Exploratory Data Analysis (EDA) Report: Indian Climate 2025

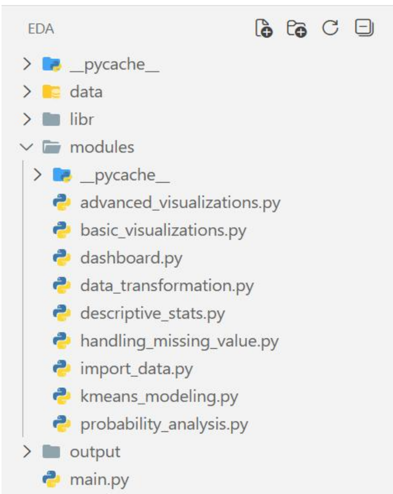
Exploratory data analysis (EDA) is crucial for understanding your climate data, identifying weather patterns, and generating insights that can inform environmental policy or health advisories. In this project, we implement critical steps for performing EDA on the Indianclimate.csv dataset sourced from Kaggle. This iterative procedure involves summarizing, visualizing, and exploring information to find patterns, anomalies, and relationships between weather variables and air quality.

Procedure Exploratory Data Analysis

1. Understand the Problem and the Data
2. Import and Inspect the Data
3. Handling Missing Values
4. Explore Data Characteristics
5. Perform Data Transformation
6. Visualize Data Relationships
7. Handling Outliers
8. Communicate Findings and Insights

Dataset: Indianclimate.csv from Kaggle

Project Structure:



1. Understand the Problem and the Data

The objective of this project is to analyze climate variations across major Indian cities and identify relationships between temperature, humidity, rainfall, and Air Quality Index (AQI).

Variables in our Dataset:

- **Numerical:** Temperature (Max, Min, Avg), Humidity, Rainfall, Wind Speed, AQI, Pressure, Cloud Cover.
- **Categorical:** City, State, AQI_Category.
- **Temporal:** Date

2.Import and Inspect the Data

File Name: import_data.py

The dataset is loaded into a Pandas DataFrame using the read_csv() function. This allows us to access, manipulate, and analyze the climate data efficiently.

The function load_data() performs the following tasks:

- Displays the **first five rows** of the dataset using head() to give a quick overview of the climate records for Indian cities.
- Uses info() to inspect the **structure of the dataset**, including:
  - Column names
  - Data types of each variable
  - Count of non-null values
  - Total number of entries

This step helps in identifying missing values, incorrect data formats, and the overall size of the dataset, which is essential before proceeding to data cleaning and transformation.

```
Microsoft Windows [Version 10.0.26200.7171]
(c) Microsoft Corporation. All rights reserved.

C:\Users\MISBA T\Desktop\EDA>"C:/Users/MISBA T/Desktop/EDA/libr/Scripts/activate.bat"

(libr) C:\Users\MISBA T\Desktop\EDA>libr\Scripts\activate

(libr) C:\Users\MISBA T\Desktop\EDA>python main.py

First 5 rows of dataset:

         Date       City        State  Temperature_Max (°C)  Temperature_Min (°C)  ...  Wind_Speed (km/h)  AQI  AQI_Category  Pressure (hPa)  Cloud_Cover (%)
0  2024-01-01     Mumbai  Maharashtra                  32.5                   NaN  ...                3.3  259          Poor          1020.3             62.1
1  2024-01-01      Delhi        Delhi                  25.4                  10.7  ...                9.0  130      Moderate          1008.4             46.0
2  2024-01-01  Bengaluru    Karnataka                   NaN                  30.8  ...                6.6   54  Satisfactory          1008.0             61.3
3  2024-01-01    Chennai   Tamil Nadu                  37.2                  30.4  ...                9.0  176      Moderate           993.4             70.0
4  2024-01-01    Kolkata  West Bengal                  27.4                  17.5  ...                9.2   97  Satisfactory          1008.2             56.9

[5 rows x 13 columns]
```

```
Dataset Structure:

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7310 entries, 0 to 7309
Data columns (total 13 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   Date                 7310 non-null   object
 1   City                 7310 non-null   object
 2   State                7310 non-null   object
 3   Temperature_Max (°C)  7308 non-null   float64
 4   Temperature_Min (°C)  7307 non-null   float64
 5   Temperature_Avg (°C)  7310 non-null   float64
 6   Humidity (%)         7310 non-null   float64
 7   Rainfall (mm)        7310 non-null   float64
 8   Wind_Speed (km/h)    7310 non-null   float64
 9   AQI                  7310 non-null   int64
 10  AQI_Category         7310 non-null   object
 11  Pressure (hPa)       7310 non-null   float64
 12  Cloud_Cover (%)      7310 non-null   float64
dtypes: float64(8), int64(1), object(4)
memory usage: 742.6+ KB
None
------------------------------------------------------------
```

3. Handling Missing Values and Outliers

File Name:handling_missing_value.py

3.1 Detection of Missing Values

The dataset was first inspected for missing entries using isnull().sum() and notnull().sum(). It was found that Temperature_Max (°C) had 2 missing values and Temperature_Min (°C) had 3 missing values. All other columns, including humidity, rainfall, and AQI, were complete. This step was crucial to identify gaps that could affect the quality of analysis.

```
Missing values (isnull) before cleaning:
Date                    0
City                    0
State                   0
Temperature_Max (°C)    2
Temperature_Min (°C)    3
Temperature_Avg (°C)    0
Humidity (%)            0
Rainfall (mm)           0
Wind_Speed (km/h)       0
AQI                     0
AQI_Category            0
Pressure (hPa)          0
Cloud_Cover (%)         0
dtype: int64

Non-missing values (notnull) before cleaning:
Date                    7310
City                    7310
State                   7310
Temperature_Max (°C)    7308
Temperature_Min (°C)    7307
Temperature_Avg (°C)    7310
Humidity (%)            7310
Rainfall (mm)           7310
Wind_Speed (km/h)       7310
AQI                     7310
AQI_Category            7310
Pressure (hPa)          7310
Cloud_Cover (%)         7310
dtype: int64
```

3.2 Mean Imputation

To handle the missing numeric values, the mean of each respective column was used for imputation. This approach ensures that the dataset size remains unchanged while maintaining realistic distributions for temperature values. After this step, all numeric columns had zero missing values, making the dataset ready for further processing.

```
Missing values after filling numeric columns with mean:
Date                  0
City                  0
State                 0
Temperature_Max (°C)  0
Temperature_Min (°C)  0
Temperature_Avg (°C)  0
Humidity (%)          0
Rainfall (mm)         0
Wind_Speed (km/h)     0
AQI                   0
AQI_Category          0
Pressure (hPa)        0
Cloud_Cover (%)       0
dtype: int64

Removed 0 duplicate rows, if any.

Missing values after cleaning and removing duplicates:
Date                  0
City                  0
State                 0
Temperature_Max (°C)  0
Temperature_Min (°C)  0
Temperature_Avg (°C)  0
Humidity (%)          0
Rainfall (mm)         0
Wind_Speed (km/h)     0
AQI                   0
AQI_Category          0
Pressure (hPa)        0
Cloud_Cover (%)       0
dtype: int64
```

3.3 Duplicate Removal

The dataset was checked for duplicate rows using drop_duplicates(). No duplicates were found, confirming that each observation represented a unique record of city-wise climate data. This step ensured the integrity and reliability of the dataset.

```
Missing values after cleaning and removing duplicates:
Date                  0
City                  0
State                 0
Temperature_Max (°C)  0
Temperature_Min (°C)  0
Temperature_Avg (°C)  0
Humidity (%)          0
Rainfall (mm)         0
Wind_Speed (km/h)     0
AQI                   0
AQI_Category          0
Pressure (hPa)        0
Cloud_Cover (%)       0
dtype: int64

Non-missing values after cleaning and removing duplicates:
Date                  7310
City                  7310
State                 7310
Temperature_Max (°C)  7310
Temperature_Min (°C)  7310
Temperature_Avg (°C)  7310
Humidity (%)          7310
Rainfall (mm)         7310
Wind_Speed (km/h)     7310
AQI                   7310
AQI_Category          7310
Pressure (hPa)        7310
Cloud_Cover (%)       7310
dtype: int64
```

3.4 Final Dataset

After handling missing values, duplicates, and outliers, the dataset size was reduced from 7310 to 6270 records. The cleaned dataset, free from missing entries and extreme distortions, was saved as cleaned_dataset.csv. This dataset is now fully prepared for visualization, clustering, and predictive modeling tasks.

```
Cleaned dataset saved as 'cleaned_dataset.csv'
```

4.Explore Data Characteristics

File Name:descriptive_stats.py

The statistical characteristics of the cleaned Indian Climate dataset to understand the typical range and variability of weather across cities.

Using **descriptive statistics**, we calculated the following for all numeric features.

- **Mean** – Average value of the dataset
- **Median** – Middle value when data is arranged in order
- **Mode** – Most frequently occurring value
- **Standard Deviation** – Measure of how spread out the values are
- **Minimum** – Smallest value in the dataset
- **Maximum** – Largest value in the dataset

The **Temperature_Max, Temperature_Min, and Temperature_Avg** columns showed mean values of 35.01°C, 25.08°C, and 30.05°C respectively, with moderate variability, reflecting typical daily temperatures in India. **Humidity (%)** averaged 62.69% with a standard deviation of 18.67%, indicating variation between humid coastal regions and drier inland areas. **Rainfall (mm)** had a mean of 1.71 but a median of 0, showing that heavy rain events are infrequent. **Wind_Speed (km/h)** averaged 13.54 km/h, while **AQI** had a mean of 193.97, highlighting pollution levels across cities. **Pressure (hPa)** and **Cloud_Cover (%)** showed moderate variability, with means of 1007.36 hPa and 52.66%, respectively. Overall, these descriptive statistics provide a clear picture of climate distributions, identify typical ranges, and highlight the spread or volatility in temperature, humidity, rainfall, wind speed, AQI, pressure, and cloud cover, laying the foundation for further visualization and analysis.

```
----- Descriptive Statistics -----

Column: Temperature_Max (°C)
Mean: 35.01
Median: 35.00
Mode: 26.7
Standard Deviation: 5.79
Minimum: 25.0
Maximum: 45.0
----------------------------------------
Column: Temperature_Min (°C)
Mean: 25.08
Median: 25.10
Mode: 19.9
Standard Deviation: 6.47
Minimum: 10.1
Maximum: 39.8
----------------------------------------
Column: Temperature_Avg (°C)
Mean: 30.05
Median: 30.10
Mode: 37.9
Standard Deviation: 5.97
Minimum: 17.6
Maximum: 42.3
----------------------------------------
```

```
Column: Humidity (%)
Mean: 62.69
Median: 62.70
Mode: 71.9
Standard Deviation: 18.67
Minimum: 30.0
Maximum: 95.0
----------------------------------------
Column: Rainfall (mm)
Mean: 1.71
Median: 0.00
Mode: 0.0
Standard Deviation: 3.12
Minimum: 0.0
Maximum: 15.6
----------------------------------------
Column: Wind_Speed (km/h)
Mean: 13.54
Median: 13.60
Mode: 5.4
Standard Deviation: 6.53
Minimum: 2.0
Maximum: 25.0
----------------------------------------
Column: AQI
Mean: 193.97
Median: 194.00
Mode: 307
Standard Deviation: 89.15
Minimum: 40
Maximum: 349
----------------------------------------

Column: Pressure (hPa)
Mean: 1007.36
Median: 1007.30
Mode: 1009.6
Standard Deviation: 10.11
Minimum: 990.0
Maximum: 1025.0
----------------------------------------
Column: Cloud_Cover (%)
Mean: 52.66
Median: 52.80
Mode: 58.5
Standard Deviation: 27.35
Minimum: 5.0
Maximum: 100.0
----------------------------------------
```

5. Perform Data Transformation

File Name:data_transformation.py

Transform numeric features to a common scale for accurate analysis and distance-based modeling like K-Means.

5.1 Min-Max Normalization: Min-Max Normalization rescales all numeric features into the range **0 to 1** using MinMaxScaler, preserving the original distribution while bringing all attributes to a common scale.

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

5.2 Z-Score Standardization: Z-Score Standardization transforms numeric features to have a **mean of 0 and standard deviation of 1** using StandardScaler, making the data suitable for clustering and machine-learning algorithms.

$$Z = \frac{X - \mu}{\sigma}$$

Normalized Data (Min-Max scaling):

	Temperature_Max (°C)	Temperature_Min (°C)	Temperature_Avg (°C)	Humidity (%)	Rainfall (mm)	Wind_Speed (km/h)	AQI	Pressure (hPa)	Cloud_Cover (%)
0	0.375000	0.501924	0.307692	0.732308	0.000000	0.056522	0.708738	0.865714	0.601053
1	0.020000	0.020202	0.020243	0.832308	0.000000	0.304348	0.291262	0.525714	0.431579
2	0.497618	0.696970	0.663968	0.292308	0.237179	0.200000	0.045307	0.514286	0.592632
3	0.610000	0.683502	0.655870	0.064615	0.608974	0.304348	0.440129	0.097143	0.684211
4	0.120000	0.249158	0.198381	0.033846	0.583333	0.313043	0.184466	0.520000	0.546316

Standardized Data (Z-score scaling):

	Temperature_Max (°C)	Temperature_Min (°C)	Temperature_Avg (°C)	Humidity (%)	Rainfall (mm)	Wind_Speed (km/h)	AQI	Pressure (hPa)	Cloud_Cover (%)
0	-0.432709	-0.012014	-0.812055	0.798699	-0.545888	-1.567948	0.729431	1.280251	0.345016
1	-1.658621	-2.224452	-2.001830	1.146811	-0.545888	-0.695474	-0.717651	0.102915	-0.243614
2	-0.009274	0.883785	0.662596	-0.732995	0.638454	-1.062832	-1.570195	0.063341	0.315767
3	0.378811	0.821930	0.629081	-1.525620	2.494990	-0.695474	-0.201638	-1.381121	0.633847
4	-1.313294	-1.172909	-1.264504	-1.632731	2.366953	-0.664861	-1.087835	0.083128	0.154900

6. Visualize Data Relationships

File Name:basic_visualizations.py ,advanced_visualizations.py

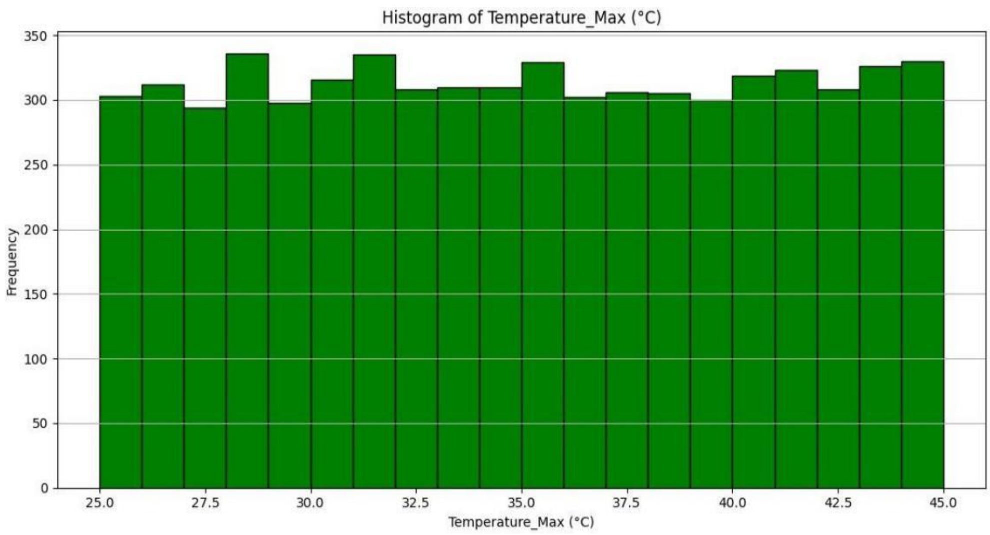
The purpose of visualization is to transform numerical climate data into graphical formats so that trends, variations, and relationships between weather conditions and AQI can be easily interpreted.

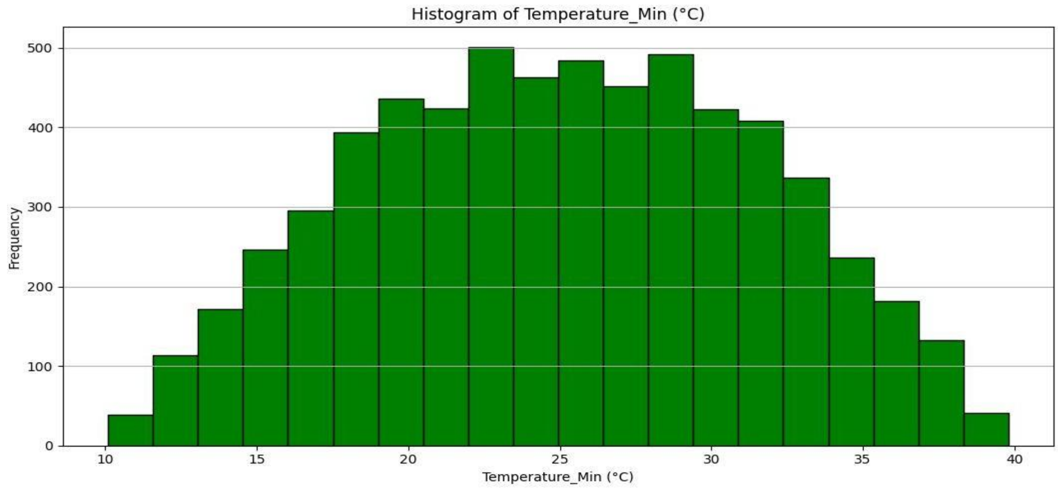
6.1 Univariate Analysis:

This analysis focuses on **one variable at a time** to understand its distribution and individual behavior.

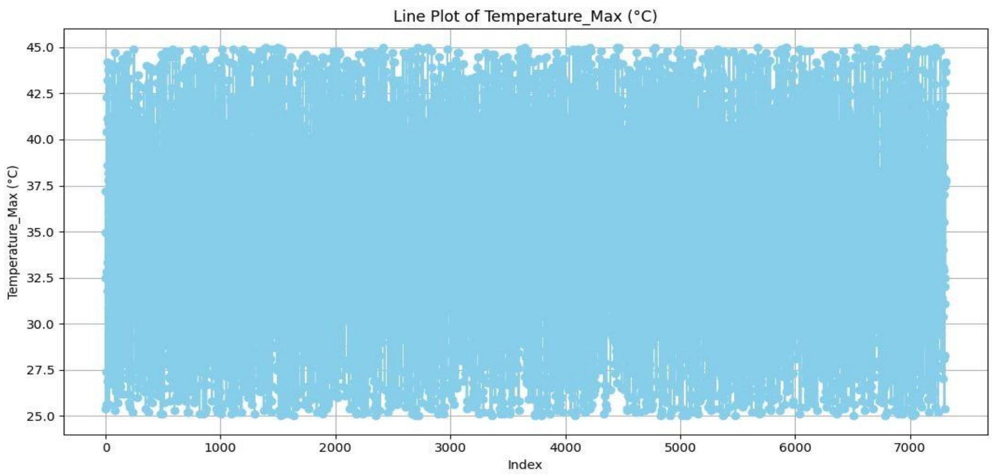
Charts:

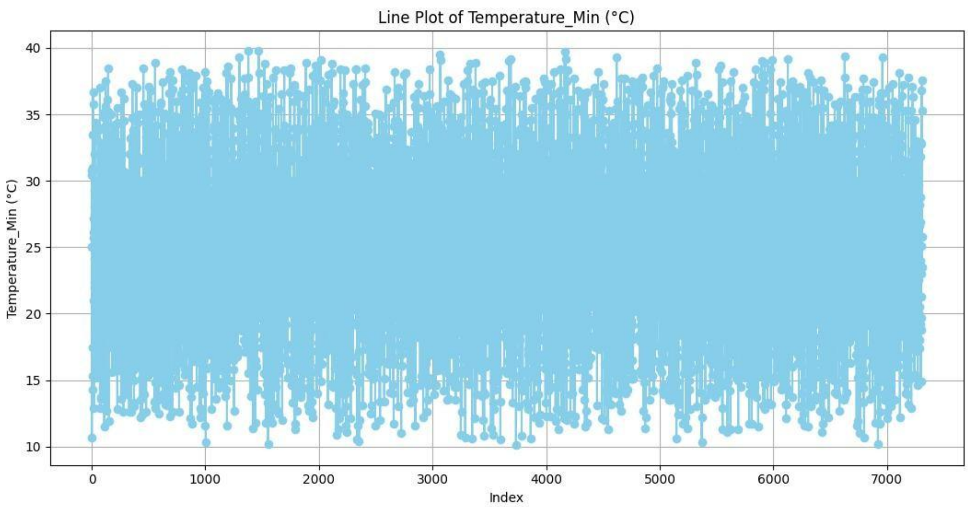
6.1.1 Histogram – to show the frequency distribution of AQI and temperature values.



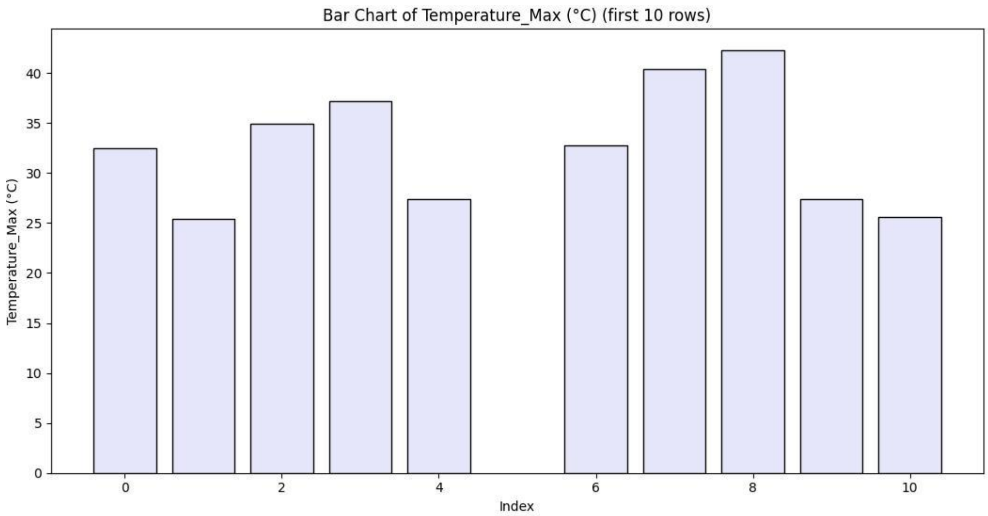


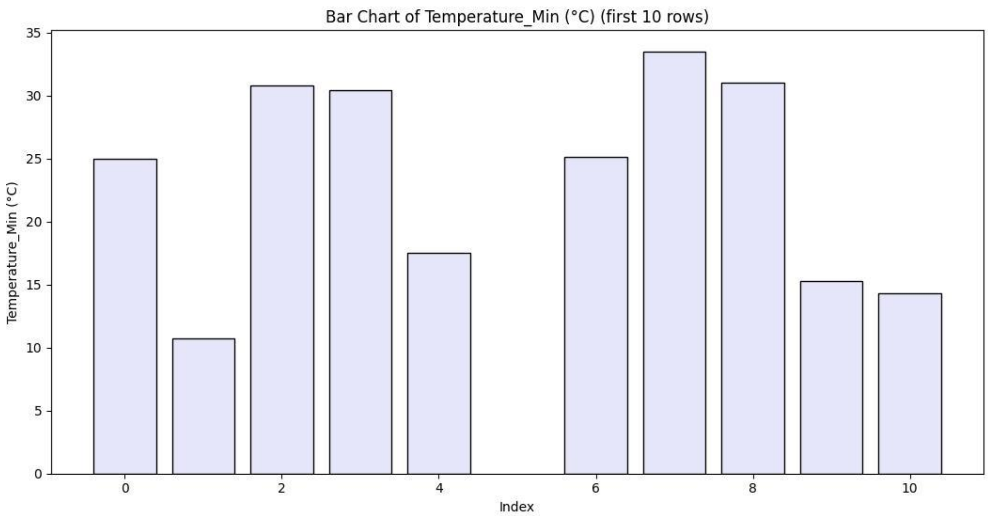
6.1.2 Line Plot – to observe trends of individual climate variables over records.





6.1.3 Bar Chart – to compare values of the first few observations for each numeric feature



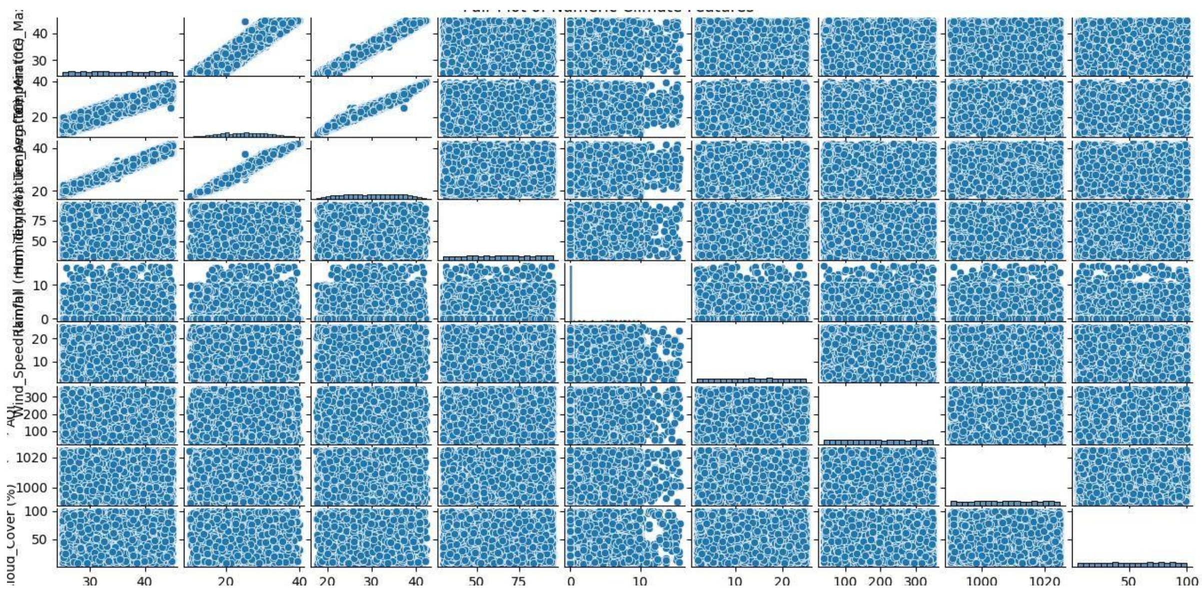


6.2 Multivariate Analysis:

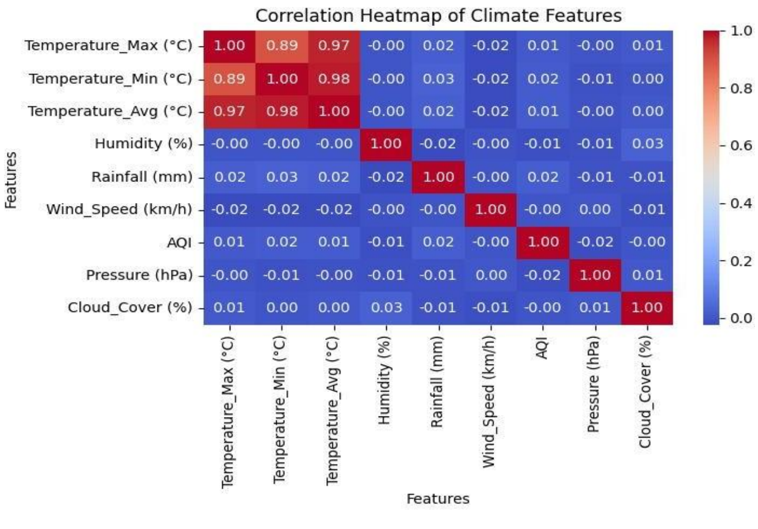
This analysis evaluates interactions between **multiple climate variables** together.

Charts:

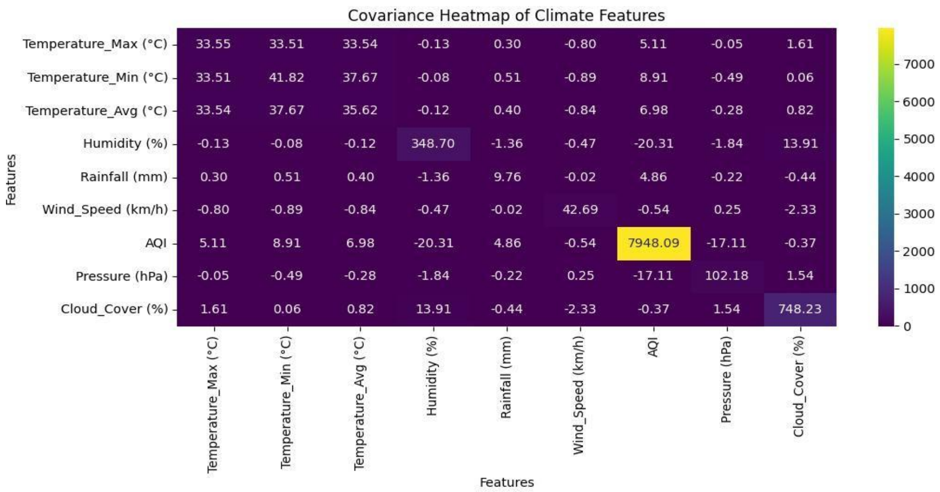
6.2.1 Pair Plot – to visualize relationships among all numeric features.



6.2.2 Correlation Heatmap – shows the strength and direction of relationships between variables.



6.2.3 Covariance Heatmap – measures how features such as Wind Speed, Cloud Cover, and AQI vary together.



7. Handling Outliers

File Name:handling_missing_value.py

Outliers in numeric columns were identified using the Interquartile Range (IQR) method. While most features had no extreme values, Rainfall (mm) contained 1040 outliers representing unusually heavy rainfall days. These extreme values were removed to prevent skewed analysis and maintain statistical consistency.

Logic:

Calculate Q1(25th percentile) and Q3(75th percentile).

Define bounds: Lower = Q1 - 1.5 *IQR; Upper = Q3 + 1.5*IQR

```
Handling outliers in column: Temperature_Max (°C)
Number of outliers in Temperature_Max (°C): 0
After removing outliers, Temperature_Max (°C) stats:
count    7310.000000
mean       34.952367
std         5.781145
min        25.000000
25%        30.000000
50%        34.900000
75%        40.000000
max        45.000000
Name: Temperature_Max (°C), dtype: float64

Handling outliers in column: Temperature_Min (°C)
Number of outliers in Temperature_Min (°C): 0
After removing outliers, Temperature_Min (°C) stats:
count    7310.000000
mean       25.007144
std         6.475377
min        10.100000
25%        19.900000
50%        25.000000
75%        30.000000
max        39.800000
Name: Temperature_Min (°C), dtype: float64
```

```
Handling outliers in column: Temperature_Avg (°C)
Number of outliers in Temperature_Avg (°C): 0
After removing outliers, Temperature_Avg (°C) stats:
count    7310.000000
mean       29.980192
std         5.966698
min        17.600000
25%        25.000000
50%        30.000000
75%        35.000000
max        42.300000
Name: Temperature_Avg (°C), dtype: float64

Handling outliers in column: Humidity (%)
Number of outliers in Humidity (%): 0
After removing outliers, Humidity (%) stats:
count    7310.000000
mean       62.653516
std        18.680171
min        30.000000
25%        46.400000
50%        62.700000
75%        78.700000
max        95.000000
Name: Humidity (%), dtype: float64
```

```
Handling outliers in column: Rainfall (mm)
Number of outliers in Rainfall (mm): 1040
After removing outliers, Rainfall (mm) stats:
count    6270.000000
mean        1.705407
std         3.124348
min         0.000000
25%         0.000000
50%         0.000000
75%         2.200000
max        15.600000
Name: Rainfall (mm), dtype: float64

Handling outliers in column: Wind_Speed (km/h)
Number of outliers in Wind_Speed (km/h): 0
After removing outliers, Wind_Speed (km/h) stats:
count    6270.000000
mean       13.543636
std         6.533668
min         2.000000
25%         7.925000
50%        13.600000
75%        19.100000
max        25.000000
Name: Wind_Speed (km/h), dtype: float64
```

```
Handling outliers in column: AQI
Number of outliers in AQI: 0
After removing outliers, AQI stats:
count    6270.000000
mean      193.974960
std        89.152035
min        40.000000
25%       117.000000
50%       194.000000
75%       270.000000
max       349.000000
Name: AQI, dtype: float64
```

```
Handling outliers in column: Pressure (hPa)
Number of outliers in Pressure (hPa): 0
After removing outliers, Pressure (hPa) stats:
count    6270.000000
mean     1007.359777
std        10.108377
min       990.000000
25%       998.700000
50%      1007.300000
75%      1016.200000
max      1025.000000
Name: Pressure (hPa), dtype: float64

Handling outliers in column: Cloud_Cover (%)
Number of outliers in Cloud_Cover (%): 0
After removing outliers, Cloud_Cover (%) stats:
count    6270.000000
mean       52.663238
std        27.353833
min         5.000000
25%        29.100000
50%        52.800000
75%        76.300000
max       100.000000
Name: Cloud_Cover (%), dtype: float64
```

8.Communicate Findings and Insights

File Name:dashboard.py,

The purpose of this step is to present the analyzed Indian Climate dataset in an interactive and meaningful way so that users can easily understand climate trends, air quality variations, and regional patterns.

8.1 Interactive Dashboard:

A Dash-based web dashboard was developed to display climate trends dynamically, making the analysis accessible to non-technical users.

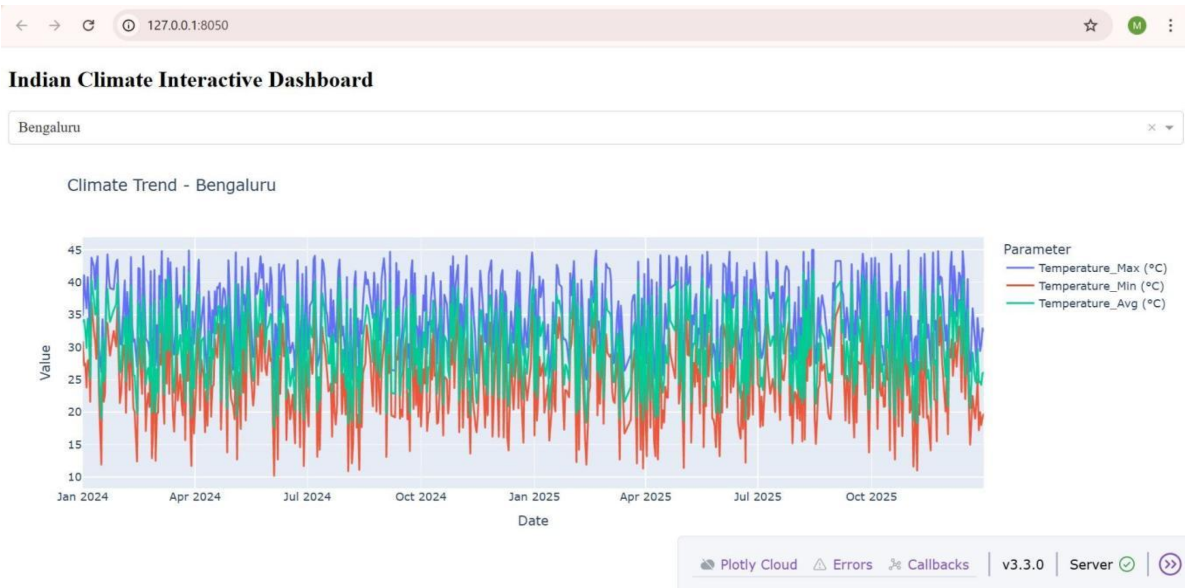
City Dropdown:

A dropdown menu allows users to select a specific **city**, enabling real-time filtering of climate records.

Trend Analysis:

Interactive **line graphs** display the progression of major numeric climate parameters such as Temperature, Rainfall, and AQI over time for the selected city.

```
Dash is running on http://127.0.0.1:8050/

 * Serving Flask app 'dashboard'
 * Debug mode: on
```

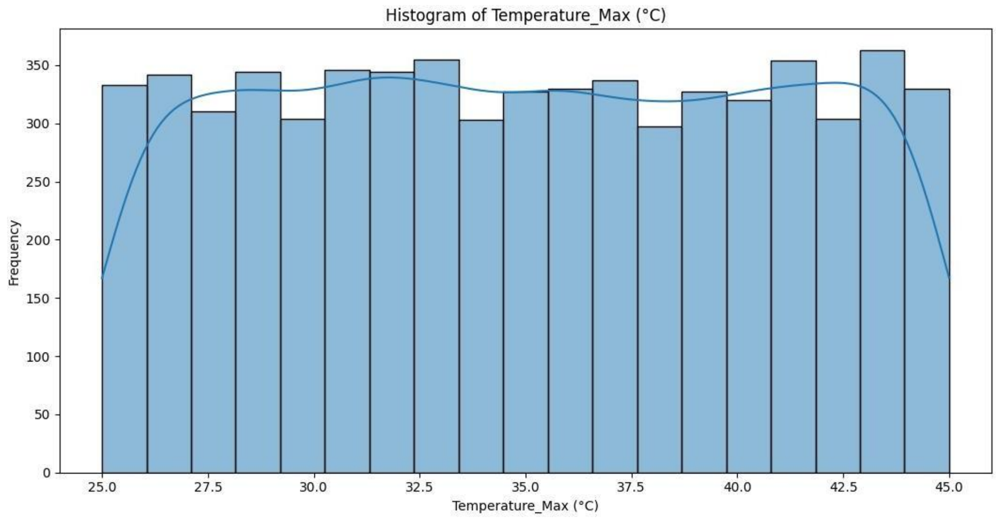


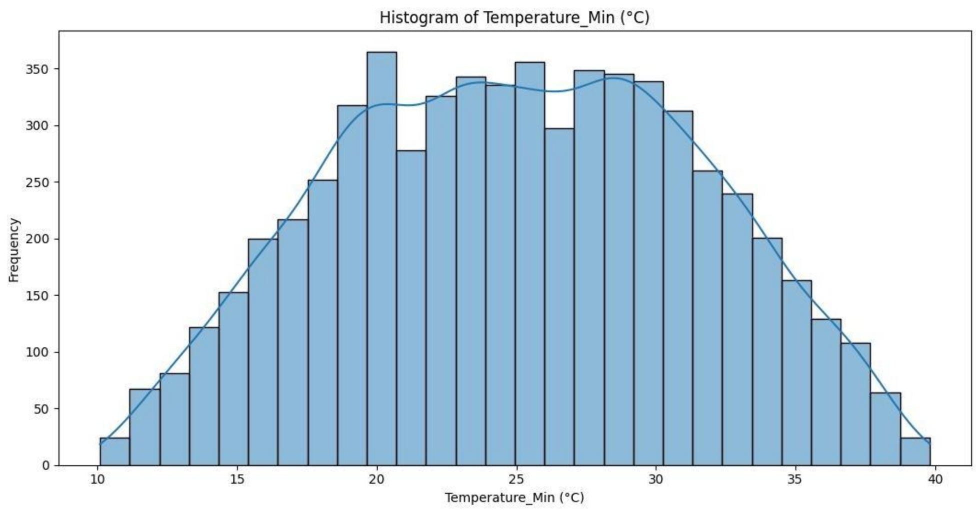
8.2 Probability Analysis:

File Name:probability_analysis.py

Range and variance were calculated for key numeric variables to understand their spread and variability, and histograms with KDE curves were plotted to observe probability distributions.

```
----- Probability Analysis -----

Column: Temperature_Max (°C)
Range: 20.0
Variance: 33.55

Column: Temperature_Min (°C)
Range: 29.699999999999996
Variance: 41.82
```





8.3 K-Means Clustering:

File Name:kmeans_modeling.py

The standardized numeric dataset was clustered into **three segments** using the K-Means algorithm.

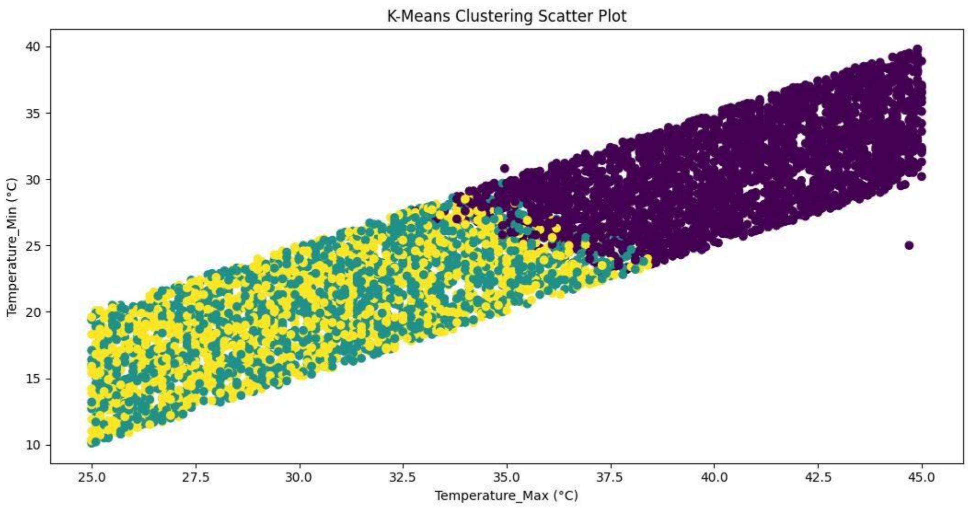
```
----- K-Means Clustering -----

Cluster Count:
Cluster
0    2864
2    1712
1    1694
Name: count, dtype: int64
```

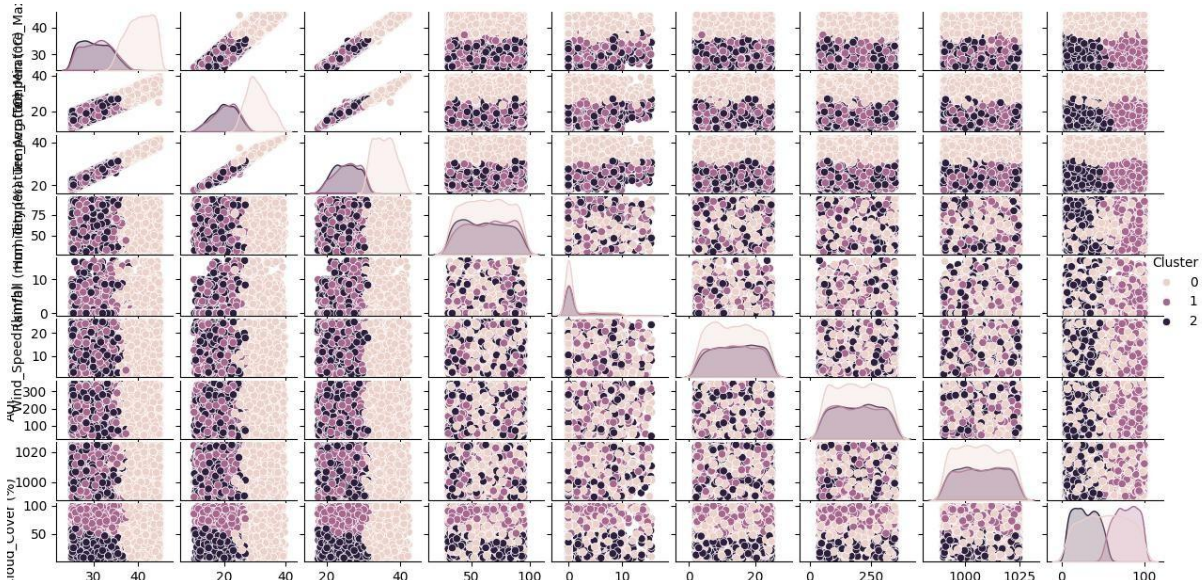
8.3.1 Cluster Visualization:

A **scatter plot** and **pair plot** were generated to visually distinguish clusters and observe inter-feature relationships.

Scatter plot - Bivariate Analysis:

This analysis examines the relationship between **two variables simultaneously**.





Kmeans-Pairplot

8.3.2 Interpretation:

Cluster-wise means were computed to interpret environmental behavior across different climate zones.

--- Cluster Interpretation ---

Cluster	Temperature_Max (°C)	Temperature_Min (°C)	Temperature_Avg (°C)	Humidity (%)	Rainfall (mm)	Wind_Speed (km/h)	AQI	Pressure (hPa)	Cloud_Cover (%)
0	40.341603	30.827447	35.587291	62.875524	1.710300	13.347242	194.811103	1007.090642	52.396753
1	30.781641	20.453137	25.615821	64.153247	1.664817	13.431228	192.370720	1007.739315	77.345396
2	30.260311	20.061040	25.159404	60.919276	1.737383	13.983411	194.163551	1007.434463	28.686390

CONCLUSION:

This project successfully performed an end-to-end Exploratory Data Analysis on the Indian Climate dataset. The raw data was cleaned by handling missing values, removing outliers, and eliminating inconsistencies. Descriptive statistics and visualizations revealed important patterns in temperature, rainfall, humidity, and AQI across Indian cities. Data transformation techniques ensured fair comparison among features for accurate modeling. Interactive dashboards enabled real-time exploration of city-wise climate trends. K-Means clustering identified distinct climate zones with unique environmental behaviors. Overall, the study provides meaningful insights that can support environmental monitoring and data-driven decision making.