# Weather Modeling using the Quadratic Model – Iterative Process Model

## Overview

A weather modeling system using a quadratic regression approach can be effectively developed using the Iterative Process Model. This method allows for gradual enhancement through multiple development cycles, where each cycle builds upon the previous one with refined features, improved accuracy, and user feedback incorporation.

## Scope

The objective is to predict rainfall (in mm) and humidity (in %) over a 24-hour period using a quadratic model. Each iteration aims to enhance the prediction accuracy and user experience using historical time-based data.

## Inputs

• Time (in hours): time = [0, 4, 8, 12, 16, 20]

• Rainfall (in mm): rainfall = [0, 2, 8, 15, 10, 3]

• Humidity (in %): humidity = [95, 85, 70, 60, 75, 90]

## Iteration Phases

### Iteration 1: Basic Model Setup

• Requirement Gathering: Identify the core functionality – basic prediction using quadratic regression.

• Design: Apply y = a·t² + b·t + c formula to fit both rainfall and humidity curves.

• Implementation: Use Python with NumPy to perform polynomial fitting.

• Testing: Use a small dataset and validate with MAE or RMSE.

• Feedback: Gather initial user feedback on usability and prediction trend.

### Iteration 2: Visualization Integration

• Enhancement: Integrate Matplotlib for graphical plots of predictions.

• Improved Design: Display scatter and line plots to compare actual vs. predicted data.

• Testing: Ensure plots are accurate and readable.

• Feedback: Users provide input on the graph clarity and data representation.

### Iteration 3: Accuracy Improvement

• Refinement: Tune the model with updated or extended datasets.

• Verification: Compare predictions with newer real-world data.

• Testing: Evaluate error margin (±2 mm rainfall, ±5% humidity).

• Feedback: Improve model coefficients based on performance.

### Iteration 4: Deployment and Feedback Loop

• Deployment: Launch on Google Colab, GitHub, or a simple web interface.

• User Acceptance Testing (UAT): Final testing by target users to ensure predictions and graphs meet expectations.

• Maintenance: Prepare for future updates by documenting feedback and maintaining code quality.

## System Design Summary

• Input: Time, rainfall, humidity

• Processing: Polynomial fitting

• Output: Numerical predictions + visual plots

## Code

# Step 1: Input Data

time = np.array([0, 4, 8, 12, 16, 20])  # Time in hours

rainfall = np.array([0, 2, 8, 15, 10, 3])     # Rainfall in mm

humidity = np.array([95, 85, 70, 60, 75, 90]) # Humidity in %

# Step 2: Fit quadratic models

rain\_coeff = np.polyfit(time, rainfall, 2)

hum\_coeff = np.polyfit(time, humidity, 2)

a\_r, b\_r, c\_r = rain\_coeff

a\_h, b\_h, c\_h = hum\_coeff

print(f"\nDeveloped Quadratic Model for Rainfall:")

print(f"R(t) = {a\_r:.4f}t² + {b\_r:.4f}t + {c\_r:.4f}\n")

print(f"Developed Quadratic Model for Humidity:")

print(f"H(t) = {a\_h:.4f}t² + {b\_h:.4f}t + {c\_h:.4f}\n")

# Step 3: Predict for every hour from 0 to 24

t\_values = np.arange(0, 25, 1)

predicted\_rainfall = a\_r \* t\_values\*\*2 + b\_r \* t\_values + c\_r

predicted\_humidity = a\_h \* t\_values\*\*2 + b\_h \* t\_values + c\_h

print("Predicted Rainfall (mm) and Humidity (%) for 24 Hours:\n")

for t, rain, hum in zip(t\_values, predicted\_rainfall, predicted\_humidity):

    print(f"At {t:02d}:00 hrs -> Rainfall: {rain:.2f} mm, Humidity: {hum:.2f} %")

# Step 4: Plot (if matplotlib is available)

try:

    import matplotlib.pyplot as plt

    plt.figure(figsize=(12, 6))

    # Rainfall Plot

    plt.subplot(1, 2, 1)

    plt.scatter(time, rainfall, color='blue', label='Original Rainfall Data', zorder=5)

    plt.plot(t\_values, predicted\_rainfall, color='red', linestyle='--', label='Rainfall Prediction')

    plt.title('Rainfall Prediction using Quadratic Model')

    plt.xlabel('Time (Hours)')

    plt.ylabel('Rainfall (mm)')

    plt.xticks(np.arange(0, 25, 2))

    plt.grid(True)

    plt.legend()

    # Humidity Plot

    plt.subplot(1, 2, 2)

    plt.scatter(time, humidity, color='green', label='Original Humidity Data', zorder=5)

    plt.plot(t\_values, predicted\_humidity, color='orange', linestyle='--', label='Humidity Prediction')

    plt.title('Humidity Prediction using Quadratic Model')

    plt.xlabel('Time (Hours)')

    plt.ylabel('Humidity (%)')

    plt.xticks(np.arange(0, 25, 2))

    plt.grid(True)

    plt.legend()

    plt.tight\_layout()

    plt.show()

except ImportError:

    print("\nNOTE: 'matplotlib' is not installed. Skipping graph display.")

    print("To install it, run: pip install matplotlib")

**Output**

A screenshot of a computer

AI-generated content may be incorrect.

## Conclusion

Using the Iterative Process Model, the weather prediction system evolves through multiple refinement cycles, ensuring continuous improvement in accuracy, performance, and user experience. This model allows flexibility to incorporate feedback, extend functionality, and respond to changing requirements efficiently.