🌍 Project Summary: Environmental Pollution Data Analysis

This project analyzes environmental pollution data—specifically focusing on air quality indicators such as PM2.5, PM10, NO2, CO, O3, and SO2—to understand trends, correlations, and predict pollution levels using machine learning.

📂 Dataset Overview

The dataset contains daily pollution metrics for multiple cities, with the following columns:

Date – Observation date

City – City name (e.g., CityA, CityB)

PM2.5 – Fine particulate matter concentration (µg/m³)

PM10, NO2, CO, O3, SO2 – Other air pollutants

📌 Sample Entry:

Date, City, PM2.5, PM10, NO2, CO, O3, SO2

2023-01-01, CityA, 12, 34, 23,1.2, 45, 7

2023-01-01, CityB, 18, 40, 30,1.5, 50, 10

🧹 Step 1–3: Data Loading and Cleaning

Loaded dataset using pandas.

Checked for missing values and filled them using the mean of each column.

Ensured date format consistency and set 'Date' as the index.

📊 Step 4: Exploratory Data Analysis (EDA)

✅ Statistical Summary

Used .describe() to explore min, max, mean, and standard deviation of pollution levels.

✅ Correlation Matrix

Computed and visualized correlation among pollutants.

Found potential linear relationships (e.g., PM2.5 vs PM10).

✅ Visual Insights

1. Heatmap of correlations — highlights strong/weak dependencies.

2. Distribution plot of PM2.5 — helps detect skewness/outliers.

3. Time series of PM2.5 — trends and seasonality over time.

4. Boxplot by City — city-wise pollution comparisons.

5. Multiple time series — pollutant-wise trends in one view.

🤖 Step 5: Machine Learning (Regression)

Used Random Forest Regressor to predict PM2.5 based on other pollutants.

✅ Model Pipeline

Features: PM10, NO2, CO, O3, SO2

Target: PM2.5

Train-Test Split: 80% train, 20% test

Model: RandomForestRegressor with 100 trees

✅ Evaluation

Used Mean Squared Error (MSE) to evaluate predictions.

Displayed feature importance, revealing which pollutants most influence PM2.5.

📈 Feature Importance Example

(Varies based on data, but typically):

PM10 0.45

NO2 0.25

CO 0.15

O3 0.10

SO2 0.05

🧠 Insights & Applications

PM10 and NO2 often show strong correlation with PM2.5.

Time series reveal spikes in pollution levels, possibly due to events (e.g., festivals, traffic).

Cities can be compared to identify high-risk zones.

Machine learning models help predict future pollution and assist policy planning.

📁 Next Steps / Suggestions

Fill missing dataset rows (like the incomplete entry on 2023-01-02 for CityB).

Include weather features (like wind, humidity) for better accuracy.

Deploy model into a dashboard using Streamlit or Plotly Dash.

Add real-time sensor integration for live pollution monitoring.

Source code

Environmental Pollution Data Analysis

Step 1: Import Libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error
```

Step 2: Load the Dataset

Replace 'pollution_data.csv' with your dataset file path

```
df = pd.read_csv('pollution_data.csv')
```

Display the first few rows of the dataset

```
print("Dataset Head:")
print(df.head())
```

Step 3: Data Cleaning

Check for missing values

```
print("\nMissing Values:")
print(df.isnull().sum())
```

Fill missing values with the mean (or any other strategy)

```
df.fillna(df.mean(), inplace=True)
```

Step 4: Exploratory Data Analysis (EDA)

Summary statistics

```
print("\nSummary Statistics:")
print(df.describe())
```

Correlation matrix

```
print("\nCorrelation Matrix:")
corr_matrix = df.corr()
```

```
print(corr_matrix)

Visualize correlation matrix

plt.figure(figsize=(10, 6))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()

Distribution of PM2.5

plt.figure(figsize=(8, 6))
sns.histplot(df['PM2.5'], kde=True)
plt.title('Distribution of PM2.5')
plt.show()

Convert 'Date' column to datetime and set as index

df['Date'] = pd.to_datetime(df['Date'])
df.set_index('Date', inplace=True)

Trend of PM2.5 over time

df['PM2.5'].plot()
plt.title('PM2.5 Over Time')
plt.xlabel('Date')
plt.ylabel('PM2.5 (µg/m³)')
plt.show()

Boxplot of pollutants by city
```

```
plt.figure(figsize=(12, 6))
sns.boxplot(x='City', y='PM2.5', data=df)
plt.title('PM2.5 Levels by City')
plt.show()
```

Time series of multiple pollutants

```
df[['PM2.5', 'PM10', 'NO2', 'CO', 'O3', 'SO2']].plot(subplots=True, figsize=(12, 10))
plt.suptitle('Time Series of Pollutants')
plt.show()
```

Step 5: Machine Learning Model (Optional)

Prepare data for machine learning

```
X = df[['PM10', 'NO2', 'CO', 'O3', 'SO2']]
y = df['PM2.5']
```

Split the data into training and testing sets

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

Train a Random Forest model

```
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
```

Predict and evaluate

```
y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
```

```
print(f'\nMean Squared Error: {mse}')
```

Feature importance

```
feature_importance = pd.Series(model.feature_importances_, index=X.columns)
feature_importance.sort_values(ascending=False, inplace=True)
print("\nFeature Importance:")
print(feature_importance)
```

Plot feature importance

```
plt.figure(figsize=(10, 6))
sns.barplot(x=feature_importance, y=feature_importance.index)
plt.title('Feature Importance')
plt.xlabel('Importance Score')
plt.ylabel('Features')
plt.show()
```

Data set

```
Date,City,PM2.5,PM10,NO2,CO,O3,SO2
2023-01-01,CityA,12,34,23,1.2,45,7
2023-01-02,CityA,15,36,25,1.3,47,8
2023-01-03,CityA,14,35,24,1.1,46,7
2023-01-01,CityB,18,40,30,1.5,50,10
2023-01-02,CityB,20,42,
```