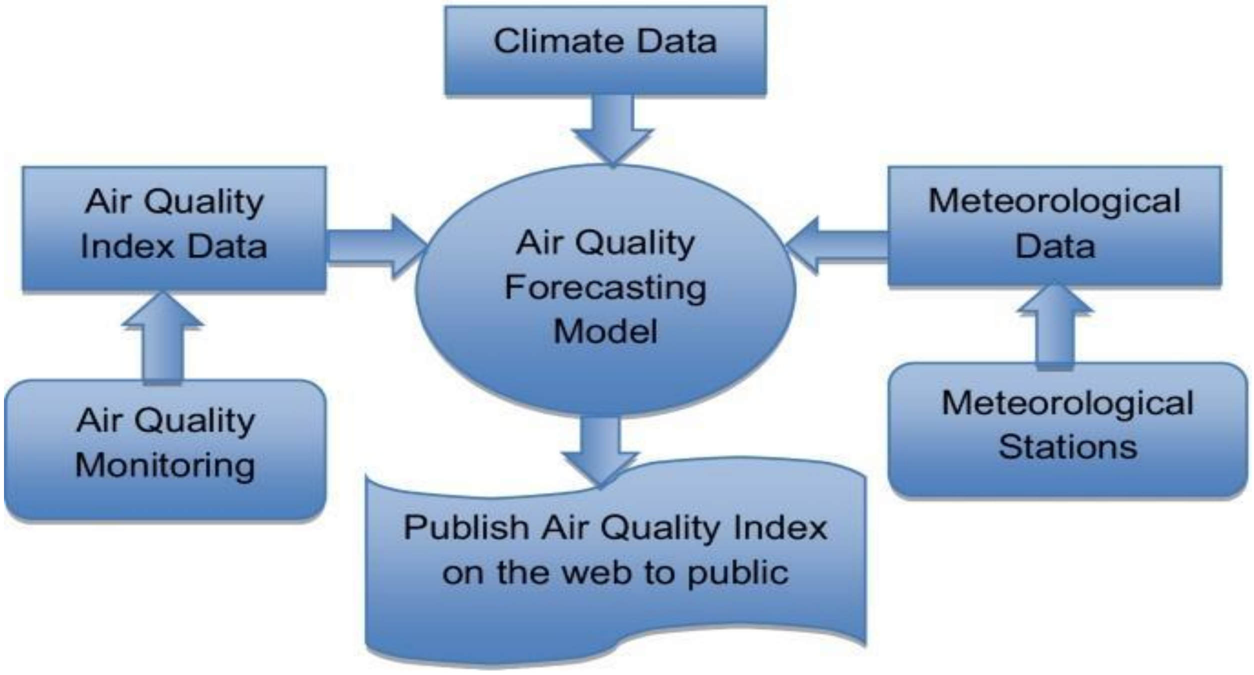
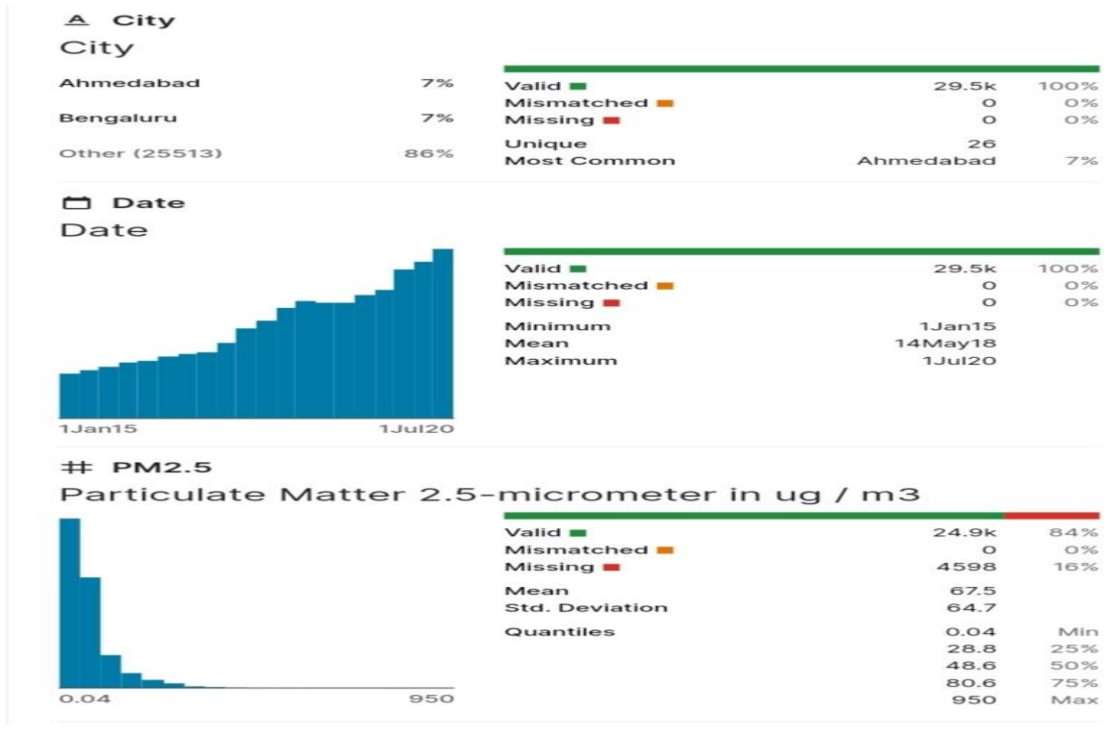
Air Quality Analysis & Prediction



Introduction :

The world we live in is being rapidly automated and emerging technologies like Cloud, Internet of Things, and so forth are being continuously integrated into Concepts such as Smart Cities to provide a high level of comfort to the residents With minimum human intervention [10]. A major challenge faced by corporations Of developed cities is to control and regulate air quality. With the advent of modern Air quality monitoring and pollution control systems, a novel prediction framework Aids the process of finding effective solutions to complex problems. This project aims to predict the air quality band for PM2.5 using present and Historical pollution data in combination with predicted weather data which is Readily available. To solve this problem, firstly, exploratory data analysis will be Conducted on available weather and pollution datasets to discover the correlation Between different features. After employing suitable data cleaning and feature Engineering methods based on the observations made, the feasibility of using Different machine learning techniques such as classification and regression models Will be analysed.

Give Data set :



City
City

Ahmedabad	7%
Bengaluru	7%
Other (25513)	86%

Valid 🟩	29.5k	100%
Mismatched 🟧	0	0%
Missing 🟥	0	0%
Unique	26	
Most Common	Ahmedabad	7%

Date
Date

Valid 🟩	29.5k	100%
Mismatched 🟧	0	0%
Missing 🟥	0	0%
Minimum	1Jan15	
Mean	14May18	
Maximum	1Jul20	

PM2.5
Particulate Matter 2.5-micrometer in ug / m3

Valid 🟩	24.9k	84%
Mismatched 🟧	0	0%
Missing 🟥	4598	16%
Mean	67.5	
Std. Deviation	64.7	
Quantiles	0.04	Min
	28.8	25%
	48.6	50%
	80.6	75%
	950	Max

Necessary step to follow:

1.Import Libraries:

Start by importing the necessary libraries:

Program:

Import pandas as pd

Import matplotlib.pyplot as plt

From sklearn.model_selection import train_test_split

From sklearn.linear_model import LinearRegression

2.Load the Dataset

To load an air quality dataset into your Python program, you can use the Pandas library. Here's a basic example of how to do it:

Program:

Data = pd.read_csv('Air_quality_dataset.csv')

Pd. Read()

3.Exploratory Data Analysis (EDA)

Air quality dataset for analysis and prediction. Here’s a simplified guide to perform EDA on an air quality dataset

4.Features of air quality

Air quality is good, the air is clear and contains only small amounts of solid particle and chemical pollutants. Poor air quality, which contains high levels of pollutants, is often hazy and dangerous to health and the environment.

Importance of loading and processing dataset:

- ### Loading the Dataset:

Data Source: Determine where the dataset is coming from. Is it from a file, a database, an API, or some other source?

Data Format: Understand the format of the dataset, whether it’s a CSV file, Excel spreadsheet, JSON, or in a structured database.

Data Size: Be aware of the size of the dataset. Large datasets may require more memory and processing power.

Data Quality: Check for missing values, duplicates, and data errors. Decide on how to handle these issues.

Timestamps: If the dataset includes a time component, ensure that timestamps are correctly formatted and, if needed, convert them to a consistent time zone.

➢ Data Processing:

Data Cleaning: Clean the data by removing duplicates, handling missing values (imputation), and addressing outliers. Ensure data quality and consistency.

Feature Engineering: Create new features or derive additional information from the existing features if it helps with your analysis or prediction.

Scaling and Normalization: Standardize or normalize numerical features to prevent issues related to varying scales.

Categorical Variables: Encode categorical variables into numerical format if necessary, using methods like one-hot encoding or label encoding.

Time Series Handling: If the data is time-series data, consider techniques such as resampling, interpolation, or time-based feature extraction.

Dimensionality Reduction: If you have a high-dimensional dataset, apply dimensionality reduction techniques like Principal Component Analysis (PCA) or feature selection to reduce complexity.

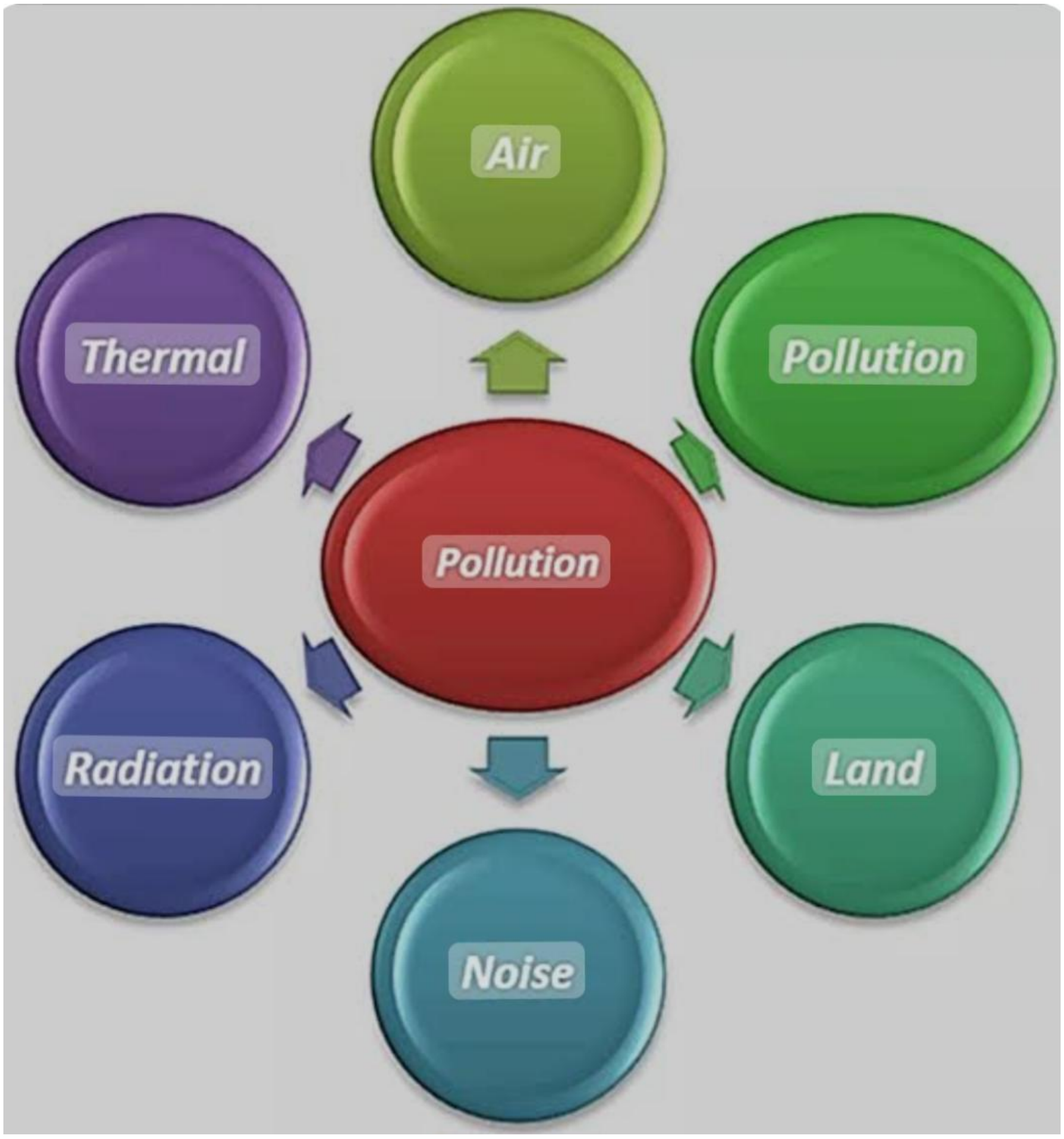
Data Splitting: Split the dataset into training and testing sets for machine learning models. Consider using techniques like cross-validation.

Handling Imbalanced Data: If you're working with classification and your classes are imbalanced, consider techniques like oversampling, undersampling, or using evaluation metrics that account for class imbalances.

Data Integration: If you have multiple datasets or sources, ensure that you integrate and merge them effectively.

Data Transformation: Depending on your analysis or prediction task, you may need to transform the data, such as taking logarithms or performing other mathematical operations.

Air Quality Processing flow Chart :



Program:

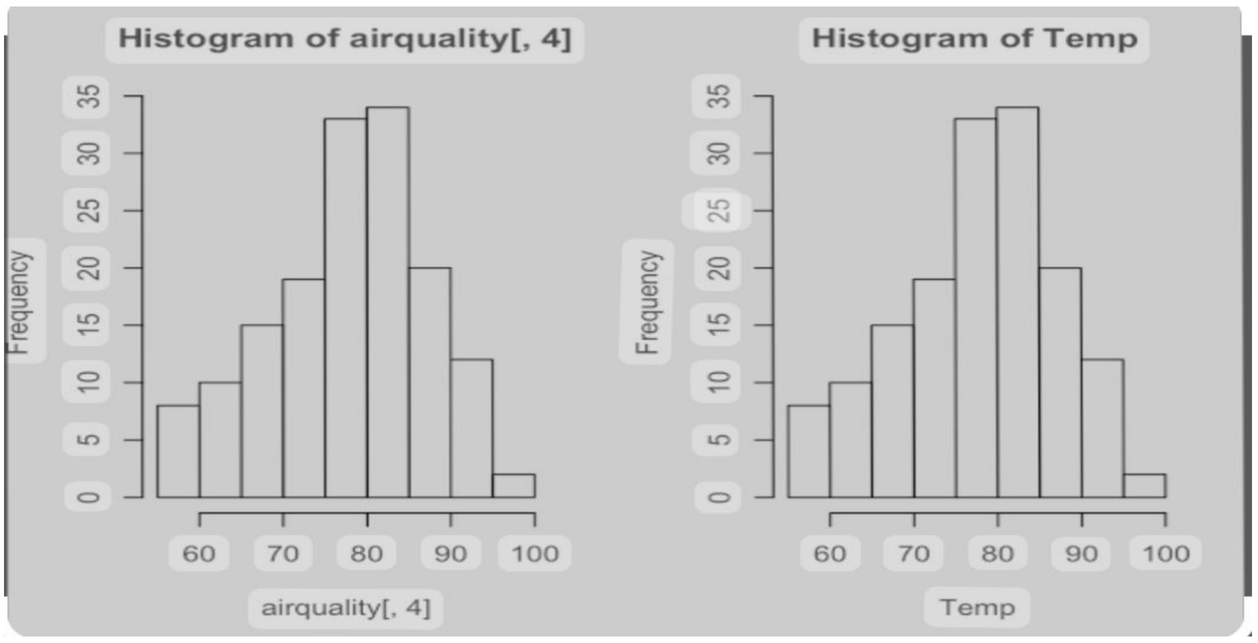
```
# Import necessary libraries

Import pandas as pd

Import numpy as np

# Load the air quality dataset

Data = pd.read_csv('your_air_quality_data.csv')

# Data cleaning and preprocessing

# Handle missing values

Data = data.dropna()  # Or use imputation methods if suitable

# Remove duplicates

Data = data.drop_duplicates()

# Feature engineering

# Create time-based features

Data['date'] = pd.to_datetime(data['date'])

Data['hour'] = data['date'].dt.hour

Data['day_of_week'] = data['date'].dt.dayofweek

# Data transformation

# Normalize data if needed

From sklearn.preprocessing import MinMaxScaler

Scaler = MinMaxScaler()

Data[['feature1', 'feature2']] = scaler.fit_transform(data[['feature1', 'feature2']])

# Log transformation for skewed data

Data['feature3'] = np.log(data['feature3'])

# Outlier detection

# Implement outlier detection methods such as Z-score, IQR, or visualization techniques

# Data splitting for model development

From sklearn.model_selection import train_test_split

X = data[['feature1', 'feature2', 'hour', 'day_of_week']]

Y = data['target_variable']
```

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

# Further model development and analysis

# Now you can use X_train, X_test, y_train, and y_test to build and evaluate your air quality prediction models.

# The above is a simplified example. You should adapt the steps to your specific dataset and analysis needs.
```

Loading Dataset:

```
Data = pd.read_csv('your_air_quality_dataset.csv')
```

Output:



Processing Dataset:

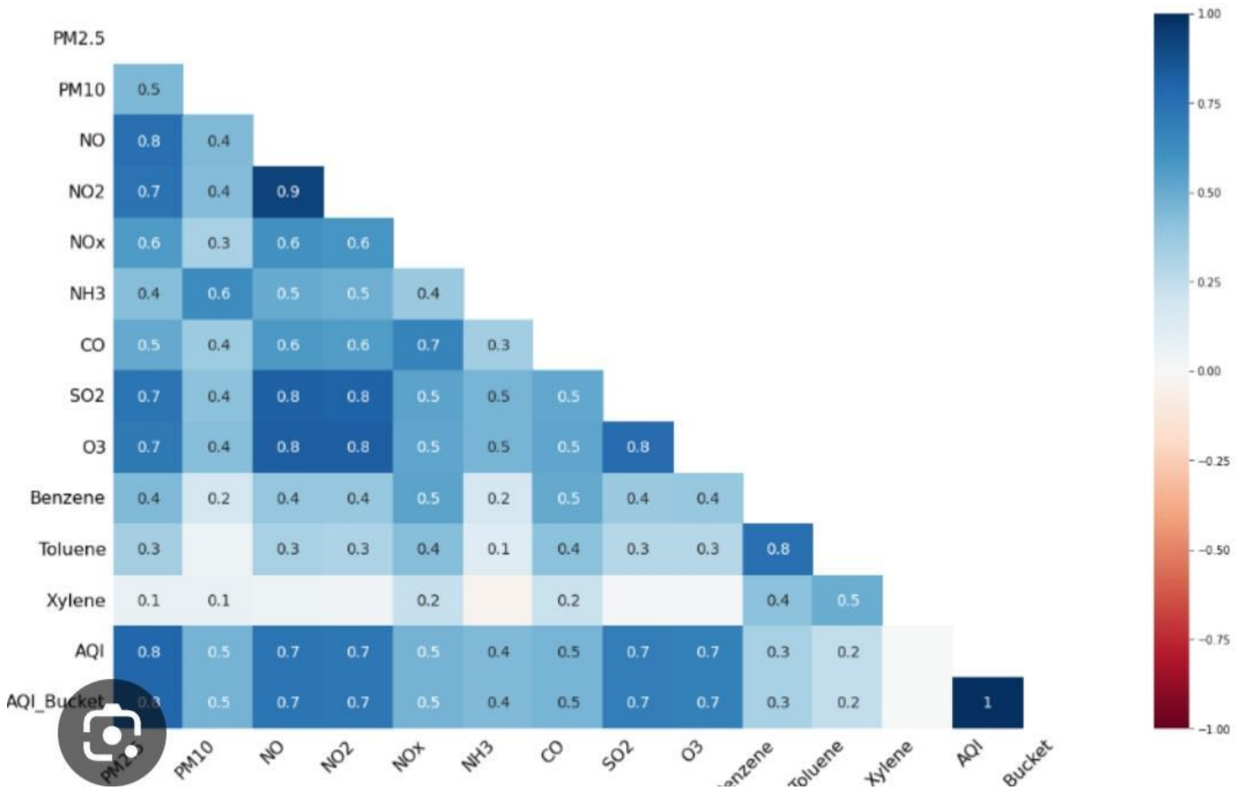
- ❖ Air quality management refers to all the activities a regulatory authority undertakes to help protect human health and the environment from the harmful effects of air pollution.

- ❖ The process of managing air quality can be illustrated as a cycle of inter-related elements.

Visualisation and Pre-Processing of Data:

```
# Load the air quality dataset

Data = pd.read_csv('your_air_quality_dataset.csv')

# Data Visualization

# Plot a histogram of a specific feature (e.g., PM2.5 concentration)

Plt.figure(figsize=(10, 6))

Plt.hist(data['PM2.5'], bins=20, color='blue', alpha=0.7)

Plt.title('Histogram of PM2.5 Concentration')

Plt.xlabel('PM2.5 Concentration')

Plt.ylabel('Frequency')

Plt.show()
```

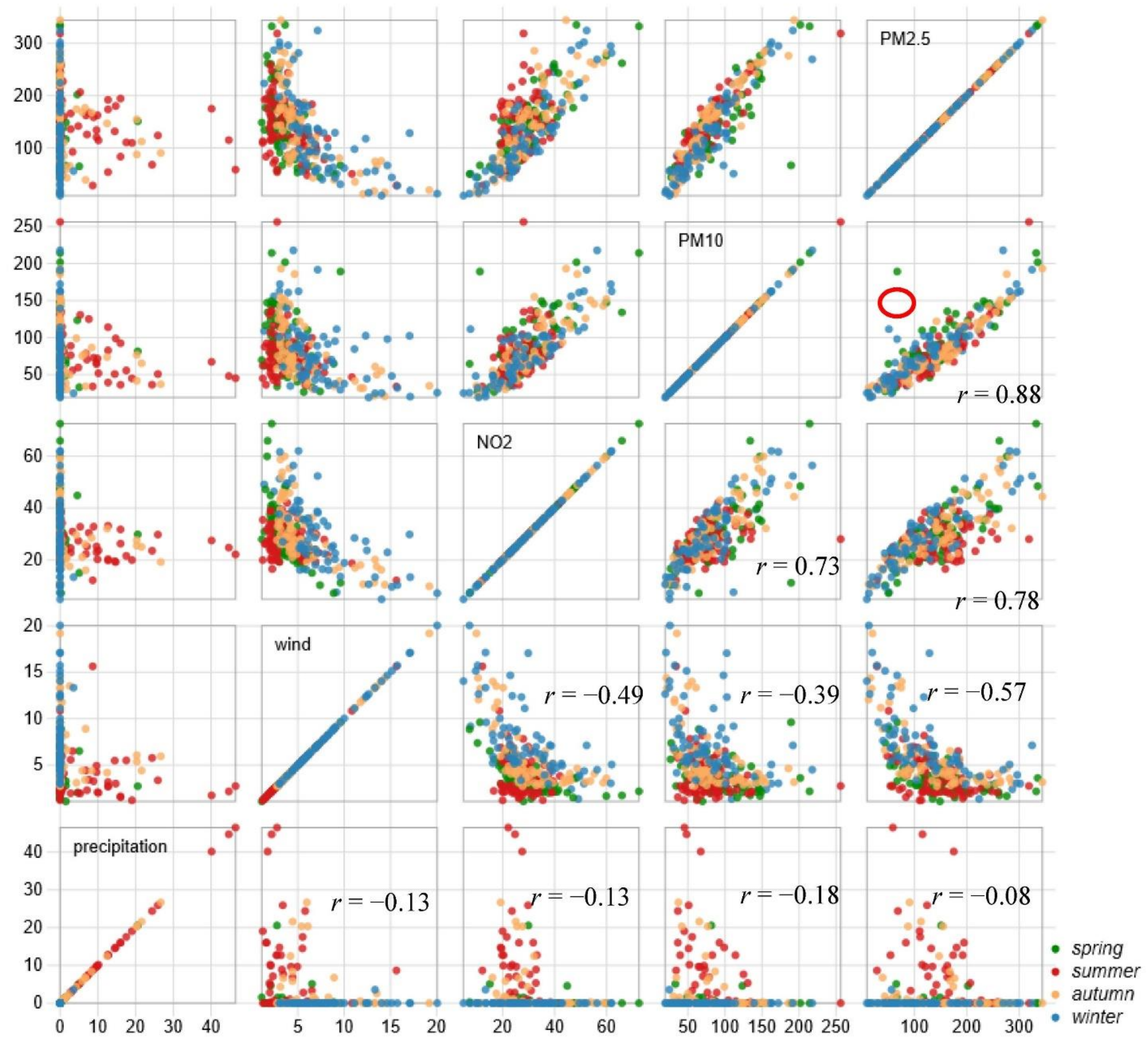
✓ Output:



Visualising Correlation:

```
Import pandas as pd

Import matplotlib.pyplot as plt

Import seaborn as sns


# Load the air quality dataset

Data = pd.read_csv(‘your_air_quality_dataset.csv’)

# Calculate the correlation matrix

Correlation_matrix = data.corr()

# Create a heatmap to visualize the correlations

Plt.figure(figsize=(10, 8))

Sns.heatmap(correlation_matrix, annot=True, cmap=’coolwarm’, linewidths=0.5)

Plt.title(‘Correlation Heatmap for Air Quality Variables’)

Plt.show()

Data = pd.read_csv(‘your_air_quality_data.csv’)

# Data cleaning and preprocessing

# Handle missing values

Data = data.dropna()  # Or use imputation methods if suitable

# Remove duplicates

Data = data.drop_duplicates()

Data = pd.read_csv(‘your_dataset.csv’)

Pd. Read()

Data.read()

Read. Air_quality (‘air-quality.csv’)
```

✓ Output



Some common data pre-processing tasks include:

1. Handling Missing Values: Identify and handle missing data, either by removing rows with missing values or imputing missing values using methods like mean, median, or interpolation.
2. Data Cleaning: Remove duplicates from the dataset if they exist .Check for and correct data errors, such as outliers or inaccuracies.
3. Feature Engineering: Create new features or derive additional information from existing features, such as time-based features (day of the week, time of day), aggregations (daily averages), or ratios (e.g., the ratio of pollutants).
4. Data Transformation: Normalize or standardize numerical features if they have different scales to avoid biases in some machine learning algorithms. Log transformation can be applied to skewed data to make it more normally distributed.

5. Categorical Variables: Encode categorical variables into numerical format, using techniques like one-hot encoding or label encoding if your machine learning models require it.
6. Time Series Handling (if applicable):For time-series air quality data, consider resampling, interpolation, and time-based feature extraction.
7. Dimensionality Reduction (if needed):Apply dimensionality reduction techniques like Principal Component

IOT

Define:

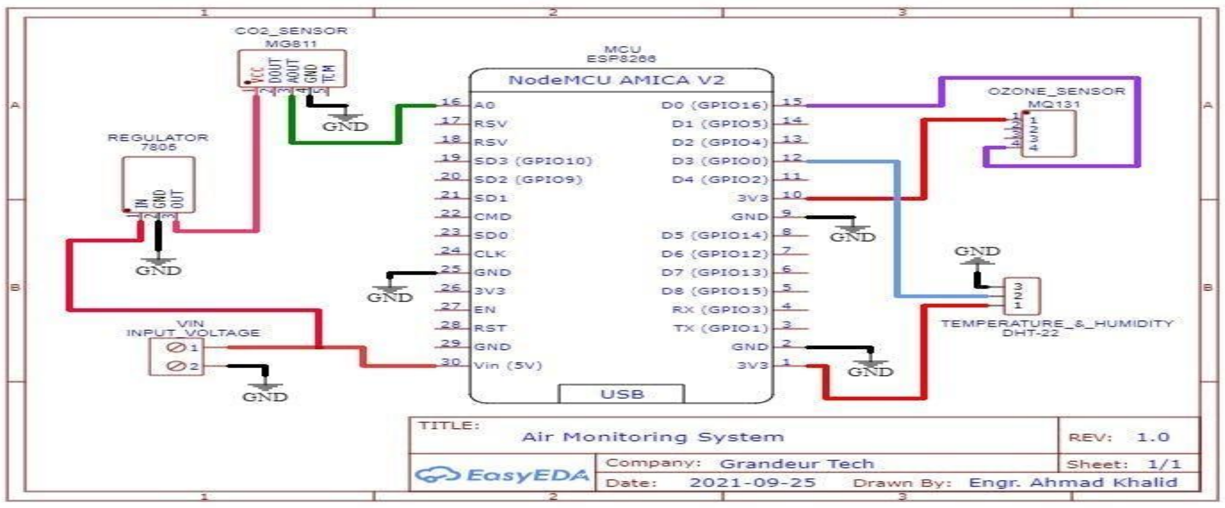
IOT is a network of interconnected physical devices, vehicles, and objects that collect and exchange data via the internet.

Script:

```
Import requests


Def get_air_quality(api_key, latitude, longitude):
  Base_url = https://www.airnowapi.org/aq/observation/latLong/current/
  Params = {
    “latitude”: latitude,
    “longitude”: longitude,
    “format”: “json”,
    “distance”: 25,  # You can adjust the search radius
    “API_KEY”: api_key,  # Get your API key from the AirNow website
  }

  Response = requests.get(base_url, params=params)

  If response.status_code == 200:
    Data = response.json()
    If data:
      # Extract relevant air quality information (e.g., AQI)
      Aqi = data[0][“AQI”]
      Category = data[0][“Category”][“Name”]
```

```
            Print(f”Air Quality Index (AQI): {aqi}”)
            Print(f”Category: {category}”)
        Else:
            Print(“No air quality data available for this location.”)
    Else:
        Print(“Failed to retrieve air quality data.”)


If __name__ == “__main__”:
    # Replace with your API key and coordinates (latitude, longitude)
    Api_key = “YOUR_API_KEY”
    Latitude = 37.7749
    Longitude = -122.4194


    Get_air_quality(api_key, latitude, longitude)
```

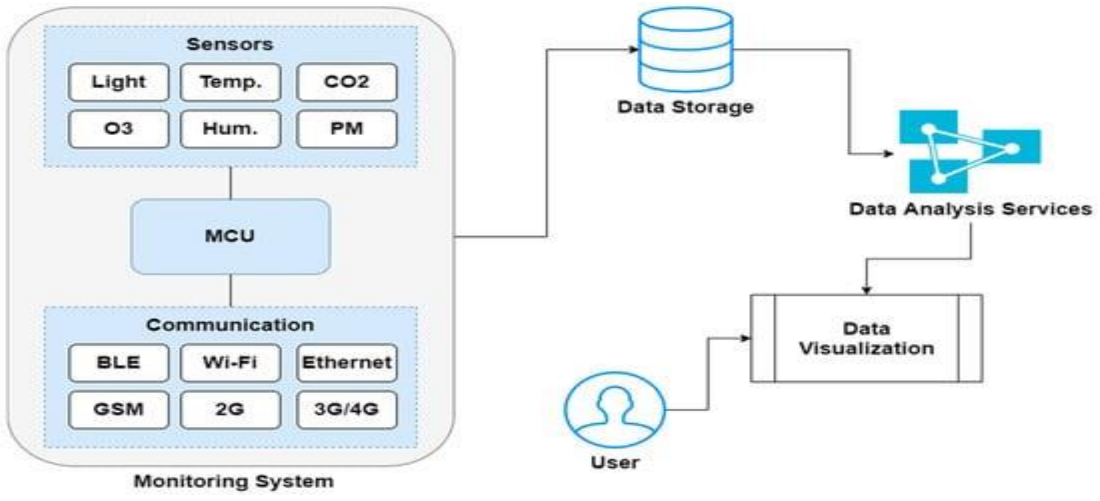
IOT Project Requirements:


CO2_SENSOR
MG811
VCC DOUT AOUT GND TCM
GND
MCU
ESP8266
NodeMCU AMICA V2
A0 RSV RSV SD3 (GPIO10) SD2 (GPIO9) SD1 CMD SD0 CLK GND 3V3 EN RST GND Vin (5V)
D0 (GPIO16) D1 (GPIO5) D2 (GPIO4) D3 (GPIO0) D4 (GPIO2) 3V3 GND D5 (GPIO14) D6 (GPIO12) D7 (GPIO13) D8 (GPIO15) RX (GPIO3) TX (GPIO1) GND 3V3
USB
OZONE_SENSOR
MQ131
REGULATOR
7805
IN GND OUT
VIN
INPUT_VOLTAGE
TEMPERATURE_&_HUMIDITY
DHT-22
GND
TITLE: Air Monitoring System
REV: 1.0
EasyEDA
Company: Grandeur Tech
Sheet: 1/1
Date: 2021-09-25
Drawn By: Engr. Ahmad Khalid

Requirements:

- Clear Objective
- Hardware Components
- Power Source
- Connectivity
- Data Collection
- User Interface

Flow chart:



Conclusion:

- Throughout this project, several models which can predict Pm2.5 levels and Classify them into different pollution bands were experimented and their Performance was successfully evaluated. The exploratory data analysis and Feature engineering methods implemented for the prediction models Revealed interesting correlations between weather and pollution data. We Obtained several notable outcomes from the predictive models that are Worth being discussed.
- Different approaches to handle null values yielded varied performance from Each of the models, however simply dropping the records that had null Values seemed to be the best approach. Between obtaining the AQI by Predicting the PM2.5 values and using a classifier to predict the AQI band Straight away, the classifier seemed to perform better. A regression model Could be used for applications in data analytics, but it is concluded that Classifier models perform better for air quality prediction.