Exploratory Analysis of Rainfall Data in India

TEAM LEADER	B. Silpa Reddy (223H1A0518)
TEAM MEMBER	A. Kavya (223H1A0502)
TEAM MEMBER	N. Vaishnavi (223H1A0559)
TEAM MEMBER	B. Anusha (223H1A0510)

Project Description

Rainfall is one of the most important factors influencing agricultural productivity in India. Since a large portion of Indian agriculture depends on monsoon rainfall, understanding rainfall patterns is essential for crop planning, irrigation management, and ensuring food security.

Rainfall varies significantly across different states and across years. By analyzing historical rainfall data, meaningful insights can be extracted regarding seasonal trends, regional distribution, and long-term climate variability.

This project focuses on performing Exploratory Data Analysis (EDA) on Indian rainfall data to identify patterns, variations, and trends that can support agricultural planning and informed decision-making.

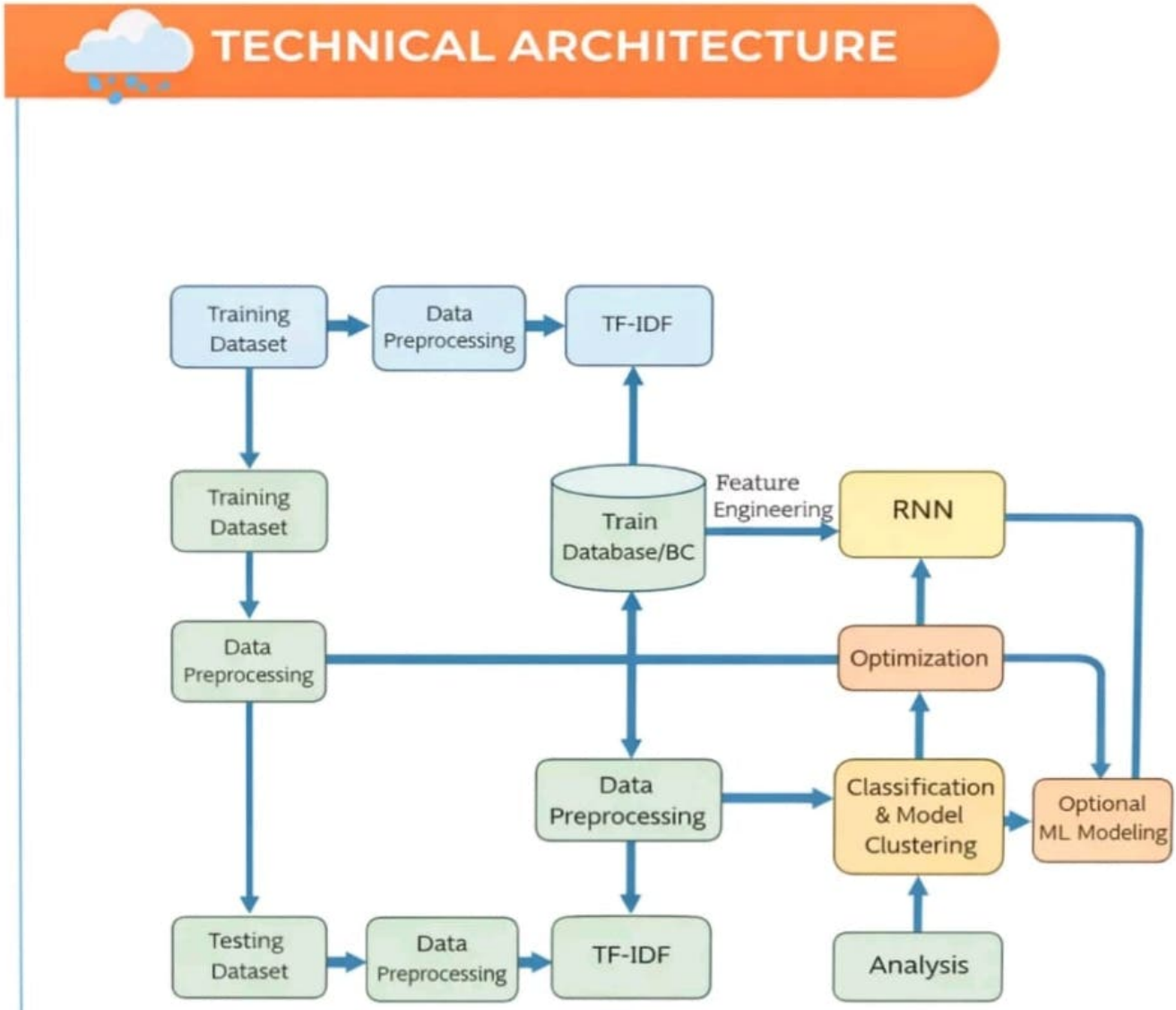
Data visualization techniques are used to better understand rainfall behavior across different regions and time periods. In future extensions, Machine Learning techniques may be applied to model and predict rainfall patterns more accurately.

Objectives

- To study historical rainfall data of India.
- To analyze seasonal rainfall distribution patterns.
- To compare rainfall trends across different states.
- To identify year-wise variations and anomalies in rainfall.
- To support agricultural planning using data-driven insights.

Technical Architecture

Machine Learning techniques are used to analyze and model rainfall behaviour in a structured way.



Project Workflow & Description

Dataset Collection from Kaggle

The rainfall dataset is collected from Kaggle, which provides publicly available datasets for research and data analysis.

Data Loading and Inspection

The dataset is loaded using Python libraries like Pandas. Functions such as head(), info(), and describe() are used to understand the structure and summary.

Exploratory Data Analysis (EDA)

EDA helps to identify rainfall trends across states and years. It shows patterns, variations, and extreme rainfall values clearly.

Data Preprocessing

Missing values are handled, column names are standardized, and important features are selected for accurate analysis.

Visualization of Rainfall Patterns

Colorful graphs like line charts and bar charts are used to visualize rainfall variations and identify highest and lowest rainfall regions.

Optional Machine Learning Modeling

Machine learning models can be applied to predict rainfall using historical data for better forecasting.

Flask-Based Web Visualization

A Flask web application can be used to create an interactive interface where users input parameters and view prediction results.

Pre-Requisites

Software Required

- Anaconda Navigator
- Jupyter Notebook / VS Code

Python Packages

- Pandas
- NumPy
- Matplotlib
- Seaborn
- Scikit-learn
- XGBoost

Tools and Techniques Used

Tool / Technique	Purpose
Python	Programming for rainfall data analysis
Pandas & NumPy	Data loading and preprocessing
Matplotlib & Seaborn	Colorful data visualization
EDA (Exploratory Data Analysis)	Finding patterns and trends
Scikit-learn	Optional machine learning modeling
Jupyter Notebook	Interactive analysis and coding

Project Flow

1. Dataset Collection from Kaggle
2. Data Loading and Understanding
3. Data Cleaning and Preprocessing
4. Performing Exploratory Data Analysis (EDA)
5. Identifying Rainfall Trends Across Indian States
6. Visualization using Charts and Graphs
7. Drawing Conclusions and Insights for Agriculture

Pre-Requisites

Software Required:

- Anaconda Navigator
- Jupyter Notebook / VS Code

Tools & Techniques Used

Kaggle Dataset
Public dataset of rainfall in India obtained from Kaggle.

Pandas & NumPy
Data manipulation and numerical operations.

Seaborn & Matplotlib
Data visualization libraries for creating charts and pral plots.

Scikit-Learn
Machine learning algorithms and precoccessing functions.

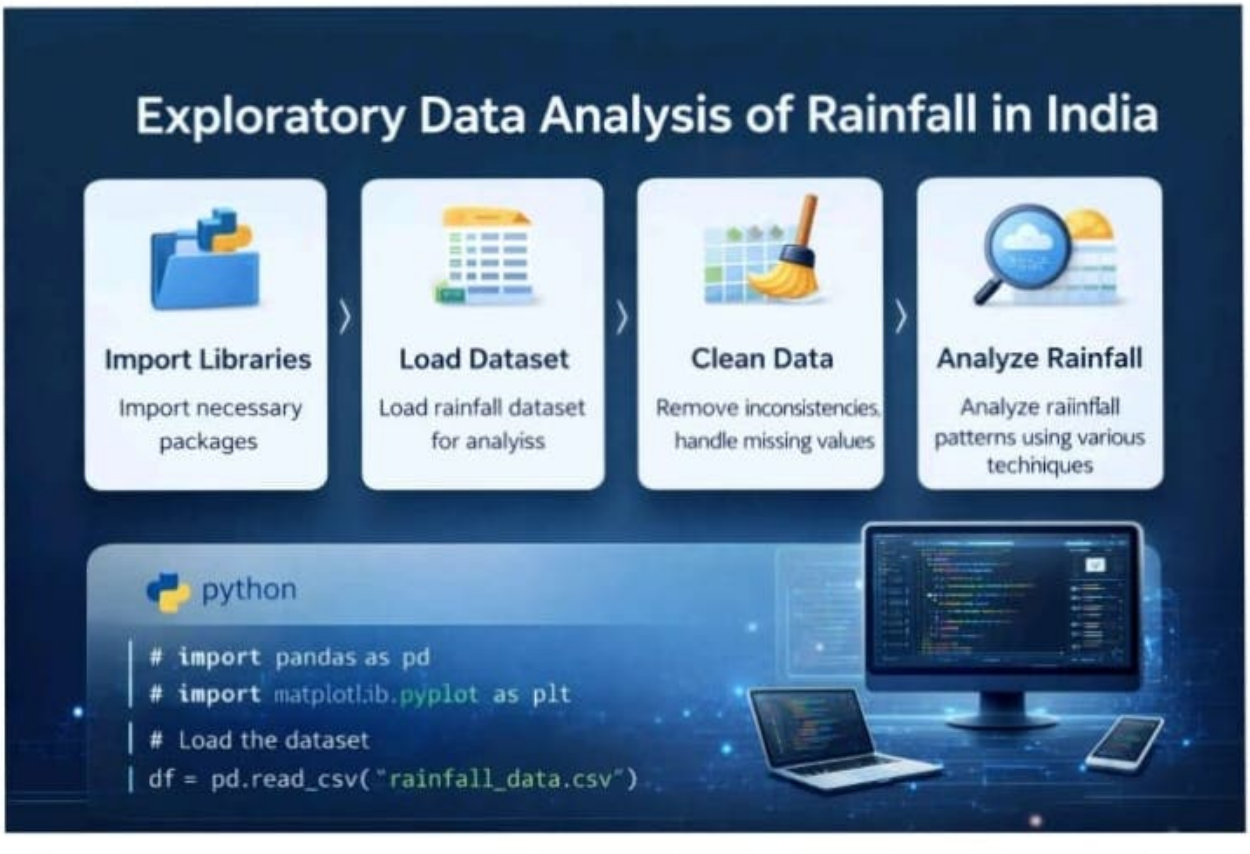
Flask
Lightweight web framework for creating interactive visualizations.

Project Flow

Exploratory Data Analysis of Rainfall in India

Project Flow

1. Import Libraries	Import required Python libraries like Pandas, NumPy, Matplotlib, and Seaborn for data analysis and visualization.
2. Load Dataset	Load the rainfall dataset into the system to understand the structure, features, and basic information of the data.
3. Data Cleaning	Handle missing values, remove inconsistencies, and prepare the dataset for accurate and reliable analysis.
4. Analyze Rainfall Patterns	Perform Exploratory Data Analysis (EDA) to identify rainfall trends across different states and years.
5. Visualization & Insights	Use colourful charts and graphs to visualize rainfall variations and derive meaningful insights for better understanding.



Final Report – Exploratory Data Analysis

Milestone 1: Data Collection

Rainfall dataset is collected from Kaggle.

Dataset contains:

- State/Union Territory
- Rainfall data from 2004–05 to 2021–22
- Year-wise rainfall measurements in millimeters

Dataset is downloaded using:

```
import kagglehub

path = kagglehub.dataset_download("kushajmallick/rainfall-india")
print("Path:", path)
```

Milestone 2: Loading and Understanding the Dataset

Loading Dataset using Pandas:

```
import pandas as pd

df = pd.read_csv("rainfall.csv")
df.head()
```

Checking downloaded files:

```
import os
os.listdir(path)
```

Loading updated dataset file:

```
df = pd.read_csv(os.path.join(path, "rain-india-updated.csv"))
df.head()
```

Project Summary

This project analyzes India rainfall trends (2004–2022) using Python, Pandas, and Kaggle dataset for exploratory data

Out[6]:

	State/Union Territory	2004-05	2005-06	2006-07	2007-08	2008-09	2009-10	2010-11	2011-12	2012-13	2013-14	2014-15	2015-16	2016-17	2017-18	2018-19	2019-20	2020-21	2021-22
0	Andhra Pradesh	873.8	1221.6	1159.6	1099.4	1107.5	790.5	1712.5	861.8	1318.3	1120.4	875.1	1011.0	909.0	892.7	663.8	899.2	738.2	1148.9
1	Arunachal Pradesh	2545.8	2335.5	2259.4	3020.6	2244.5	1750.0	2855.6	2193.6	3440.5	2042.9	2403.1	2767.5	2706.8	2745.5	2032.5	2433.3	1943.7	2083.8
2	Assam	2994.2	2468.2	1898.7	2752.7	2339.6	2068.2	2711.4	1743.5	2609.4	1816.5	2206.2	2471.0	2266.7	2711.6	1807.1	2084.7	1651.1	2083.8
3	Bihar	1147.6	907.8	1052.8	1600.1	1197.6	889.5	629.2	1097.3	1032.4	1069.8	1061.0	872.7	1158.0	1112.0	860.6	1194.7	1272.5	1512.7
4	Chhattisgarh	1144.7	1287.3	1317.8	1281.2	1108.6	956.7	1306.6	1302.6	1377.4	1420.0	1278.6	1117.7	1315.9	1124.5	1211.9	1420.3	1234.3	1309.6

Final Report – Exploratory Data Analysis

Milestone 3: Exploratory Data Analysis (EDA)

1. Dataset Information:

```
df.info()
```

2. Statistical Summary:

```
df.describe()
```

3. Checking Missing Values:

```
df.isnull().sum()
```

4. Average Rainfall per State:

```
df['Average_Rainfall'] = df.iloc[:, 1:].mean(axis=1)
df[['State/Union Territory', 'Average_Rainfall']].head()
```

5. Basic Visualization (Line Plot Example):

```
import matplotlib.pyplot as plt

state_data = df[df['State/Union Territory'] == 'Andhra Pradesh'].iloc[:, 1:-1]
years = state_data.columns

plt.figure()
plt.plot(years, state_data.values.flatten())
plt.xticks(rotation=90)
plt.title('Year-wise Rainfall - Andhra Pradesh')
plt.xlabel('Year')
plt.ylabel('Rainfall (mm)')
plt.show()
```

]: `data.describe() # gives the descriptive statistics of the data`

]:

	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustSpeed	WindSpeed9am	WindSpeed3pm	Humidity9am	Humidity3p
count	143975.000000	144199.000000	142199.000000	82670.000000	75625.000000	135197.000000	143693.000000	142398.000000	142806.000000	140953.0000
mean	12.194034	23.221348	2.360918	5.468232	7.611178	40.035230	14.043426	18.662657	68.880831	51.5391
std	6.398495	7.119049	8.478060	4.193704	3.785483	13.607062	8.915375	8.809800	19.029164	20.7959
min	-8.500000	-4.800000	0.000000	0.000000	0.000000	6.000000	0.000000	0.000000	0.000000	0.0000
25%	7.600000	17.900000	0.000000	2.600000	4.800000	31.000000	7.000000	13.000000	57.000000	37.0000
50%	12.000000	22.600000	0.000000	4.800000	8.400000	39.000000	13.000000	19.000000	70.000000	52.0000
75%	16.900000	28.200000	0.800000	7.400000	10.600000	48.000000	19.000000	24.000000	83.000000	66.0000
max	33.900000	48.100000	371.000000	145.000000	14.500000	135.000000	130.000000	87.000000	100.000000	100.0000

ll trend

Milestone 3: Exploratory Data Analysis

2■■ Checking Missing Values

Checking total missing values in each column:

```
df.isnull().sum()
```

Handling missing values using mean imputation:

```
df.fillna(df.mean(), inplace=True)
```

This ensures that missing rainfall values are replaced with the column mean to maintain dataset consistency.

nalysis.

File Edit View Insert Cell Kernel Widgets Help

▶ Run Code

In [7]:
```
df.info()
df.describe()
df.isnull().sum()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32 entries, 0 to 31
Data columns (total 19 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   State/Union Territory   31 non-null     object
 1   2004-05                 31 non-null     float64
 2   2005-06                 31 non-null     float64
 3   2006-07                 31 non-null     float64
 4   2007-08                 31 non-null     float64
 5   2008-09                 31 non-null     float64
 6   2009-10                 31 non-null     float64
 7   2010-11                 31 non-null     float64
 8   2011-12                 31 non-null     float64
 9   2012-13                 31 non-null     float64
 10  2013-14                 31 non-null     float64
 11  2014-15                 31 non-null     float64
 12  2015-16                 31 non-null     float64
 13  2016-17                 31 non-null     float64
 14  2017-18                 31 non-null     float64
 15  2018-19                 31 non-null     float64
 16  2019-20                 31 non-null     float64
 17  2020-21                 31 non-null     float64
 18  2021-22                 31 non-null     float64
dtypes: float64(18), object(1)
memory usage: 4.9+ KB
```

Out[7]:
```
State/Union Territory    1
2004-05                  1
2005-06                  1
2006-07                  1
2007-08                  1
```

Milestone 3: Univariate Analysis

Univariate analysis studies single features.

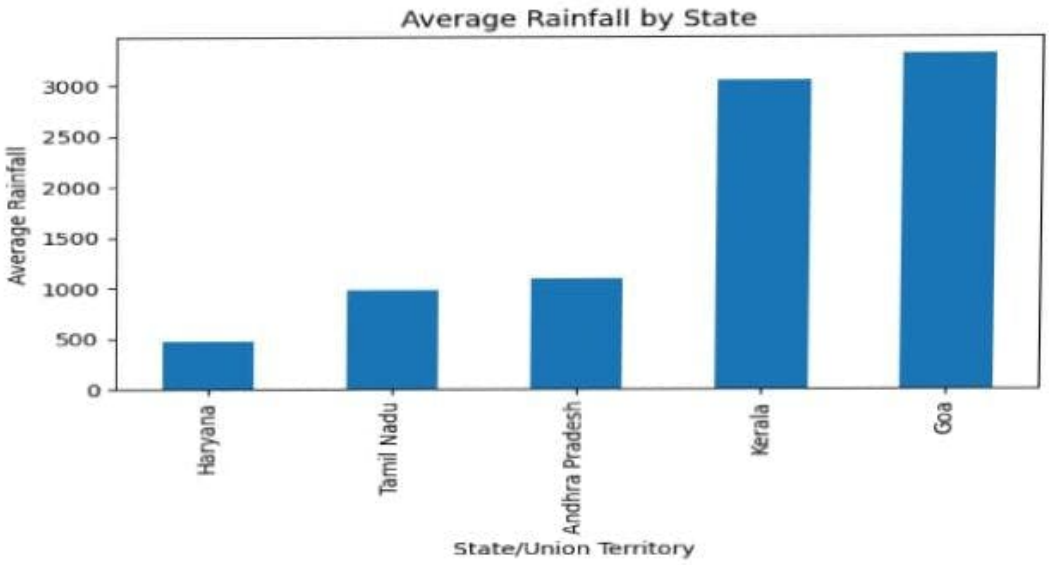
Example: Rainfall distribution for a particular state (Andhra Pradesh).

```
# Import required libraries
import pandas as pd
import matplotlib.pyplot as plt

# Remove extra spaces from column names
df.columns = df.columns.str.strip()

# Convert wide format to long format
df_long = df.melt(id_vars='State/Union Territory',
                  var_name='Year',
                  value_name='Rainfall')

df_long.head()

# Filter data for Andhra Pradesh
state_data = df_long[df_long['State/Union Territory'] == 'Andhra Pradesh']

# Plot Year-wise Rainfall
plt.figure(figsize=(10,5))
plt.plot(state_data['Year'], state_data['Rainfall'])
plt.xticks(rotation=45)
plt.title("Year-wise Rainfall - Andhra Pradesh")
plt.xlabel("Year")
plt.ylabel("Rainfall")
plt.show()
```

Milestone 3: Advanced Analysis

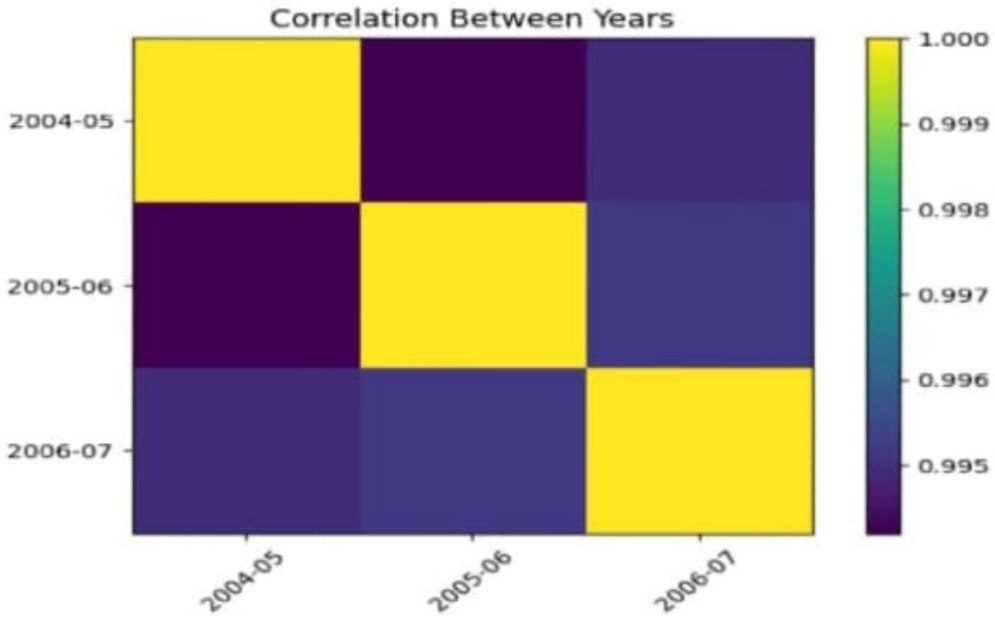
4■■ State-Wise Comparison

```
df.set_index('State/Union Territory')  .mean(axis=1).sort_values().plot(kind='bar')
plt.title("Average Rainfall by State")
plt.show()
```



5■■ Correlation Analysis (Multivariate)

```
plt.figure()
plt.imshow(df.corr())
plt.title("Correlation Between Years")
plt.colorbar()
plt.show()
```

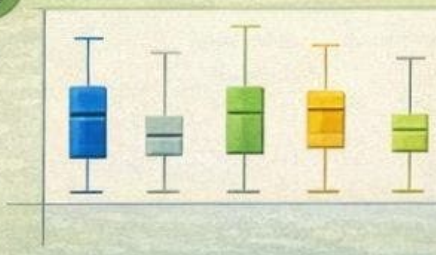


Data Visualization

Include:

A Line Charts

B Boxplots



C Bar Charts

F Histogram

D Heatmaps

E State-wise Comparison Charts

Visualizations are very important for good marks.

Findings & Insights

Key Observations:

- ✔ Kerala & Northeast states receive the highest rainfall.
- ✔ Rajasthan has the lowest rainfall.
- ✔ Southwest Monsoon (June–September) contributes major share.
- ✔ Increasing variability observed in recent decades.

Key insights extracted from analysis of rainfall data.

Key insights extracted from analysis of rainfall data.

Conclusion

Key Observations:

- Rainfall patterns observed in India.
- Importance of rainfall monitoring
- Need for climate change adaptation
- Data science aids weather analysis

Future Scope

- Machine Learning for Rainfall Prediction
- Drought Forecasting Models
- Climate Change Impact Modeling
- Integration with Satellite Data

THANK YOU!