test and Sen's Slope test.

4.1.1 Mann Kendall Test:

The Mann-Kendall test is a non-parametric statistical method used to detect trends in time series data, making it a valuable tool for analyzing rainfall patterns. is employed to assess whether there are significant trends in historical rainfall data, providing insights into long-term precipitation patterns. It is calculated by comparing the number of positive and negative differences between consecutive data points in the time series. A positive test statistic indicates an increasing trend, while a negative test statistic indicates a decreasing trend. The significance of the test statistic is determined by comparing it to a critical value, which is based on the number of data points in the time series and the desired level of significance.

4.1.2 Sen's Slope:

Sen's slope estimator is a widely utilized method for estimating the magnitude and direction of trends in time series data, particularly in rainfall analysis. It is a non-parametric approach that computes the median of all possible pairwise slopes between data points within a time series. Its non-parametric nature makes it robust against outliers and does not require the data to conform to a specific distribution. It provides a robust and reliable means of quantifying trends in rainfall data, enabling comprehensive analyses of climate variability and change

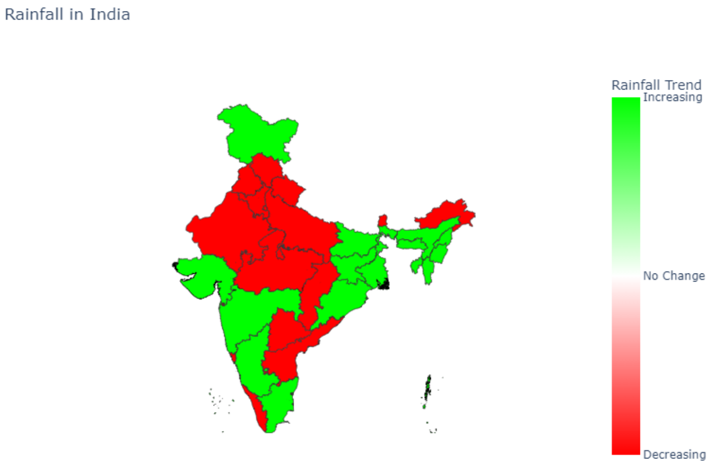


Figure 4.1: Predicted Trends

Out of 29 states in India, 25 states are giving the same results as provided by the Mann Kendall test and the Sen's Slope test. The trend exhibited by the states of India using the proposed method and by using Mann Kendall methods is depicted by Figure 4.1 and 4.2 respectively.

4.2 Implementation of LSTM for Rainfall Forecasting

The objective of this implementation is to build and evaluate a deep-learning model for forecasting rainfall. We used a dataset that contains month-wise data for each subdivision from year 1901 - 2015. Figure 4.1 depicts the sample of the dataset used for forecasting the rainfall. The deep learning model was built using LSTM, a special type of recurrence neural network

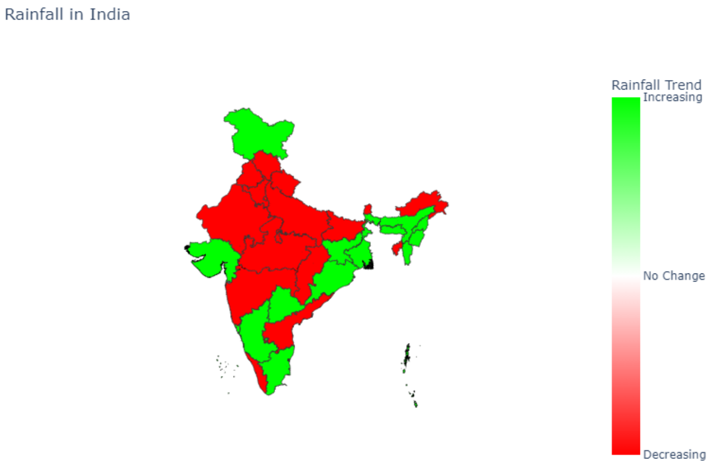
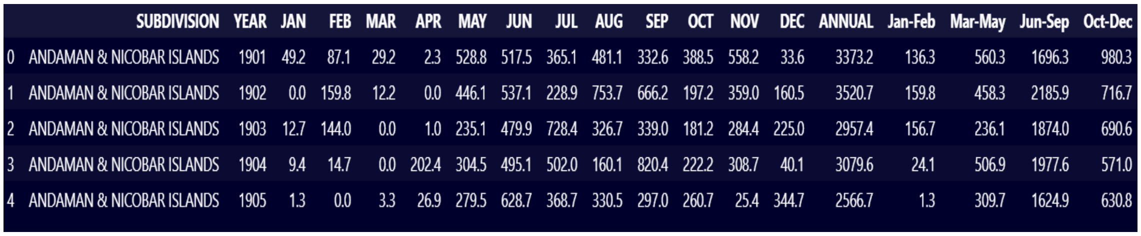


Figure 4.2: Actual Trends



	SUBDIVISION	YEAR	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	ANNUAL	Jan-Feb	Mar-May	Jun-Sep	Oct-Dec
0	ANDAMAN & NICOBAR ISLANDS	1901	49.2	87.1	29.2	2.3	528.8	517.5	365.1	481.1	332.6	388.5	558.2	33.6	3373.2	136.3	560.3	1696.3	980.3
1	ANDAMAN & NICOBAR ISLANDS	1902	0.0	159.8	12.2	0.0	446.1	537.1	228.9	753.7	666.2	197.2	359.0	160.5	3520.7	159.8	458.3	2185.9	716.7
2	ANDAMAN & NICOBAR ISLANDS	1903	12.7	144.0	0.0	1.0	235.1	479.9	728.4	326.7	339.0	181.2	284.4	225.0	2957.4	156.7	236.1	1874.0	690.6
3	ANDAMAN & NICOBAR ISLANDS	1904	9.4	14.7	0.0	202.4	304.5	495.1	502.0	160.1	820.4	222.2	308.7	40.1	3079.6	24.1	506.9	1977.6	571.0
4	ANDAMAN & NICOBAR ISLANDS	1905	1.3	0.0	3.3	26.9	279.5	628.7	368.7	330.5	297.0	260.7	25.4	344.7	2566.7	1.3	309.7	1624.9	630.8

Figure 4.3: Dataset

(RNN), specifically designed to forecast time series data and capture long tern trends. The LSTM model was trained on this dataset, and its performance was evaluated on a test set.

1. **Model Setup:** The implementation of an LSTM (Long Short-Term Memory) model for rainfall forecasting begins with the necessary imports and data preprocessing steps. The imports include libraries such as NumPy for numerical operations, Pandas for data manipulation,

Matplotlib and Seaborn for visualization, Keras and TensorFlow for building the neural network model. Firstly, the data is scaled using the MinMaxScaler from scikit-learn. After normalization, a function is defined to organize the dataset for training the LSTM model. This function takes two parameters: the dataset itself and a parameter determining the number of past time steps considered for predictions. Inside this function, two lists are initialized to hold input sequences and their corresponding output values. A loop iterates through the dataset, selecting sequences of data points equal to the specified number of past time steps. Each sequence becomes an input, and the data point immediately following it becomes the output. This process generates pairs of input-output sequences, forming the basis for training the LSTM model. The resulting input-output pairs are stored separately and split to train and test data sets.

2. **Model Execution:** The execution starts with the initialization of the sequential model. The Sequential model is a linear stack of layers, which allows for easy construction of deep learning models. Two LSTM layers are added to the model. The first LSTM layer has 64 units and is configured to return sequences, meaning it outputs the full sequence of hidden states for each input time step. The second LSTM layer has 32 units and is configured to return only the output of the last time step. Dropout layers with a dropout rate of 0.2 are added after each LSTM layer. Dropout is a regularization technique that randomly sets a fraction of input units to zero during training to prevent overfitting. A dense output layer with one unit is added to the model.

Layer (type)	**Output Shape**	**Param #**
lstm_1 (LSTM)	((None, 1, 64))	18,176
dropout_1 (Dropout)	(None, 1, 64)	36896
lstm_2 (LSTM)	(None, 32)	12,416
dropout_ (Dropout)	(None, 32)	0
dense (Dense)	(None, 1)	33

Table 4.1: Model Details

This layer produces the final output prediction for each input sequence. The model is compiled using the Adam optimizer with a learning rate of 0.0001, mean squared error (MSE) as the loss function, and accuracy as the metric to monitor during training. Callbacks are defined to monitor the validation loss during training. EarlyStopping is used to stop training if the validation loss stops improving for a specified number of epochs (patience), and ModelCheckpoint is used to save the best model weights based on the training loss. Once training is complete, the best model weights are loaded from the saved checkpoint file. The summary in Table 4.1 provides a comprehensive overview of the model's architecture, showcasing the layer configurations, and output shapes.

3. **Outcome:** After training the model, predictions are made on the training and test datasets using the prediction method of the best-trained model. Then the Root Mean Squared Error (RMSE) is calculated for both the training and test predictions using the mean_squared_error

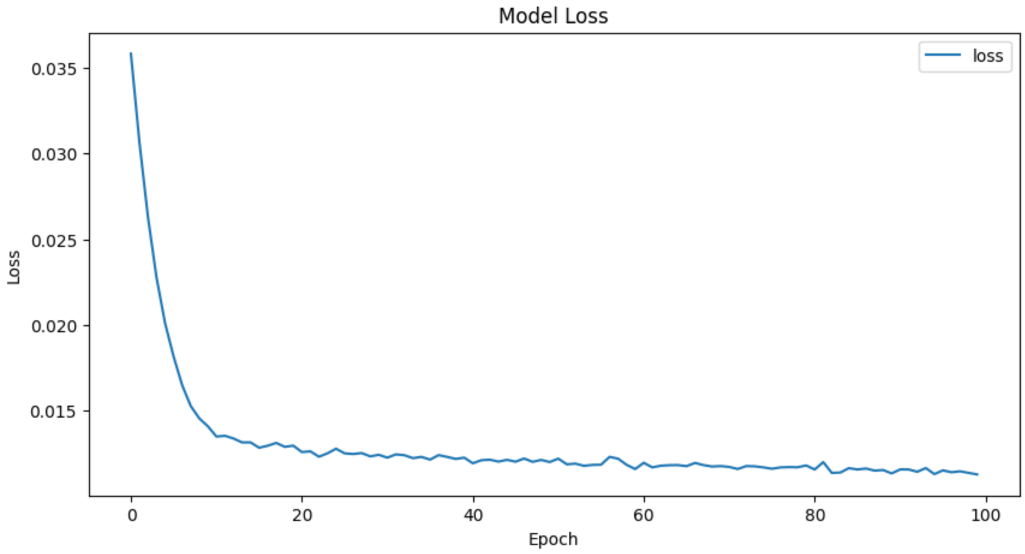


Figure 4.4: Model Loss vs Epoch Graph

function from scikit-learn. RMSE is a measure of the average deviation of the predicted values from the actual values. The RMSE values for training and testing datasets are 88.72 and 132.01, whereas the existing benchmark model achieved an RMSE of 245.30. Figure 4.4 shows the model loss vs epochs graph during the training time. The model accurately captured the trends during the training and testing which are depicted in Figures 4.5 and 4.6 respectively.

4.3 Dimensionality Reduction of Districts

As mentioned in the methodology of implementing Principal Component Analysis in section 3.2.3 we apply it to the districts of a state to reduce the correlation and dimensionality while predicting for large datasets. The

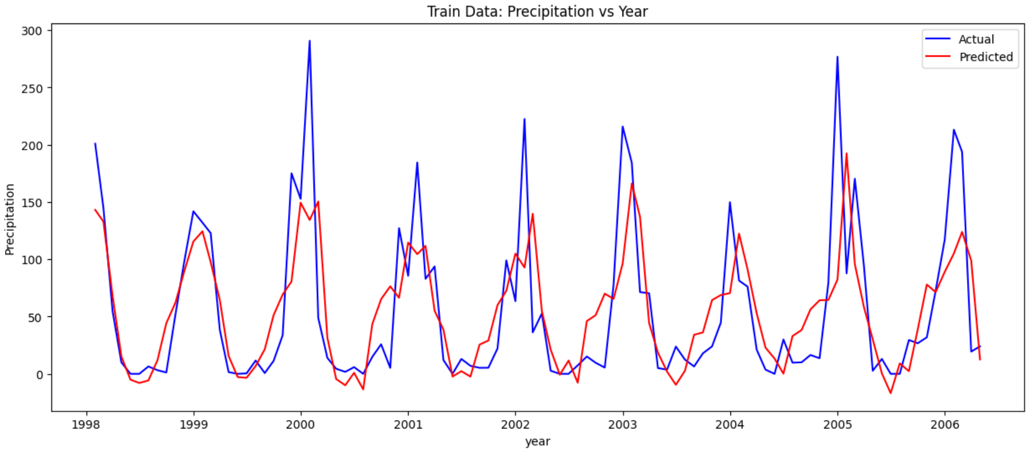


Figure 4.5: Forecasting Training Data

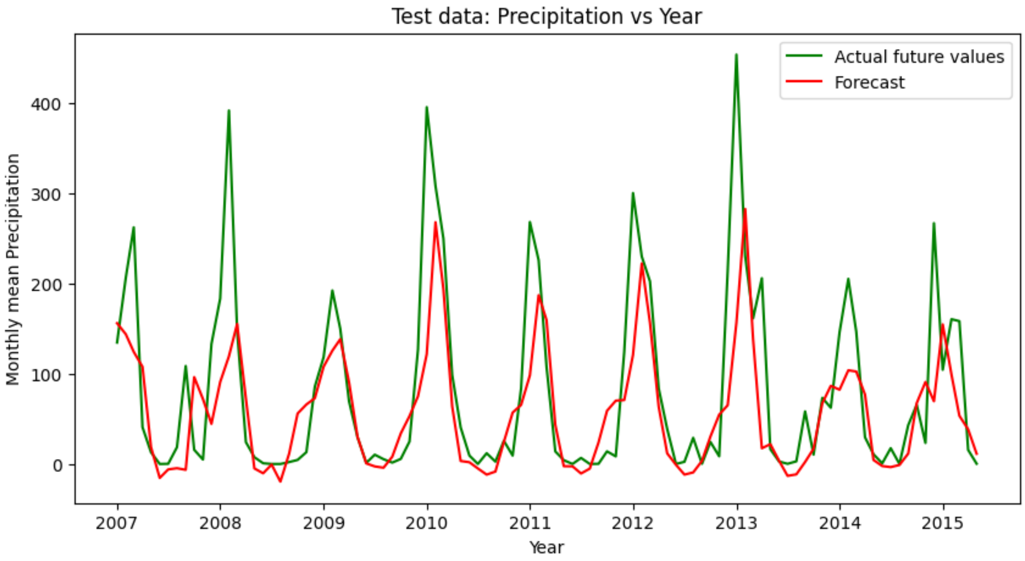


Figure 4.6: Forecasting Test Data

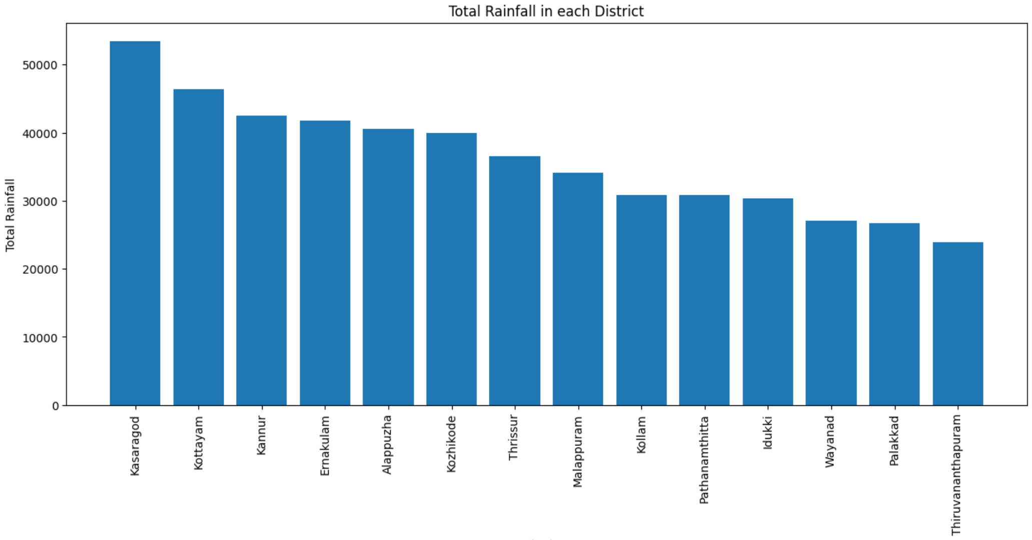
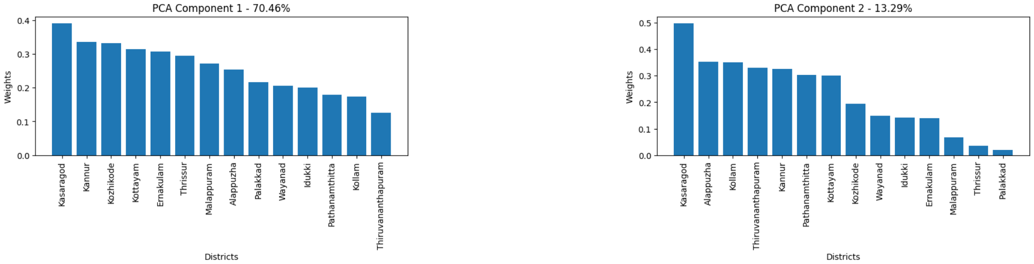


Figure 4.7: Rainfall of all Districts in Kerala

results after applying PCA are given below.

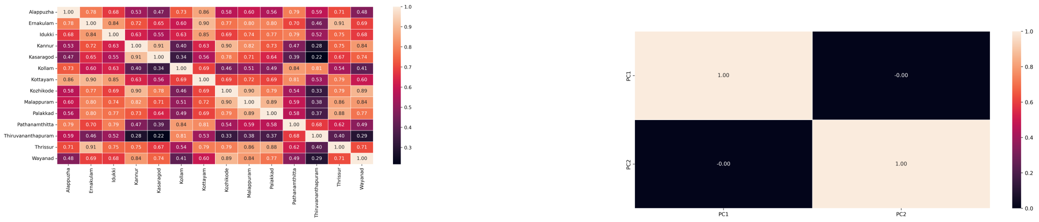
Figure 4.7 depicts the total amount of rainfall received by all the districts of Kerala. It has 14 features to consider for training and could not produce correct results. So by applying PCA, the number of features is reduced to two features depicted in Figure 4.8. Even in the principal components, we can observe that Kasargod receives the highest amount of rainfall. Thus even



(a) Principal Component 1

(b) Principal Component 2

Figure 4.8: Reduced Dimensionality from 14 to 2



(a) Original Features

(b) Reduced Features

Figure 4.9: Removal of Correlation

if we reduce the dimensions, the important features are regained. As shown in Figure 4.9 the correlation between the features is also removed.

Chapter 5

Conclusion and Future Works

In conclusion, the comprehensive analysis of rainfall patterns in India undertaken in this project has provided valuable insights into the dynamics of precipitation across the country. Through the application of the proposed methodology for trend detection, significant trends in rainfall were identified and validated for 25 states using statistical methods such as Mann-Kendall and Sens's Slope over the period from 2009 to 2023. This trend analysis serves as a foundational understanding of how precipitation is evolving over time, crucial for climate assessment and water resource management.

Furthermore, the investigation into district-level features within each state using Principal Component Analysis (PCA) has highlighted pivotal districts that exert significant influence on rainfall patterns. Understanding the role of these key districts is essential for localized water resource management, agricultural planning, and disaster preparedness initiatives. By identifying districts with outsized influence on rainfall, policymakers and stakeholders can prioritize resources and interventions more effectively.

The integration of Long Short-Term Memory (LSTM) models for rainfall forecasting has demonstrated promising results in capturing the complex temporal dependencies inherent in precipitation data. The LSTM model, trained on a dataset spanning 115 years, offers a robust framework for generating accurate predictions, thereby aiding in proactive decision-making and risk mitigation strategies.

Looking ahead, there are several avenues for future research and development in the field of rainfall analysis in India. Firstly, the refinement and validation of forecasting models, including the exploration of ensemble techniques and hybrid models, can further enhance the accuracy and reliability of predictions. Additionally, incorporating additional environmental and socio-economic variables into the analysis can provide a more comprehensive understanding of the factors influencing rainfall variability.

Moreover, there is a need for ongoing monitoring and evaluation of rainfall patterns, particularly in the context of climate change. Continuous data collection and analysis will enable policymakers to adapt strategies and interventions in response to evolving climatic conditions, ensuring the sustainability and resilience of water resources in India.

In conclusion, this study lays the groundwork for future research endeavors aimed at understanding and managing rainfall variability in India, with implications for climate resilience, water security, and sustainable development.