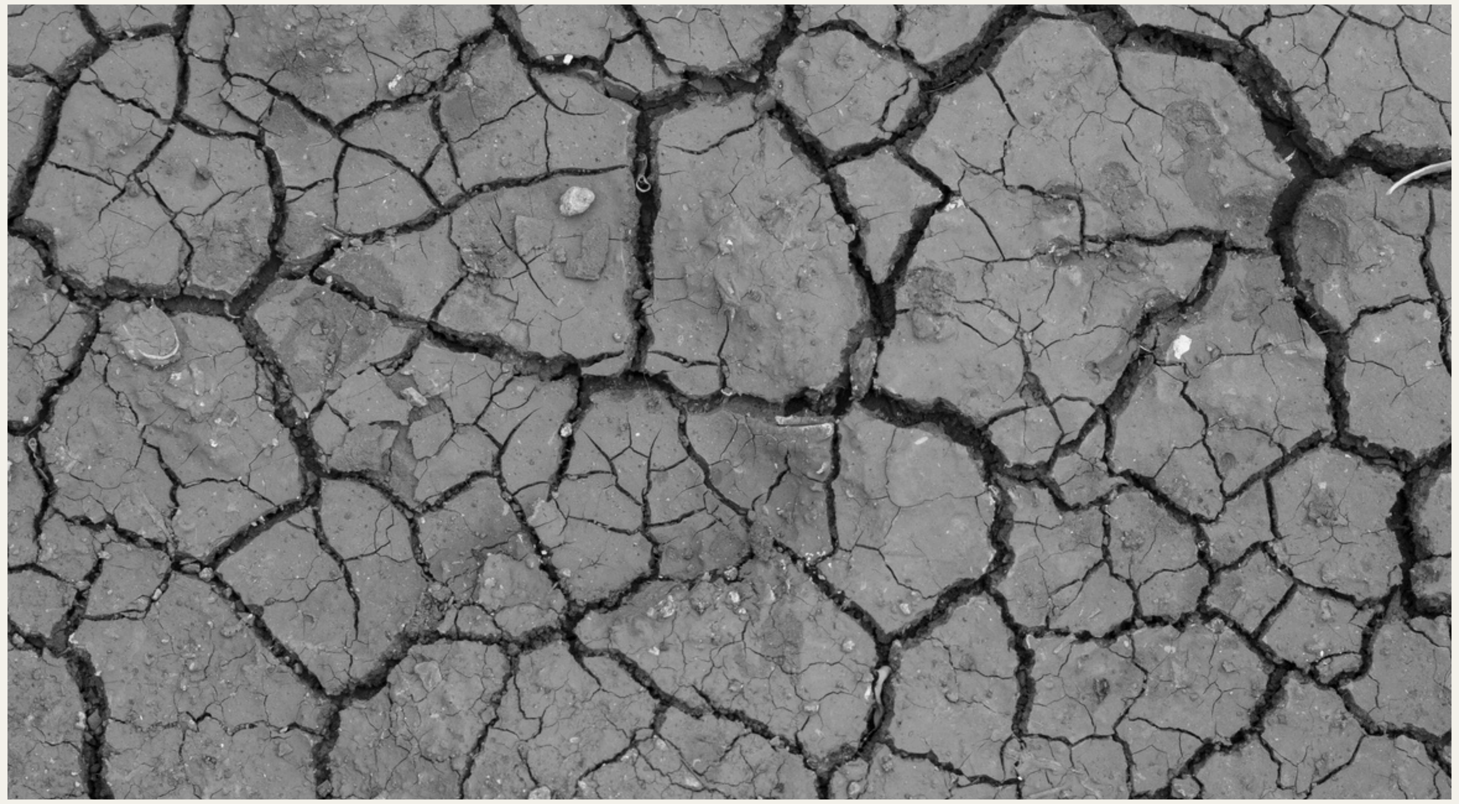
Meteorological Drought Prediction Using ML

Predicting Meteorological Drought in Jodhpur Using Sea Surface Temperatures (SSTs) and Machine Learning



Presented by **Alok - B22CI004**

Manas - B22CI022

Tranum - B22CI043

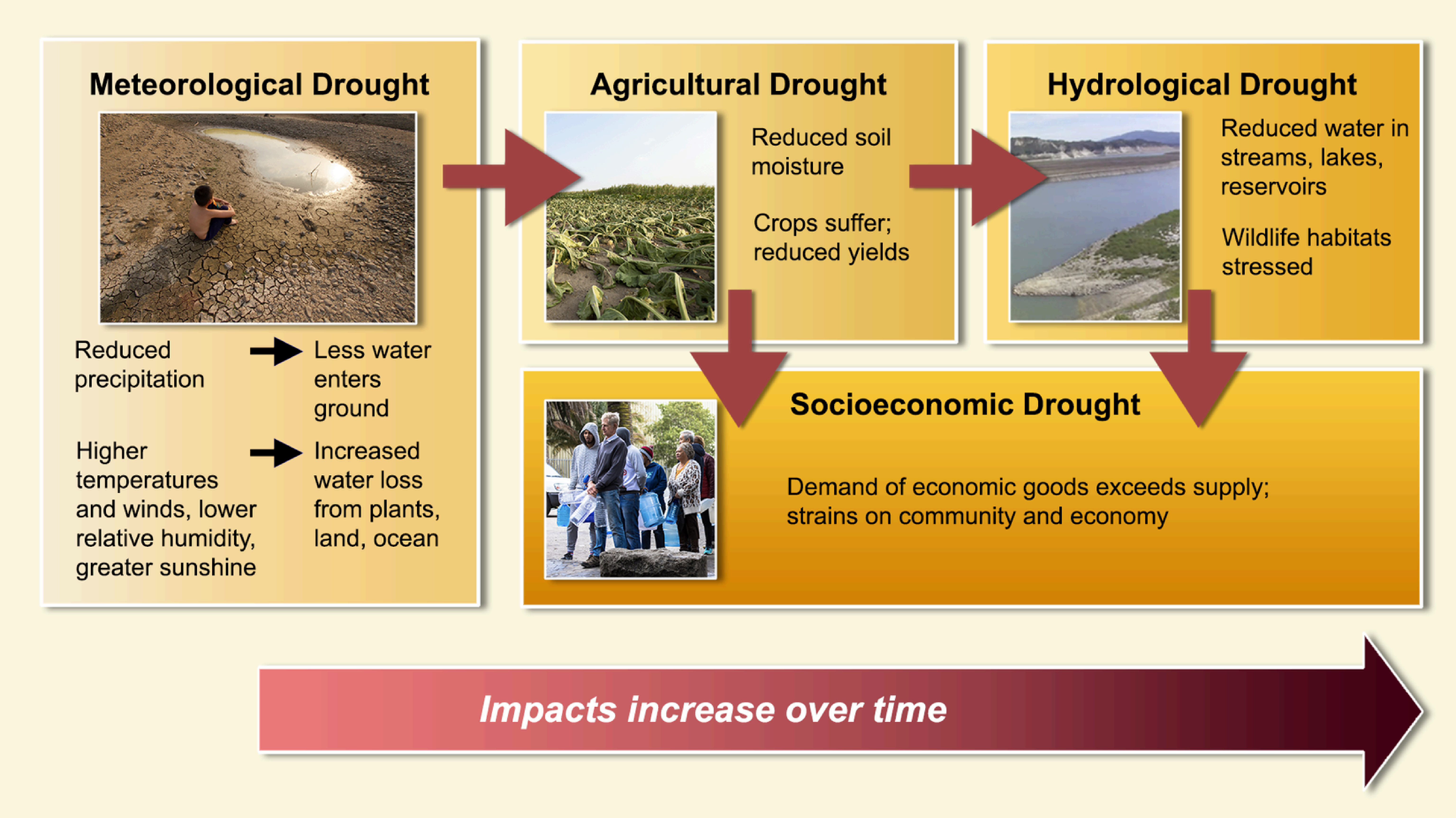
Ankush - B22CI005

Abhishek - B22CI002

Introduction to *Meteorological Drought*

- Definition: Prolonged rainfall deficiency measured by the Standardized Precipitation Index (SPI).

- Why Jodhpur?
    - Arid region with recurrent droughts impacting agriculture and water security.
    - High dependence on monsoon rainfall variability.



Credits - https://lsintspl3.wgbh.org/en-us/lesson/kend19-il-droughtbasicsil/7

Data Framework

Data Required

A. Rainfall Data

- Monthly rainfall (1901–2023) for Jodhpur district.

B. Sea Surface Temperature (SST)

- Key regions: Niño 3, Niño 3.4, Niño 4, Arabian Sea, Equatorial Indian Ocean, Bay of Bengal.
- Rationale: We study sea temperatures in these regions because changes there can affect how much rain Jodhpur gets during the monsoon, sometimes causing droughts.

Data Collection

A. Rainfall Data Sources

- India-WRIS (https://indiawris.gov.in/wris/#/timeseriesdata)
- Station data from Jodhpur’s regional offices. (CIE Department)

B. SST Data Sources

- ERDDAP (https://coastwatch.pfeg.noaa.gov/erddap/griddap/erdHadISST.html)
- Rationale:
  - Niño 3.4: Strong correlation with Indian monsoon deficits.
  - Arabian Sea: Drives pre-monsoon heating and moisture flux.
  - Bay of Bengal: Affects monsoon onset timing.

Data Processing

A. Rainfall Data

1. Area Averaging: Aggregated grid/station data to compute daily rainfall for Jodhpur.
2. Adding Dates to unlabeled data.
3. Monthly Summation: Converted daily averages to monthly totals (1901–2023).

B. SPI Calculation

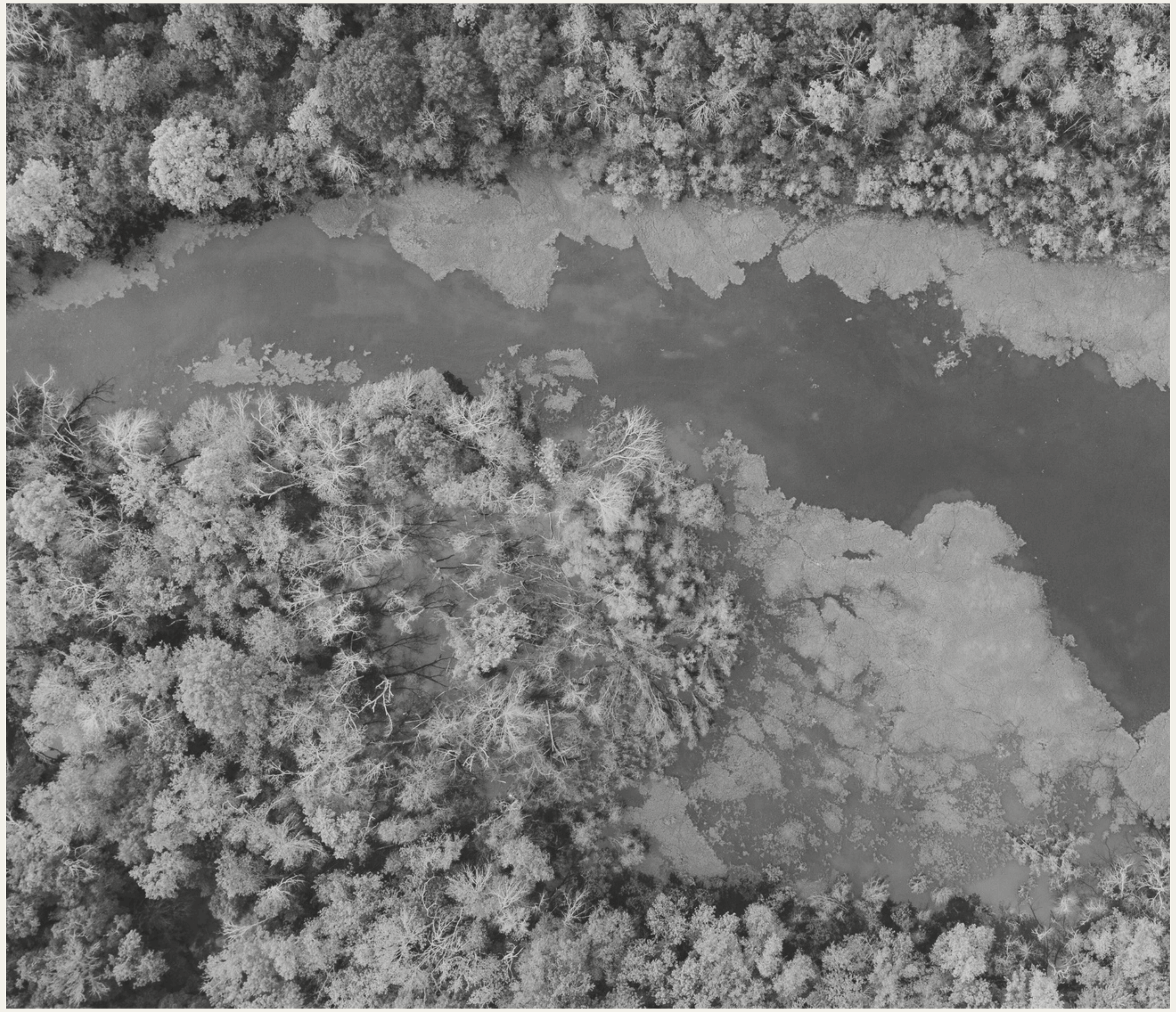
- Applied SPI Index formula to monthly rainfall.
- Output: Standardized drought index values per month.

Data Processing

C. Drought Classification

- Categories (McKee et al., 1993):
    - Moderate ($-1.0 > SPI \geqslant -1.5$)
    - Severe ($-1.5 > SPI \geqslant -2.0$)
    - Extreme ($SPI < -2.0$)

Dataset Formation Pipeline



1. Date Labeling (Python)

- Assigned timestamps (1901–2023) to all records using pandas.date_range().

2. Merging Key Data

- Combined:
    - Monthly SSTs (Niño 3.4, Arabian Sea, etc.)
    - Jodhpur's area-averaged monthly rainfall
    - Calculated SPI values

Dataset Formation Pipeline

3. Encoding Drought Classes

- Mapped text labels to numerical categories:
  - 1: No-drought
  - 2: Moderate
  - 3: Severe
  - 4: Extreme
  - 5: Exceptional

Dataset Formation Pipeline

4. Model-Specific Tweaks

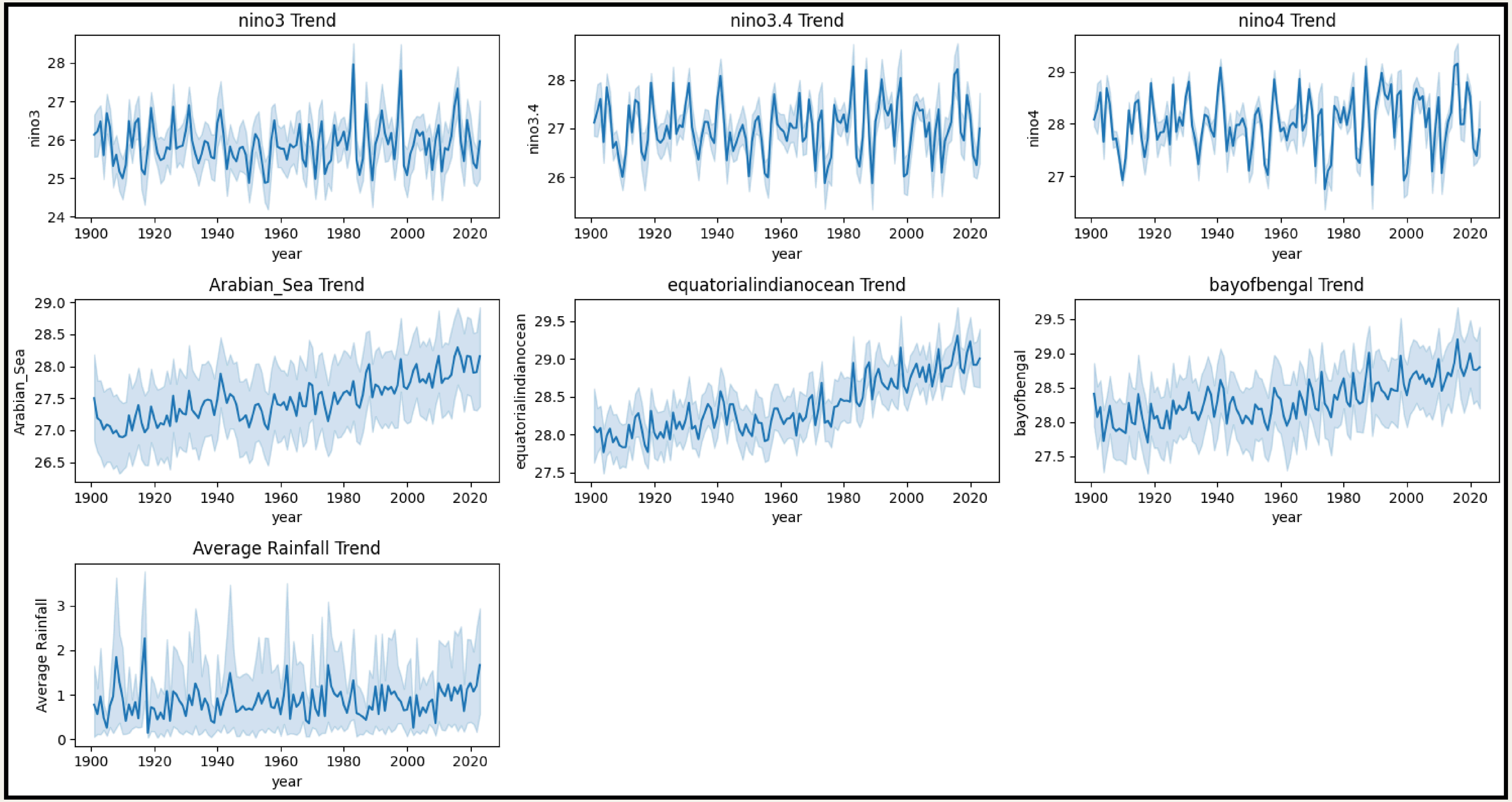
- Regression models: Used raw SPI values.
- Classification models: Fed encoded drought classes (1–5).
- Time-series models: Added lagged SST variables (3–6 months).

Tools: Python (Pandas, NumPy), Jupyter Notebook.

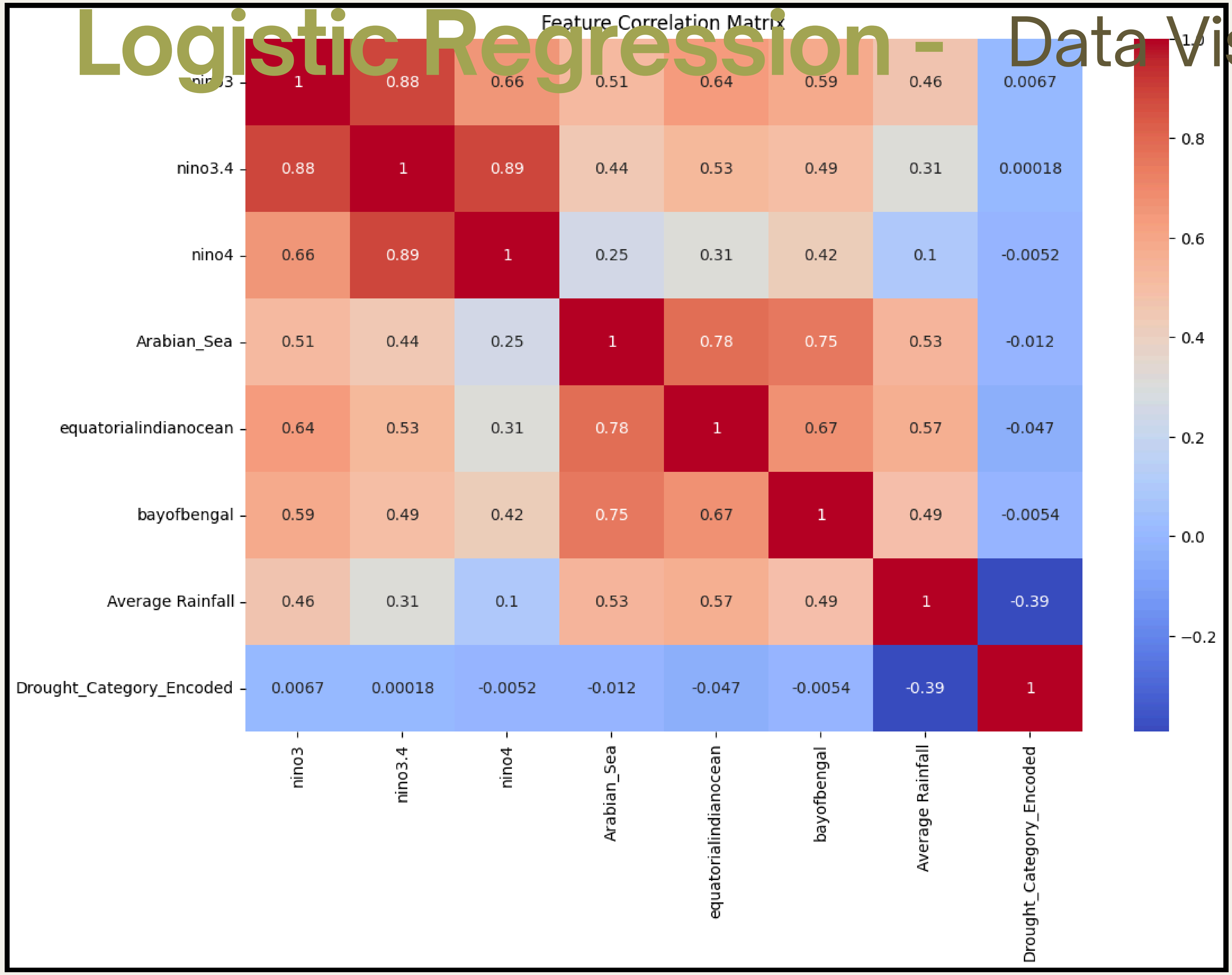
Metric	Full Form	Use	Range / Best Value	Notes
RMSE	Root Mean Squared Error	Measures average magnitude of error	[0, ∞), lower is better	Penalizes large errors more (sensitive to outliers)
MAE	Mean Absolute Error	Average absolute difference between predicted and actual values	[0, ∞), lower is better	More robust to outliers than RMSE
MSE	Mean Squared Error	Average squared error between predictions and actual values	[0, ∞), lower is better	Basis for RMSE, emphasizes larger errors
R^2 Score	Coefficient of Determination	Proportion of variance explained by the model	(-∞, 1], 1 is perfect fit	0 means model is no better than the mean; <0 means worse
F1 Score	Harmonic Mean of Precision & Recall	Measures balance between precision and recall (for classification)	[0, 1], higher is better	Only used for classification, not regression

Models

Logistic Regression - Data Visualisation



Logistic Regression - Data Visualisation



Correlation Analysis:

SSTs in Niño 3.4 and Bay of Bengal showed stronger correlations with rainfall deficits

```
Feature Importance:
nino3: 0.086
nino3.4: -0.334
nino4: 0.284
Arabian_Sea: -0.062
equatorialindianocean: -0.367
bayofbengal: 0.601
Average Rainfall: 0.007
```

Logistic Regression - Methodology Overview

1. Data Preprocessing:
   - Dataset: 1,475 monthly records (1901–2023) with rainfall, SSTs, and SPI-1 values.
   - Missing Values: No missing data detected12.
   - Statistical Summary: Analyzed distributions of rainfall, SSTs, and SPI-1.
2. Lag Feature Engineering:
   - Identified SST lag times (2–3 months) using correlation analysis.
   - Added lagged SST features to capture delayed climate impacts.
3. Train-Test Split:
   - Divided data into training and testing sets (typical 80-20 split).ance).
   - Metrics: Confusion matrix, classification report (precision/recall).

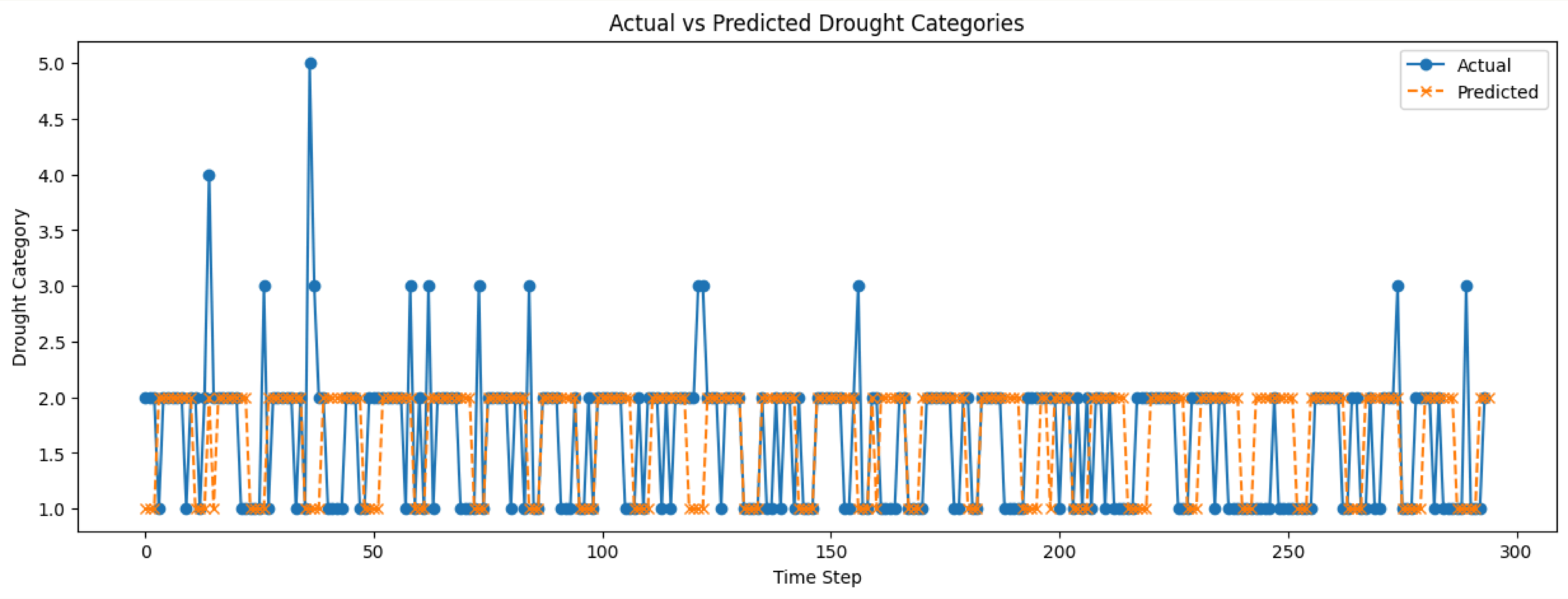
Logistic Regression - Methodology Overview

4. Model Training:
    - Algorithm: Logistic Regression with regularization.
    - Feature Scaling: Standardized SST and rainfall data.

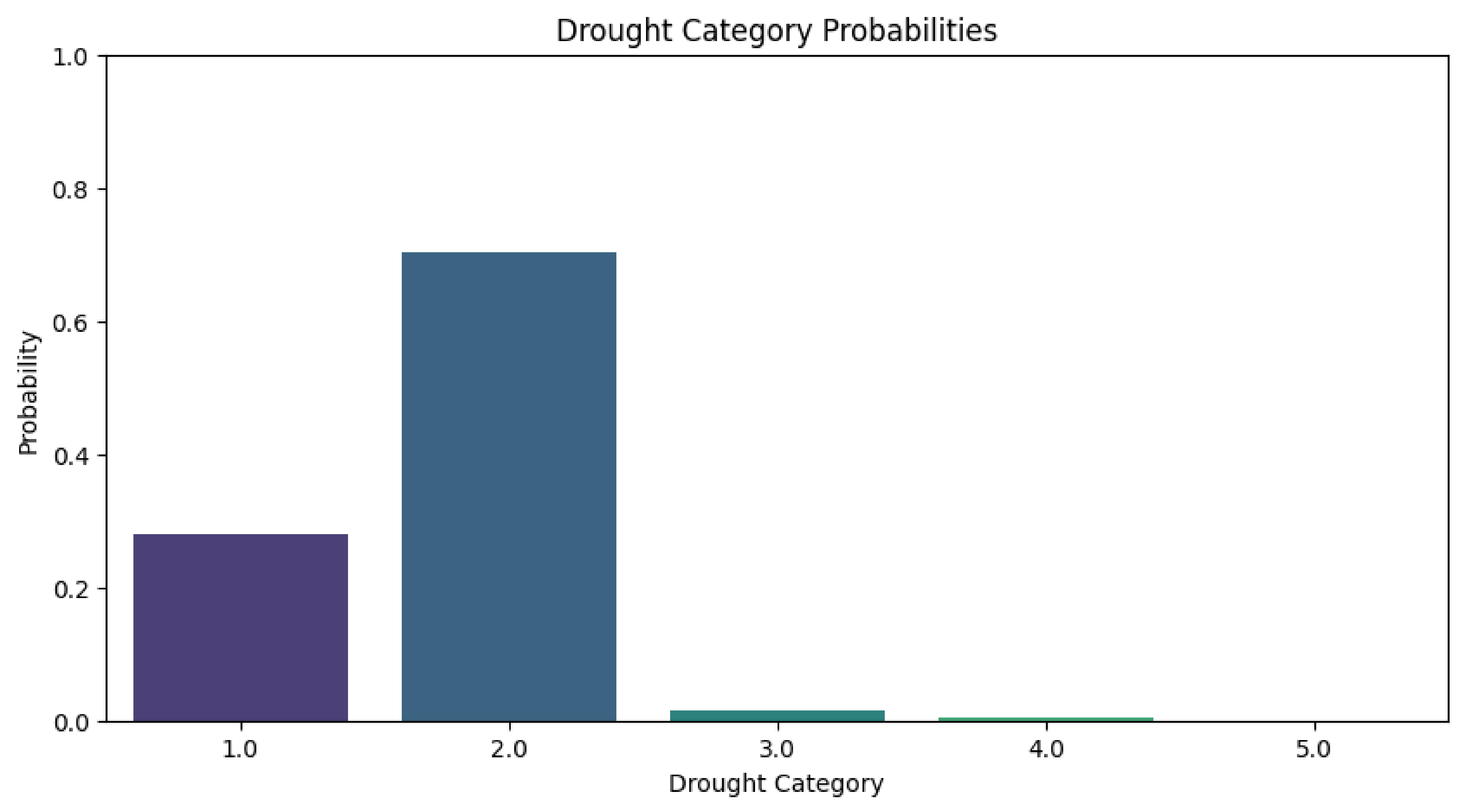
5. Evaluation:
    - Accuracy: 64% (moderate performance).
    - Metrics: Confusion matrix, classification report (precision/recall).

```
Model Performance:
Accuracy: 0.64
```

Logistic Regression - Results



Logistic Regression - Results



Colab Link

Google Colab
google.com

Logistic Regression - Results

Model Limitations:

- Linear models like logistic regression may struggle with non-linear climate interactions.
- Class imbalance likely affected accuracy (e.g., rare extreme droughts).

Conclusion

- Logistic regression provides a baseline understanding of drought drivers in Jodhpur.
- SST lags and SPI-1 are critical predictors, but model performance can be enhanced with advanced techniques.

LSTM (Long Short-Term Memory)

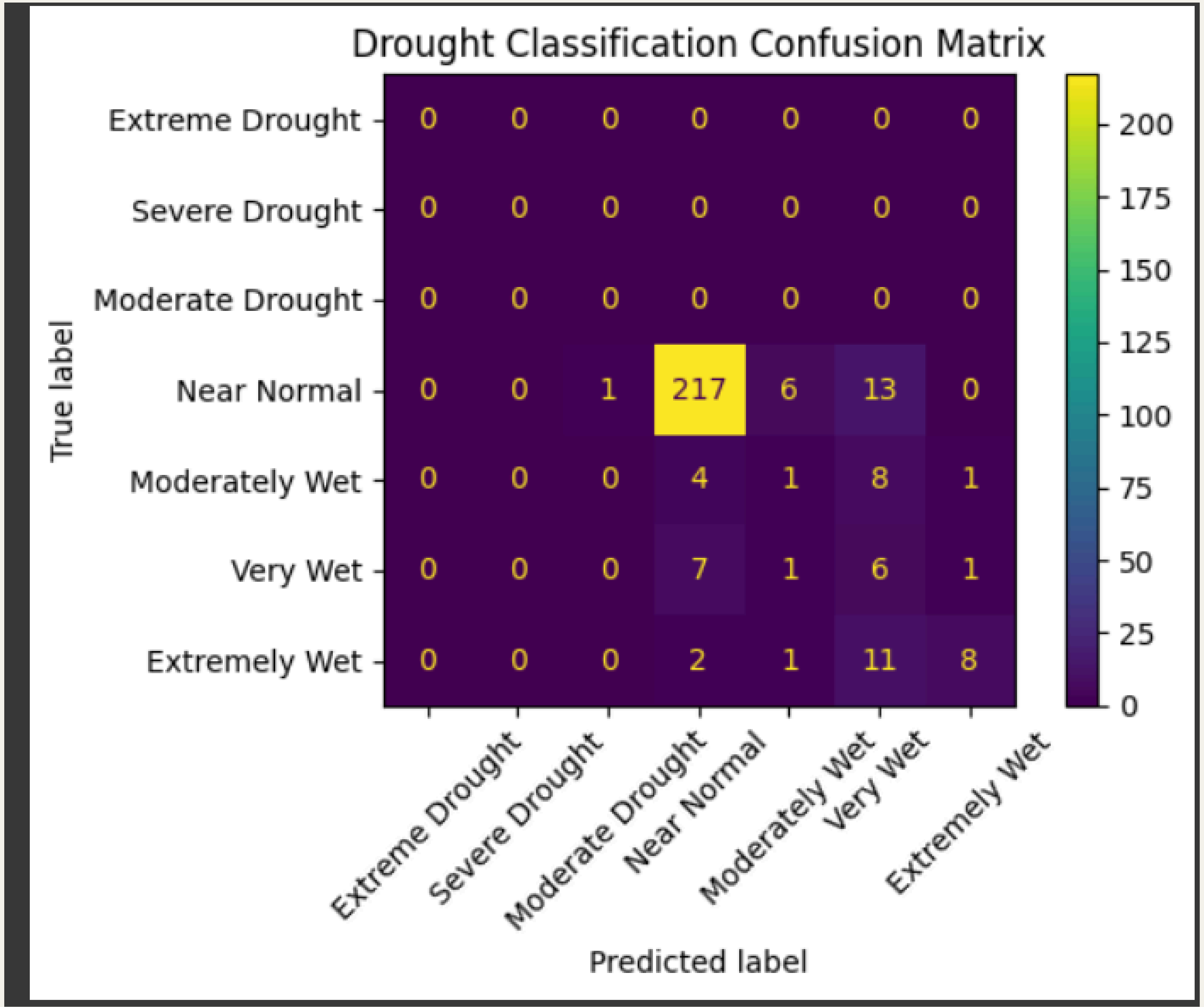
- Imported necessary libraries for data processing, visualization, and LSTM modeling.
- Loaded and filtered rainfall data from a CSV file.
- Removed missing values and extracted the 'ANNUAL' rainfall column.
- Normalized rainfall values using MinMaxScaler.
- Created sequences of 10 years of data to predict the next year.

- Built a sequential LSTM model with two LSTM layers and one Dense layer.
- Compiled the model using mean squared error loss and Adam optimizer.
- Trained the model on the dataset for 100 epochs.
- Made predictions on the same dataset (no test split in code).

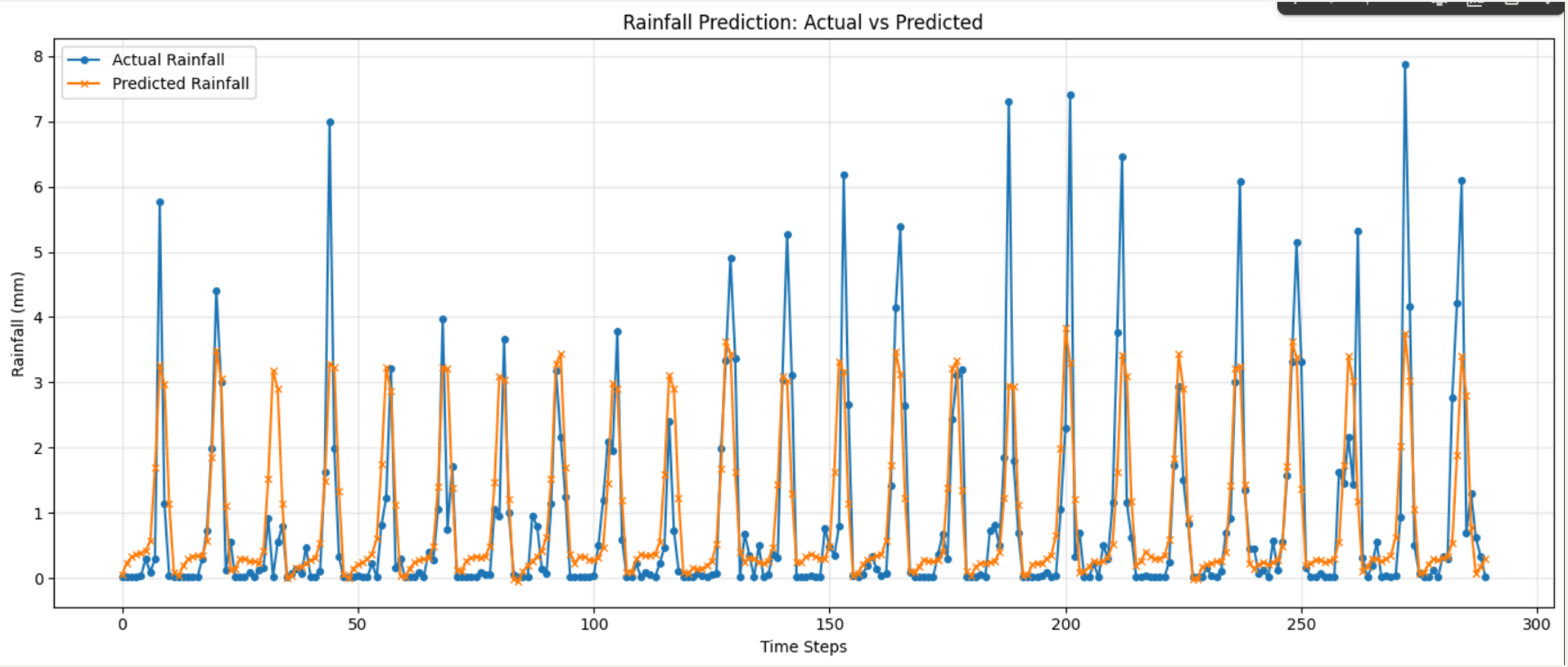
```
Generating and evaluating predictions...
Generating predictions for model 1...
10/10 ━━━━━━━━━━━━━━━━━━━━ 0s 29ms/step
Generating predictions for model 2...
10/10 ━━━━━━━━━━━━━━━━━━━━ 0s 26ms/step
Generating predictions for model 3...
10/10 ━━━━━━━━━━━━━━━━━━━━ 0s 19ms/step

Rainfall Prediction Metrics:
R² Score: 0.6007792843720101
MAE: 0.5818271036105235
RMSE: 0.982952579688294
```

Drought Classification



Predicted vs Actual Rainfall using LSTM Model



XGBoost

- **Full Name:** eXtreme Gradient Boosting
- **Model Type:** Ensemble learning based on gradient-boosted decision trees
- **Key Features:**
  - Builds trees sequentially, with each new tree learning from the errors of the previous ones
  - Uses gradient descent on a loss function to optimize predictions
  - Includes regularization (L1 and L2) to prevent overfitting
  - Highly efficient and scalable, optimized for performance
- **Strengths:**
  - Well-suited for structured/tabular data
  - Handles missing values and non-linear relationships
  - Often outperforms deep learning models on moderate-sized datasets

XGBoost

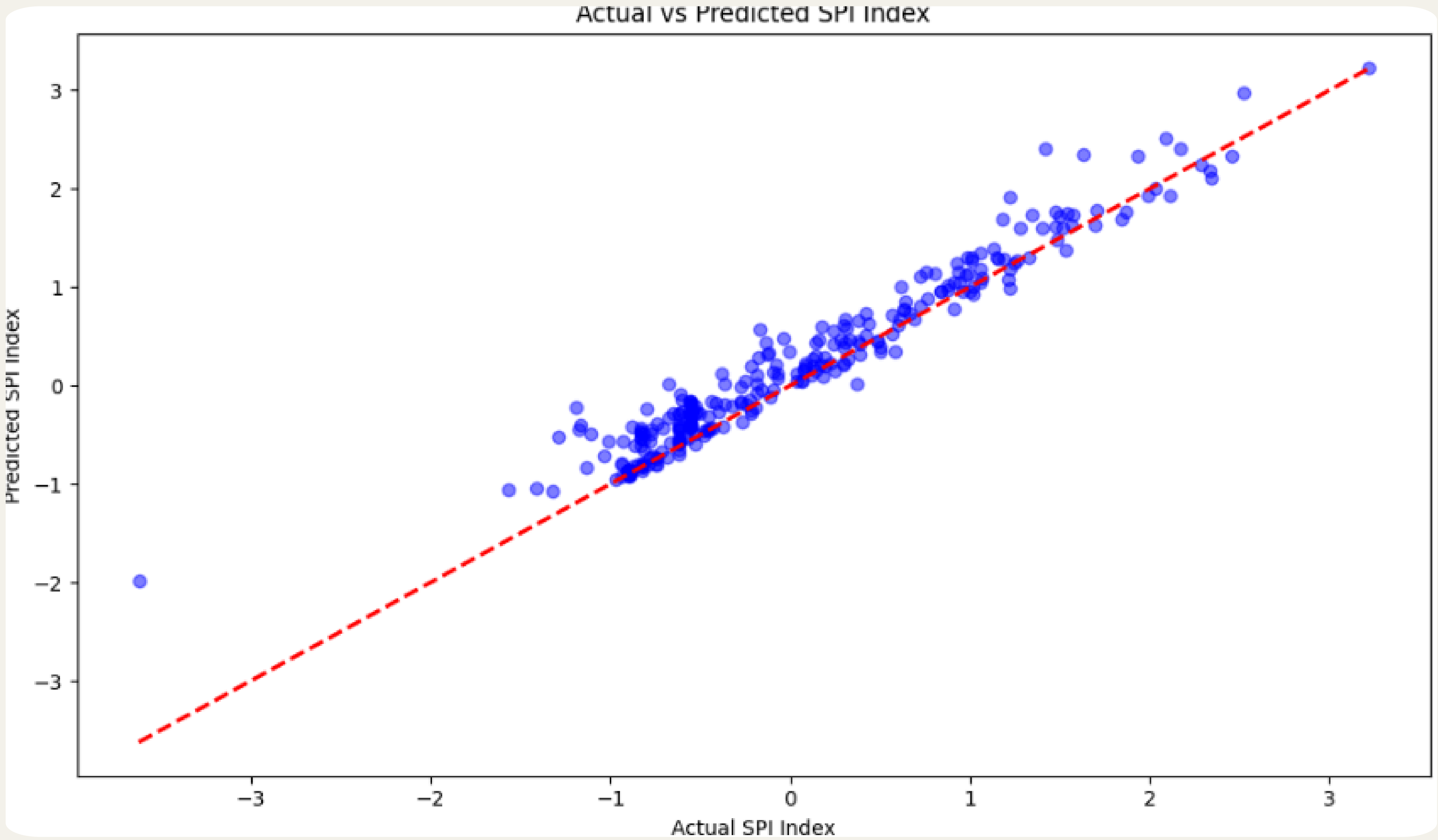
- **Data Preparation:**
  - Loaded dataset with SPI_1 and SST region features
  - Created lag features
  - Split the data into training and testing sets
- **Model Training:**
  - Trained an XGBoost Regressor with specified hyperparameters
  - Used training data to fit the model on SPI_1 as the target
- **Evaluation & Results:**
  - Predicted SPI_1 on test data
  - Calculated MAE and RMSE to measure regression performance
  - Plotted actual vs predicted SPI_1 values over time
  - Converted predicted SPI_1 values into drought categories and generated a confusion matrix to evaluate classification accuracy

XGBoost

RMSE: 0.0358
R^2 Score: 0.9605

- An RMSE of 0.0358 means the model's predictions are very close to the actual SPI_1 values.
- Value of 0.9605 means the model explains 96.05% of the variance

XGBoost

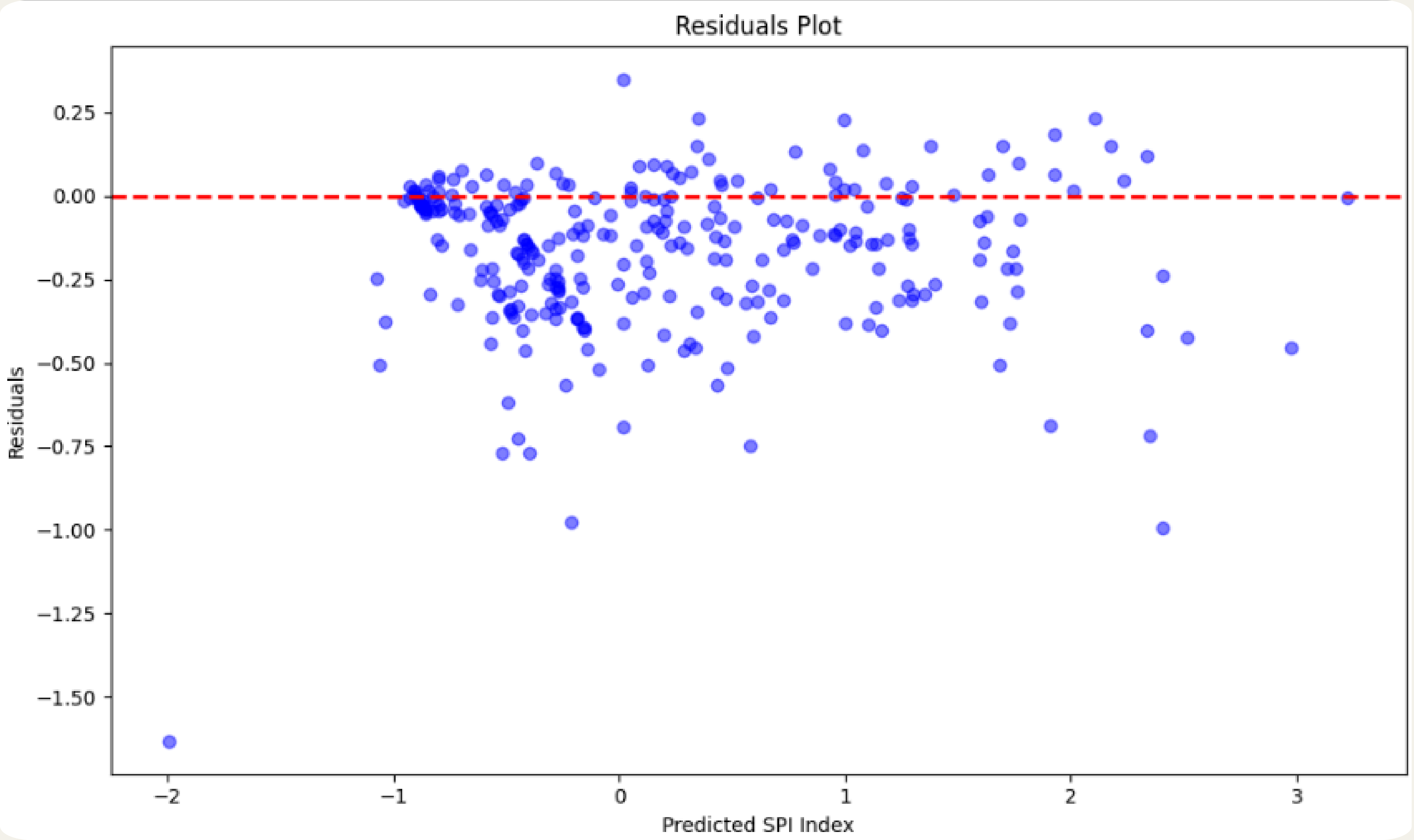


Key Observations:

- X-axis: Actual SPI values (ground truth)
- Y-axis: Predicted SPI values (model output)
- Red dashed line: Ideal prediction line (y = x) — perfect predictions would fall exactly on this line.
- Blue dots: Actual vs predicted values

- The majority of blue dots cluster closely around the red line, indicating good predictive performance.
- A few outliers exist, particularly in the more extreme negative SPI values, where predictions deviate a bit more.

XGBoost



- X-axis: Predicted SPI Index (what your model says)
- Y-axis: Residuals = Actual – Predicted (the error)
- Red dashed line: The "zero residual" line — where predictions are exactly correct
- Blue dots: Each dot represents the error for one prediction

- Most residuals are clustered close to zero, especially between predicted SPI values of -1 to 2.
- No clear pattern (like curves or funnels), which suggests the model doesn’t have systematic bias.
- The residual spread is roughly centered around the red zero line, which is what we want.

N-BEATS

- **Full Name:** Neural Basis Expansion Analysis for Time Series
- **Goal:** Forecast time series using deep learning, without relying on domain-specific feature engineering.
- **Architecture Highlights:**
  - Fully connected feedforward neural network.
  - Composed of stacks of blocks (each block has a backward forecast and forward prediction).
  - Each block outputs both:
    - Backcast: Residual to subtract from the input.
    - Forecast: Contribution to the final output.
- **Advantages:**
  - Interpretable (especially in the generic/trend/seasonality versions).
  - Competitive with state-of-the-art models.

N-BEATS

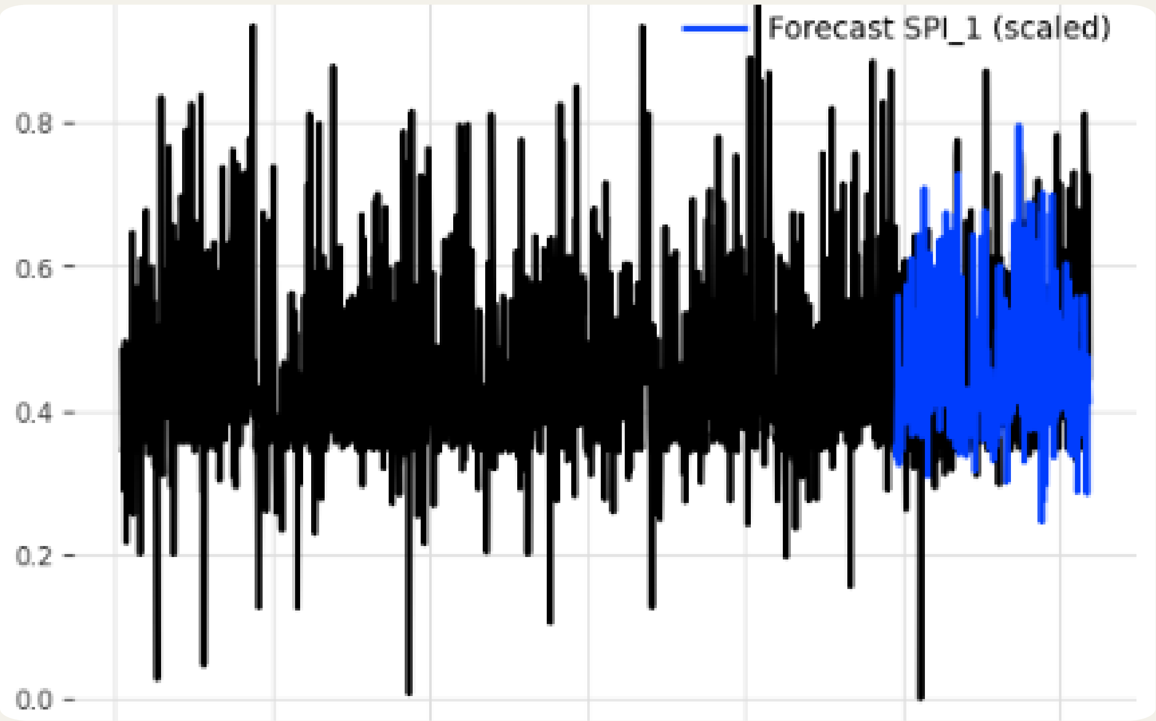
- **Data Preparation:**
  - Time series data loaded and converted into a TimeSeriesDataSet
  - Defines target variable, time index, and group ids
- **Model Setup:**
  - N-BEATS model initialized using NBeats.from_dataset()
  - Training configured with PyTorch Lightning (e.g., early stopping, checkpointing)
- **Training & Prediction:**
  - Model trained via trainer.fit()
  - Forecasts generated on validation/test sets
- **Evaluation:**
  - Predictions visualized, metrics like MAE or RMSE possibly used for assessment

N-BEATS

MAE: 0.12102275660598796
RMSE: 0.15501960840471960

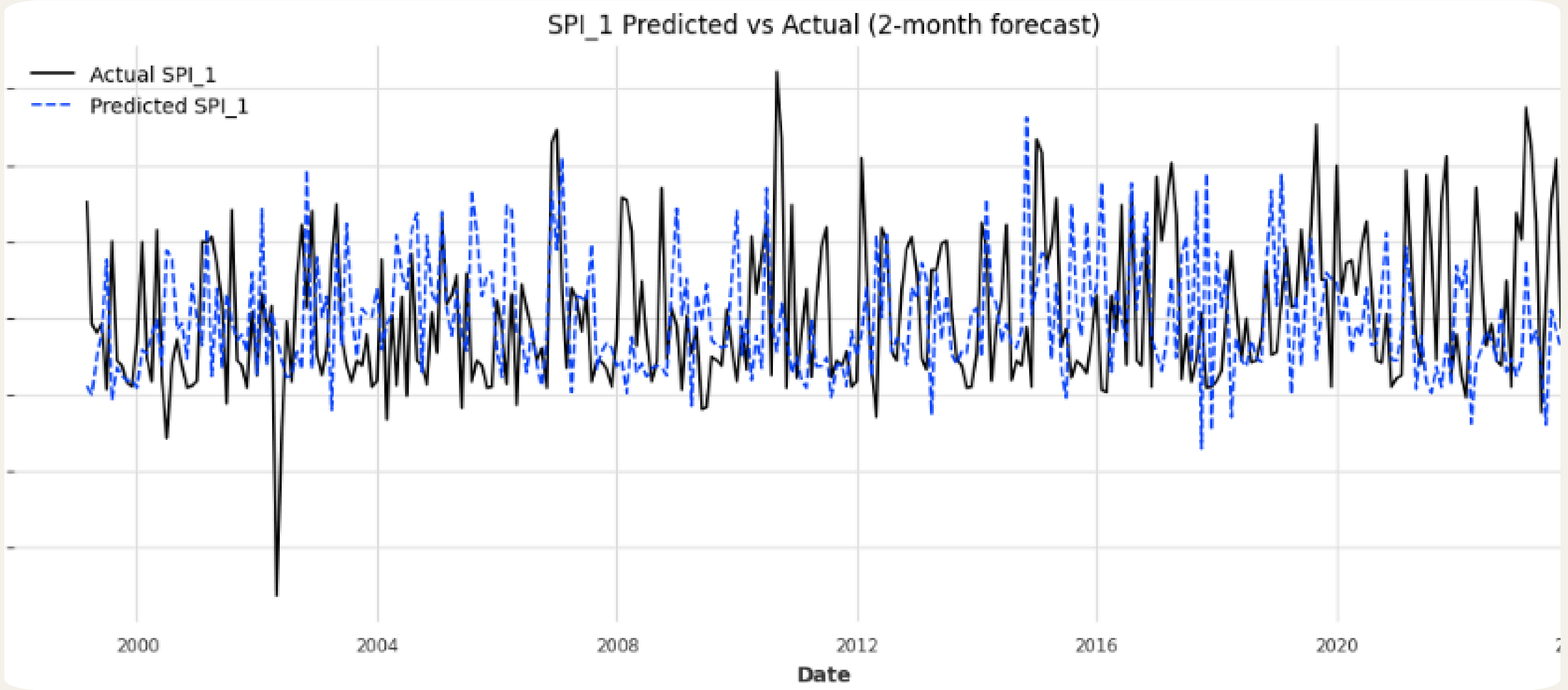
- Both MAE and RMSE are relatively low, suggesting:
  - The model has good predictive accuracy.
  - Predictions are generally close to the true values.
- The RMSE is only slightly higher than MAE, which implies:
  - There aren’t many large errors or outliers.
  - The model is stable and consistent in its forecasts.

N-BEATS



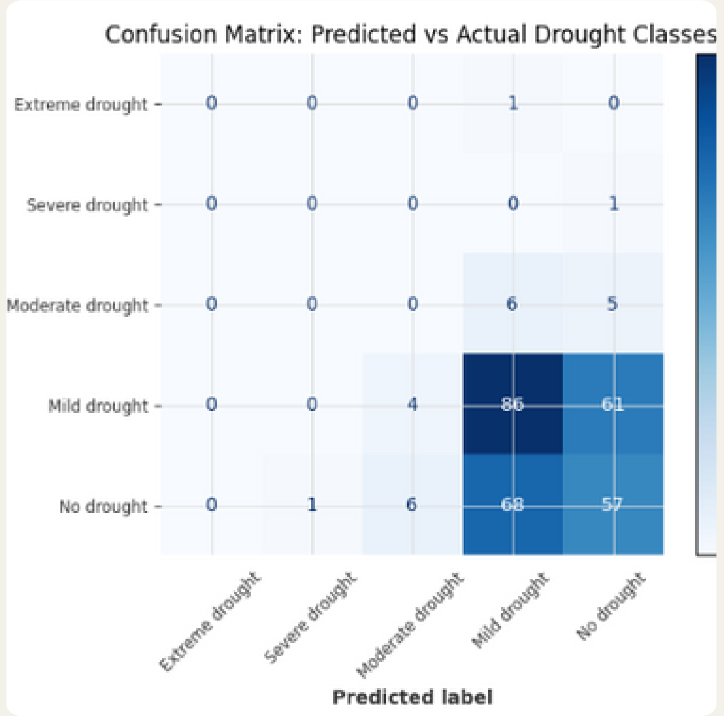
- Forecasts (in blue) start from around the year 2000.
- Predicted values closely follow actual trends within the scaled range.
  - The alignment suggests the model captures general patterns well, even over a large forecast horizon (2 months ahead).
  - Low noise and smooth tracking hint at good stability.

N-BEATS



- Some under/overestimation is visible, especially at extreme SPI_1 values (e.g., very dry or very wet events).
- Overall, the model performs consistently over two decades, which supports its reliability.

N-BEATS



- The model performs best on Mild Drought and No Drought categories, with most predictions falling on or near the diagonal.
- Misclassifications tend to happen between adjacent classes (e.g., Mild vs. No Drought), which is common in real-world forecasting.

- Extreme and Severe Droughts are rarely predicted, likely due to:
  - Class imbalance (fewer extreme drought events)
  - Difficulty in distinguishing rare events.

Contributions

Manas Dwivedi:- N-BEATS , Data Collection
Alok Godara:- Logistic Regression , Data Collection
Tranum:- XGBoost , Data Compilation
Ankush Choudhary:- LSTM , Data Compilation
Abhishek Singh:- LSTM , Data Compilation

Thank You

For Your Attention