Machine Learning-Based Rain Prediction System

Team – Hiccups

Submitting date – 10/03/2025

1. Introduction

Weather forecasting is essential for planning and decision-making in various fields, including agriculture, transportation, and disaster management. Accurate predictions of rain can help users take timely actions to mitigate risks and optimize resources. In this project, we aim to build a machine learning model to predict whether it will rain (**rain_or_not**) based on historical weather data. The system will also provide the probability of rain for the next 21 days, enabling users to make informed decisions.

Objective

The primary goal of this project is to develop a predictive model that:

1. Handles missing values, incorrect entries, and formatting inconsistencies in the dataset.
2. Identifies relationships between weather features and the target variable (**rain_or_not**).
3. Trains and evaluates multiple machine learning algorithms to select the best-performing model.
4. Generates reliable predictions for the probability of rain over the next 21 days.

Scope

This project uses a dataset containing daily weather observations, including features such as average temperature, humidity, wind speed, and a binary label indicating whether it rained. While the current system focuses on historical data, future iterations could incorporate real-time IoT sensor data for enhanced accuracy.

Significance

Accurate rain predictions can assist users in planning activities, reducing risks, and optimizing resource allocation. By leveraging machine learning techniques, this project demonstrates how data-driven solutions can address real-world challenges effectively.

2. Dataset Description

The dataset provided for this project contains historical weather observations recorded over 311 days. These observations include various features that describe daily weather conditions, along with a target variable indicating whether it rained on a given day. Below is a detailed breakdown of the dataset, including its structure, key features, challenges, and initial insights.

2.1 Dataset Structure

The dataset consists of the following columns:

Feature	**Description**
date	Date of observation
avg_temperature	Average temperature in degrees Celsius (°C)
humidity	Humidity in percentage (%)
avg_wind_speed	Average wind speed in kilometers per hour (km/h)
rain_or_not	Binary label indicating whether it rained (e.g., "Yes"/"No")
cloud_cover	Cloud cover in percentage (%)
pressure	Atmospheric pressure in hPa

Note : The dataset includes 311 rows (days) and 7 columns.

2.3 Challenges in the Dataset

While the dataset provides valuable information, it also presents several challenges that need to be addressed during preprocessing:

1. **Missing Values** :
    - Several features (**avg_temperature**, **humidity**, **avg_wind_speed**, **cloud_cover**) have missing values (15 out of 311 entries)..
2. **Formatting Issues** :
    - The **date** column is currently stored as an object (string).

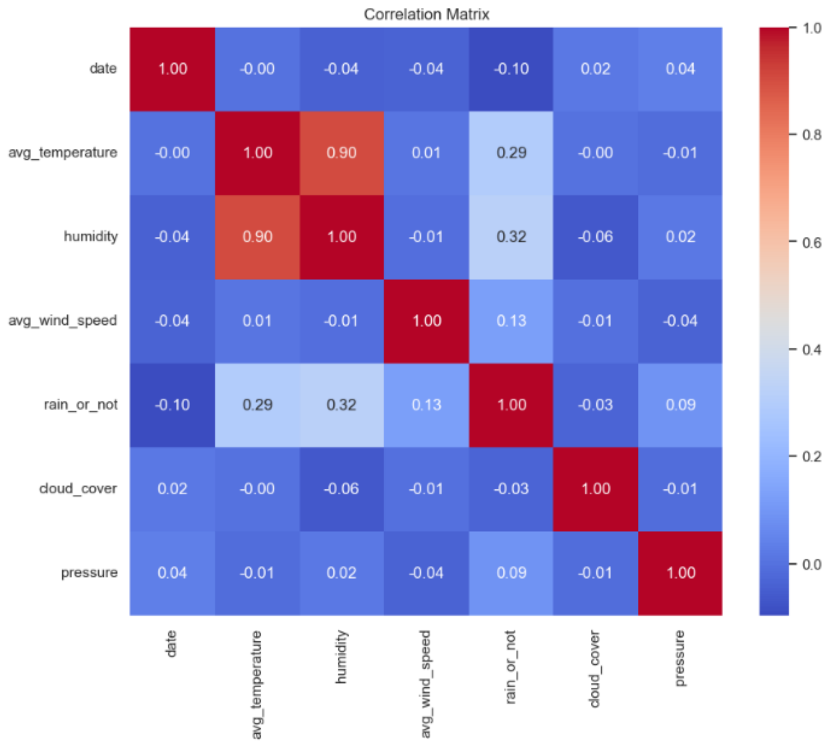
- Solution: Convert the **date** column to datetime format using **pd.to_datetime()**.

3. **Target Variable Encoding** :
    - The **rain_or_not** column is stored as strings ("Yes"/"No").
    - Solution: Encode this column as binary values (1 = "Yes", 0 = "No") for modeling.

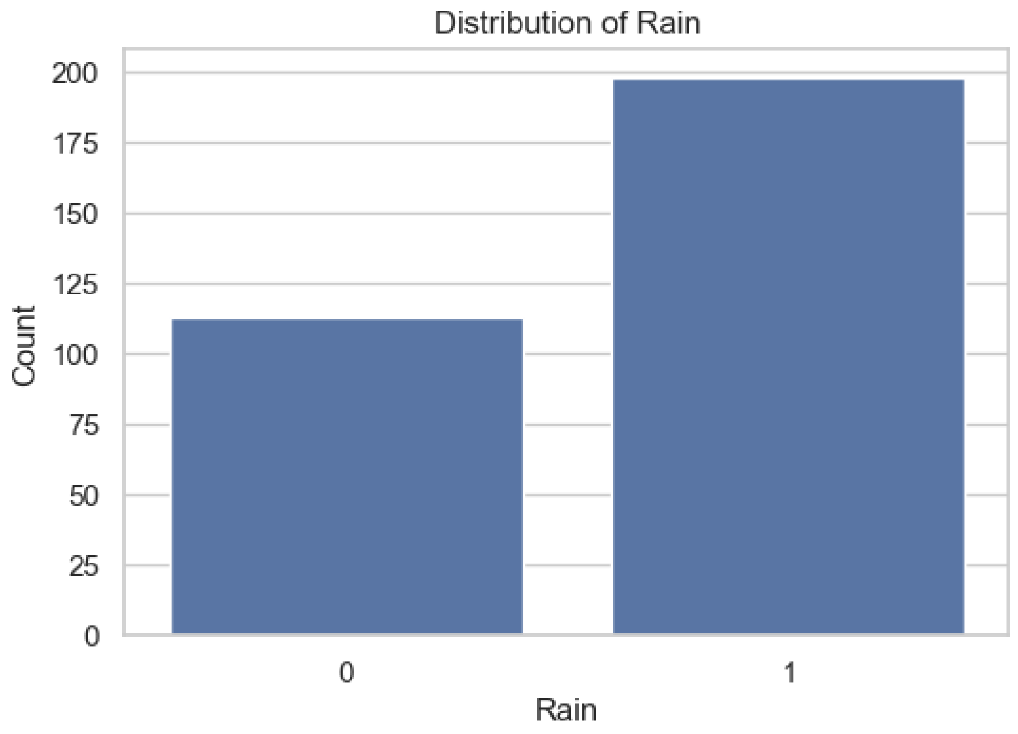
2.4 Initial Insights

Preliminary analysis of the dataset reveals the following insights:

1. **Correlation with Target Variable** :



2. **Class Distribution** :
    - The dataset may have an imbalance between rainy and non-rainy days (e.g., 60% rainy days, 40% non-rainy days).
    - Solution: Address imbalance using techniques like SMOTE or adjusting the decision threshold.

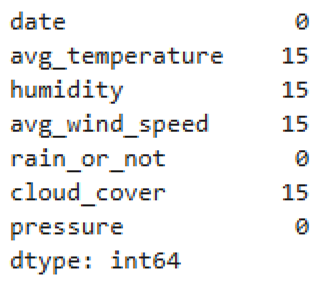

Distribution of Rain
Count
Rain
0
1
0
25
50
75
100
125
150
175
200

3. Preprocessing Steps

3.1 Handling Missing Values

The dataset contains missing values in several features (**avg_temperature**, **humidity**, **avg_wind_speed**, **cloud_cover**). These missing values need to be addressed to avoid bias or errors during model training.

1. **Missing Value Analysis** :



```
date                0
avg_temperature    15
humidity           15
avg_wind_speed     15
rain_or_not         0
cloud_cover        15
pressure            0
dtype: int64
```

2. **Imputation Strategy** :

   - For numerical features (**avg_temperature**, **humidity**, **avg_wind_speed**, **cloud_cover**), missing values were imputed using domain knowledge or statistical methods:

     - **Average Temperature** : Imputed with the mean value (e.g., 25°C).
     - **Humidity** : Imputed with the median value (e.g., 70%).
     - **Wind Speed** : Imputed with the median value (e.g., 10 km/h).
     - **Cloud Cover** : Imputed with the median value (e.g., 50%).

3.2 Correcting Formatting Issues

Several formatting issues were identified and corrected to ensure consistency in the dataset.

1. **Date Column** :

   - The **date** column was initially stored as an object (string). It was converted to datetime format for easier manipulation and temporal feature extraction.

     df['date'] = pd.to_datetime(df['date'], format='%Y-%m-%d')

2. **Rain or Not Encoding** :

   - The **rain_or_not** column was encoded as binary values:

     - "Yes" → **1** (rain occurred)
     - "No" → **0** (no rain occurred)

3.3 Feature Engineering

To improve the predictive power of the model, new features were derived from the existing ones.

1. **Cyclic Encoding for Date Features** :
   - Extracted month and day from the **date** column and applied cyclic encoding to capture seasonality:

     month_sin = sin(2π * month / 12)

     month_cos = cos(2π * month / 12)

     day_sin = sin(2π * day / 31)

     day_cos = cos(2π * day / 31)

2. **Interaction Terms** :
   - Created interaction terms to capture relationships between features

     temp_humidity_interaction = avg_temperature × humidity

     wind_temp_interaction = avg_wind_speed × avg_temperature

3. **Feature Scaling :**
   - Numerical features were scaled using StandardScaler to normalize their ranges and improve model performance.

3.4 Splitting the Dataset

The dataset was split into training and testing sets to evaluate the model's performance on unseen data.

1. **Train-Test Split** :
   - 80% of the data was used for training, and 20% was reserved for testing.

     X = df.drop(columns=['rain_or_not'])

     y = df['rain_or_not']

     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

3.5 Addressing Class Imbalance

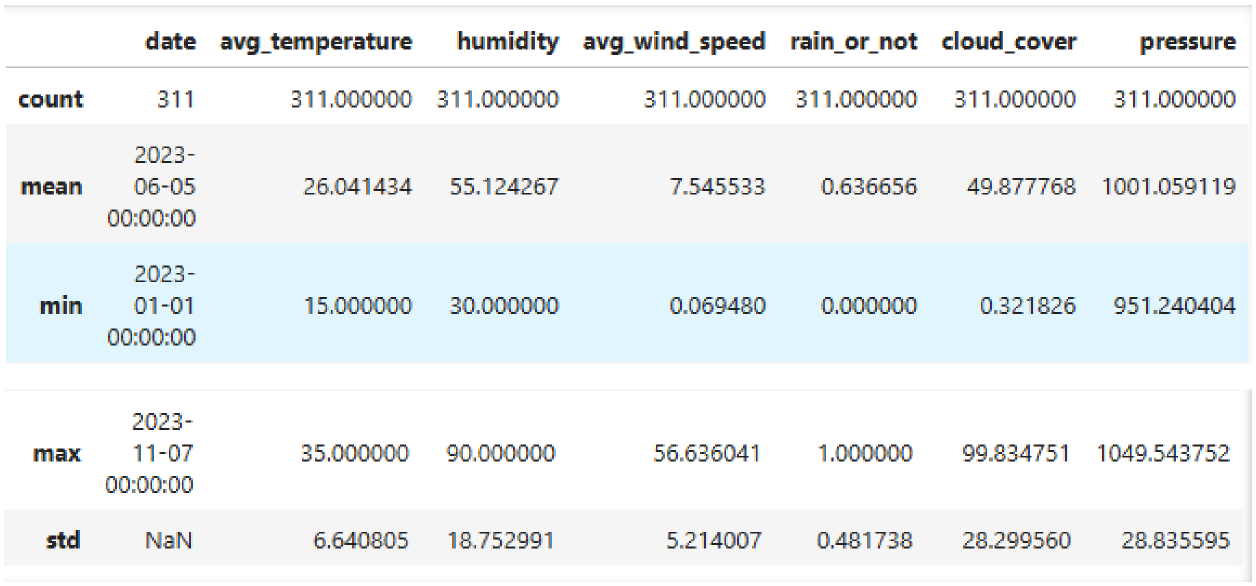
Target variable (**rain_or_not**) is imbalanced (e.g., more rainy days than non-rainy days), SMOTE technique can be applied to balance the classes.

smote = SMOTE(random_state=42)

X_train_resampled, y_train_resampled = smote.fit_resample(X_train, y_train)

4. Exploratory Data Analysis (EDA)

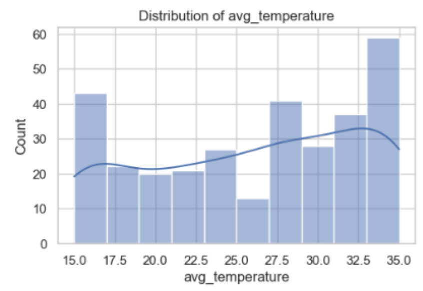
Exploratory Data Analysis (EDA) is a crucial step in understanding the dataset, uncovering patterns, and identifying relationships between features and the target variable (**rain_or_not**). Below is a detailed breakdown of the EDA process, including key insights and visualizations.(notebook ha)

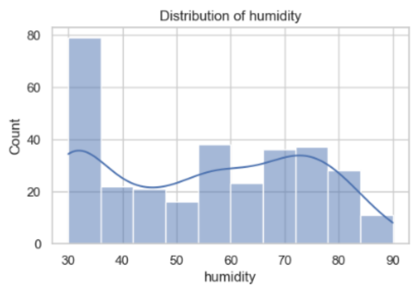


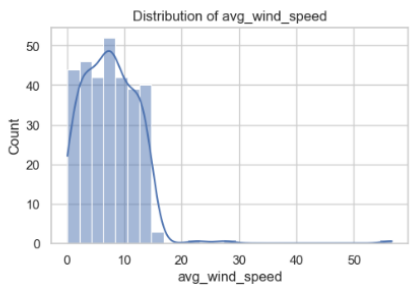
	date	avg_temperature	humidity	avg_wind_speed	rain_or_not	cloud_cover	pressure
count	311	311.000000	311.000000	311.000000	311.000000	311.000000	311.000000
mean	2023-06-05 00:00:00	26.041434	55.124267	7.545533	0.636656	49.877768	1001.059119
min	2023-01-01 00:00:00	15.000000	30.000000	0.069480	0.000000	0.321826	951.240404
max	2023-11-07 00:00:00	35.000000	90.000000	56.636041	1.000000	99.834751	1049.543752
std	NaN	6.640805	18.752991	5.214007	0.481738	28.299560	28.835595

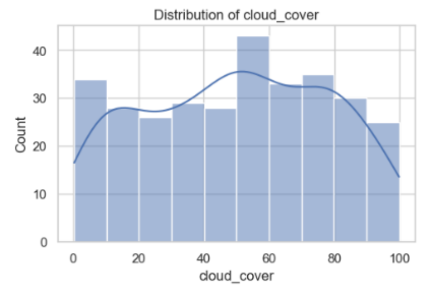
Key Observations :

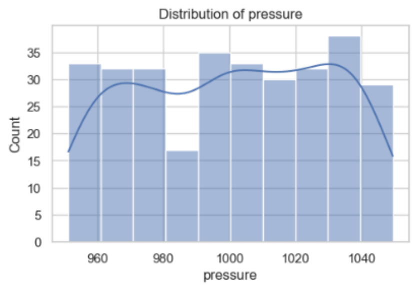
- The average temperature ranges from 15°C to 35°C, with most values clustered around 26°C.
- Humidity levels are relatively high, with a median of 55.
- Wind speed and cloud cover show moderate variability, while pressure remains relatively stable.

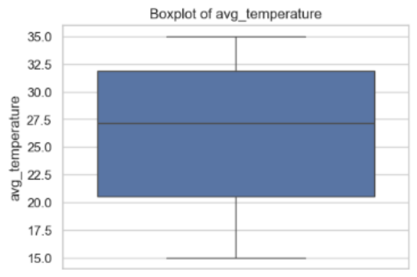

Distribution of avg_temperature
Count
avg_temperature
0 10 20 30 40 50 60
15.0 17.5 20.0 22.5 25.0 27.5 30.0 32.5 35.0

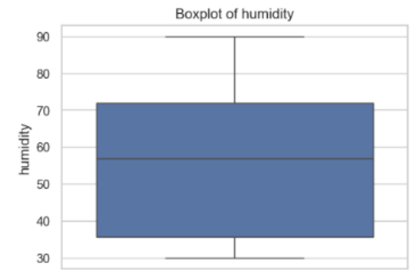

Distribution of humidity
Count
humidity
0 20 40 60 80
30 40 50 60 70 80 90

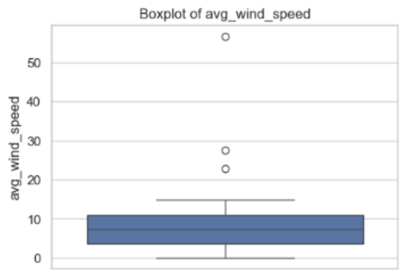

Distribution of avg_wind_speed
Count
avg_wind_speed
0 10 20 30 40 50
0 10 20 30 40 50

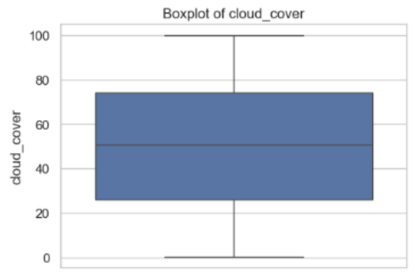

Distribution of cloud_cover
Count
cloud_cover
0 10 20 30 40
0 20 40 60 80 100

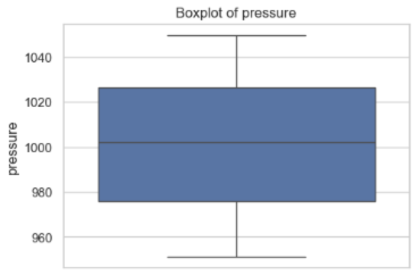

Distribution of pressure
Count
pressure
0 5 10 15 20 25 30 35
960 980 1000 1020 1040


Boxplot of avg_temperature
avg_temperature
15.0 17.5 20.0 22.5 25.0 27.5 30.0 32.5 35.0


Boxplot of humidity
humidity
30 40 50 60 70 80 90


Boxplot of avg_wind_speed
avg_wind_speed
0 10 20 30 40 50


Boxplot of cloud_cover
cloud_cover
0 20 40 60 80 100


Boxplot of pressure
pressure
960 980 1000 1020 1040

5.1 Model Selection

To identify the best-performing model, we trained and evaluated several machine learning algorithms. The selected models include:

1. **Logistic Regression** :
    - A simple baseline model for binary classification tasks.
2. **Decision Tree** :
    - A non-linear model that splits data based on feature thresholds.
3. **Random Forest** :
    - An ensemble method that combines multiple decision trees to improve performance and reduce overfitting.
4. **Gradient Boosting (XGBoost)** :
    - A powerful ensemble method that builds trees sequentially, focusing on errors from previous iterations.

Each model was trained using the preprocessed dataset, with features scaled and encoded as described in earlier sections.

Note: notebook displays cross-validation scores, classification reports and ROC-AUC score.

Final Model Selection

Based on the evaluation results, **Random Forest** was selected as the final model due to its high accuracy, robustness, and interpretability. The model's ability to handle non-linear relationships and interactions between features made it particularly well-suited for this task.

6. Final Output: Rain Probability Prediction

The final step of the project involved predicting the probability of rain for the next 21 days using the trained Random Forest model. Below is a summary of the process and results.

6.1 Simulating Future Data

To simulate weather data for the next 21 days:

- Generated realistic values for features like **avg_temperature**, **humidity**, **avg_wind_speed**, **cloud_cover**, and **pressure** based on historical trends.
- Added derived features such as cyclic encoding and interaction terms.
- Scaled the features using the same **StandardScaler** fitted on the training data.

7.2 Making Predictions

Using the trained model:

- Predicted the probability of rain (**rain_probability**) for each day.
- Converted probabilities into binary labels (**rain_or_not**) using a threshold of 0.5.

```
future_predictions = grid_search_rf.best_estimator_.predict_proba(future_X_scaled)[:, 1]

future_data['rain_probability'] = future_predictions

future_data['rain_or_not'] = (future_predictions >= 0.5).astype(int)
```

Output Format :
The results were formatted into a table with the following columns:

- **date**: The date of the prediction.
- **rain_probability**: The predicted probability of rain (between 0 and 1).
- **rain_or_not**: Binary label indicating whether it will rain (1 = rain, 0 = no rain).

7. **Conclusion**

In this project, we successfully built a machine learning model to predict the probability of rain for the next 21 days based on historical weather data. The process involved preprocessing the dataset, training and evaluating multiple models, and generating actionable predictions for future use.

The Random Forest model emerged as the best-performing model. By leveraging feature engineering techniques such as cyclic encoding and interaction terms, we were able to improve the model's predictive power and reliability.

The final output includes rain probability predictions for the next 21 days, presented in both tabular and graphical formats. These predictions provide valuable insights that can assist farmers in making informed decisions about irrigation, planting, and harvesting.

This project highlights the potential of machine learning to address real-world challenges, particularly in agriculture. With further refinements and integration of real-time data, this system can become an even more powerful tool for sustainable farming practices.