Project Problem Statement

- Floods are among the most devastating natural disasters, causing massive environmental, economic, and human losses. Conventional flood prediction systems rely heavily on manual monitoring or delayed alerts from weather departments, which limits accuracy and response time. Furthermore, many developing regions lack cost-effective and scalable systems to analyze continuous weather variations.

- To address this challenge, an intelligent flood risk prediction system is needed that leverages Machine Learning (ML) to analyze real-time weather parameters such as rainfall, humidity, temperature, and atmospheric pressure. This system can predict the probability of flood occurrence in specific regions and alert users in advance, enabling timely preventive measures. The solution supports sustainable disaster management by improving community preparedness and reducing the environmental and economic impact of floods.

Dataset:

• DatasetName: Flood Prediction Dataset (Weather and Flood History Data)

About Dataset:

The dataset contains historical weather records combined with flood event data for multiple regions. It includes parameters such as rainfall (mm), humidity (%), temperature (°C), pressure (hPa), and wind speed (m/s), along with labels indicating whether a flood occurred (Low, Moderate, or High risk).

This dataset enables the development of supervised ML models such as Random Forest or Logistic Regression, which can learn from past data to forecast future flood risks.

Real-time data from OpenWeather API is also integrated for live prediction, allowing continuous monitoring and early warnings

Source: • Kaggle (Flood Prediction Dataset)
• OpenWeatherMap API (for real-time updates)
• Government Open Data Platforms (IMD, NOAA)

Flood Prediction Dataset

Next Steps:

1. Collect & Prepare Dataset
   - Download historical flood datasets from Kaggle or government data portals.
   - Combine with real-time weather data from OpenWeather API.
   - Preprocess the data: handle missing values, normalize parameters, and create labels for flood risk levels (Low/Moderate/High).
2. Train the CNN Model
   - Use algorithms like Random Forest, Logistic Regression, or XGBoost.
   - Split dataset into training and testing sets (e.g., 80:20).
   - Train the model to recognize flood-risk patterns using weather features.
3. Evaluate & Test the Model
   - Evaluate model performance with metrics such as accuracy, precision, recall, and F1-score.
   - Validate with unseen test data and adjust hyperparameters for improved accuracy.
4. Build the Web Interface
   - Connect OpenWeather API to fetch live temperature, rainfall, humidity, and pressure.
   - Feed data into the trained ML model to generate live flood risk predictions with probability scores.
5. Test the Complete Application
   - Develop a simple web or desktop dashboard that displays:
   - Flood Risk Level (Low/Moderate/High)
   - Probability Score (e.g., 78%)
   - Weather Parameters in Real-Time
   - Optionally include map visualization or alert notifications.
6. Deploy & Document
   - Deploy the model locally or on a cloud platform.
   - Prepare documentation including dataset details, model design, performance metrics, and sustainability impact.
   - Highlight how the system supports environmental sustainability by promoting early warning, risk mitigation, and public safety.