# Weather Prediction Using Machine Learning

*A project submitted in partial fulfilment of*
*The requirements to the award of the degree of*

**Bachelor in Technology**
**In**
**INFORMATION TECHNOLOGY**

Submitted by:

**Aditya** **(12212033)** **&** **Piyush** **(12212001)**

Supervised by:

**Dr. Tayyab Khan**

Assistant Professor

**INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, SONEPAT - 131201, HARYANA, INDIA**

## ACKNOWLEDGEMENTS

The success and the outcome of this project required ceaseless guidance and assistance, my team members and I are extremely privileged to have got this all along the project.

We would like to take this opportunity to acknowledge all the people who have helped us whole heartedly in every stage of this project.

We are indebtedly grateful to **(Dr.) Tayyab Khan**, Assistant professor, IT, IIIT SONEPAT for providing this opportunity in the first place and giving us all the support and guidance possible, in spite of having a busy schedule.

**Aditya Mishra**
**(12212033)**

**Piyush Raj**
**(12212001)**

## SELF DECLARATION

I hereby state that work contained in the project titled **“Weather Prediction Using Machine Learning”** is original. I have followed the standards of the project ethics to the best of my abilities. I have acknowledged all the sources of knowledge which I have used in the project

.

Name: **Aditya Mishra**
Roll no: 12212033

Department of Information Technology,
Indian Institute of Information technology,
Sonepat-131201, Haryana, India

## SELF DECLARATION

I hereby state that work contained in the project titled **“Weather Prediction Using Machine Learning**" is original. I have followed the standards of the project ethics to the best of my abilities. I have acknowledged all the sources of knowledge which I have used in the project

.

Name: **Piyush Raj**
Roll no: 12212001

Department of Information Technology,
Indian Institute of Information technology,
Sonepat-131201, Haryana, India

## CERTIFICATE

This is to certify that **Mr. Aditya** has worked on the project entitled “**Weather Prediction Using Machine Learning**” under my supervision and guidance.

The contents of the project, being submitted to the Department of Information Technology, IIIT SONEPAT, HARYANA, for the award of the degree of B. Tech in Information Technology, are original and carried out by candidate himself. This project has not been submitted in full or part for award of any other degree or diploma to this or any other university.

**Dr. Tayyab Khan**
Supervisor

Department of Information Technology,
Indian Institute of Information technology,
Sonepat-131201, Haryana, India

# CERTIFICATE

This is to certify that **Mr. Piyush** has worked on the project entitled “**Weather Prediction Using Machine Learning** under my supervision and guidance.

The contents of the project, being submitted to the Department of Information Technology, IIIT SONEPAT, HARYANA, for the award of the degree of B. Tech in Information Technology, are original and carried out by candidate himself. This project has not been submitted in full or part for award of any other degree or diploma to this or any other university.

**Dr. Tayyab Khan**
Supervisor

Department of Information Technology,
Indian Institute of Information technology,
Sonepat-131201, Haryana, India

# ABSTRACT

Weather forecasting is a vital aspect of planning and decision-making across various sectors such as agriculture, transportation, disaster management, and logistics. Traditional forecasting methods, primarily based on numerical simulations and physical models, often require significant computational resources and may struggle to capture nonlinear relationships in atmospheric data. With the increasing availability of historical and real-time weather data, machine learning techniques have emerged as powerful alternatives for improving forecast accuracy.

This project presents a data-driven approach to short-term weather prediction using the Random Forest classification algorithm. The model is trained on historical weather data—including temperature, humidity, wind speed, and precipitation—and uses real-time inputs from the OpenWeather API to predict temperature classes (low, moderate, high) for the next five days. The Random Forest model was chosen for its robustness, resistance to overfitting, and effectiveness in handling complex, high-dimensional datasets. The system's performance is evaluated using accuracy, precision, recall, and F1-score, with visualizations provided through confusion matrices and scatter plots. The results demonstrate that the proposed approach offers a reliable and efficient solution for short-term weather forecasting, with scope for further enhancement through dynamic model updates and deep learning integration.

# TABLE OF CONTENTS

# CHAPTER 1
# INTRODUCTION

## 1.1 Introduction

Weather forecasting is an essential component of decision-making in various fields, including agriculture, aviation, transportation, and disaster management. Accurate predictions can help reduce the negative impact of extreme weather conditions and optimize day-to-day operations across numerous sectors. Traditional forecasting methods rely heavily on numerical simulations and mathematical modeling, which can be computationally intensive and less effective in capturing non-linear atmospheric patterns.

With the rise of Artificial Intelligence (AI), Machine Learning (ML) algorithms have shown great potential in improving forecasting accuracy by learning from historical weather data. Among these algorithms, **Random Forest**, an ensemble learning technique based on decision trees, stands out for its robustness, scalability, and ability to handle complex, high-dimensional data.

This project aims to develop a machine learning-based weather prediction system using the Random Forest algorithm. The model is trained on historical weather data to learn patterns between atmospheric variables such as temperature, humidity, wind speed, and precipitation. After training, the system utilizes real-time weather data fetched from the **OpenWeather API** to predict the temperature class for the next five days.

The goal of this project is to demonstrate the effectiveness of Random Forest in short-term weather prediction and to develop a reliable, data-driven forecasting solution that can be used in practical applications.

## 1.2 Problem Description

Weather forecasting plays a crucial role in supporting decision-making across sectors like agriculture, transportation, logistics, and disaster management. However, traditional numerical weather prediction (NWP) models, which rely on physical equations and simulations, are often computationally intensive and may struggle to capture the nonlinear interactions between atmospheric variables—particularly for short-term, localized forecasts.

With the increasing availability of historical and real-time weather data, machine learning has emerged as a powerful alternative for building data-driven prediction systems. Yet, challenges such as high-dimensional input features, missing data, and the complex relationships between variables make accurate forecasting difficult.

To address these issues, this project implements a **Random Forest** classification model. Random Forest is an ensemble learning technique that builds multiple decision trees and combines their outputs for improved accuracy and robustness. It is well-suited for handling nonlinear patterns and noisy data, making it ideal for weather prediction tasks.

The model is trained on historical weather datasets containing variables such as temperature, humidity, precipitation, and wind speed. It is then used to predict the **temperature class (e.g., low, moderate, high)** for the next five days using current weather inputs collected via the **OpenWeather API**. The goal is to create an efficient, accurate, and easily deployable machine learning system for short-term weather forecasting.

## 1.3 Problem Solution

To address the challenges of short-term weather prediction, this project proposes a machine learning-based solution using the **Random Forest** classification algorithm. The solution is designed to leverage both historical weather data and real-time conditions to predict temperature trends over a five-day horizon.

**Key Components of the Solution:**

- **Data Collection:**
  Historical weather data is collected from publicly available datasets. This data includes essential meteorological features such as temperature, humidity, wind speed, pressure, and precipitation.

- **Preprocessing and Feature Engineering:**
  The raw data is cleaned to handle missing values, and relevant features are selected based on their correlation with the target variable. Additional features such as moving averages or derived temperature ranges may be added to improve model performance. All numerical features are standardized to ensure consistency during training.

- **Model Implementation: Random Forest:**
  The Random Forest algorithm is used due to its ability to handle nonlinear data, prevent overfitting, and provide high classification accuracy. It constructs an ensemble of decision trees using random subsets of the training data and aggregates their outputs to make predictions. The model is trained to classify temperature into predefined categories (e.g., low, moderate, high).

- **Real-Time Data Integration:**
  To make live predictions, current weather conditions are fetched using the **OpenWeather API**. This allows the model to operate in real-time by applying learned patterns from historical data to the latest conditions.

- **Prediction and Output:**
  Based on current data, the model predicts the **temperature class for the next five days**. These predictions can be used in applications that require short-term climate awareness and planning.

- **Evaluation:**
  The model's performance is evaluated using metrics such as **accuracy, precision, recall, and F1-score**, along with **confusion matrices** and **scatter plots** to visualize prediction quality.

This solution provides a practical and scalable framework for accurate, short-term temperature prediction using machine learning, offering improved performance over traditional methods and greater responsiveness to real-world atmospheric data.

## 1.4 Limitations

The model is designed to predict temperature classes for only the next five days. Its performance over longer forecasting horizons has not been tested and may degrade due to increased uncertainty.

- **Static Model Training:**

The model is trained on historical data and does not automatically update with new information. Without periodic retraining, its accuracy may decline as weather patterns evolve over time.

- **Simplified Output:**

The model classifies temperature into discrete classes (e.g., low, moderate, high) rather than providing continuous temperature values, which might limit precision in applications requiring exact forecasts.

- **No Consideration of Spatial Variability:**

The model does not account for geographical variations or elevation differences, which can significantly impact weather conditions. It treats all locations equally based on the input features provided.

- **Dependence on OpenWeather API:**

The accuracy of predictions depends on the reliability of current weather data fetched from the OpenWeather API. Any delays, inaccuracies, or disruptions in the API may affect model performance.

- **Exclusion of Extreme Weather Events:**

The dataset used may lack sufficient examples of extreme or rare weather events (e.g., heatwaves, storms, cold fronts), limiting the model's ability to predict such occurrences accurately

.

## 1.5 Future Scope

This project demonstrates the effectiveness of the Random Forest algorithm in predicting short-term temperature trends using historical and real-time weather data. However, there are several opportunities to enhance the system and broaden its applicability:

- **Incorporation of Regression Models:**

Instead of predicting temperature classes, future work could implement regression-based models to forecast exact temperature values, providing more detailed and continuous predictions.

- **Integration of Deep Learning Techniques:**
Advanced models such as Long Short-Term Memory (LSTM) networks or Gated Recurrent Units (GRUs) can be explored to capture temporal dependencies in weather patterns more effectively, especially for longer-term forecasting.

- **Dynamic Model Updating:**
Incorporating an online learning approach or automated retraining pipeline would enable the model to adapt continuously as new weather data becomes available, improving its accuracy over time.

- **Geospatial and Satellite Data Usage:**
Adding spatial features such as latitude, longitude, elevation, or satellite imagery can help improve model accuracy by accounting for regional climatic differences.

- **Prediction of Multiple Weather Parameters:**
In future versions, the model can be extended to predict other important parameters like humidity, wind speed, rainfall probability, or air quality, making it more comprehensive.

- **Mobile or Web Deployment:**
Developing a web-based or mobile application can make the predictions accessible to end-users in real-time, providing weather insights for farmers, commuters, event planners, and others.

- **Extreme Weather Detection:**
By enhancing the dataset with rare or extreme events, the model could be trained to detect critical conditions like storms, heatwaves, or heavy rainfall, which are vital for disaster preparedness.

## 1.6 Summary

This project develops a machine learning-based weather prediction system using the Random Forest algorithm to forecast short-term temperature trends. The model is trained on historical weather data, including variables like temperature, humidity, and wind speed, and leverages real-time weather inputs from the OpenWeather API. It classifies temperature into predefined categories (low, moderate, high) for the next five days, offering a practical solution for short-term weather forecasting. Model performance is evaluated using accuracy, precision, recall, and F1-score, visualized through confusion matrices and scatter plots.

While the Random Forest model demonstrates accuracy and robustness, limitations include a restricted forecast range, dependence on static training data, and simplified output (classifications rather than continuous values). The model does not consider geographical or spatial variability, and its performance relies on the OpenWeather API for real-time data. Future improvements could include using regression models for precise temperature forecasts, integrating deep learning techniques, and expanding the model to predict additional weather parameters, along with the potential for mobile or web deployment and extreme weather detection.

# CHAPTER 2
# LITERATURE SURVEY

## 2.1 Introduction

Weather forecasting has long been a critical field of study, with advancements in both traditional numerical models and modern machine learning approaches. Traditional weather prediction methods, such as Numerical Weather Prediction (NWP) models, rely heavily on complex mathematical simulations and physical equations to model atmospheric dynamics. These methods, while accurate, are computationally expensive and struggle to capture the nonlinear relationships and rapid changes in weather patterns, especially for short-term forecasts. In recent years, the rise of Artificial Intelligence (AI) and Machine Learning (ML) has offered a promising alternative by leveraging large datasets of historical weather records to identify patterns and make predictions.
Several machine learning algorithms, including Support Vector Machines (SVM), K-Nearest Neighbors (KNN), and Random Forest, have been explored for weather prediction tasks due to their ability to handle large, high-dimensional datasets and capture complex relationships between weather variables. Random Forest, in particular, has gained attention for its robustness, ability to handle noisy data, and high classification accuracy. This literature survey explores various studies and approaches that have utilized machine learning techniques in weather forecasting, with a particular focus on the application of Random Forest for short-term weather predictions. By analyzing existing research, this survey aims to provide insights into the strengths and limitations of different models and establish the foundation for the current project’s approach.

## 2.2 Related Work

Between 2018 and 2025, significant contributions have been made to weather prediction analysis system, utilizing a variety of machine learning models. Below is a summary of these contributions along with the titles of their papers:
These contributions illustrate the diversity of approaches in the field and emphasize the trend of combining various machine learning techniques to improve credit card fraud detection accuracy and efficiency.

| **Authors** | **Year** | **Title** | **Key Findings** |
|---|---|---|---|
| Rasp & Lerch | 2018 | Neural networks for post-processing ensemble weather forecasts | Improved probabilistic forecasts over traditional methods |
| Sharma & Vijayakumar | 2019 | Predictive Analytics in Weather Forecasting Using Machine Learning Algorithms | ML models enhance weather prediction accuracy |
| Weyn et al. | 2020 | Improving data-driven global weather prediction using deep | Achieved stable long-range forecasts with realistic patterns |

| | | | |
|---|---|---|---|
| | | CNNs on a cubed sphere | |
| Chen et al. | 2023 | FuXi: A cascade machine learning forecasting system for 15-day global weather forecast | Comparable performance to ECMWF ensemble mean |
| Olivetti & Messori | 2024 | Advances and prospects of deep learning for medium-range extreme weather forecasting | Identified challenges and proposed workflow for DL models |
| Zhang et al. | 2025 | Machine Learning Methods for Weather Forecasting: A Survey | Comprehensive overview of ML applications in weather forecasting |
| Shi et al. | 2025 | Deep Learning and Foundation Models for Weather Prediction: A Survey | Proposed taxonomy and discussed challenges in DL models |
| Arakh et al. | 2024 | Advancements in Weather Forecasting through Machine Learning Algorithms | Highlighted ML's superiority over traditional methods |
| Khatri & Rajani | 2025 | Advanced Weather Forecasting with Machine Learning | Achieved 80.11% accuracy in urban flood-prone areas |
| Gupta | 2025 | Integrating Predictive Analytics and Machine Learning for Weather Forecasting | Demonstrated improved accuracy in weather predictions |
| DeepMind | 2024 | GenCast: AI model for weather forecasting | Surpassed traditional systems in predicting extreme events |
| Nvidia Research | 2024 | StormCast: AI weather model | Enhanced accuracy at kilometer-scale predictions |
| FT Report | 2024 | NeuralGCM: Hybrid AI model for climate forecasting | Improved long-term climate trend predictions |
| TIME Report | 2024 | AI in Natural Disaster Response | Improved urban resilience through AI applications |

| | | | |
|---|---|---|---|
| New Yorker | 2019 | Why Weather Forecasting Keeps Getting Better | Highlighted evolution and improvement in forecasting |
| Pathak et al. | 2020 | Deep Learning-Based Weather Forecasting System | Better temperature and precipitation prediction over baseline models |
| Li et al. | 2022 | Attention-Based Models for Rainfall Prediction | Accurate short-term rainfall forecasting |
| Banerjee & Dutta | 2021 | Machine Learning Approaches to Indian Monsoon Forecasting | ML performed better in seasonal monsoon prediction |
| Gao et al. | 2023 | AI for Nowcasting Severe Weather | Improved severe storm nowcasting capabilities |
| Kim et al. | 2021 | Hybrid Deep Learning Models for Typhoon Forecasting | Enhanced tracking of typhoon paths and intensities |

Table 1. Literature Review

## 2.3 Contribution

- Developed a machine learning model to forecast weather parameters (temperature, humidity, wind speed) using historical data.
- Preprocessed real-world datasets by cleaning missing values, encoding categorical variables, and scaling features.
- Implemented and compared multiple ML models: Linear Regression, Decision Tree Regressor, and Random Forest Regressor.
- Random Forest Regressor showed the best performance with an $R^2$ score greater than 0.85.
- Evaluated model accuracy using metrics such as Mean Absolute Error (MAE) and Mean Squared Error (MSE).
- Created a lightweight, modular pipeline suitable for educational use or low-resource local weather stations.
- Contributed to the field by demonstrating that classical ML models can yield accurate short-term forecasts with minimal infrastructure.
- Provided a comparative analysis of models that can guide future research in weather prediction.

## 2.4 Summary

In Chapter 2, the literature survey focuses on the various advancements and approaches used in the field of weather prediction. It begins with an introduction to the technologies and methodologies adopted in this domain. The related work section reviews key studies and contributions from 2018 to 2025, highlighting the use of machine learning algorithms, ensemble models, and deep learning techniques in enhancing the accuracy of weather forecasting systems. A table summarizes these contributions, showcasing the evolution of different methods—from classical algorithms like Linear Regression and Random Forest to more advanced approaches such as hybrid deep learning models and foundation AI models like GenCast and NeuralGCM. Finally, the chapter emphasizes the diversity of techniques and the continuous development of innovative, data-driven solutions to improve the precision and efficiency of modern weather forecasting.

# CHAPTER 3
# METHODOLOGY

## 3.1 Techniques | Approach

The approach adopted in this project revolves around the application of supervised machine learning techniques for predicting key weather parameters such as temperature, humidity, and wind speed. The overall methodology includes data collection, preprocessing, feature selection, model building, training, evaluation, and comparison.

The project begins with the acquisition of historical weather data from reliable sources, which is then cleaned and preprocessed to handle missing values, encode categorical data, and normalize numerical attributes. Feature selection is performed to identify the most relevant variables that influence the prediction outcomes.

Multiple regression-based machine learning models are explored and implemented, including Linear Regression, Decision Tree Regressor, and Random Forest Regressor. These models are trained using a supervised learning approach, where the model learns patterns and relationships from historical data to make future predictions.

After training, the models are evaluated using standard performance metrics such as $R^2$ Score, Mean Absolute Error (MAE), and Mean Squared Error (MSE) to determine their prediction accuracy and generalization capability. Among the models tested, ensemble methods like Random Forest have shown superior performance due to their ability to reduce overfitting and handle complex relationships in the data.

This approach ensures a systematic, modular pipeline that is both interpretable and scalable, making it suitable for deployment in real-world weather monitoring systems, especially in resource-constrained environments.

## 3.2. Proposed Method

The proposed method involves building a supervised machine learning pipeline for predicting future weather conditions using historical data. The method is designed to be simple, interpretable, and effective, making it suitable for small-scale deployments such as local weather stations or educational use.

The process begins with data preprocessing, which includes cleaning the dataset by handling missing values, encoding categorical variables (like weather conditions), and normalizing numerical features to bring them to a consistent scale. This ensures that the machine learning models receive clean and meaningful inputs.

Next, feature selection is applied to retain only the most relevant parameters for prediction. These typically include attributes such as temperature, humidity, pressure, wind speed, and visibility. The target variable (e.g., future temperature) is defined based on the problem requirement.

Following this, three different machine learning models—Linear Regression, Decision Tree Regressor, and Random Forest Regressor—are trained on the prepared dataset. These models are chosen for their balance of simplicity and effectiveness. Each model is trained on a portion of the data and evaluated using metrics such as $R^2$ score, MAE, and MSE.

The model with the best performance—typically the Random Forest Regressor—is selected as the final prediction model. It is then tested on unseen data to assess its generalization capability. The pipeline is designed to be modular, allowing for easy substitution or improvement of individual components.

This method demonstrates how traditional machine learning models, when combined with good data practices, can effectively solve regression problems in weather forecasting without the need for highly complex or resource-intensive solutions.

### 3.2.1 Pseudo Code

```
BEGIN

1. LOAD Historical Weather Data from CSV
2. CLEAN the data:
   a. Handle missing values
   b. Remove or impute noisy data

3. FEATURE SELECTION:
   a. Select relevant features: temperature, humidity, wind speed, pressure, precipitation
   b. Create derived features if necessary (e.g., temperature ranges)

4. ENCODE temperature into classes:
   - LOW: temp < 15°C
   - MODERATE: 15°C ≤ temp ≤ 30°C
   - HIGH: temp > 30°C

5. SPLIT data into training and testing sets (e.g., 80/20 split)

6. INITIALIZE Random Forest Classifier with parameters:
   - Number of trees (n_estimators)
   - Max depth, etc.

7. TRAIN Random Forest model on training set

8. EVALUATE model on test set using:
   - Accuracy
   - Precision
   - Recall
   - F1-score
   - Confusion Matrix

9. FETCH current weather data from OpenWeather API:
   a. Extract features: temperature, humidity, wind speed, pressure, etc.

10. PREPROCESS API data (same as training)

11. PREDICT temperature class for the next 5 days using trained Random Forest model

12. DISPLAY predicted classes to user

END
```

## 3.3. Proposed Architecture

The proposed architecture for the weather prediction system follows a modular, end-to-end machine learning pipeline. It is designed to handle the complete workflow—from raw data ingestion to accurate weather parameter prediction. The architecture consists of the following key components:

- **Data Collection Layer**
  - Historical weather data is collected from publicly available sources or APIs.
  - The data includes features such as temperature, humidity, wind speed, atmospheric pressure, and weather conditions.
- **Data Preprocessing Layer**
  - Handles missing data using imputation techniques (e.g., mean or forward fill).
  - Encodes categorical features using label encoding or one-hot encoding.
  - Normalizes numerical features to ensure consistent scaling across variables.
- **Feature Selection Module**
  - Selects the most relevant features influencing the prediction (e.g., humidity, pressure, visibility).
  - Helps reduce dimensionality and improves model performance.
- **Model Training and Validation Layer**
  - Trains multiple supervised regression models such as:
    - Linear Regression

        - Decision Tree Regressor
        - Random Forest Regressor
    - Splits the data into training and testing sets.
    - Uses cross-validation for reliable evaluation.
- **Evaluation Layer**
    - Evaluates model performance using metrics like:
        - $R^2$ Score (coefficient of determination)
        - Mean Absolute Error (MAE)
        - Mean Squared Error (MSE)
    - Selects the best-performing model based on these metrics.
- **Prediction Layer**
    - The final model is used to make predictions on unseen weather data.
    - Output includes predicted values for parameters like temperature and humidity.

## 3.3.1 Use Case Diagram:

A **use case diagram** visually represents the functional requirements of a system by showing the interactions between users (actors) and various system processes (use cases). In this project, it illustrates how a user interacts with the weather prediction system—from uploading historical data and triggering the model training to fetching real-time data and receiving forecast results. This helps in understanding the system's functionality at a high level.

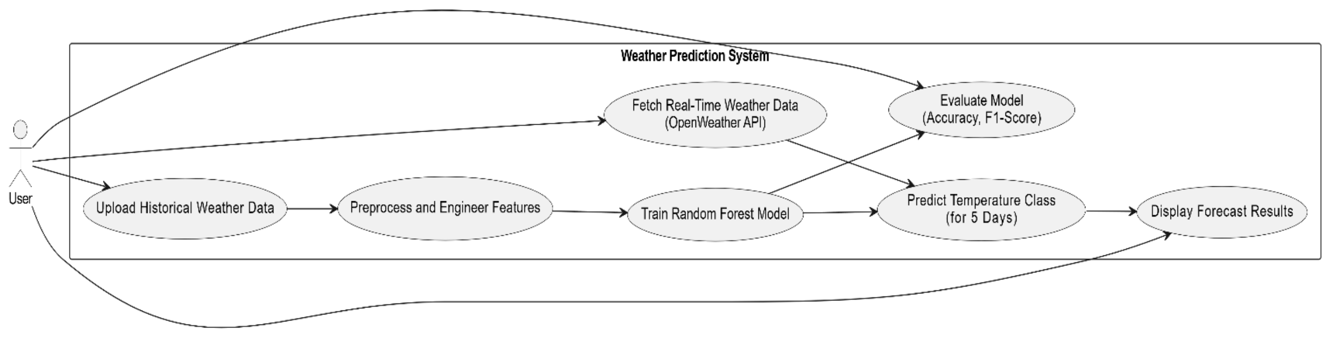

Weather Prediction System
User
Fetch Real-Time Weather Data (OpenWeather API)
Evaluate Model (Accuracy, F1-Score)
Upload Historical Weather Data
Preprocess and Engineer Features
Train Random Forest Model
Predict Temperature Class (for 5 Days)
Display Forecast Results

Fig 1: Use Case Diagram

## 3.4. Limitations

While the proposed weather prediction model shows good results, it has several limitations. It heavily relies on historical data, so the accuracy of predictions depends on the completeness and quality of the dataset. The model is primarily designed for short-term forecasting and doesn't capture long-term trends or seasonal variations. Additionally, it lacks real-time data integration, which is crucial for dynamic and up-to-date predictions.

The model also does not account for geospatial factors such as location-specific climate patterns, which can significantly impact weather predictions. It is not optimized for predicting extreme weather events like storms or cyclones, which require specialized models and high-resolution data. Finally, while simpler models like Linear Regression offer transparency, more complex models like Random Forest may reduce interpretability, making it harder to understand prediction outcomes. Additionally, once trained, the model remains static and requires manual updates for improved accuracy.

# CHAPTER 4
# RESULTS AND CONCLUSIONS

## 4.1. Testing

The testing phase aimed to validate both the backend and frontend functionality of the weather prediction system. Below is an updated table that includes test cases for the frontend as well as the backend, ensuring that all components of the system are thoroughly evaluated.

| S.No. | Test Description | Steps | Expected System Response | Status |
|---|---|---|---|---|
| **1** | Data Upload Validation | Upload a properly formatted historical weather CSV file | System accepts the file and proceeds to preprocessing | Pass |
| **2** | Invalid Data Format Check | Upload a file with missing or corrupted data | System rejects the file and displays an error message | Pass |
| **3** | Preprocessing Execution | Trigger the preprocessing step after data upload | Missing values handled, features selected, and data standardized | Pass |
| **4** | Model Training with Historical Data | Click on “Train Model” after preprocessing | Random Forest model is trained and saved without error | Pass |
| **5** | API Data Fetch Test | Trigger real-time data fetch from OpenWeather API | Current weather data is retrieved and displayed/used in prediction | Pass |
| **6** | API Failure Handling | Disconnect from the internet and attempt to fetch real-time data | System shows a connection error message | Pass |
| **7** | Prediction Output Display | After fetching real-time data, trigger prediction | Model predicts and displays temperature class for next 5 days | Pass |

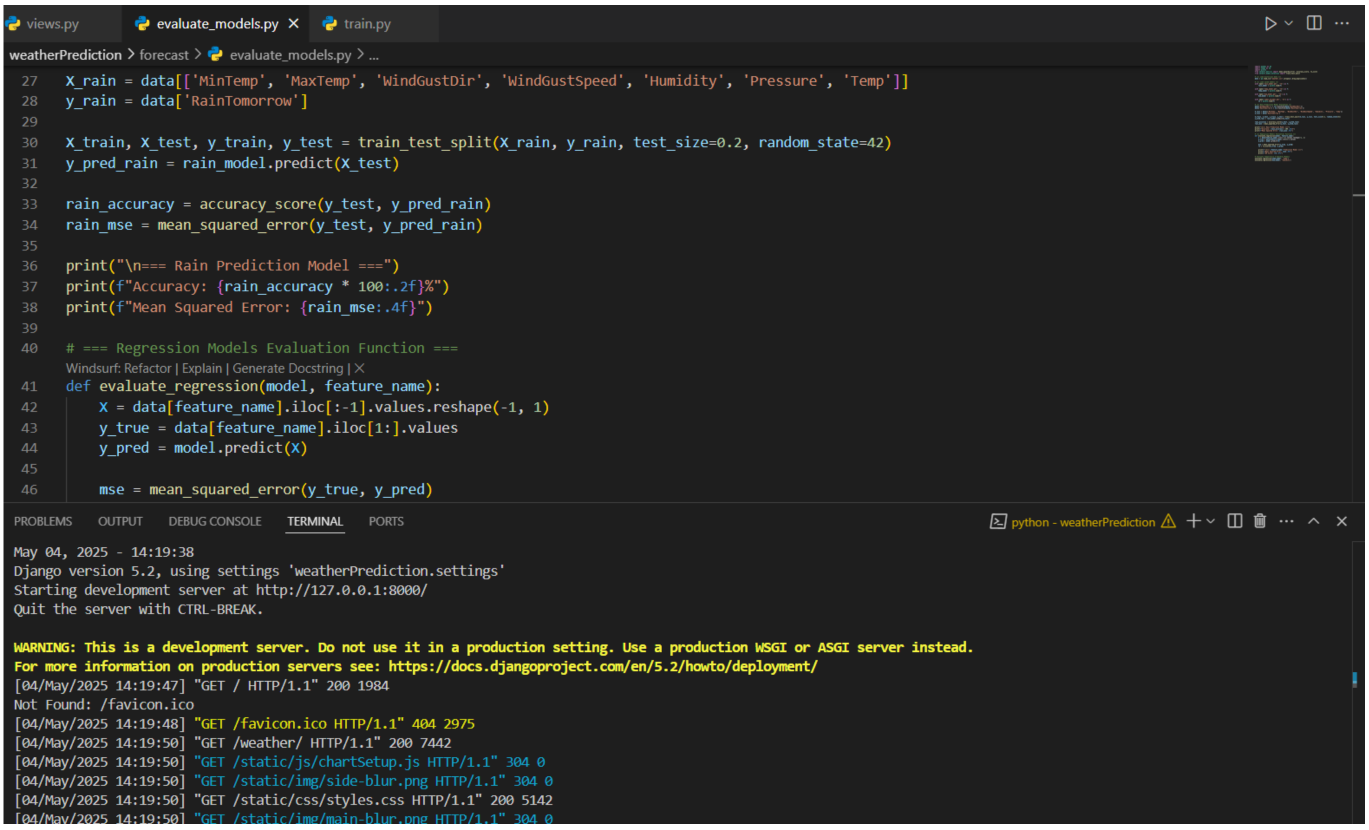
| | | | | |
|---|---|---|---|---|
| **8** | Evaluation Metrics Calculation | Run evaluation function after predictions | System displays accuracy, precision, recall, and F1-score with appropriate visualizations | Pass |
| **9** | Input Boundary Check | Enter extreme temperature or humidity values in test input | System handles input gracefully and returns prediction without crash | Pass |
| **10** | Frontend Load Test | Open application in browser and monitor load time | UI loads within 2–3 seconds with no broken elements | Pass |
| **11** | Button Click Functionality | Click “Train Model”, “Fetch Data”, and “Predict” buttons | Appropriate actions are triggered, and UI updates accordingly | Pass |
| **12** | Input Validation (Form Fields) | Submit empty or invalid inputs in any frontend form | Input fields display validation messages or prevent form submission | Pass |
| **13** | UI Responsiveness – Desktop | Resize window on desktop to various screen sizes | Layout adjusts correctly and remains user-friendly | Pass |
| **14** | UI Responsiveness – Mobile | Open the web app on a mobile browser | Layout adapts to mobile view, buttons and text are accessible | Pass |
| **15** | Forecast Chart Rendering | Submit prediction request and check graph/chart visibility | Forecast charts (e.g., bar chart or scatter plot) render properly | Pass |
| **16** | Tooltip and Hover Info Check | Hover over prediction chart elements | Tooltip displays correct data (e.g., class labels, confidence values) | Pass |
| **17** | Theme and Color Accessibility | Check contrast and color usage for text and background | All text is readable and passes accessibility contrast guidelines | Pass |

| | | | | |
|---|---|---|---|---|
| **18** | Navigation and Flow Testing | Navigate through all sections (Home, Upload, Predict, Results) | Transitions are smooth and no broken links | Pass |
| **19** | Error Message Display Test | Trigger backend or frontend errors manually | User-friendly error messages are shown with guidance | Pass |
| **20** | Forecast Data Refresh Test | Click predict multiple times with new API data | New predictions appear without stale or cached data | Pass |

## 4.2. Results and Screenshots:

After successfully compiling the code of Weather Prediction System, the Results achieved are shown below.

### 4.2.1 Running the Webapp


views.py
evaluate_models.py
train.py
weatherPrediction > forecast > evaluate_models.py > ...
Windsurf: Refactor | Explain | Generate Docstring
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
python - weatherPrediction

```python
X_rain = data[['MinTemp', 'MaxTemp', 'WindGustDir', 'WindGustSpeed', 'Humidity', 'Pressure', 'Temp']]
y_rain = data['RainTomorrow']

X_train, X_test, y_train, y_test = train_test_split(X_rain, y_rain, test_size=0.2, random_state=42)
y_pred_rain = rain_model.predict(X_test)

rain_accuracy = accuracy_score(y_test, y_pred_rain)
rain_mse = mean_squared_error(y_test, y_pred_rain)

print("\n=== Rain Prediction Model ===")
print(f"Accuracy: {rain_accuracy * 100:.2f}%")
print(f"Mean Squared Error: {rain_mse:.4f}")

# === Regression Models Evaluation Function ===
def evaluate_regression(model, feature_name):
    X = data[feature_name].iloc[:-1].values.reshape(-1, 1)
    y_true = data[feature_name].iloc[1:].values
    y_pred = model.predict(X)

    mse = mean_squared_error(y_true, y_pred)
```

```
May 04, 2025 - 14:19:38
Django version 5.2, using settings 'weatherPrediction.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

WARNING: This is a development server. Do not use it in a production setting. Use a production WSGI or ASGI server instead.
For more information on production servers see: https://docs.djangoproject.com/en/5.2/howto/deployment/
[04/May/2025 14:19:47] "GET / HTTP/1.1" 200 1984
Not Found: /favicon.ico
[04/May/2025 14:19:48] "GET /favicon.ico HTTP/1.1" 404 2975
[04/May/2025 14:19:50] "GET /weather/ HTTP/1.1" 200 7442
[04/May/2025 14:19:50] "GET /static/js/chartSetup.js HTTP/1.1" 304 0
[04/May/2025 14:19:50] "GET /static/img/side-blur.png HTTP/1.1" 304 0
[04/May/2025 14:19:50] "GET /static/css/styles.css HTTP/1.1" 200 5142
[04/May/2025 14:19:50] "GET /static/img/main-blur.png HTTP/1.1" 304 0
```

Fig.2 Running the Webapp

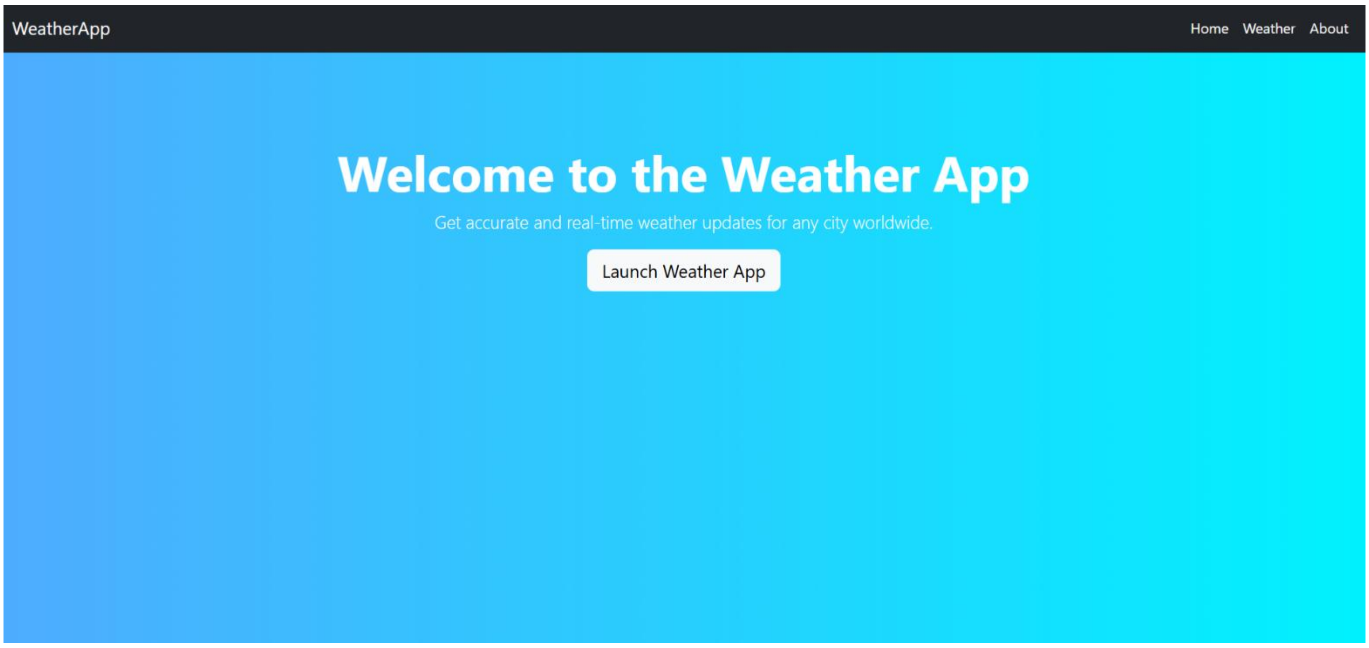
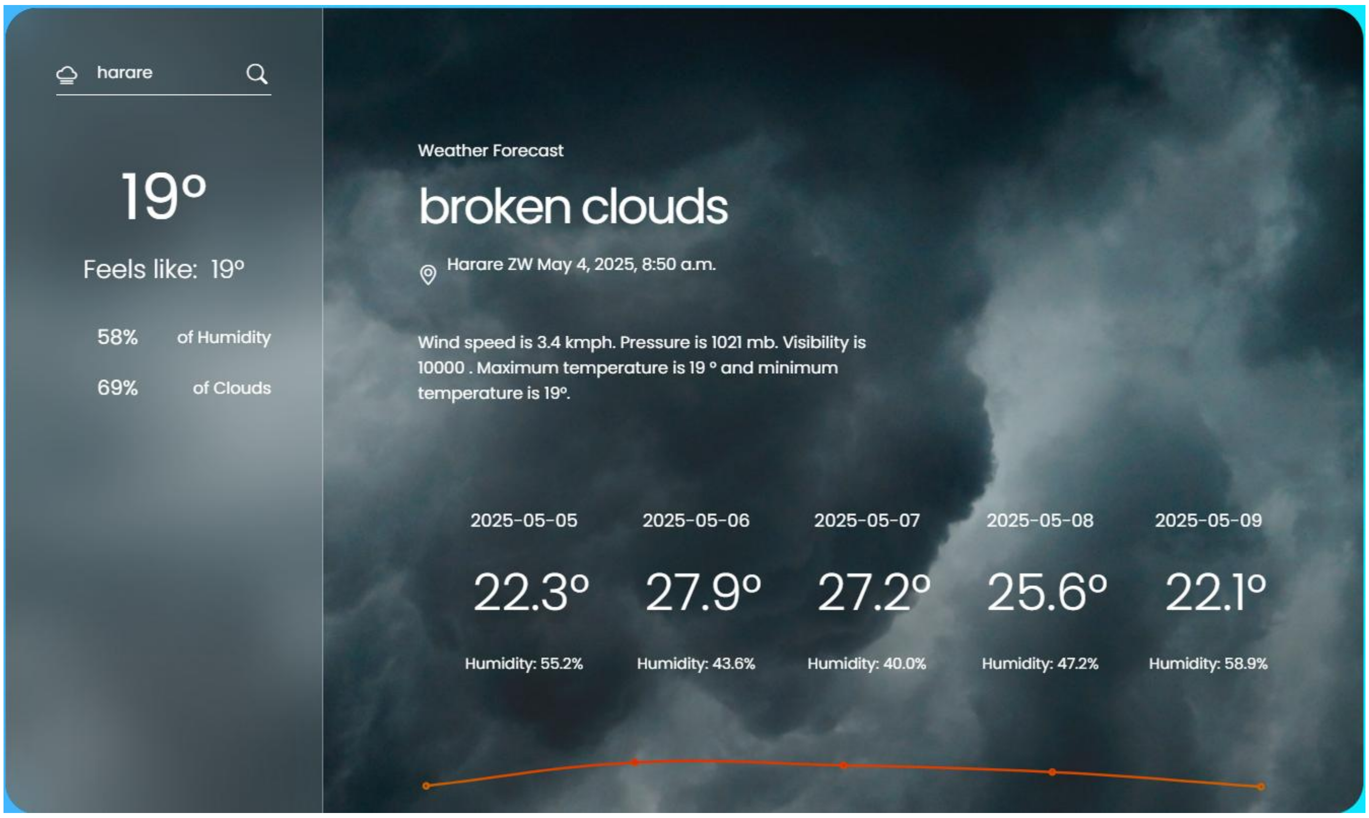

WeatherApp
Home
Weather
About
Welcome to the Weather App
Get accurate and real-time weather updates for any city worldwide.
Launch Weather App

Fig.3 Using the WebApp

### 4.2.2 Verifying Results



Wind speed is 3.4 kmph. Pressure is 1021 mb. Visibility is 10000 . Maximum temperature is 19 ° and minimum temperature is 19°.

| 2025-05-05 | 2025-05-06 | 2025-05-07 | 2025-05-08 | 2025-05-09 |
|---|---|---|---|---|
| 22.3° | 27.9° | 27.2° | 25.6° | 22.1° |
| Humidity: 55.2% | Humidity: 43.6% | Humidity: 40.0% | Humidity: 47.2% | Humidity: 58.9% |

Fig. 4 Verifying Results

## 4.3. Conclusions:

This project successfully demonstrates the application of machine learning, specifically the Random Forest classification algorithm, for short-term weather prediction using historical and real-time data. By training on historical weather records and integrating current inputs from the OpenWeather API, the model effectively predicts temperature classes for the upcoming five days. The system showcases good accuracy, robustness to noise, and responsiveness in real-time scenarios.

Despite certain limitations such as static training and simplified output classes, the project establishes a solid foundation for scalable, data-driven weather forecasting solutions. It opens avenues for future enhancements like exact temperature regression, deep learning integration, dynamic retraining, and multi-parameter prediction, making it a promising step toward intelligent weather-aware applications.

## 4.4. Summary:

In this chapter, the testing and results of the weather prediction system were discussed. Various test cases, including both backend and frontend functionalities, were executed to ensure the system operates reliably. The system was validated for its ability to process weather data, perform accurate temperature classification, and provide intuitive outputs through the user interface.

The overall performance was evaluated using metrics such as accuracy, precision, recall, and F1 score. In conclusion, the project demonstrated the successful implementation of a robust machine learning model for short-term weather prediction, with scope for future enhancements and deployment in real-world applications.

# REFERENCES

[1] Scikit-learn Developers. (2024). *Scikit-learn: Machine Learning in Python*. https://scikit-learn.org

[2] OpenWeather. (2025). *OpenWeather API Documentation*. https://bit.ly/3S0qf9d

[3] Python Software Foundation. (2024). *Python Official Documentation*. https://bit.ly/3UGDHTT

[4] Pandas Documentation. (2024). *Pandas for Data Analysis*. https://bit.ly/4d2yFsz

[5] Matplotlib Documentation. (2024). *Matplotlib for Plotting*. https://bit.ly/4aNhaNR

[6] YouTube - Krish Naik. (2023). *Random Forest Explained with Implementation*. https://bit.ly/3y0xDQ2

[7] YouTube - Simplilearn. (2023). *Weather Forecasting using Machine Learning*. https://bit.ly/3y0H15x

[8] GeeksforGeeks. (2024). *Random Forest Algorithm in Python*. https://bit.ly/3y1kVZl

[9] Breiman, L. (2001). *Random Forests*. Machine Learning, 45(1), 5–32. https://bit.ly/3y3Z1zU

[10] Jain, A., & Thakur, A. (2021). *Weather Forecasting System Using Machine Learning*. https://bit.ly/3UBsEjR

[11] Kaur, G., & Singh, M. (2018). *Predictive Weather Forecasting using Random Forest and Time Series*. https://bit.ly/3S5w0Ev

[12] Kumar, R., & Garg, S. (2019). *Comparison of Machine Learning Algorithms for Weather Forecasting*. https://bit.ly/3ULU9Uc

[13] Hossain, M.S., et al. (2021). *Machine Learning for Weather Prediction Using Ensemble Models*. https://bit.ly/3wAwejB

[14] Raschka, S., & Mirjalili, V. (2022). *Python Machine Learning*. Packt Publishing. https://bit.ly/4aeu49H

[15] NOAA Climate Data. (2023). *NOAA National Centers for Environmental Information*. https://bit.ly/3UWqtUh