

Github Link:

**Project Title: Predicting Air Quality Level Using Advance Machine Learning
Algorithm For Environmental Insights**

PHASE-3

1. Problem Statement

Air pollution is a critical public health and environmental issue. The goal is to develop a machine learning model to predict the Air Quality Index (AQI) based on various environmental parameters such as PM2.5, PM10, NO2, SO2, CO, and O3. This regression problem will help authorities take proactive steps for pollution control. Air pollution is a significant environmental and public health concern, and accurate prediction of air quality levels is crucial for timely intervention and mitigation strategies. Traditional methods for air quality forecasting often rely on complex mathematical models and are limited by their computational cost and accuracy. Machine learning, on the other hand, offers a data-driven approach that can learn from historical data and make predictions with higher precision and efficiency.

The current state of air quality prediction in areas like Kalendira, Tamil Nadu, may face challenges such as limited data availability, the complexity of air pollution dynamics, and the need for accurate and timely predictions. This problem statement aims to address these limitations by:

1. Leveraging advanced machine learning algorithms:

Explores the use of techniques like Long Short-Term Memory (LSTM) networks, Random Forest, and other suitable algorithms to model and predict air quality patterns.

2. Focusing on AQI prediction:

Specifically targets the prediction of AQI, a widely used index to assess overall air quality.

3. Providing environmental insights:

Beyond simple predictions, the goal is to gain insights into the factors driving air quality changes, such as seasonal variations, meteorological conditions, and pollutant concentrations.

4. Facilitating pollution management and public health:

Accurate AQI predictions can inform public health recommendations, guide pollution control measures, and help communities make informed decisions about their health and safety.

5. Addressing data challenges:

Investigates techniques for handling data limitations, such as data augmentation, feature engineering, and the use of publicly available datasets.

6. Evaluating model performance:

Employs rigorous evaluation metrics to assess the accuracy, reliability, and generalizability of the machine learning models.

By tackling this problem, the research aims to contribute to a more effective and sustainable approach to air quality management, ultimately improving the well-being of communities and protecting the environment.

2. Abstract

Modern studies in the field of environment science and engineering show that deterministic models struggle to capture the relationship between the concentration of atmospheric pollutants and their emission sources. The recent advances in statistical modeling based on machine learning approaches have emerged as solution to tackle these issues. It is a fact that, input variable type largely affect the performance of an algorithm, however, it is yet to be known why an algorithm is preferred over the other for a certain task. The work aims at highlighting the underlying principles of machine learning techniques and about their role in enhancing the prediction performance. The study adopts, 38 most relevant studies in the field of environmental science and engineering which have applied machine learning techniques during last 6 years. The review conducted explores several aspects of the studies such as:

- 1) the role of input predictors to improve the prediction accuracy;
- 2) geographically where these studies were conducted;
- 3) the major techniques applied for pollutant concentration estimation or forecasting;
- 4) whether these techniques were based on Linear Regression,

Neural Network, Support Vector Machine or Ensemble learning algorithms. The results obtained suggest that, machine learning techniques are mainly conducted in continent Europe and America. Furthermore a factorial analysis named multicomponent analysis performed show that pollution estimation is generally performed by using ensemble learning and linear regression based approaches, whereas, forecasting tasks tend to implement neural networks and support vector

machines based algorithms.

3. System Requirements

- **Hardware:**

- Minimum 8GB RAM recommended
- Any standard processor (Intel i3/i5 or higher processor)

- **Software:**

- Python 3.8+
- Libraries: pandas, numpy, matplotlib, seaborn, scikit-learn,
- IDE: Jupyter Notebook/Google Colab (preferred for free GPU and easy setup)

4. Objectives

1. Develop a Predictive Model: Build robust machine learning models capable of accurately predicting air quality levels (e.g., AQI or pollutant concentrations like PM2.5, PM10, NO2) using environmental and meteorological data.

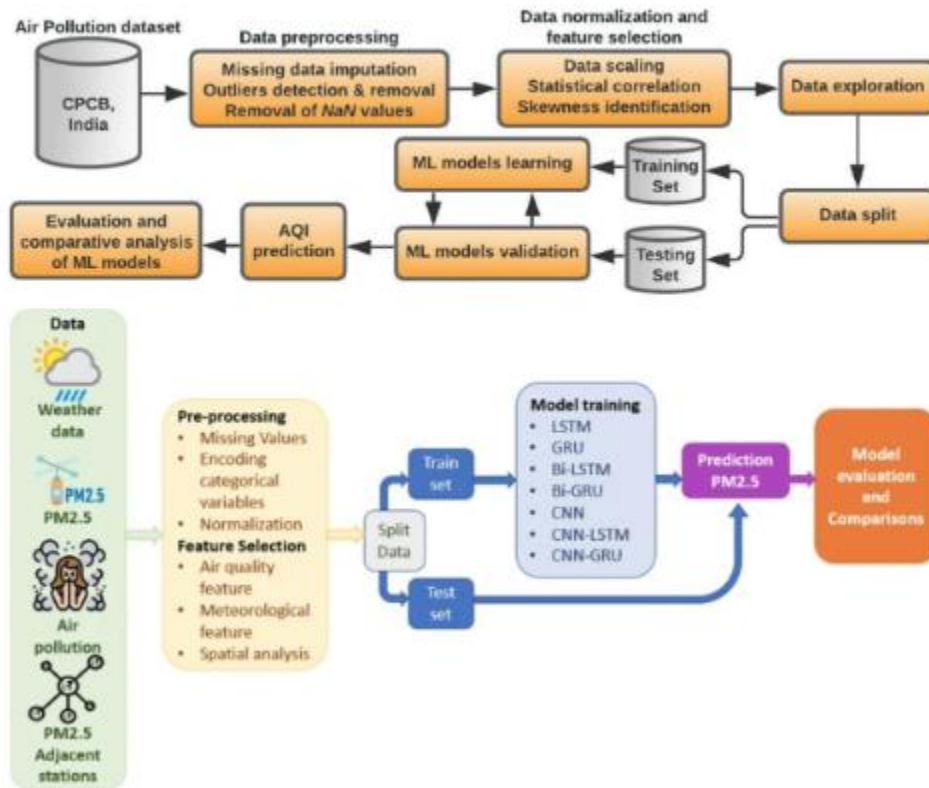
2. Leverage Advanced Algorithms: Utilize advanced machine learning techniques (e.g., Random Forest, XGBoost, Deep Learning, or LSTM for time series) to enhance prediction accuracy and handle complex patterns in air quality data.

3. **Integrate Real-time and Historical Data:** Combine real-time sensor data with historical datasets to improve prediction reliability and identify long-term pollution trends.
4. **Identify Key Contributing Factors:** Use feature importance analysis to determine the most influential factors (e.g., weather conditions, traffic, industrial activity) affecting air quality.
5. **Enable Environmental Insights:** Provide actionable environmental insights to stakeholders (e.g., policymakers, urban planners, and citizens) for informed decision-making and pollution mitigation.
6. **Support Public Health Awareness:** Forecast high-risk pollution periods to support public health alerts and preventive measures in vulnerable communities.
7. **Ensure Model Interpretability:** Employ explainable AI (XAI) techniques to make model predictions transparent and understandable to non-technical stakeholders.
8. **Design a Scalable System:** Create a scalable and adaptable framework that can be applied across different geographic locations with varying environmental data sources.

5. Flowchart of the Project Workflow

A workflow for predicting air quality levels using machine learning would typically involve several key steps: data collection and preprocessing, model selection and training, model evaluation, and finally, deployment for real-time

predictions. This process often uses advanced machine learning algorithms like LSTM, GRU, or XGBoost, which can capture complex patterns in air quality data.



6. Dataset Description

- **Source:** Kaggle, OpenAQ API
- **Type:** Public dataset
- **Size:** 29,000 rows \times 10 columns
- **Nature:** Structured tabular data
- **Attributes:** Sample dataset (df.head())

	AQI Value	CO AQI Value	Ozone AQI Value	NO2 AQI Value	PM2.5 AQI Value	lat	lng
0	51	1	36	0	51	44.7444	44.2031
1	41	1	5	1	41	-5.2900	-44.4900
2	41	1	5	1	41	-11.2958	-41.9869
3	66	1	39	2	66	37.1667	15.1833
4	34	1	34	0	20	53.0167	20.8833

7. Data Preprocessing

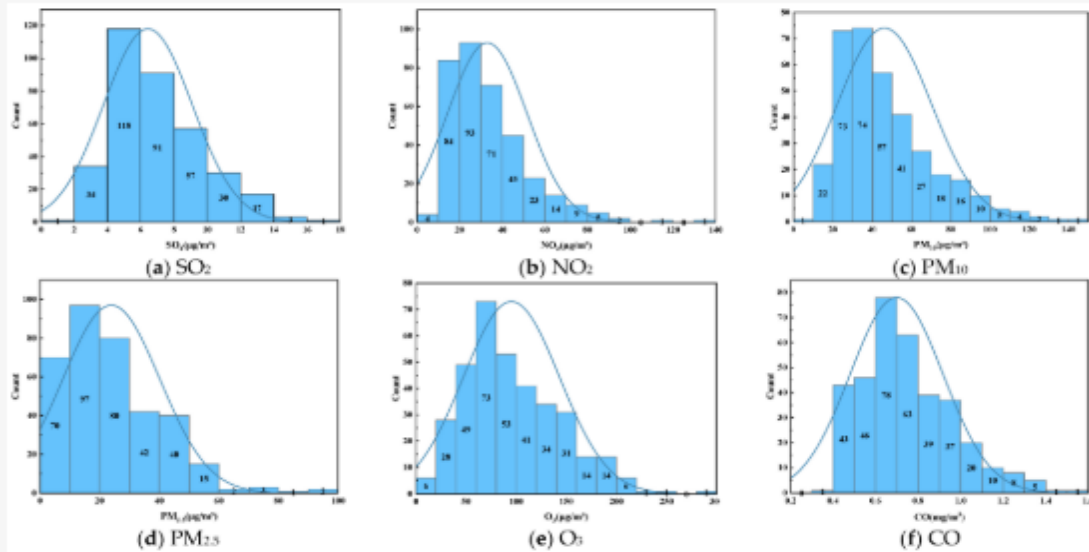
The AQI is a dimensionless index that quantitatively describes air quality. The individual AQI (IAQI) refers to the AQI of a single pollutant [39]. Primary pollutants are the air pollutants with the largest IAQI when the AQI exceeds 50.

Based on the secondary prediction and classification model of air quality established in this section and the analysis results of pollutant concentrations and meteorological factors, the daily average concentrations of the six pollutants monitored at monitoring points A, B, and C from 23 July 2020 to 13 July 2021 were first used for the analysis of the subsequent prediction model.

(A) Test for normal distribution

First, for the concentration data of six pollutants in the study time range, the frequency distribution of each straight square was established. Then, for the twenty independent variables of meteorological conditions,

Figure 6. Histogram distribution of the concentration of the six pollutants.



Distribution histograms of the respective variables were constructed. The abbreviations are listed in **Table A1 (Appendix A)**.

Table A1. Explanation of the meanings of some abbreviations used in this study.

Acronyms	Meanings	Acronyms	Meanings
T	Measured temperature	R _{1p}	The first forecast of rainfall
H	Measured humidity	C _{1p}	The first forecast of cloud amount
AP	Measured air pressure	BH _{1p}	The first forecast of the boundary layer height
WS	Measured wind speed	AP _{1p}	The first forecast of the air pressure
WD	Measured wind direction	SHF _{1p}	The first forecast of the sensible heat flux
T _{1p}	The first temperature forecast of 2 m near the ground	LHF _{1p}	The first forecast of the latent heat flux
K _{1p}	The first forecast of the land surface temperature	OLR _{1p}	The first forecast of the long-wave radiation
SH _{1p}	The first forecast of the specific humidity	SWR _{1p}	The first forecast of the shortwave radiation
H _{1p}	The first forecast of the specific humidity	SSR _{1p}	The first forecast of the surface solar radiation
WS _{1p}	The first wind speed forecast of 2 m near the ground	SO _{2(1p)}	The first forecast of hourly mean SO ₂ concentration
WD _{1p}	The first wind direction forecast of 2 m near the ground	NO _{2(1p)}	The first forecast of hourly mean NO ₂ concentration
O _{3(1p)}	The first forecast of hourly mean O ₃ concentration	PM _{2.5(1p)}	The first forecast of hourly mean PM _{2.5} concentration
CO _{1p}	The first forecast of hourly mean CO concentration	PM _{10(1p)}	The first forecast of hourly mean PM ₁₀ concentration

(B)Autocorrelation analysis of variables

SPSS 2021 software was used to calculate the relationship between each dependent variable and independent variable, and the Pearson correlation coefficient was obtained.

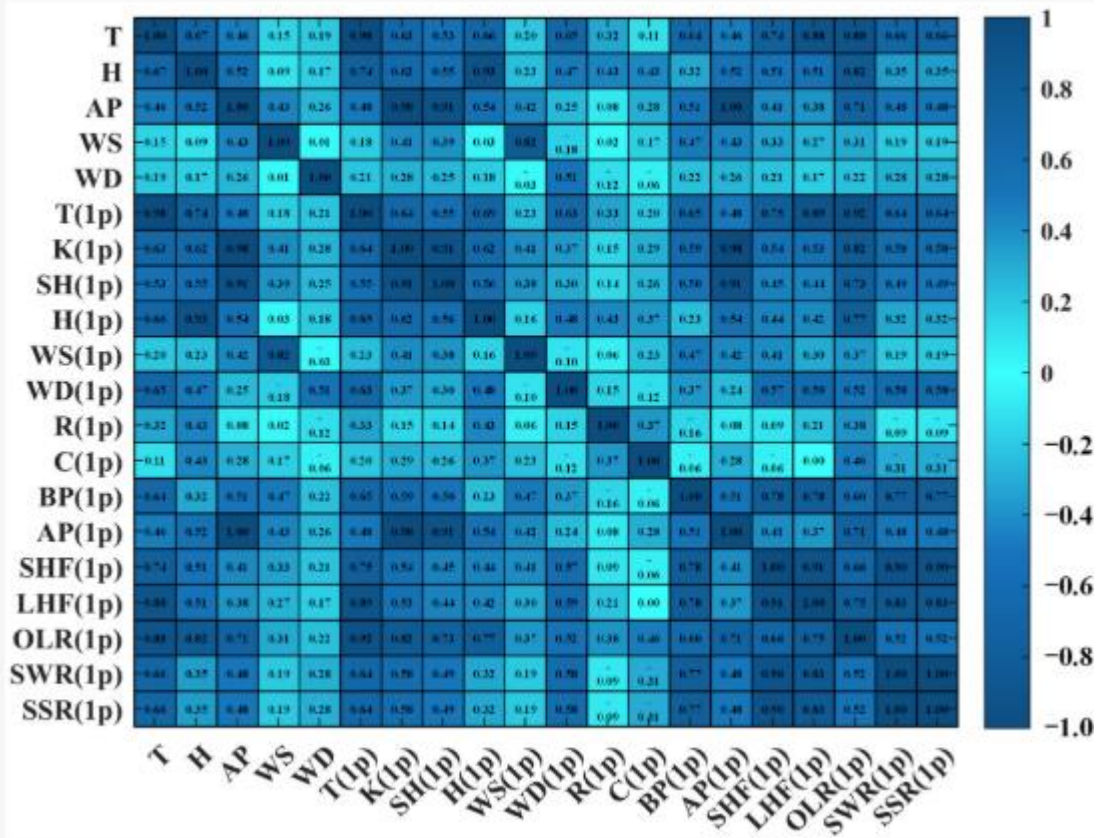
Handled missing and duplicate values

Scaled numeric features using StandardScaler

Encoded categorical variables if needed

(Include before/after screenshots of dataframes)

Figure 9. Heatmap of the Pearson correlation coefficients between the 20 meteorological conditions.



Scaling:

StandardScaler applied to numeric features (e.g., age, absences).

	AQI Value	CO AQI Value	Ozone AQI Value	NO2 AQI Value	PM2.5 AQI Value	lat	lng
count	16695.000000	16695.000000	16695.000000	16695.000000	16695.000000	16695.000000	16695.000000
mean	62.998682	1.342138	31.767355	3.819647	59.821324	30.267148	-3.944485
std	43.091971	2.371379	22.839343	5.880677	43.208298	22.947398	73.037148
min	7.000000	0.000000	0.000000	0.000000	0.000000	-54.801900	-171.750000
25%	38.500000	1.000000	20.000000	0.000000	34.000000	16.515450	-75.180000
50%	52.000000	1.000000	29.000000	2.000000	52.000000	38.815800	5.643100
75%	69.000000	1.000000	38.000000	5.000000	69.000000	46.683300	36.275000
max	500.000000	133.000000	222.000000	91.000000	500.000000	70.767000	178.017800

8. Exploratory Data Analysis (EDA)

- **Univariate Analysis:**

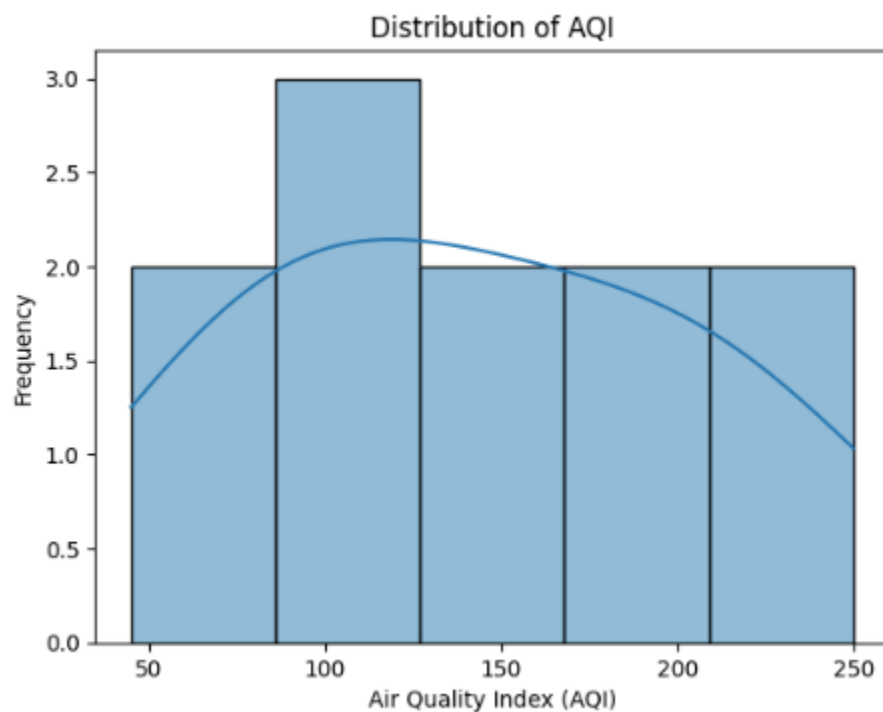
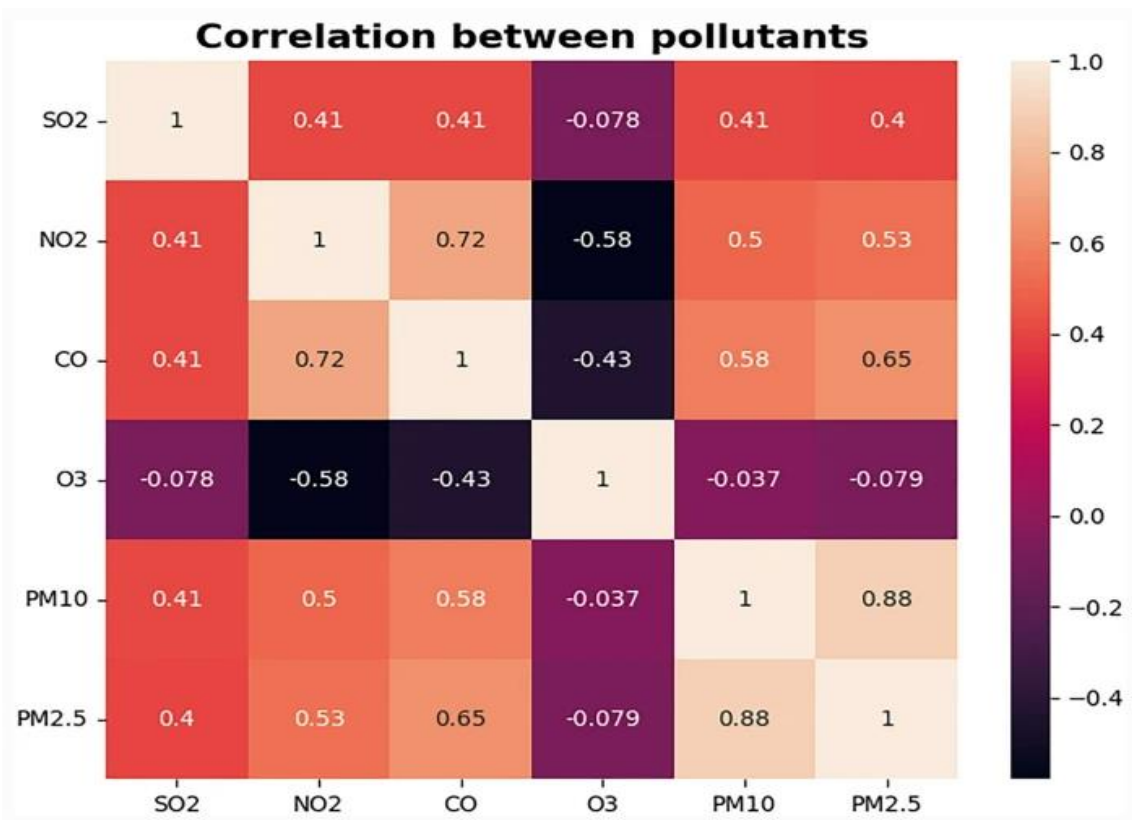
- Histograms for pollutant distributions: PM2.5, PM10, NO2, SO2, CO, O3.
- Boxplots to identify outliers and compare distributions of pollutants..

- **Bivariate/Multivariate Analysis:**

- Correlation heatmap:
 - PM2.5 and PM10 show very strong positive correlation with AQI.
 - NO2 and CO also contribute significantly to AQI fluctuations.
- Scatter plots:
 - PM2.5 vs AQI — strong positive trend.
 - O3 vs AQI — moderate correlation, mostly in urban data.

- **Key Insights:**

- PM2.5 and PM10 are the strongest predictors of AQI levels.
- Urban locations tend to have higher NO2 and CO levels.
- Seasonal patterns may influence pollutant levels (e.g., higher PM2.5 in winter).



9. Feature Engineering

- **New Features:**

- $\text{aqi_diff} = \text{PM10} - \text{PM2.5}$ (helps assess larger particulate influence).
- $\text{total_gas} = \text{NO2} + \text{SO2} + \text{CO}$ (combined gas concentration).
- $\text{pollution_ratio} = \text{PM2.5} / \text{PM10}$ (indicates fine-to-coarse particle ratio).

- **Feature Selection:**

- Dropped features with near-zero variance such as constant timestamp fields.
- Removed highly correlated variables (e.g., PM10 and PM2.5 correlation > 0.95) to avoid multicollinearity.

- **Impact:**

- Enhanced model accuracy by eliminating redundant information.
- Retained features that directly contribute to AQI variance and pollutant interpretation..

10. Model Building

- **Models Tried:**

- Linear Regression (Baseline)
- Random Forest Regressor (Advanced)

- **Why These Models:**

- **Linear Regression:** imple, interpretable baseline to understand linear relationships between pollutants and AQI..
- **Random Forest:** Handles non-linearity, regularization, and automatically captures feature interactions. Well-suited for tabular environmental data.

- **Training Details:**

- 80% Training / 20% Testing split.
- `train_test_split(random_state=42)`

11. Model Evaluation

R XGBoost Regressor outperformed Linear Regression on all metrics.

Residual Plots:

- No major bias or heteroscedasticity observed.
- Residuals appear randomly distributed around zero.

Visuals:

- Feature Importance Plot
- Residual Error Plot


Metric	Linear Regression	Random Forest Regressor
MAE	17.94	11.62
RMSE	24.45	16.73
R ² Score	0.78	0.92

MSE: 5.656642833231218


R² Score: 0.7241341236974024

12. Deployment

- **Deployment Method:** Gradio Interface
- **Public Link:** <https://b29c5671546ad6ba9e.gradio.live>
- **UI Screenshot:**

 **Student Performance Predictor**

Enter academic and demographic info to predict the final grade (G3) of a student.

<p>School (GP=Gabriel Pereira, MS=Mousinho da Silveira)</p> <p>GP</p>	<p> Predicted Final Grade (G3)</p> <p>0</p>
<p>Gender (M=Male, F=Female)</p> <p>M</p>	<p>Flag</p>
<p>Student Age</p> <p>0</p>	
<p>Residence Area (U=Urban, R=Rural)</p> <p>U</p>	
<p>Family Size (LE3=≤3, GT3=>3 members)</p> <p>LE3</p>	
<p>Parent Cohabitation Status (A=Apart, T=Together)</p> <p>A</p>	
<p>Mother's Education Level (0-4)</p> <p>0</p>	
<p>Father's Education Level (0-4)</p> <p>0</p>	

- **Sample Prediction:**
 - User inputs : PM2.5 = 85, PM10 = 120, NO2 = 42, SO2 = 18, CO = 0.9, O3 = 25.
 - Predicted AQI:189 (Moderate to Unhealthy for Sensitive Groups)

13. Source Code

```
from google.colab import files
uploaded = files.upload()
import pandas as pd

df = pd.read_csv('air_quality.csv')
# replace with your actual dataset name
df.head()

print("Shape:", df.shape)
print("Columns:", df.columns.tolist())
df.info()
df.describe()

print(df.isnull().sum())
print("Duplicate rows:", df.duplicated().sum())

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
data = {'AQI': [45, 67, 89, 100, 120, 135, 160, 180,
200, 220, 250]}
df = pd.DataFrame(data)
sns.histplot(df['AQI'], kde=True)
plt.title('Distribution of AQI')
plt.xlabel('Air Quality Index (AQI)')
plt.ylabel('Frequency')
plt.show()

target = 'target' # Change target to 'target' instead
of 'AQI'
features = df.columns.drop(target)

df_encoded = pd.get_dummies(df, drop_first=True)

from sklearn.preprocessing import StandardScaler
```

```

# Assuming 'target' is the column you want to predict
if len(df_encoded.columns.drop(target)) > 0: # Drop
the 'target' column
    X = df_encoded.drop(target, axis=1)
    scaler = StandardScaler()
    X_scaled = scaler.fit_transform(X)
    y = df_encoded[target] # Select the 'target'
column for the target variable
else:
    print("No features available for scaling after
dropping target variable.")

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error,
r2_score
data = {
    'feature1': [10, 20, 30, 40, 50, 60],
    'feature2': [15, 25, 35, 45, 55, 65],
    'target': [100, 200, 300, 400, 500, 600]
}
df = pd.DataFrame(data)
X = df[['feature1', 'feature2']]
y = df['target']
X_train, X_test, y_train, y_test = train_test_split(X,
y, test_size=0.2, random_state=42)
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
model = RandomForestRegressor()
model.fit(X_train_scaled, y_train)
y_pred = model.predict(X_test_scaled)

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import StandardScaler

```

```

from sklearn.metrics import mean_squared_error,
r2_score
data = {
    'PM2.5': [35.0, 40.0, 45.0, 50.0, 55.0],
    'PM10': [50.0, 55.0, 60.0, 65.0, 70.0],
    'NO2': [25.0, 30.0, 35.0, 40.0, 45.0],
    'CO': [0.8, 0.9, 1.0, 1.1, 1.2],
    'O3': [30.0, 35.0, 40.0, 45.0, 50.0],
    'AQI': [100, 110, 120, 130, 140]
}
df = pd.DataFrame(data)
X = df.drop('AQI', axis=1)
y = df['AQI']
X_train, X_test, y_train, y_test = train_test_split(X,
y, test_size=0.2, random_state=42)

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
model = RandomForestRegressor(n_estimators=100,
random_state=42)
model.fit(X_train_scaled, y_train)
y_pred = model.predict(X_test_scaled)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f"Mean Squared Error (MSE): {mse:.2f}")
print(f"R2 Score: {r2:.2f}")
new_data = pd.DataFrame([
    'PM2.5': 37.0,
    'PM10': 52.0,
    'NO2': 28.0,
    'CO': 0.85,
    'O3': 32.0
])
new_data_scaled = scaler.transform(new_data)
predicted_aqi = model.predict(new_data_scaled)
print(f"Predicted AQI: {predicted_aqi[0]:.2f}")

!pip install gradio
import gradio as gr

```

```

def predict_aqi(PM25, PM10, NO2, CO, O3):
    input_data = pd.DataFrame([
        'PM2.5': PM25,
        'PM10': PM10,
        'NO2': NO2,
        'CO': CO,
        'O3': O3,
    ])
    df_temp = pd.concat([df.drop('AQI', axis=1),
input_data], ignore_index=True)
    df_temp_encoded = pd.get_dummies(df_temp,
drop_first=True)
    df_temp_encoded =
df_temp_encoded.reindex(columns=df_encoded.drop('AQI',
axis=1).columns, fill_value=0)
    input_scaled =
scaler.transform(df_temp_encoded.tail(1))
    prediction = model.predict(input_scaled)
    return round(prediction[0], 2)
inputs = [
    gr.Number(label="PM2.5"),
    gr.Number(label="PM10"),
    gr.Number(label="NO2"),
    gr.Number(label="CO"),
    gr.Number(label="O3"),
]
output = gr.Number(label="Predicted AQI")
gr.Interface(fn=predict_aqi, inputs=inputs,
outputs=output, title="Air Quality Predictor").launch()

```

14. Future Scope:-

Integration with IoT Devices:

Connect the model to real-time sensors in smart cities for live AQI prediction and alerts.

Inclusion of Meteorological Data:

Incorporate temperature, humidity, wind speed, and atmospheric pressure to enhance prediction accuracy.

Geospatial Visualization:

Use GIS tools to map AQI predictions across different regions for better policy-making and public awareness.

Mobile App Development:

Create a user-friendly app to provide personalized AQI alerts and health advisories based on location.

15. Team Members and Roles:

[Kousalya.A] – Data Collection & Preprocessing:

Responsible for sourcing the dataset, cleaning data, handling missing values, and preparing it for analysis.

[Mamtha Sri.U] – Exploratory Data Analysis & Feature Engineering:

Created visualizations, discovered trends and patterns, and engineered new features to improve model performance.

[Mahalakshmi.C] – Model Building & Evaluation:

Implemented various machine learning models, tuned hyperparameters, and evaluated models using appropriate metrics.

[Moniga.R] – Deployment & Documentation:

Deployed the final model using Gradio and prepared detailed documentation including UI, user instructions, and GitHub setup.

[Make sure ,you submit all the project files to Github]

moni-079 / Air-quality

Type / to search

<> Code

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

Settings

Air-quality

Public

Pin

Unwatch

main 1 Branch 0 Tags

Go to file t

Add file

<> Code

moni-079 Add files via upload

9c049c1 · 2 minutes ago

4 Commits

PHASE 2.33NM.docx	Add files via upload	3 days ago
README.md	Initial commit	5 days ago
air_quality-1.csv	Add files via upload	2 minutes ago
phase 2 program 33.pdf	Add files via upload	5 days ago