Weather Forecasting Using Time Series Models

PM Accelerator Mission:

The **PM Accelerator** program is dedicated to providing **industry-leading tools and education** to individuals from diverse backgrounds, ensuring equal opportunities for future product management leaders. By offering **access to industry leaders, AI-driven product management skills, and a strong professional ecosystem**, the initiative empowers aspiring and experienced professionals to leverage **data-driven decision-making** in their respective fields.

---

1. Introduction:

a. Project Overview

Climate change and weather variability have made **temperature forecasting** a crucial field of study. This project aims to **analyze global weather data** and develop a reliable forecasting model using **time-series techniques**. The study covers:

- **Data Cleaning & Preprocessing** – Handling missing values, outlier detection, and data normalization.
- **Exploratory Data Analysis (EDA)** – Identifying key trends, correlations, and seasonal patterns.
- **Forecasting Model Implementation** – Using **ARIMA (AutoRegressive Integrated Moving Average)**.
- **Model Evaluation** – Assessing performance using **Mean Absolute Error (MAE) & Root Mean Squared Error (RMSE)**.
- **Insights & Conclusions**– Interpreting findings.

b. Objective of the Study

The primary objective is to **predict temperature trends** using time-series forecasting techniques and analyze the impact of various weather parameters.

c. Dataset Description

- **Total Records:** 50,085
- **Total Features:** 41
- **Time Period:** Continuous weather data

d. Key Variables in Dataset

- **Temperature (Celsius & Fahrenheit)** – Target variable for forecasting.
- **Precipitation (mm & inches)** – Measures rainfall at different locations.
- **Wind Speed (mph & kph)** – Affects temperature variations.
- **Humidity & Air Quality** – Indicators of climate conditions.
- **Timestamps** – Essential for time-series analysis.

---

2. Data Cleaning and Preprocessing:

a. Handling Missing Values:

- The dataset was **analyzed for missing values**, and **no missing data** was detected.
- Therefore, **no imputation was necessary** for data integrity.

```
1 df.isnull().sum()
```

```
country                          0
location_name                    0
latitude                         0
longitude                        0
timezone                         0
last_updated_epoch               0
last_updated                     0
temperature_celsius              0
temperature_fahrenheit           0
condition_text                   0
wind_mph                         0
wind_kph                         0
wind_degree                      0
wind_direction                   0
pressure_mb                      0
pressure_in                      0
precip_mm                        0
precip_in                        0
humidity                         0
cloud                            0
feels_like_celsius               0
feels_like_fahrenheit            0
visibility_km                    0
visibility_miles                 0
uv_index                         0
gust_mph                         0
gust_kph                         0
air_quality_Carbon_Monoxide      0
air_quality_Ozone                0
air_quality_Nitrogen_dioxide     0
air_quality_Sulphur_dioxide      0
air_quality_PM2.5                0
air_quality_PM10                 0
air_quality_us-epa-index         0
air_quality_gb-defra-index       0
sunrise                          0
sunset                           0
moonrise                         0
moonset                          0
moon_phase                       0
moon_illumination                0
dtype: int64
```

b. Outlier Detection:

To ensure **data reliability**, outliers were identified using the **Interquartile Range (IQR) method**.

Interquartile Range (IQR) Method

- The IQR method detects outliers by calculating:

$$Q_1 = 25th\ Percentile \qquad Q_3 = 75th\ Percentile$$

$$IQR = Q_3 - Q_1$$

$$Lower\ Bound = Q_1 - 1.5 * IQR \qquad Upper\ Bound = Q_3 + 1.5 * IQR$$

- Any value **outside this range** was flagged as an **outlier**.

```python
# Detect outliers using IQR method
Q1 = df[numerical_cols].quantile(0.25)
Q3 = df[numerical_cols].quantile(0.75)
IQR = Q3 - Q1

# Define outliers as values outside of 1.5 * IQR range
outliers = ((df[numerical_cols] < (Q1 - 1.5 * IQR)) | (df[numerical_cols] > (Q3 + 1.5 * IQR))).sum()

outliers
```

```
latitude                         0
longitude                     3853
last_updated_epoch               0
temperature_celsius           1544
temperature_fahrenheit        1518
wind_mph                       543
wind_kph                       654
wind_degree                      0
pressure_mb                   3343
pressure_in                   3434
precip_mm                     9564
precip_in                     8043
humidity                         0
cloud                            0
feels_like_celsius            1312
feels_like_fahrenheit         1314
visibility_km                 9314
visibility_miles              9231
uv_index                         1
gust_mph                      1009
gust_kph                      1022
air_quality_Carbon_Monoxide   4602
air_quality_Ozone              980
air_quality_Nitrogen_dioxide  6538
air_quality_Sulphur_dioxide   7396
air_quality_PM2.5             4183
air_quality_PM10              5100
air_quality_us-epa-index      3519
air_quality_gb-defra-index    4879
moon_illumination                0
dtype: int64
```

c. Outlier Handling:

Two methods of Outlier Handling were tried out to determine which worked the best:

C.1. Winsorization

✔ Winsorization is a technique that **limits extreme values by replacing them with the nearest boundary value** at a certain percentile (e.g., **1st and 99th percentiles**).
✔ This method **retains the total number of observations** while **reducing the impact of extreme values**.

🔹 **How Winsorization Works:**

- If a value is **below the 1st percentile**, it is **replaced** with the **1st percentile value**.
- If a value is **above the 99th percentile**, it is **replaced** with the **99th percentile value**.

🔹 **Limitations of Winsorization:**

- **Some extreme values may still remain** because it only replaces values at a given percentile threshold.
- **Cannot guarantee complete removal of all outliers**, as some may fall just inside the defined percentile range.

Before handling the Outliers:

latitude	0
longitude	3853
last_updated_epoch	0
temperature_celsius	1544
temperature_fahrenheit	1518
wind_mph	543
wind_kph	654
wind_degree	0
pressure_mb	3343
pressure_in	3434
precip_mm	9564
precip_in	8043
humidity	0
cloud	0
feels_like_celsius	1312
feels_like_fahrenheit	1314
visibility_km	9314
visibility_miles	9231
uv_index	1
gust_mph	1009
gust_kph	1022
air_quality_Carbon_Monoxide	4602
air_quality_Ozone	980
air_quality_Nitrogen_dioxide	6538
air_quality_Sulphur_dioxide	7396
air_quality_PM2.5	4183
air_quality_PM10	5100
air_quality_us-epa-index	3519
air_quality_gb-defra-index	4879
moon_illumination	0
dtype: int64	

After handling the Outliers:

latitude	0
longitude	3853
last_updated_epoch	0
temperature_celsius	1451
temperature_fahrenheit	1438
wind_mph	543
wind_kph	654
wind_degree	0
pressure_mb	3343
pressure_in	3434
precip_mm	9564
precip_in	8043
humidity	0
cloud	0
feels_like_celsius	1308
feels_like_fahrenheit	1310
visibility_km	9314
visibility_miles	9231
uv_index	0
gust_mph	1009
gust_kph	1022
air_quality_Carbon_Monoxide	4601
air_quality_Ozone	980
air_quality_Nitrogen_dioxide	6538
air_quality_Sulphur_dioxide	7395
air_quality_PM2.5	4183
air_quality_PM10	5100
air_quality_us-epa-index	3519
air_quality_gb-defra-index	4879
moon_illumination	0
dtype: int64	

C.2. Clipping(Capping)

✔ Clipping (also known as **capping**) involves **replacing all values beyond a fixed threshold with the threshold itself**.
✔ Unlike Winsorization, **clipping strictly removes all outliers** beyond the set boundary.

🔹 **How Clipping Works:**

- If a value is **less than a lower threshold**, it is **replaced with the lower threshold**.
- If a value is **greater than an upper threshold**, it is **replaced with the upper threshold**.

🔹 **Advantages of Clipping:**

- **Ensures no extreme values remain** beyond the given range.
- **More effective than Winsorization** in handling extreme cases.

🔹 **Limitations of Clipping:**

- **May distort data distribution** if thresholds are not carefully chosen.
- **Can remove significant variations** that might be relevant to predictions.

Before dealing with Outliers

latitude	0
longitude	3853
last_updated_epoch	0
temperature_celsius	1544
temperature_fahrenheit	1518
wind_mph	543
wind_kph	654
wind_degree	0
pressure_mb	3343
pressure_in	3434
precip_mm	9564
precip_in	8043
humidity	0
cloud	0
feels_like_celsius	1312
feels_like_fahrenheit	1314
visibility_km	9314
visibility_miles	9231
uv_index	1
gust_mph	1009
gust_kph	1022
air_quality_Carbon_Monoxide	4602
air_quality_Ozone	980
air_quality_Nitrogen_dioxide	6538
air_quality_Sulphur_dioxide	7396
air_quality_PM2.5	4183
air_quality_PM10	5100
air_quality_us-epa-index	3519
air_quality_gb-defra-index	4879
moon_illumination	0
dtype: int64	

After dealing with Outliers

latitude	0
longitude	0
last_updated_epoch	0
temperature_celsius	0
temperature_fahrenheit	0
wind_mph	0
wind_kph	0
wind_degree	0
pressure_mb	0
pressure_in	0
precip_mm	0
precip_in	0
humidity	0
cloud	0
feels_like_celsius	0
feels_like_fahrenheit	0
visibility_km	0
visibility_miles	0
uv_index	0
gust_mph	0
gust_kph	0
air_quality_Carbon_Monoxide	0
air_quality_Ozone	0
air_quality_Nitrogen_dioxide	0
air_quality_Sulphur_dioxide	0
air_quality_PM2.5	0
air_quality_PM10	0
air_quality_us-epa-index	0
air_quality_gb-defra-index	0
moon_illumination	0
dtype: int64	

d. Skewness Analysis:

- The **skewness of all columns** was calculated to determine **the distribution of data in each feature**.
- Formula for skewness is:

$$S = \frac{n * \Sigma \left(\frac{X_i - X_m}{s}\right)^3}{(n-1)\,(n-2)}$$

$X_i = Individual\ Datapoints$

$X_m = Mean\ of\ Data$

$s = Standard\ Deviation$

$n = Number\ of\ Datapoints$

- **Observation:** Most columns exhibited **high skewness**, meaning they were **not normally distributed**.
- **Impact on Outlier Handling:** Since the data was skewed, traditional outlier removal techniques (like Z-score-based removal) **could have caused excessive data loss**.

- Since, it was observed that most of the columns were highly skewed and so, winsorization method was applied to handle the extreme outliers. This was because the clipping method worked best with data which is normally distributed whereas winsorization worked best with skewed data.

	Skewness
latitude	-0.305923
longitude	0.001597
last_updated_epoch	-0.015773
temperature_celsius	-0.880714
temperature_fahrenheit	-0.880644
wind_mph	131.319481
wind_kph	131.362487
wind_degree	0.151367
pressure_mb	107.361091
pressure_in	107.433063
precip_mm	19.497899
precip_in	19.342718
humidity	-0.480423
cloud	0.248067
feels_like_celsius	-0.880972
feels_like_fahrenheit	-0.880677
visibility_km	1.274328
visibility_miles	0.738403
uv_index	0.395813
gust_mph	96.608178
gust_kph	96.603356
air_quality_Carbon_Monoxide	13.384502
air_quality_Ozone	1.247881
air_quality_Nitrogen_dioxide	3.238307
air_quality_Sulphur_dioxide	-145.269162
air_quality_PM2.5	10.582185
air_quality_PM10	18.822872
air_quality_us-epa-index	1.678655
air_quality_gb-defra-index	2.002708
moon_illumination	0.035702

e. Data Normalization (Scaling):

After handling outliers, the dataset was **scaled** using **Min-Max Scaling**:

- Since different variables had **different ranges** (e.g., temperature in Celsius, air quality in arbitrary index values), **Min-Max Scaling** was applied:

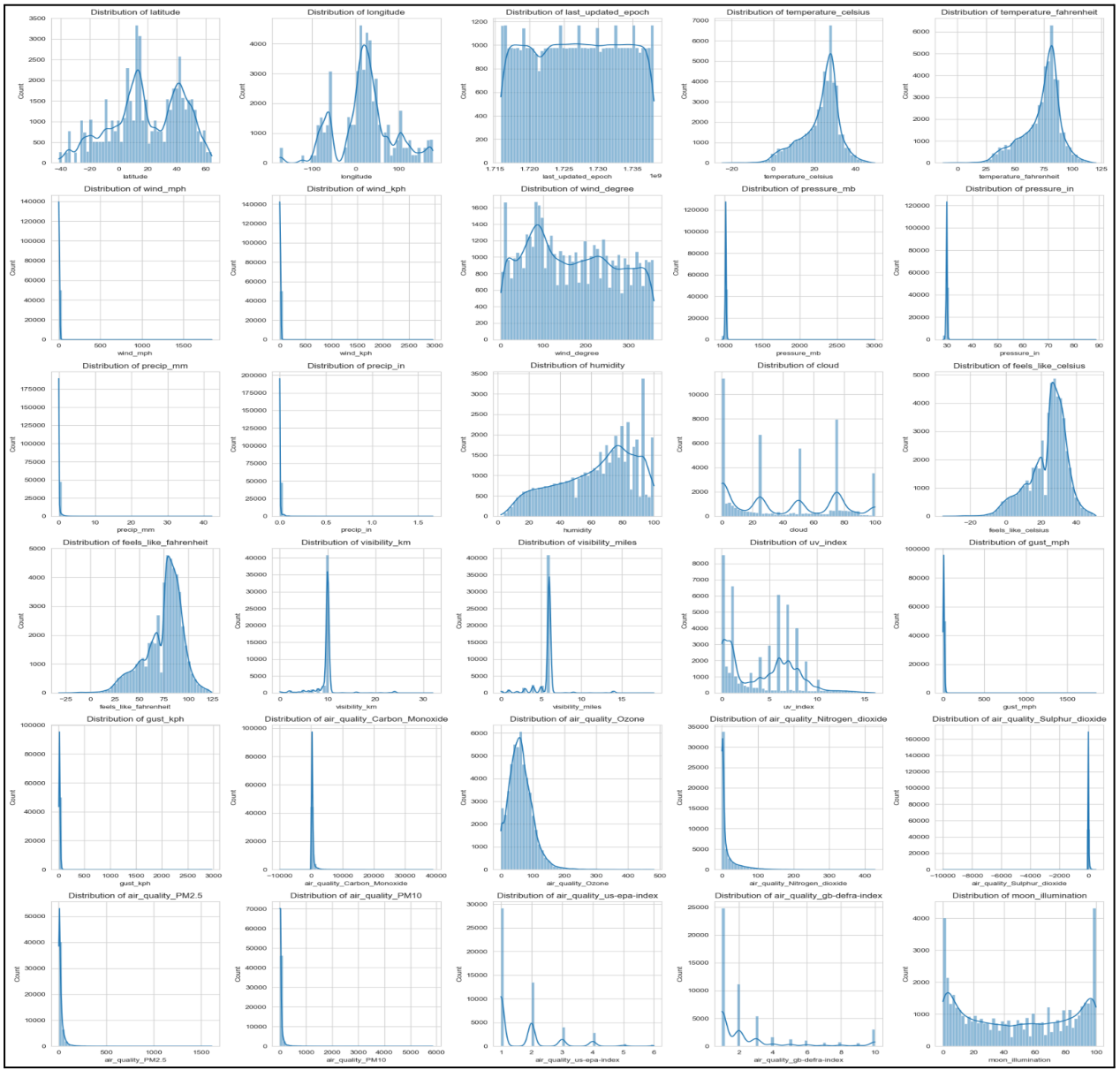
$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

- **Benefit:** Ensures **all numerical features are between 0 and 1**, improving model performance and preventing bias toward larger numerical ranges.

---

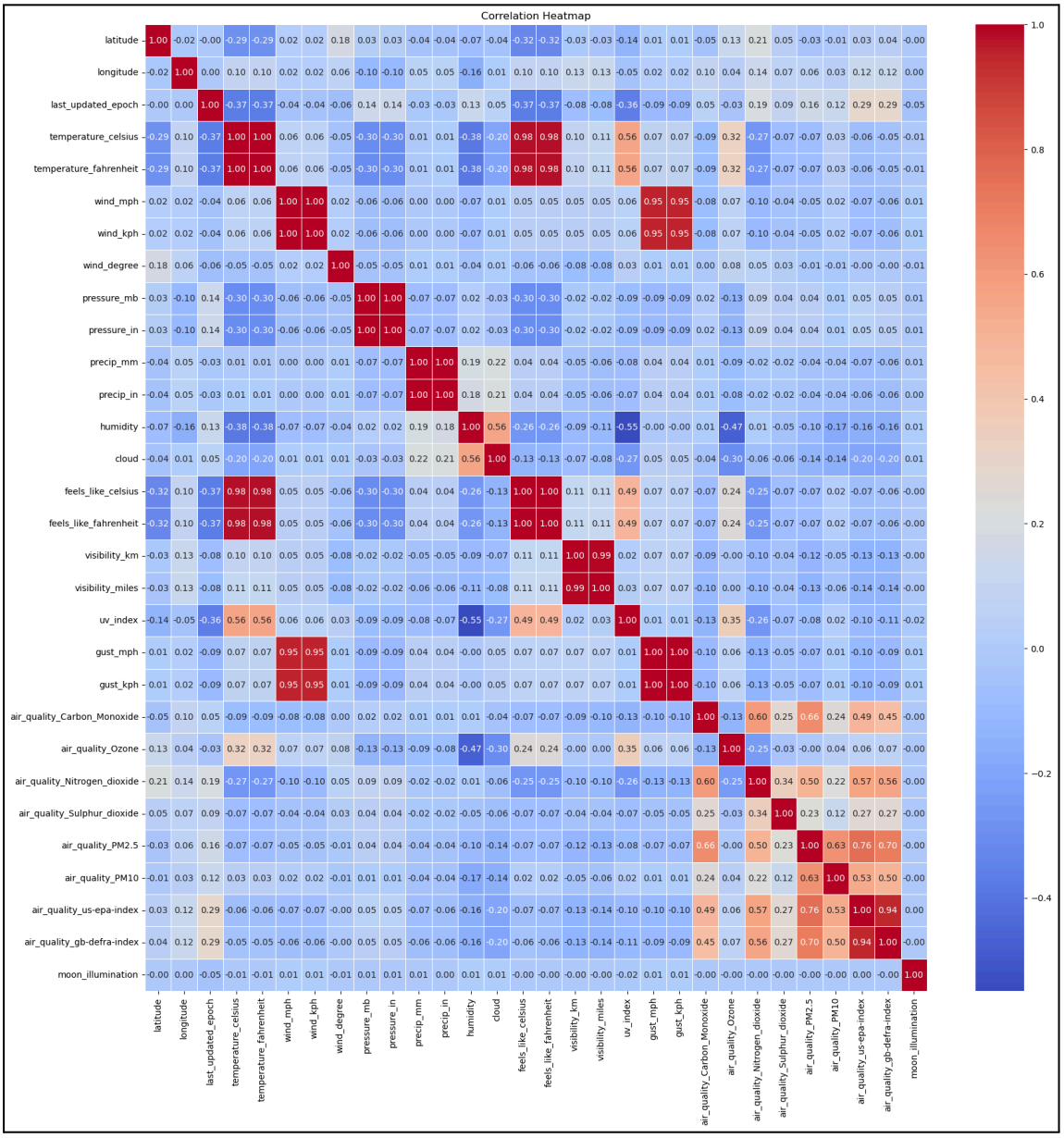
3. Exploratory Data Analysis (EDA):

a. Plotting Graphs of all Features to obtain trends:



- **Temperature:** Normally distributed with some skewness in extreme values.
- **Wind Speed & Gusts:** Right-skewed, meaning most locations have low wind speeds, but extreme wind speeds exist.
- **Humidity & Cloud Cover:** Bimodal, with peaks at low and high values, indicating distinct dry and humid regions.
- **UV Index:** Skewed towards lower values, likely due to locations experiencing varied sunlight exposure.
- **PM2.5, PM10, CO, NO2, SO2:** Skewed right, with a majority of locations having low pollution, but some highly polluted regions.
- **EPA and DEFRA Indexes:** Mostly concentrated in lower values (indicating acceptable air quality), but some spikes in unhealthy levels.
- **Moon Illumination:** Nearly uniform, as expected from cyclic moon phases.

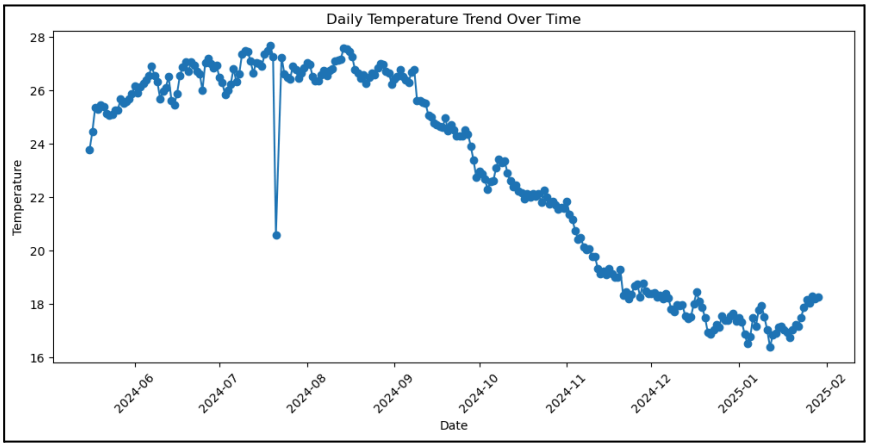
b. Correlation Matrix:



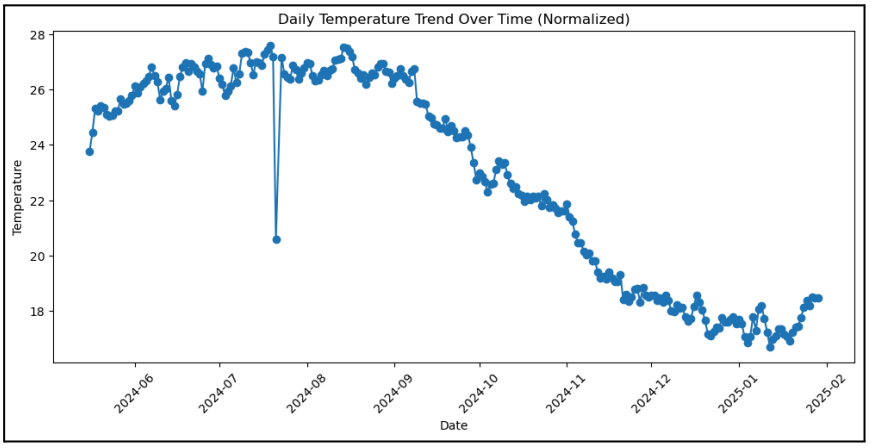
- **Highly Correlated:**
  - ☐ **Wind Speed & Gusts** (~0.95) → Strong winds lead to strong gusts.
  - ☐ **Humidity & Cloud Cover** (~0.56) → High humidity generally leads to cloud formation.
  - ☐ **Air Pollution Components(CO, NO2, PM2.5, PM10) and Air Quality Indices**(~0.5-0.76)**:** Presence of Air pollutants lead to increase in air quality indices as expected.
- **Inverse Correlations:**
  - ☐ **Temperature & Humidity** (~-0.26) → Higher temperatures tend to reduce humidity.
  - ☐ **Cloud Cover & Visibility** (~-0.1) → More cloud cover often reduces clear visibility.
- **Weak/No Correlation:**
  - ☐ **Air Pollution Components (PM2.5, PM10, CO, NO2, SO2)** (~0.2-0.7) → It would be expected that pollutants of one kind would lead to an increase in another but that isn't the case statistically.
  - ☐ **Visibility & Air Pollution (PM2.5, PM10, CO, etc.)** (~-0.1-0) → It would be expected that pollutants would lead to decrease in visibility however statistically, visibility doesn't depend much on pollution.
  - ☐ **Moon Phases & Weather**(~0)**:** As expected, moon phases do not influence weather.
  - ☐ **Wind Direction & All other factors**(~0)**:** No significant relationship.

c. Temperature Trends over time:

Before handling outliers



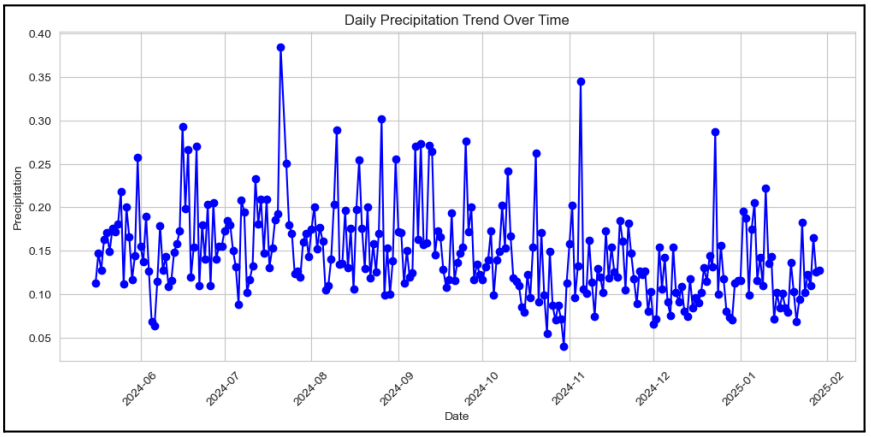
After handling outliers (winsorization)



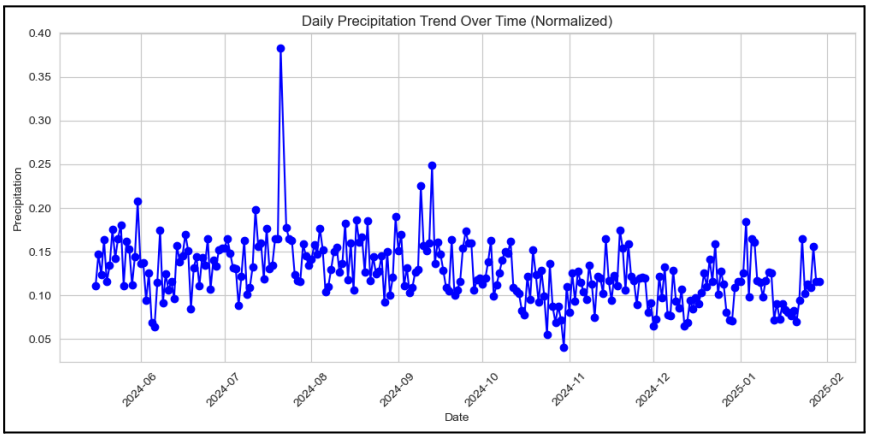
- The plot reveals a **seasonal pattern**, where temperature values **rise and fall periodically**, indicating **recurring trends**.
- The temperature was high in **June-September** likely due to the summer **season** in most of the parts of Earth whereas the temperatures were low between **September and February** likely due to the winter **season**.
- The data suggests the presence of **long-term trends**, which may indicate gradual climate shifts or yearly variations.
- Short-term **fluctuations** were also observed, likely due to **daily weather changes, atmospheric conditions, or local environmental factors**.

d. Precipitation Trends over time:

Before handling outliers



After handling outliers (winsorization)

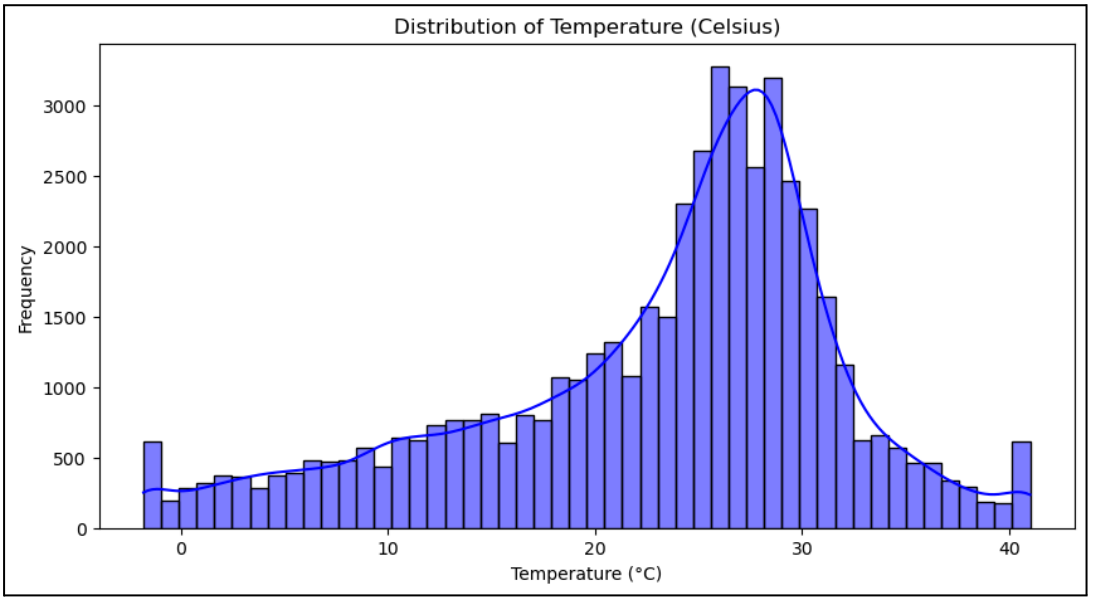


- Unlike temperature, **precipitation does not follow a smooth trend** and instead appears **sporadic**, with frequent **zero or very low values** and occasional spikes indicating heavy rainfall events. This aligns with the **skewed precipitation distribution observed in the histogram**, where most values were close to zero.
- The precipitation trend shows **sudden, sharp increases** on specific days, likely due to **localized storms, seasonal monsoons, or atmospheric disturbances**.

- Precipitation **does not exhibit a clear long-term trend**. Since **precipitation does not follow a smooth trend**, traditional forecasting models like **ARIMA may struggle** with accuracy.

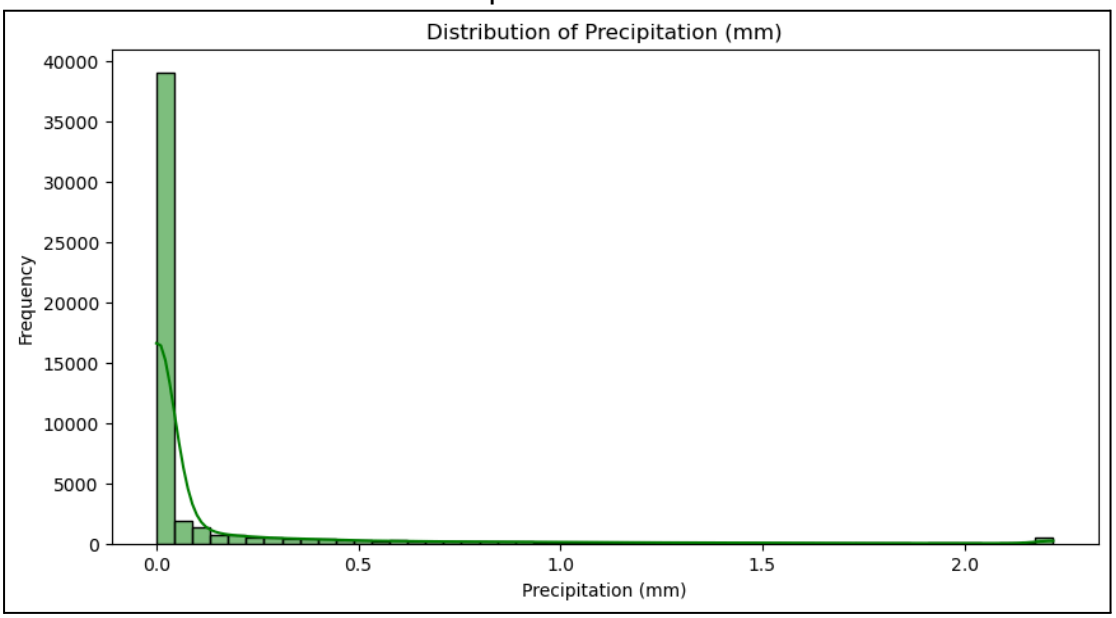
e. Data Visualizations:

- Distribution of Temperature:



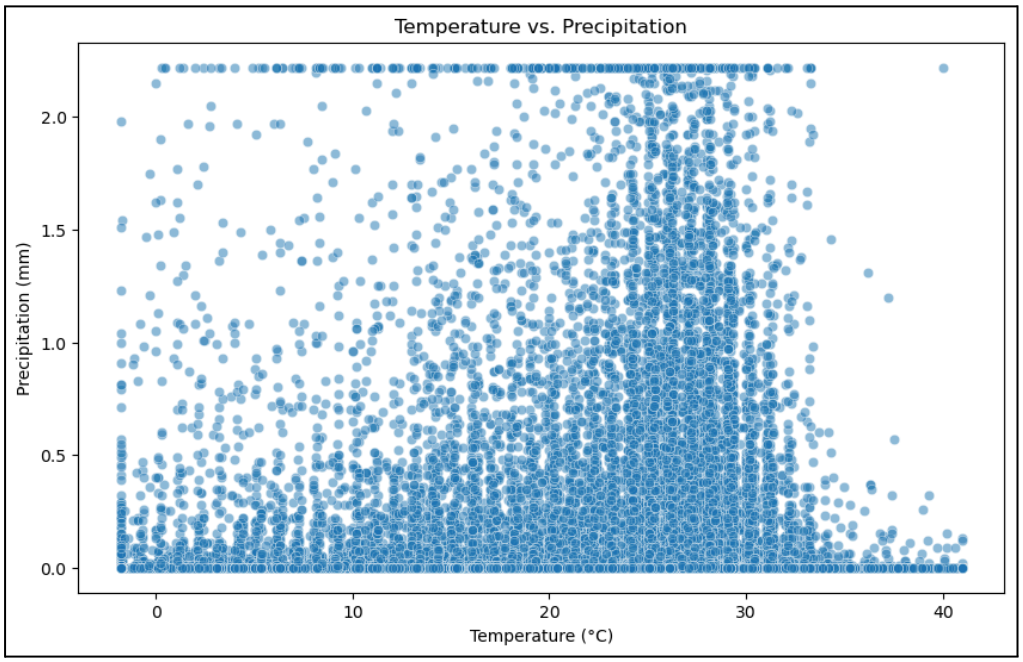
The dataset is left-skewed but not much. This indicates that the dataset doesn't exhibit very extreme variations in data.

- Distribution of Precipitation:



The precipitation data was **highly skewed**, meaning most recorded values were **low or near zero**, while a few extreme values represented heavy rainfall events. This suggests that **rainfall is not evenly distributed**, with dry periods being far more common than high-rainfall periods.

- Temperature vs Precipitation:



A scatter plot was used to visualize the relationship between **temperature and precipitation**. The plot **showed no strong correlation**, meaning that **higher temperatures do not necessarily lead to increased precipitation** or vice versa. This means that there are several other factors that affect precipitation in an area.

---

4. Forecasting Model:

a. Model used and why?

In our case, the ARIMA (AutoRegressive Integrated Moving Average) model was used to predict future temperatures. It is a widely used statistical approach for **time-series forecasting**, particularly for **predicting numerical values over time**.

This model was used in our case because of following reasons:

- **Strong Short-Term Predictive Performance** – Since temperature values tend to fluctuate based on recent trends, ARIMA effectively captures short-term variations.
- **Effectiveness for Non-Seasonal Data** – While temperature exhibits **some seasonal behavior**, ARIMA can still provide **reliable short-term forecasts** by differencing the data to remove trends and making it stationary.

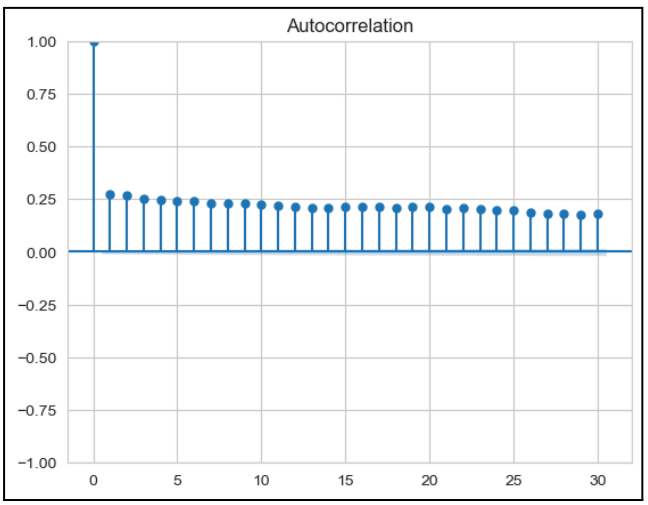
b. Model Implementation:

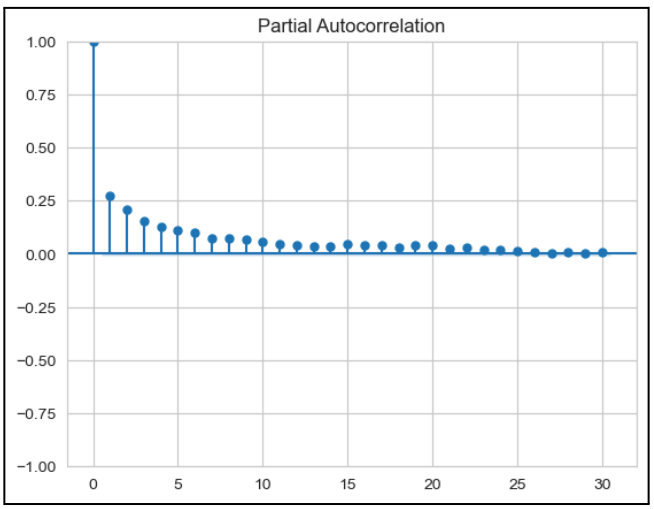
```python
# Split Data
train_data = time_series_daily[:'2025-01-22']
test_data = time_series_daily['2025-01-23':'2025-01-29']

# Fit ARIMA Model
arima_model = ARIMA(train_data, order=(5,1,2))
fitted_arima_model = arima_model.fit()

# Forecast for test period
forecast_steps = len(test_data)
forecast_arima = fitted_arima_model.forecast(steps=forecast_steps)

# Extract actual vs. predicted values
actual_values_arima = test_data.values.flatten()
predicted_values_arima = forecast_arima.values

# Get min and max temperature from the original dataset (before scaling)
#temp_min = df_copy["temperature_celsius"].min()
#temp_max = df_copy["temperature_celsius"].max()


# Compute Corrected Metrics after inverse transformation
mae_arima_corrected = mean_absolute_error(actual_values_arima, predicted_values_arima)
rmse_arima_corrected = np.sqrt(mean_squared_error(actual_values_arima, predicted_values_arima))

print("Corrected ARIMA MAE:", mae_arima_corrected)
print("Corrected ARIMA RMSE:", rmse_arima_corrected)
print("Average Forecasted Temperatures for Jan 23 - Jan 29:\n", predicted_values_arima)
print("Actual Temperatures for Jan 23 - Jan 29:\n", actual_values_arima)
```

- To train the ARIMA model, the dataset was split into:
  - **Training Data:** All the data before **January 23, 2025** were used to train the model on historical temperature patterns.
  - **Test Data:** Data from **January 23 - January 29, 2025** were used to evaluate how well the model performs on unseen future data.

- Model was configured with the following **parameters (p, d, q):**
  - **p (AutoRegressive Component) = 5** → The model uses the last **5 temperature values** to predict the next value.
  - **d (Differencing) = 1** → The data was **differenced once** to remove trends and make it stationary.
  - **q (Moving Average Component) = 2** → The model incorporates information from **the last 2 forecast errors** to adjust predictions.

These parameters were chosen based on **analysis of the autocorrelation function (ACF) and partial autocorrelation function (PACF) plots**.





c. Output prediction:

The **ARIMA model was used to generate temperature forecasts** for the test period **(January 23 - January 29, 2025)**.
These predictions were later evaluated against **actual recorded temperatures** using **Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE)** to assess the model's accuracy.

---

5. Model Evaluation:

Comparison of **actual vs predicted** temperatures were done for **January 23** till **January 29.** These values were then used to calculate the error metrics.

a. Performance Metrics used:

- **Mean Absolute Error (MAE):** Measures the **average absolute difference** between actual and predicted values.

$$MAE = \frac{\Sigma|y_i - y_p|}{n}$$

$y_i = Actual\ value$ $\quad\quad$ $y_p = Predicted\ Value$

$n = Total\ Number\ of\ observations$

- **Root Mean Squared Error (RMSE):** Captures how much the **predicted values deviate from the actual values**.

$$RMSE = \left(\frac{\Sigma|y_i - y_p|^2}{n}\right)^{0.5}$$

$y_i = Actual\ value$ $\quad\quad$ $y_p = Predicted\ Value$

$n = Total\ Number\ of\ observations$

b. Model Evaluation:

The error metrics were calculated:

MAE (Lower is better)	RMSE (Lower is better)
0.9678221548340323	1.002382024281664

ARIMA Model provided a reasonable short-term forecast. The error metrics are quite decent and it can be said that the model worked well.

6. Insights and Observations:

a. Key Observations:

- **ARIMA** was effective for **short-term temperature predictions.**
- **No strong relationship** was found between **temperature** & **precipitation.** A **scatter plot** analysis of **temperature** vs. **precipitation** revealed that higher temperatures do not necessarily lead to increased precipitation.
- **Seasonality** was detected, meaning advanced seasonal models could improve forecasting accuracy. The **temperature trend analysis and ACF/PACF plots** showed a **repeating pattern over specific intervals**, indicating the presence of **seasonality** in the data.