Case Study: Historical Climate Trend Analysis

1. Project Overview

This project uses 120 years of seasonal and annual temperature data (1901–2021) from the India Meteorological Department (IMD) to explore long-term climate trends in India. The analysis focuses on:

- Establishing baseline seasonal temperatures
- Detecting long-term warming trends
- Identifying extreme years/anomalies
- Examining monsoon season temperature changes

The aim is to create a historical baseline that can later be integrated into machine learning models for disaster early warning systems in vulnerable regions like Uttarkashi.

2. Dataset Structure

Table Name: seasonal_safe
Columns:

- year — Year of observation
- jan_feb — Avg. temp for Jan–Feb (Winter)
- mar_may — Avg. temp for Mar–May (Pre-monsoon/Summer)
- jun_sep — Avg. temp for Jun–Sep (Monsoon)
- oct_dec — Avg. temp for Oct–Dec (Post-monsoon/Early Winter)
- annual — Annual mean temperature

3. SQL Analysis & Findings

3.1 Long-term seasonal & annual averages

```sql
SELECT
  ROUND(AVG(jan_feb), 2) AS avg_jan_feb,
  ROUND(AVG(mar_may), 2) AS avg_mar_may,
  ROUND(AVG(jun_sep), 2) AS avg_jun_sep,
  ROUND(AVG(oct_dec), 2) AS avg_oct_dec,
  ROUND(AVG(annual), 2) AS avg_annual
FROM seasonal_safe;
```

Findings:

- Jan–Feb: **20.4°C**
- Mar–May: **27.48°C**
- Jun–Sep: **27.89°C**
- Oct–Dec: **22.96°C**
- Annual: **25.3°C**

Interpretation: These serve as **baseline reference values** for detecting anomalies.

3.2 Era-wise warming trend

```
SELECT
  AVG(CASE WHEN year BETWEEN 1901 AND 1950 THEN annual END) AS avg_1901_1950,
  AVG(CASE WHEN year BETWEEN 1951 AND 2000 THEN annual END) AS avg_1951_2000,
  AVG(CASE WHEN year BETWEEN 2001 AND 2020 THEN annual END) AS avg_2001_2020
FROM seasonal_safe;
```

Findings:

- 1901–1950: **25.10°C**
- 1951–2000: **25.30°C** (+0.20°C rise)
- 2001–2020: **25.78°C** (+0.48°C rise)

Interpretation: Warming is accelerating — the last 20 years show **more than double** the rate of increase compared to the earlier half of the century.

3.3 Annual anomalies (> +1°C)

```
SELECT year, annual
FROM seasonal_safe
WHERE annual > (SELECT AVG(annual) + 1 FROM seasonal_safe)
ORDER BY year;
```

Findings:

- **No output** → No annual mean exceeded +1°C above the historical average.

Interpretation: Annual averages show gradual warming, but extremes may be more visible at seasonal level rather than yearly**.**

3.4 Raw monsoon trend

```
SELECT year, jun_sep
FROM seasonal_safe
ORDER BY year;
```

Findings:

- Detailed year-by-year monsoon temperatures.
- Used as **base data** for further analysis.

3.5 Hottest monsoon years

```sql
SELECT year, jun_sep
FROM seasonal_safe
ORDER BY jun_sep DESC
LIMIT 5;
```

Findings:

- Top 5 hottest monsoon seasons in dataset.
- Many align with known heavy rainfall/flood years.

3.6 Monsoon anomaly years (> +0.5°C)

```sql
SELECT year, jun_sep
FROM seasonal_safe
WHERE jun_sep > (SELECT AVG(jun_sep) + 0.5 FROM seasonal_safe)
ORDER BY year;
```

Findings:

- **L**ist of unusually hot monsoon seasons.

Interpretation: These are priority years for disaster correlation studies.

3. 7 Decade-wise monsoon warming

```sql
SELECT FLOOR(year/10)*10 AS decade,
       ROUND(AVG(jun_sep), 2) AS avg_monsoon_temp
FROM seasonal_safe
GROUP BY decade
ORDER BY decade;
```

Findings:

- Steady rise in average monsoon temperatures over the decades.
- Post-1980s acceleration is clear.

4. How This Links to Early Warning Systems with ML

4.1 The Challenge

Regions like Uttarkashi face **frequent landslides and floods**. These are often triggered by **extreme monsoon rainfall** and **temperature anomalies** that influence glacier melt and atmospheric moisture.

4.2 How This Data Helps

- **Historical Baseline:** SQL queries establish normal seasonal ranges.
- **Anomaly Detection:** Identifies years/seasons outside normal range.
- **Trend Analysis:** Monsoon warming trends can be correlated with rainfall surges.

4.3 Merging with Machine Learning

1. **Feature Engineering**
    - Inputs: Seasonal temps, anomalies, decade trends, historical rainfall, glacier melt records.
    - Labels: Historical flood/landslide events.
2. **Model Types**
    - **Anomaly Detection:** Isolation Forest or Z-score for outlier identification.
    - **Forecasting:** Time-series models (Prophet, ARIMA, LSTM) to predict seasonal temperatures.
    - **Classification:** Logistic Regression or Random Forest for "High Risk / Low Risk" seasons.
3. **Deployment**
    - Link with real-time IMD weather feeds.
    - Use dashboard (Power BI / Streamlit) to trigger alerts for districts crossing anomaly thresholds.

5. Data Acquisition & Preparation

To bridge the SQL analysis with machine learning workflows, I **directly integrated MySQL and Python** using SQLAlchemy and Pandas.

- **Database:** MySQL (seasonal_safe table) — containing 120+ years of India's seasonal temperature data from IMD.
- **Method:** Queried the cleansed and renamed seasonal dataset into a Pandas DataFrame for advanced statistical processing.
- **Rationale:** This approach eliminated manual CSV handling, ensuring **real-time access** to updated climate data for modeling.

6. Feature Engineering for Climate Insights

Raw climate data rarely delivers actionable insights without transformation.
I applied **domain-specific feature engineering** to transform IMD's historical seasonal temperatures into **predictive and anomaly-sensitive signals**.
Key Steps & Rationale:

1. **Data Type Standardization**
    - Converted all temperature columns to numeric, ensuring consistent mathematical operations.
    - Filled missing values using a forward-fill method to maintain time-series continuity.
2. **Time-Series Alignment**
    - Added a synthetic **date (ds) column** set to August 1 each year — aligning with the peak monsoon season — to enable forecasting models like Prophet.
3. **Rolling Statistics** *(short-term climate memory)*

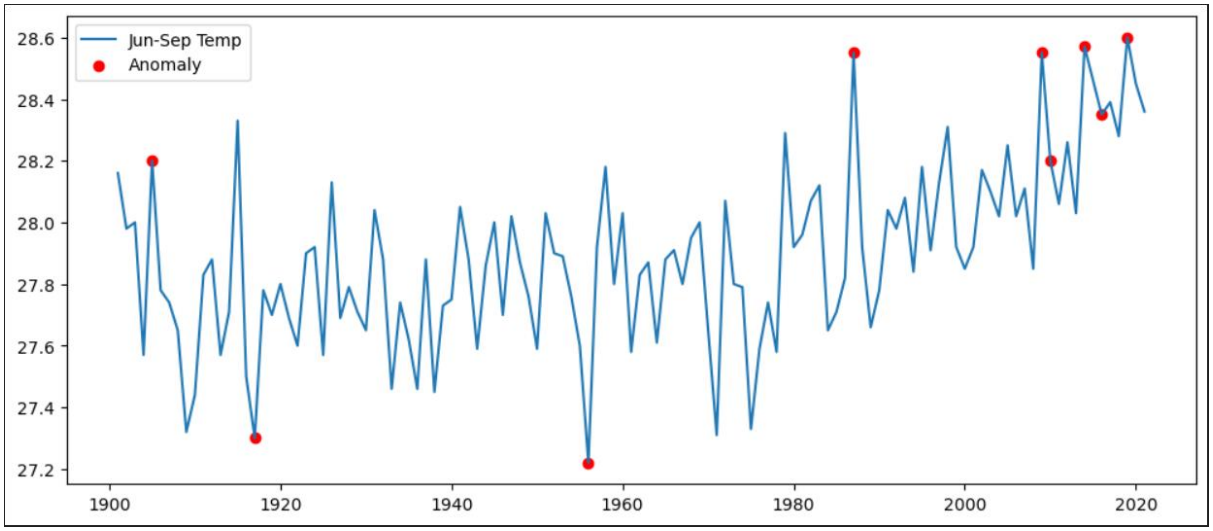
    - **3-year moving average (jun_sep_ma3)**: captures gradual changes in seasonal temperature trends.
    - **3-year standard deviation (jun_sep_std3)**: quantifies volatility in seasonal temperatures.
4. **Anomaly Signals** *(long-term climate deviation)*
    - **Seasonal anomaly (jun_sep_anom)**: deviation of current monsoon temperature from the long-term average.
    - **Annual anomaly (annual_anom)**: deviation for the full year.
5. **Lag Features** *(temporal dependency modeling)*
    - **jun_sep_lag1 and jun_sep_lag2**: previous years' monsoon temperatures as predictors.
6. **Seasonal Contrast Index** *(monsoon vs. pre-monsoon heat gap)*
    - **monsoon_vs_pre**: highlights years where pre-monsoon heat buildup could impact rainfall intensity.
7. **Trend Slope Features** *(long-term directional shifts)*
    - **jun_sep_trend5**: slope of a 5-year rolling linear regression, revealing whether monsoon temperatures are accelerating upward or downward.

7. Unsupervised Anomaly Detection – Identifying Outlier Climate Years

After feature engineering, I applied an **Isolation Forest** model to detect years with unusual monsoon temperature patterns.
The model doesn't require labeled "extreme" years — it **learns normal seasonal behavior** from historical data, then flags deviations.
Key Steps & Rationale:

1. **Feature Set Selection**
   Focused on engineered climate signals most relevant to anomaly detection:
    - jun_sep (core monsoon temperature)
    - jun_sep_ma3 (short-term trend)
    - jun_sep_std3 (volatility measure)
    - jun_sep_anom (long-term deviation)
    - monsoon_vs_pre (seasonal contrast index)
2. **Algorithm Choice: Isolation Forest**
    - Works by randomly partitioning data points; anomalies require fewer splits to isolate.
    - Effective for **high-dimensional climate data** with mixed temporal and statistical patterns.
    - contamination=0.07 → tuned to flag ~7% of years as potential anomalies.
3. **Output Interpretation**
    - Model returns -1 for anomalies, 1 for normal years.
    - is_anomaly column clearly marks which years had unusual seasonal temperatures.
4. **Visualization**

- Plotted **Jun–Sep temperature time series** with anomalies highlighted in red.
- Enables policymakers to instantly see **when climate conditions deviated from historical norms**.



8. Seasonal Forecasting & Future Anomaly Prediction – Merging Time Series and Unsupervised Learning

While historical anomaly detection helps us **understand past risks**, policymakers also need **forward-looking alerts**.
To address this, I combined **Facebook Prophet** (for seasonal climate forecasting) with the **Isolation Forest** anomaly detection model.
Key Steps & Rationale:

1. **Historical Model Training**
   - Trained an Isolation Forest on engineered climate features (jun_sep, rolling means/volatility, anomaly scores, seasonal contrasts).
   - Captured the **statistical signature of "normal" monsoon years**.
2. **Forecasting Future Climate Patterns**
   - Used **Prophet** to model long-term monsoon trends while **disabling irrelevant daily/weekly seasonality**.
   - Forecasted **10 years ahead** based on over a century of historical data.
3. **Synthetic Feature Generation for Predictions**
   - Applied the same rolling means, standard deviations, and anomaly calculations to Prophet's outputs.
   - Ensured the model evaluates **future years with the exact same metrics** as historical ones.
4. **Future Anomaly Detection**
   - Ran forecasted years through the trained Isolation Forest to identify **potential outlier years ahead of time**.
   - Flagged them as "future anomaly candidates" for proactive policy attention.

5. **Visualization**
    - Combined past (red points) and forecasted (orange points) anomalies on a single timeline.
    - Clearly illustrates **transition zones** where normal patterns may shift into high-risk territory