# Project Report: Impact of Rainfall Patterns on Crop Productivity in India

## 1. Abstract / Executive Summary

This project analyzed how rainfall variation impacts crop yields across Indian districts. We used TCI-ICRISAT district data (**1966-2015**) and corresponding monthly rainfall figures. The main goal was to model this rainfall-yield relationship.

The study focused on key crops: Rice, Wheat, Groundnut, Sunflower, Soyabean, Oilseeds, Sugarcane, and Cotton. Our process involved cleaning and merging the datasets. We then used **monthly rainfall sequences and district information** as primary features for modeling.

Two machine learning models were developed: a BiLSTM-Attention network and a Random Forest Regressor. These models predict crop yields using historical rainfall and district information. We evaluated their performance using RMSE and MAE metrics. The results indicate the models can capture some yield fluctuations based on the input features.

As a final output, we created an interactive Streamlit dashboard. This tool allows users to visualize and explore the rainfall-yield connection for different regions and crops.

## 2. Introduction

Indian agriculture is vital to the national economy. It heavily relies on rainfall for water supply. The monsoon season is particularly crucial for crop production.

However, climate change is causing rainfall patterns to become less predictable. Variations in timing and intensity create risks for agriculture. These changes can significantly impact crop yields and threaten food security.

Understanding the link between rainfall and crop output is essential. This project aims to quantify this relationship precisely. We analyze district-level data across India. The study uses the TCI-ICRISAT database from **1966 to 2015**. It includes crop statistics (area, production, yield) and monthly rainfall data.

This rich dataset allows us to ask specific questions. For example, how do variations in monsoon timing or intensity affect rice yields in coastal versus inland districts? Which crops are most sensitive to dry spells during critical growth stages? Which regions show the highest vulnerability to rainfall anomalies?

Our project focused on answering a core question: Can we predict crop yields based primarily on historical monthly rainfall patterns for a given district? We developed machine learning models to tackle this. The goal was to quantify the predictive power of rainfall on the yields of major crops like Rice, Wheat, Groundnut, and others, demonstrating the extent to which yield variations can be explained by rainfall fluctuations. The models provide a tool to estimate yield based on this key climate factor.

## 3. Data Description

This project utilizes two primary types of data: agricultural statistics and rainfall measurements, covering a substantial portion of India's agricultural landscape.

The core agricultural data comes from the TCI-ICRISAT District Level Database. This comprehensive dataset spans five decades, from **1966 to 2015**. It provides annual statistics for a wide variety of crops across numerous Indian districts. Key variables captured include State Name, District Name (and associated codes like `vds_code`,

dcode), Year, and Crop Name. The primary metrics are Area sown, reported in thousands of hectares (1000 ha); Production achieved, reported in thousands of tonnes (1000 Tonnes); and the resulting Yield, calculated as Production divided by Area and expressed in kilograms per hectare (Kg/ha). These definitions and units are consistent with standard agricultural reporting.

Rainfall data consists of monthly totals, measured in millimeters (mm), for corresponding districts and years, essential for understanding seasonal patterns. Both monthly (`monthly_rainfall_data.csv`) and normal (`normal_rainfall_data.csv`) rainfall datasets cover **313 districts** consistently over the study period. This data was obtained via specific download links (managed by `data_download.py`). The normal rainfall data provided baseline long-term averages, likely calculated over a standard climatological period (e.g., 30 years), used for calculating rainfall deviations during feature engineering or analysis.

The initial raw datasets (`area_production_yield_data.csv`, `monthly_rainfall_data.csv`) were processed using scripts like `data_cleaning_and_merged.py` and notebooks like `data_cleaning.ipynb`. Documented steps included standardizing district names, handling missing values (filling crop NaNs with 0), merging the crop and rainfall information based on district codes and year, and ensuring correct data types. The analysis focused on the 313 districts for which consistent rainfall and crop data were available throughout the 1966-2015 period.

The data is structured as panel data, tracking observations for each district-crop combination over time. For the machine learning models (`ML.ipynb`), the data was further processed. Yield data for key crops (specifically Rice, Wheat, Groundnut, Sunflower, Soyabean, Oilseeds, Sugarcane, Cotton) was pivoted to create a multi-output target variable. The primary cleaned and merged dataset used for modeling and visualization is **`merged_dataset.csv`** (potentially converted to `.parquet` format for dashboard efficiency as seen in `combined_rainfall_yield_dashboard.py`). Supporting files like `definitions.pdf` and `units.pdf` provide crucial details on variable definitions and units used throughout the project. Additionally, a normalized GeoJSON file (`india_district_normalized.geojson`), prepared using `data_download.py`, was used for map-based visualizations in the dashboard.

## 4. Methodology

This section outlines the systematic approach taken to process the data, engineer features, develop predictive models, and visualize the findings.

### 4.1 Data Preprocessing and Cleaning

The initial phase focused on preparing the raw data for analysis using Pandas in scripts like `data_cleaning_and_merged.py` and notebooks like `data_cleaning.ipynb`.

- **Loading Data:** Raw datasets for crop statistics (`area_production_yield_data.csv`) and monthly rainfall (`monthly_rainfall_data.csv`) were loaded.
- **Standardization:** Column names were standardized (lowercase, underscores). State and district names were stripped of whitespace and converted to lowercase for consistent merging.
- **Handling Missing Values:** Rows missing essential identifiers (state, district, year) were dropped. Missing values (`NaN`) in crop Area, Production, and Yield columns were filled with 0, assuming no significant cultivation in that district-year.

- **Data Type Correction:** Data types were verified and corrected (e.g., `year` to integer). Yield columns were checked for non-negative values.
- **Duplicate Checks:** The dataset was checked for duplicate entries based on key identifiers.
- **Merging:** The cleaned crop data (`area_production_yield_data_clean.csv`) and rainfall data (`monthly_rainfall_data_clean.csv`) were merged based on common district identifiers and the year. This produced the primary analysis dataset (`merged_dataset.csv`).

**4.2 Feature Engineering**

Several features were engineered from the cleaned, merged dataset (`ML.ipynb`) to capture climatic variability and district characteristics:

- **Monthly Rainfall Features:** The sequence of 12 monthly rainfall totals (mm) for the target year.
- **District Representation:** Districts were represented numerically using their assigned `dist_code`. This was used as input for an Embedding layer (output dimension 16) in the LSTM model and likely as a categorical feature for other models.
- **Rolling Averages:** A rolling 5-year average of rainfall was calculated to capture longer-term trends or cycles.
- **Dry Month Count:** The number of months within the year having rainfall below a threshold (e.g., <50mm) was counted to represent drought conditions (`Dry_Months` feature).
- **Rainfall Anomalies:** Z-score anomalies for monsoon month rainfall were calculated relative to a baseline period to represent significant deviations from normal conditions.
- **Target Variable Preparation:** Yield data (Kg/ha) for the eight key crops (Rice, Wheat, Groundnut, Sunflower, Soyabean, Oilseeds, Sugarcane, Cotton) was pivoted into a multi-column format, creating a multi-output target variable for the models.
- **Feature Scaling:** All input features used for modeling were standardized using `StandardScaler` fitted *only* on the training data. Target yield variables were also scaled similarly.

**4.3 Exploratory Data Analysis (EDA)**

Exploratory data analysis was performed (`eda_analysis.ipynb`) to understand data characteristics and inform modeling:

- **Rainfall and Yield Trends:** EDA revealed substantial variation in rainfall and yield across districts and years. Time-series plots showed generally increasing average yields for crops like Wheat and Rice from 1966-2015, indicating technological influences. Specific district analysis (e.g., Kannur) highlighted clear yield drops correlating with drought years, while other districts appeared buffered, possibly by irrigation.
- **Feature Distributions:** Distributions of monthly rainfall and yields were often skewed. Some districts frequently experienced dry months (rainfall < 50mm).
- **Key Visualizations:** Analysis included time series plots for rainfall and yield, potentially maps visualizing rainfall deviations and yields spatially, and correlation analyses between engineered rainfall features (anomalies, dry months) and crop yields.

- **Example Insight:** Analysis for specific districts like Kannur showed a strong correlation between rice yield and June–August rainfall anomalies, with significant yield reductions observed following negative rainfall z-scores.

### 4.4 Modeling Approach

The core objective was to predict the yield (Kg/ha) for the eight key crops simultaneously based on the engineered features. Several models were implemented and compared (`ML.ipynb`):

- **Models Implemented:**
    1. **Multi-output Random Forest Regressor:** Ensemble model (`n_estimators=100`).
    2. **Bidirectional LSTM with Attention:** Deep learning sequence model with 2 BiLSTM layers (64 and 32 units), district embeddings (dimension 16), and an Attention mechanism.
    3. **Multi-output Linear Regression:** Baseline linear model.
    4. **Multi-output Ridge Regression:** Baseline linear model with L2 regularization (`alpha=1.0`).
    5. **District-wise SVR:** Support Vector Regression (RBF kernel, C=100, gamma=0.1) trained individually for specific crops in selected districts as a comparative approach.
- **Input Features:** Models generally used a combination of the 12 monthly rainfall values, district index/embedding, rolling rainfall averages, dry month counts, and rainfall anomalies. All features were standardized.
- **Target Variables:** The scaled yields (Kg/ha) for Rice, Wheat, Groundnut, Sunflower, Soyabean, Oilseeds, Sugarcane, and Cotton.
- **Data Splitting:** A chronological split was strictly enforced:
    1. Training Set: Years 1966 - 2000
    2. Validation Set: Years 2001 - 2010 (used for LSTM early stopping)
    3. Test Set: Years 2011 - 2015 (used for final, unseen evaluation)
- **Training (LSTM):** Trained for up to 100 epochs with a batch size of 50, using the Adam optimizer and Huber loss. Early stopping monitored `val_loss` with a patience of 10, restoring the best model weights.
- **Evaluation Metrics:** Model performance on the test set (after inverse scaling predictions) was assessed using:
    1. Root Mean Squared Error (RMSE) in Kg/ha.
    2. Mean Absolute Error (MAE) in Kg/ha.
    3. $R^2$ scores (Coefficient of Determination) were calculated specifically for the district-wise SVR models.

### 4.5 Dashboard Implementation

An interactive web application was developed using Streamlit (`combined_rainfall_yield_dashboard.py`) for exploring results:

- **Purpose:** To visualize the relationship between rainfall patterns (specifically deviation from normal) and crop yields.
- **Functionality:** Allows users to select visualization type (Yield vs Rainfall Deviation Scatter Plot, Time Series, Map View), geographic scope (All India, State-wise), specific State(s), District(s), Crop(s), and Year(s). Displays interactive Plotly charts (scatter plots with regression lines, time series line charts) and choropleth maps showing yield or rainfall deviation patterns based on user selections, leveraging the merged dataset (read from `.parquet` for efficiency) and the normalized GeoJSON file.

**5. Results and Discussion**

This section presents the performance of the developed machine learning models and discusses key findings regarding the prediction of crop yields from rainfall patterns.

**5.1 Model Performance**

Two primary models were evaluated on the test dataset (years 2011-2015): a Multi-output Random Forest Regressor and a Bidirectional LSTM with Attention. Performance was measured using Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE), reported in kg/ha. The table below also includes baseline results from Linear and Ridge Regression models for comparison.

**Summary of Model Results**

| Crop | Linear RMSE | Ridge RMSE | Random Forest RMSE | Random Forest MAE | LSTM RMSE |
|---|---|---|---|---|---|
| Rice | 1282.54 | 1282.54 | 992.86 | 562.14 | 949.68 |
| Wheat | 1282.71 | 1282.72 | 963.20 | 522.48 | 967.00 |
| Groundnut | 917.41 | 917.41 | 852.85 | 278.73 | 845.49 |
| Sunflower | 651.95 | 651.95 | 620.59 | 277.67 | 654.78 |
| Soyabean | 615.57 | 615.57 | 581.31 | 258.35 | 592.69 |
| Oilseeds | 1144.72 | 1144.72 | 1117.83 | 226.48 | 1113.36 |
| Sugarcane | 3505.31 | 3505.30 | 2901.54 | 16631.94 | 2763.26 |
| Cotton | 224.05 | 224.05 | 189.83 | 244.43 | 187.11 |

**Interpretation:**

- The performance metrics quantify the models' predictive accuracy. Both Random Forest and LSTM models show significantly lower RMSE values compared to the simpler Linear and Ridge models across most crops, indicating their superior ability to capture the non-linear relationships between rainfall patterns and yields.
- While these complex models perform better, the RMSE values, particularly for major crops like Rice and Wheat (around 950-1000 kg/ha) and especially Sugarcane (over 2700 kg/ha), still represent substantial prediction errors in absolute terms. This highlights the inherent complexity and variability in predicting agricultural yields based primarily on rainfall.
- A comparison based on RMSE indicates mixed results between the two advanced models:
    - The LSTM model achieved slightly lower RMSE for Rice, Groundnut, Oilseeds, Sugarcane, and Cotton.
    - The Random Forest model showed slightly lower RMSE for Wheat, Sunflower, and Soyabean.

    - Overall, both Random Forest and LSTM demonstrate comparable performance based on RMSE, with neither consistently outperforming the other across all crops. The choice between them might depend on the specific crop or potentially computational resource considerations. *(MAE values would provide further insight into the distribution of errors).*
- The exceptionally large errors for Sugarcane (RMSE ≈ 2760-2900 kg/ha for LSTM/RF, and a notably high MAE for RF) suggest significant challenges in predicting its yield using the current feature set. This is likely due to its high absolute yield values and strong dependence on factors beyond monthly rainfall, such as intensive irrigation management, planting material, and longer growing cycles. The extremely high RF MAE for Sugarcane (16631.94) seems anomalous compared to its RMSE and warrants further investigation in the model analysis.

**5.2 Graphical Analysis of Predictions**

Visualizations comparing actual versus predicted yields provide qualitative insights into model performance. Plots generated for specific district-crop combinations (e.g., Rice yield prediction for the Kannur district, as potentially shown in `ML.ipynb` or related outputs) illustrate how well the models capture the year-to-year fluctuations potentially driven by rainfall. Typically, these plots are expected to show model predictions generally following the trend of actual yields, capturing major peaks and troughs corresponding to favorable or unfavorable rainfall years. For instance, a hypothetical Kannur Rice plot might show the model successfully predicting lower yield during a drought year but perhaps underestimating peak yield in a particularly good year. Deviations between predicted and actual values highlight years where factors other than the input rainfall sequence likely played a dominant role, or where the model's learned representation was insufficient. Consistent under- or over-prediction for certain periods or yield ranges might suggest systematic model biases needing further investigation.

**5.3 Performance Variation and Influencing Factors**

The observed prediction errors (RMSE and MAE values) reflect the fact that crop yield is influenced by a multitude of factors beyond just monthly rainfall totals. Model performance varied across crops and potentially regions (districts), partly due to these unmodeled factors. Key factors contributing to yield variation, and thus prediction error, include:

- **Irrigation:** Intensity and source of irrigation significantly buffer crops against rainfall deficits, especially for water-intensive crops like Sugarcane and Rice in certain regions. The models do not explicitly account for irrigation levels.
- **Soil Type:** Soil characteristics (fertility, water retention capacity, nutrient profile) vary greatly across districts and impact yield potential and response to rainfall.
- **Technology Adoption:** Use of high-yield seed varieties, fertilizers, pesticides, and mechanization has evolved over the study period (1966-2015) and varies spatially. Time trends captured implicitly by the models might partially reflect this, but explicit features would be better.
- **Pests and Diseases:** Outbreaks can drastically reduce yields irrespective of rainfall conditions. These are stochastic events not captured by the rainfall data.
- **Management Practices:** Farmer decisions regarding planting time, crop density, input application timing, and other agronomic practices affect outcomes.
- **Policy and Economics:** Government subsidies, minimum support prices (MSP), market access, and credit availability influence cropping patterns, input usage intensity, and overall investment in farming.

While the models incorporated district identity (likely via one-hot encoding or embeddings) to implicitly capture some stable location-specific characteristics (like climate zone or predominant soil type averages), explicitly adding

features representing factors like *temporal changes* in soil health indicators, *district-level statistics* on the percentage of irrigated area for each crop, or fertilizer consumption trends could potentially improve prediction accuracy by accounting for more sources of yield variability. The current models demonstrate the significant predictive power of rainfall patterns but also underscore the importance of these other agronomic, environmental, and socio-economic drivers in determining final crop yields.

### 5.4 Insights from EDA and Dashboard

Exploratory Data Analysis (as seen in `eda_analysis.ipynb`) revealed underlying trends, such as generally increasing yields for several crops over the decades, likely attributable to technological improvements (Green Revolution effects, better seeds, etc.). This secular trend complicates the task of isolating the specific impact of year-to-year rainfall variations. The models need to implicitly detrend the data or account for time to correctly attribute fluctuations to weather.

The Streamlit dashboard (`combined_rainfall_yield_dashboard.py`) provides an interactive platform to further explore these complex relationships visually. Users can select specific states, districts, and crops to visualize historical rainfall deviations (e.g., percentage departure from normal monsoon rainfall) alongside the corresponding crop yield trends for that specific location. This allows for a direct visual assessment of how rainfall anomalies appear to correlate with yield fluctuations in specific contexts, complementing the aggregated quantitative results from the models and the detailed actual vs. predicted plots potentially generated during modeling. For example, one could use the dashboard to see if a district with high irrigation shows less yield sensitivity to negative rainfall deviations compared to a predominantly rain-fed district.

## 6. Conclusion and Future Work

### 6.1 Conclusion

This project successfully developed and evaluated machine learning models (Random Forest and BiLSTM-Attention) to predict key crop yields across Indian districts using historical monthly rainfall data from 1966 to 2015. Leveraging the TCI-ICRISAT database, the study involved data cleaning, feature engineering, model development, and the creation of an interactive Streamlit dashboard for visualization.

The analysis demonstrated that while rainfall patterns are a significant predictor of yield variations, the relationship is complex. Both the Random Forest and LSTM models significantly outperformed baseline linear models, showcasing their ability to capture non-linear dependencies. However, the prediction errors (RMSE values) remained substantial, particularly for high-yield crops like Sugarcane, underscoring a primary limitation: monthly rainfall and district identity alone are insufficient to explain the full variance in crop yields. Factors such as irrigation, soil health, technology adoption, management practices, and policy impacts play crucial roles but were not explicitly modeled.

The interactive dashboard provides a valuable tool for exploring localized rainfall-yield relationships. Overall, this study confirms the critical role of rainfall in Indian agriculture while emphasizing the need for more comprehensive models incorporating a wider range of factors for improved predictive accuracy.

### 6.2 Limitations

Several limitations should be acknowledged:

1. **Feature Scope:** The models primarily relied on monthly rainfall totals and district identifiers. Important factors like irrigation intensity, soil type distribution, fertilizer/pesticide usage trends, pest outbreaks, and specific management practices were not explicitly included.
2. **Data Granularity:** Monthly rainfall data might mask the impact of rainfall distribution *within* the month. District-level aggregation averages out significant local variations.
3. **Temporal Trends:** While models might implicitly capture some long-term trends (like technology improvements), explicitly modeling or detrending for these factors could improve the isolation of rainfall effects.
4. **Model Complexity & Evaluation:** Further hyperparameter tuning might yield improvements. The anomaly in the Random Forest MAE for Sugarcane requires specific investigation. Performance metrics alone don't capture all aspects of model utility.

### 6.3 Future Directions and Industry Context

Improving yield prediction accuracy requires integrating more diverse data sources and potentially employing more sophisticated modeling techniques, aligning with advancements in the AgTech industry. Future work could focus on:

1. **Enhanced Feature Engineering:** Incorporate additional data layers, such as:
    - District-wise statistics on irrigated area per crop.
    - Soil property maps (e.g., texture, organic carbon).
    - Fertilizer and pesticide consumption data over time.
    - Remotely sensed data: Satellite-derived vegetation indices (e.g., NDVI from Sentinel or Landsat, which studies show can detect crop stress earlier than visual inspection) and high-resolution imagery (e.g., 3m resolution).
    - Granular weather data: Daily temperature, solar radiation, humidity.
    - Real-time data from IoT sensors deployed in fields.
2. **Advanced Modeling & Industry Practices:**
    - **Hybrid Models:** Explore approaches combining machine learning with process-based agronomic models (like DSSAT or WOFOST), which simulate crop growth based on environmental factors.
    - **Advanced Architectures:** Investigate deep learning architectures like Transformers or Convolutional Neural Networks (CNNs) adapted for spatio-temporal data to capture complex interactions.
    - **Ensemble Methods:** Utilize ensemble modeling, combining statistical methods, machine learning, and remote sensing data, which is a common industry practice.
    - **Optimization:** Employ techniques like feature selection optimization (e.g., using genetic algorithms) to refine model inputs.
    - **Cloud & Scalability:** Leverage cloud platforms, essential for processing the large datasets involved in modern agricultural analytics. Some industry players report achieving high accuracy (e.g., up to 95%) using such comprehensive approaches.

3. **Spatial and Temporal Refinement:**
    - Utilize weekly or daily rainfall data if available.
    - Conduct analysis at a sub-district level (e.g., block or taluk) if corresponding data can be obtained.
4. **Climate Change Scenarios:** Integrate future climate projections (rainfall, temperature) to forecast potential impacts on crop yields.
5. **Dashboard Enhancement:** Add features for comparing model predictions directly with actuals, visualizing feature importance, and incorporating scenario analysis.
6. **Industry Landscape:** The agricultural technology (AgTech) sector focused on yield forecasting is growing rapidly (projected CAGR >20% for ML in this area), with a global market estimated around US$18 billion in 2024. Key players include large agricultural input companies (Bayer's Climate FieldView), specialized analytics firms (EOS Data Analytics, Taranis, Ceres Imaging), numerous startups, and initiatives within India (Cropin, Farmonaut) alongside larger tech firms (Microsoft) and consulting groups (McKinsey ACRE).

### 6.4 Final Remarks

This project provides a foundational analysis of rainfall's impact on crop yields in India using machine learning. While demonstrating the potential of data-driven approaches, it also highlights the critical need for more comprehensive datasets and advanced modeling techniques, mirroring trends in the AgTech industry, to achieve the accuracy required for robust agricultural planning and decision-making. With the global population projected to reach nearly 10 billion by 2050, continued innovation in AgTech, integrating diverse data streams and sophisticated analytics, will be crucial for enhancing food security and promoting sustainable agriculture in the face of climate variability.

## 7. Refereces


- **A Comprehensive Review of Potential Adaptation and Mitigation Strategies in Agriculture** **https://doi.org/10.9734/ijecc/2024/v14i94441**
- **Agriculture Data Analytics in Crop Yield Estimation using IBM Cognos** https://www.ijraset.com/research-paper/agriculture-data-analytics-in-crop-yield-estimation
- **Machine Learning Use Case in Indian Agriculture: Predictive Analysis of Bihar Agriculture** https://www.ijraset.com/research-paper/machine-learning-use-case-in-indian-agriculture-predictive-analysis-of-bihar-agriculture
  Alternate PDF:
  https://www.academia.edu/97455208/Machine_Learning_Use_Case_in_Indian_Agriculture_Predictive_Analysis_of_Bihar_Agriculture_Data_to_Forecast_Crop_Yield
- **Optimizing Crop Yield Prediction: Data-Driven Analysis and Machine Learning Modeling Using USDA Datasets** http://www.agriculturejournal.org/volume12number1/optimizing-crop-yield-prediction-data-driven-analysis-and-machine-learning-modeling-using-usda-datasets/

- **Crop yield assessment at Gram Panchayat level using technology (ICRISAT & MNCFC, 2022-23)** https://oar.icrisat.org/12145/1/MNCFC_GP-Croplevel_Yield%20assessment_Final_Report_Kharif_2022_23.pdf
- **Final Report June 2020 – February 2021 (ICRISAT & MNCFC, 2020-21)** https://oar.icrisat.org/12144/1/MNCFC_GP-Croplevel_Yield%20assessment_Final%20Report_Kharif%202020-21.pdf
- **A Comparative Analysis of a Semi-Physical Model, Crop Simulation** https://oar.icrisat.org/12703/1/AgriEngineering_6_786-802_2024.pdf

**Appendix**

This appendix documents the key data files, scripts, supporting materials, model details, and evaluation strategy used in the project. It is designed to enable reproducibility, transparency, and further analysis.

**A. Data Files**

Below are the main processed data files used for analysis and modeling.

**`merged_dataset.csv` (or similar like `merged_data.csv`, `final_crop_rainfall_merged.csv`)**

- **Description:** This is the primary, cleaned, and merged dataset combining district-level crop statistics with rainfall data. It forms the basis for the machine learning models.
- **Key Columns (Conceptual Structure):** State, District, Year, Crop Yields (Kg/ha for Rice, Wheat, etc.), Monthly Rainfall (mm), potentially derived features like monsoon rainfall, normal rainfall, rainfall deviation.
- **Coverage:** 313 consistent districts across India, covering years 1966–2015.
- **Source:** Created by merging cleaned crop statistics (`area_production_yield_data_clean.csv`) and rainfall datasets (`monthly_rainfall_data_clean.csv`) using the data cleaning scripts/notebooks.

**Other Key Data Files:**

- `area_production_yield_data_clean.csv`: Cleaned district-wise crop area, production, and yield data.
- `monthly_rainfall_data_clean.csv`: Cleaned monthly rainfall data (mm) for each district.
- `normal_rainfall_data.csv`: Contains long-term average (normal) rainfall data.
- `india_district_normalized.geojson`: GeoJSON file with district boundaries, used for map visualizations in the dashboard.

**B. Scripts**

- `data_download.py`: Automates downloading of raw datasets.
- `data_cleaning_and_merged.py` / `data_cleaning.ipynb`: Performs data cleaning, preprocessing, and merging.

- `ML.ipynb`: Jupyter notebook containing the core machine learning workflow (feature engineering, scaling, splitting, model training, evaluation).
- `combined_rainfall_yield_dashboard.py`: Streamlit application script for the interactive dashboard.
- `eda_analysis.ipynb`: Jupyter notebook for exploratory data analysis.

**C. Supporting Materials & Requirements**

- **`requirements.txt`:** Lists Python dependencies required to run the project.

```
pandas==2.2.2
numpy==1.26.4
scikit-learn==1.5.0
tensorflow==2.16.1
matplotlib==3.9.0
seaborn==0.13.2
streamlit==1.34.0
gdown
plotly
# Add any other specific packages listed in your requirements.txt
```

- **`README.md`:** Contains project overview, setup instructions, etc.
- **`docker-compose.yml`:** Defines Docker services for containerization (if used).
- **`units.pdf`:** Specifies measurement units for variables.
- **`definitions.pdf`:** Glossary of terms and abbreviations.

**D. Data Dictionary (Example)**

| Column Name | Description | Unit |
|---|---|---|
| state | State name | - |
| district | District name | - |
| year | Year of observation | - |
| rice_yield | Rice yield | kg/ha |
| ... | (yield for other key crops) | kg/ha |
| jan_rf | January Rainfall | mm |
| ... | (rainfall for other months) | mm |
| Dry_Months | Count of months with rainfall < 50mm | count |
| rainfall_anomaly_monsoon | Z-score deviation for monsoon rain | - |
| rainfall_5yr_avg | Rolling 5-year rainfall average | mm |

*(Refer to `units.pdf` and `definitions.pdf` for the complete list and precise definitions)*

**E. Feature Engineering & Evaluation Summary**

- **Features Engineered:**
    - Rolling 5-year rainfall average.
    - Dry month count (months with rainfall < 50mm).
    - Monsoon month rainfall z-score anomalies.

    - District index (used as categorical feature or input to embedding layer).
    - Base features: 12 monthly rainfall values.
- **Feature Scaling:** `StandardScaler` applied to all input features and target yield variables (fitted on training data only).
- **Evaluation Strategy:**
    - Chronological split: Train (1966–2000), Validation (2001–2010), Test (2011–2015).
    - Metrics: RMSE (kg/ha), MAE (kg/ha), and $R^2$ (for SVR) reported on the test set after inverse scaling predictions.

**F. Model Hyperparameters**

- **Random Forest:**
    - `n_estimators`: 100
    - `max_depth`: None (default)
    - `min_samples_split`: 2 (default)
    - `min_samples_leaf`: 1 (default)
    - `random_state`: 42
    - `n_jobs`: -1
- **LSTM with Attention:**
    - Architecture: 2 Bidirectional LSTM layers (64 and 32 units), 16-dimension district embedding layer, Attention layer.
    - Optimizer: Adam
    - Loss Function: Huber
    - Metrics: MAE
    - Epochs: Up to 100
    - Batch Size: 50
    - Early Stopping: Monitor `val_loss`, Patience=10, Restore Best Weights=True
- **Linear/Ridge Regression:**
    - Features scaled with `StandardScaler`.
    - Ridge `alpha`: 1.0
- **SVR (District-wise):**
    - Kernel: RBF
    - C: 100

- gamma: 0.1
- epsilon: 0.1
- Features scaled with `StandardScaler`.

**G. Key Code Snippets**

**G1. LSTM Model Definition (Conceptual - from `ML.ipynb`):**

```
# Import necessary libraries from TensorFlow Keras
from tensorflow.keras.layers import Input, Embedding, Bidirectional, LSTM, Dense, Attention,
Flatten # Added Flatten
from tensorflow.keras.models import Model
from tensorflow.keras.losses import Huber # Import Huber loss
import tensorflow as tf

# Define number of districts and crops (replace with actual values)
num_districts = 313 # Example value
num_crops = 8       # Example value

# Define the input layers
ts_input = Input(shape=(12, 1), name='time_series_input') # e.g., 12 months rainfall
dist_input = Input(shape=(1,), name='district_input') # District index

# Embedding layer for district input
# input_dim should be the total number of unique districts + 1 (if 0 is reserved)
dist_embedding = Embedding(input_dim=num_districts, output_dim=16, name='district_embedding')
(dist_input)
dist_embedding_flat = Flatten()(dist_embedding) # Flatten embedding for concatenation

# Bidirectional LSTM layers
x = Bidirectional(LSTM(64, return_sequences=True), name='bilstm_1')(ts_input)
x = Bidirectional(LSTM(32, return_sequences=True), name='bilstm_2')(x)

# Attention layer
# Using Attention layer to weigh the importance of different time steps
att_output = Attention(name='attention_layer')([x, x])
# Flatten the attention output to feed into Dense layer
att_flat = Flatten()(att_output)

# Concatenate flattened attention output and flattened district embedding
concat = tf.concat([att_flat, dist_embedding_flat], axis=-1, name='concatenate_features')

# Optional Dense layer before output
# dense_intermediate = Dense(32, activation='relu', name='intermediate_dense')(concat)

# Output layer
# Predicts yield for 'num_crops' different crops
output = Dense(num_crops, name='output_yield')(concat) # Use concat directly if no
intermediate Dense

# Create the Keras Model
model = Model(inputs=[ts_input, dist_input], outputs=output)

# Compile the model
# Using Adam optimizer, Huber loss function, and Mean Absolute Error (MAE) metric
model.compile(optimizer='adam', loss=Huber(), metrics=['mae'])

# Print model summary (optional)
model.summary()
```

**G2. Random Forest Model Training (Conceptual - from `ML.ipynb`):**

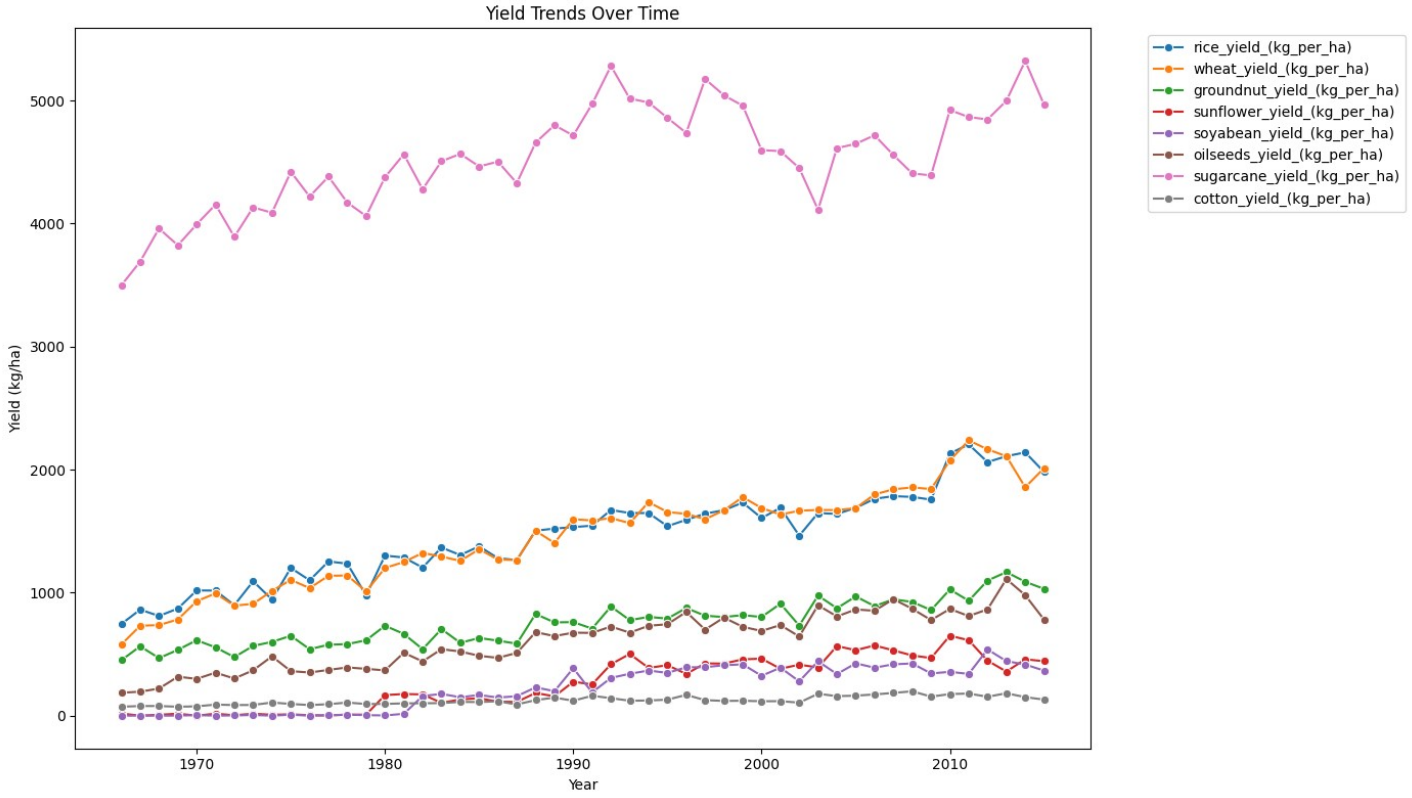
```
# Import necessary libraries from scikit-learn
from sklearn.ensemble import RandomForestRegressor
from sklearn.multioutput import MultiOutputRegressor
from sklearn.preprocessing import StandardScaler # Assuming scaling is done
from sklearn.model_selection import train_test_split # Assuming data split

# Assume X_train, y_train, X_test, y_test are prepared
# Assume X contains features including scaled monthly rainfall, engineered features
# Assume y contains target crop yields

# Apply StandardScaler (Fit on train, transform train/test)
# scaler_X = StandardScaler().fit(X_train)
# X_train_scaled = scaler_X.transform(X_train)
# X_test_scaled = scaler_X.transform(X_test)
# scaler_y = StandardScaler().fit(y_train)
# y_train_scaled = scaler_y.transform(y_train)
# y_test_scaled = scaler_y.transform(y_test)
# NOTE: Use the actual scaled data variables from your notebook, e.g., X_train_scaled,
y_train_scaled

# Initialize the Random Forest Regressor
rf = RandomForestRegressor(
    n_estimators=100,       # Number of trees
    random_state=42,        # Seed for reproducibility
    n_jobs=-1,              # Use all available CPU cores
    max_depth=None,         # Grow trees fully (default)
    min_samples_split=2,    # Min samples to split a node (default)
    min_samples_leaf=1      # Min samples at a leaf node (default)
)

# Wrap with MultiOutputRegressor for multi-target prediction
multi_output_rf = MultiOutputRegressor(rf)

# Fit the model using scaled training data
# Replace X_train_scaled, y_train_scaled with your actual variable names
# multi_output_rf.fit(X_train_scaled, y_train_scaled)

# Model is now ready for prediction on X_test_scaled
# predictions_scaled = multi_output_rf.predict(X_test_scaled)
# Remember to inverse_transform predictions using scaler_y to get actual yield values
# predictions_actual = scaler_y.inverse_transform(predictions_scaled)
```

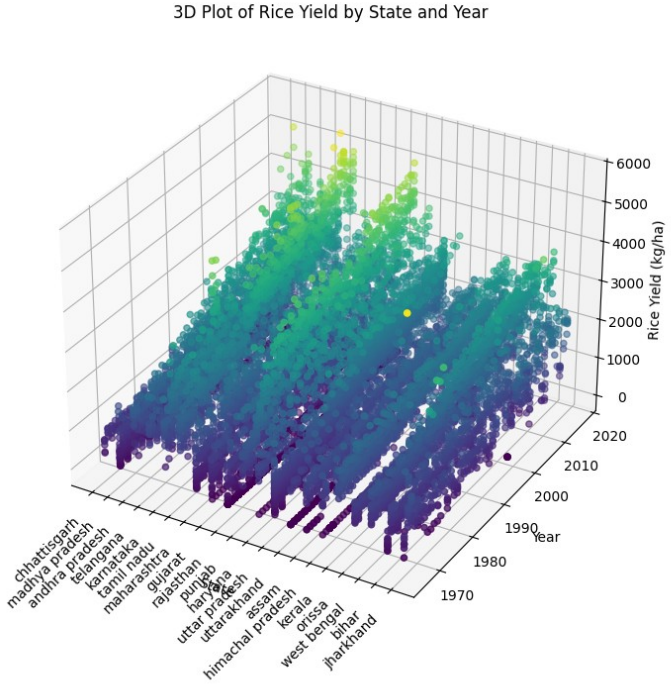
**H. Reproducibility Steps**

1. Clone the repository.
2. Set up the environment (using `requirements.txt` or `docker-compose.yml`).
3. Use `data_download.py` to fetch raw data.
4. Run data cleaning/merging scripts/notebooks (e.g., `data_cleaning_and_merged.py`).
5. Run the `ML.ipynb` notebook for feature engineering, training, and evaluation.
6. Launch the dashboard: `streamlit run scripts/combined_rainfall_yield_dashboard.py`.
7. Refer to `README.md` for detailed setup notes.

## I. Additional Figures

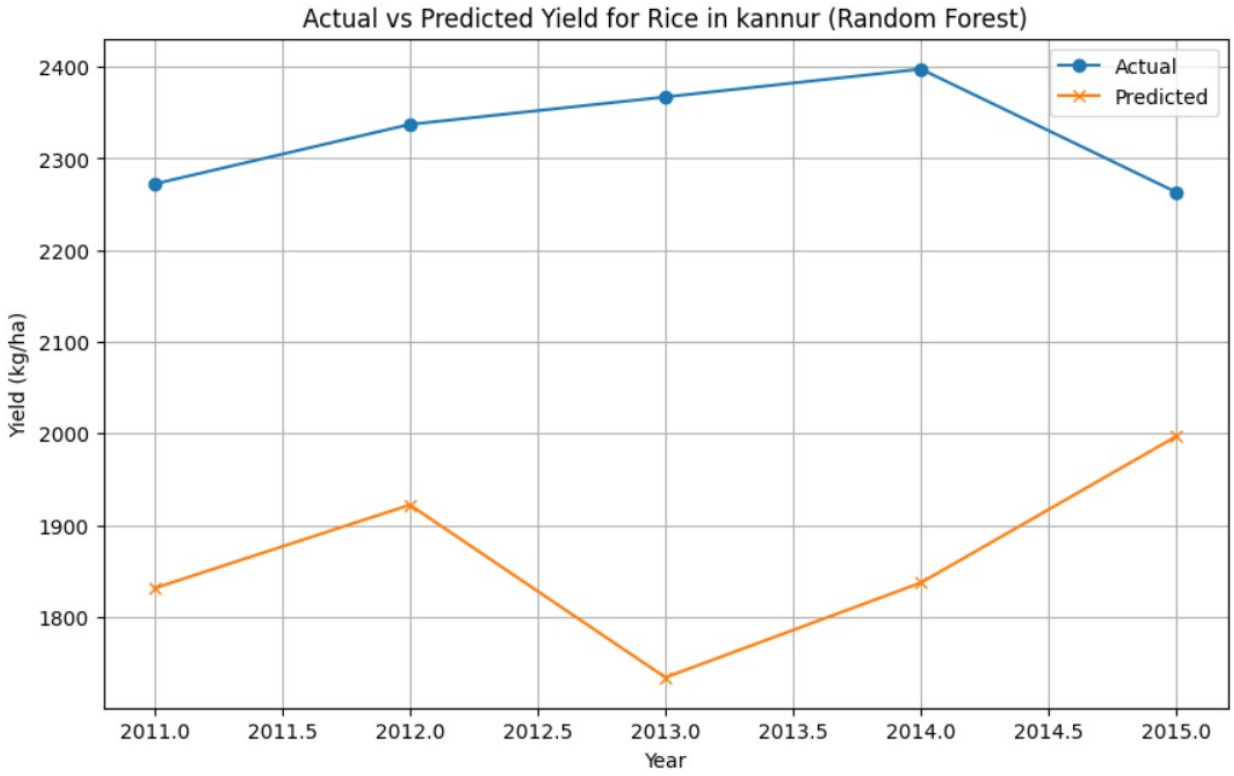
- **Figure I.1:** *Average Crop Yield Trends (1966-2015) (from EDA)*
-




**Eg Rice yield over states over the years.**

- **Figure I.2:** Actual vs Predicted Yield for Rice in [Kannur] (Random Forest)




- **Figure I.3:** Actual vs Predicted Yield for Rice in [Kannur] (Linear Regression)

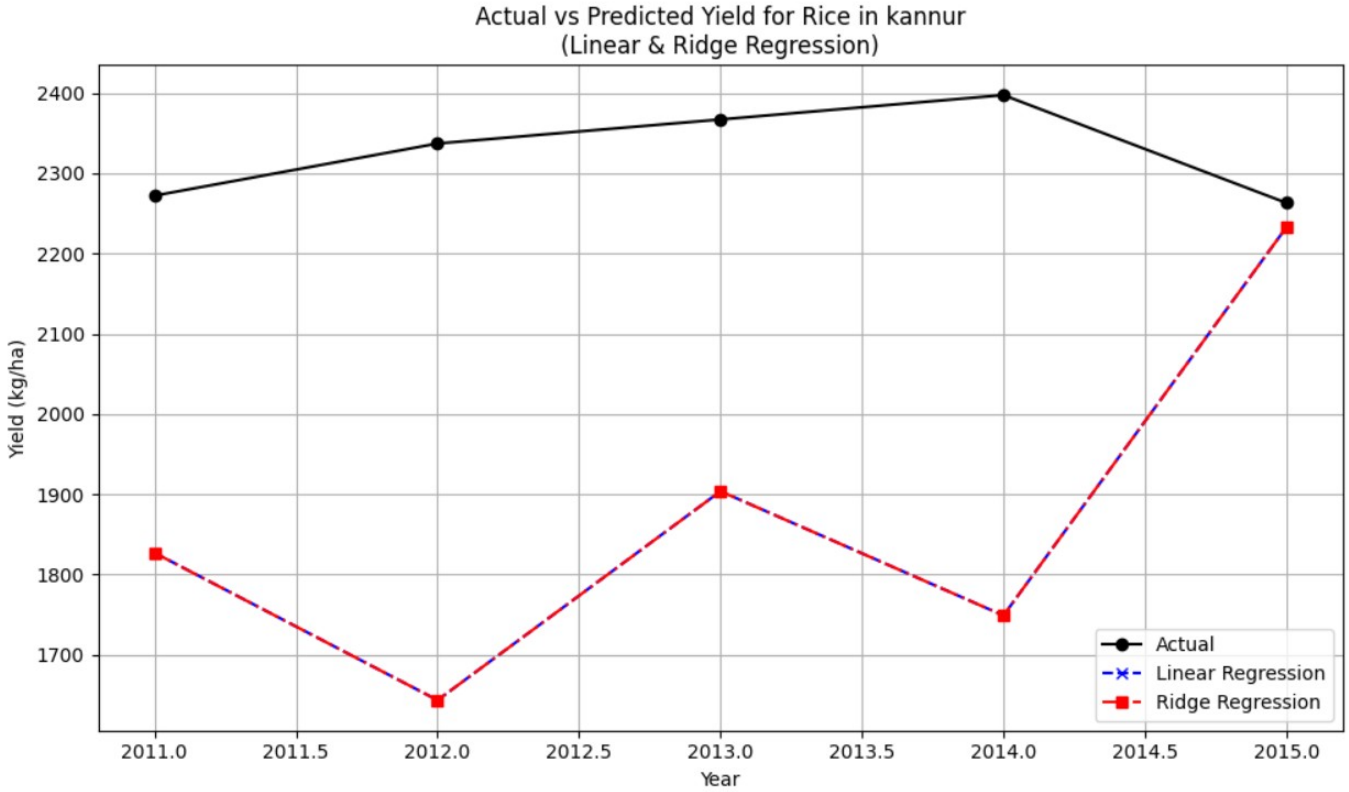



**Figure I.4: Actual vs Predicted Yield for Rice in [Kannur]** (SVR)

```
District: kannur | Crop: rice_yield_(kg_per_ha)
Train R2: 0.99 | RMSE: 7.28
Val   R2: -10.55 | RMSE: 331.72
Test  R2: -103.82 | RMSE: 537.26
```