Air Pollution Data Analysis

May 23, 2023

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

[2]:
```
dataset_path = r'C:\Users\user\Desktop\portfolio\air␣
↪pollution\death-rates-from-air-pollution.csv.csv'
data = pd.read_csv(dataset_path)
```

[3]:
```
data.head()
```

[3]:
```
        Entity Code  Year  \
0  Afghanistan  AFG  1990
1  Afghanistan  AFG  1991
2  Afghanistan  AFG  1992
3  Afghanistan  AFG  1993
4  Afghanistan  AFG  1994

   Deaths - Cause: All causes - Risk: Household air pollution from solid fuels -
Sex: Both - Age: Age-standardized (Rate)  \
0                                          370.050474
1                                          358.978418
2                                          352.766453
3                                          357.055923
4                                          362.970439

   Deaths - Cause: All causes - Risk: Ambient particulate matter pollution -
Sex: Both - Age: Age-standardized (Rate)  \
0                                           30.822693
1                                           29.826184
2                                           29.202030
3                                           29.429702
4                                           29.813259

   Deaths - Cause: All causes - Risk: Air pollution - Sex: Both - Age: Age-
standardized (Rate)  \
0                                          402.175651
1                                          390.085258
2                                          383.201196
```

```
3                                                 387.704919
4                                                 394.022027

   Deaths - Cause: All causes - Risk: Ambient ozone pollution - Sex: Both - Age:
Age-standardized (Rate)
0                                                   6.581093
1                                                   6.267613
2                                                   5.926444
3                                                   5.860345
4                                                   6.065343
```

[4]:
```
# Examine the structure of the dataset
print("Dataset Shape:")
print(data.shape)  # Prints the number of rows and columns in the dataset

# Check the column names
print("\nColumn Names:")
print(data.columns)  # Prints the column names

# Check the data types
print("\nData Types:")
print(data.dtypes)  # Prints the data types of each column

# Display the first few rows of the dataset
print("\nFirst few rows:")
print(data.head())

# Get overall statistics of the dataset
print("\nOverall Statistics:")
print(data.describe())
```

```
Dataset Shape:
(6840, 7)

Column Names:
Index(['Entity', 'Code', 'Year',
       'Deaths - Cause: All causes - Risk: Household air pollution from solid
fuels - Sex: Both - Age: Age-standardized (Rate)',
       'Deaths - Cause: All causes - Risk: Ambient particulate matter pollution
- Sex: Both - Age: Age-standardized (Rate)',
       'Deaths - Cause: All causes - Risk: Air pollution - Sex: Both - Age: Age-
standardized (Rate)',
       'Deaths - Cause: All causes - Risk: Ambient ozone pollution - Sex: Both -
Age: Age-standardized (Rate)'],
      dtype='object')

Data Types:
Entity
```

```
object
Code
object
Year
int64
Deaths - Cause: All causes - Risk: Household air pollution from solid fuels -
Sex: Both - Age: Age-standardized (Rate)    float64
Deaths - Cause: All causes - Risk: Ambient particulate matter pollution - Sex:
Both - Age: Age-standardized (Rate)        float64
Deaths - Cause: All causes - Risk: Air pollution - Sex: Both - Age: Age-
standardized (Rate)                                float64
Deaths - Cause: All causes - Risk: Ambient ozone pollution - Sex: Both - Age:
Age-standardized (Rate)                     float64
dtype: object

First few rows:
        Entity Code  Year  \
0  Afghanistan  AFG  1990
1  Afghanistan  AFG  1991
2  Afghanistan  AFG  1992
3  Afghanistan  AFG  1993
4  Afghanistan  AFG  1994

   Deaths - Cause: All causes - Risk: Household air pollution from solid fuels -
Sex: Both - Age: Age-standardized (Rate)  \
0                                          370.050474
1                                          358.978418
2                                          352.766453
3                                          357.055923
4                                          362.970439

   Deaths - Cause: All causes - Risk: Ambient particulate matter pollution -
Sex: Both - Age: Age-standardized (Rate)  \
0                                           30.822693
1                                           29.826184
2                                           29.202030
3                                           29.429702
4                                           29.813259

   Deaths - Cause: All causes - Risk: Air pollution - Sex: Both - Age: Age-
standardized (Rate)  \
0                                          402.175651
1                                          390.085258
2                                          383.201196
3                                          387.704919
4                                          394.022027

   Deaths - Cause: All causes - Risk: Ambient ozone pollution - Sex: Both - Age:
```

```
Age-standardized (Rate)
0                                                 6.581093
1                                                 6.267613
2                                                 5.926444
3                                                 5.860345
4                                                 6.065343

Overall Statistics:
              Year  \
count  6840.000000
mean   2004.500000
std       8.656074
min    1990.000000
25%    1997.000000
50%    2004.500000
75%    2012.000000
max    2019.000000

       Deaths - Cause: All causes - Risk: Household air pollution from solid
fuels - Sex: Both - Age: Age-standardized (Rate)  \
count                                          6840.000000
mean                                             72.260126
std                                              90.126896
min                                               0.004378
25%                                               1.338287
50%                                              24.129663
75%                                             134.481277
max                                             510.816365

       Deaths - Cause: All causes - Risk: Ambient particulate matter pollution -
Sex: Both - Age: Age-standardized (Rate)  \
count                                          6840.000000
mean                                             44.662894
std                                              32.808268
min                                               2.482318
25%                                              21.680712
50%                                              34.875481
75%                                              59.459187
max                                             205.579664

       Deaths - Cause: All causes - Risk: Air pollution - Sex: Both - Age: Age-
standardized (Rate)  \
count                                          6840.000000
mean                                            118.233834
std                                              90.102395
min                                               2.660529
25%                                              39.491427
50%                                              99.354684
```

```
75%                                                 186.084312
max                                                 527.894093

       Deaths - Cause: All causes - Risk: Ambient ozone pollution - Sex: Both -
Age: Age-standardized (Rate)
count                                              6840.000000
mean                                                  2.300575
std                                                   3.347720
min                                                   0.000000
25%                                                   0.624435
50%                                                   1.375799
75%                                                   2.460607
max                                                  35.429423
```

[5]:
```
# Convert 'Year' column to datetime data type
data['Year'] = pd.to_datetime(data['Year'], format='%Y')

# Perform one-hot encoding for categorical variables (if applicable)
# Example: If 'Entity' column is categorical, convert it to one-hot encoded␣
 ↪dummy variables
data_encoded = pd.get_dummies(data, columns=['Entity'])

# Perform feature engineering or create new columns (if applicable)
# Example: Calculate the sum of death rates from all causes
data_encoded['Total_Death_Rate'] = data_encoded.iloc[:, 3:].sum(axis=1)

# Drop irrelevant columns (if applicable)
# Example: Drop the 'Code' column
data_encoded = data_encoded.drop('Code', axis=1)
```

[6]:
```
# Check for missing values
missing_values = data.isnull().sum()

# Print the number of missing values for each column
print("Missing Values:")
print(missing_values)
```

```
Missing Values:
Entity
0
Code
690
Year
0
Deaths - Cause: All causes - Risk: Household air pollution from solid fuels -
Sex: Both - Age: Age-standardized (Rate)      0
Deaths - Cause: All causes - Risk: Ambient particulate matter pollution - Sex:
Both - Age: Age-standardized (Rate)           0
```

```
Deaths - Cause: All causes - Risk: Air pollution - Sex: Both - Age: Age-
standardized (Rate)                                    0
Deaths - Cause: All causes - Risk: Ambient ozone pollution - Sex: Both - Age:
Age-standardized (Rate)                          0
dtype: int64
```

[7]:
```
missing_percentage = (missing_values['Code'] / data.shape[0]) * 100
print("Missing Values Percentage:", missing_percentage)
```

```
Missing Values Percentage: 10.087719298245613
```

[8]:
```
# Set "Africa" in the 'Code' column based on condition
data.loc[data['Entity'] == 'African Region (WHO)', 'Code'] = 'Africa'
```

[9]:
```
# Set "East Asia & Pacific" in the 'Code' column based on condition
data.loc[data['Entity'] == 'East Asia & Pacific (WB)', 'Code'] = 'East Asia &␣
↪Pacific'
```

[10]:
```
# Replace null values in 'Code' column with "Eastern Mediterranean Region"␣
↪based on condition
data.loc[data['Entity'] == 'Eastern Mediterranean Region (WHO)', 'Code'].
↪fillna("Eastern Mediterranean Region", inplace=True)
```

[11]:
```
# Set "Eastern Mediterranean" in the 'Code' column based on condition
data.loc[data['Entity'] == 'Eastern Mediterranean Region (WHO)', 'Code'] =␣
↪'Eastern Mediterranean'
```

[12]:
```
# Set "UK" in the 'Code' column based on condition
data.loc[data['Entity'] == 'England', 'Code'] = 'UK'
```

[13]:
```
# Set "EU & Central Asia" in the 'Code' column based on condition
data.loc[data['Entity'] == 'Europe & Central Asia (WB)', 'Code'] = 'EU &␣
↪Central Asia'
```

[14]:
```
# Set "Europe" in the 'Code' column based on condition
data.loc[data['Entity'] == 'European Region (WHO)', 'Code'] = 'Europe'
```

[15]:
```
# Set "Latin America & Caribbean" in the 'Code' column based on condition
data.loc[data['Entity'] == 'Latin America & Caribbean (WB)', 'Code'] = 'Latin␣
↪America & Caribbean'
```

[16]:
```
# Set "MENA" in the 'Code' column based on condition
data.loc[data['Entity'] == 'Middle East & North Africa (WB)', 'Code'] = 'Middle␣
↪East and North Africa'
```

[17]:
```
# Set "North America" in the 'Code' column based on condition
data.loc[data['Entity'] == 'North America (WB)', 'Code'] = 'North America'
```

[18]:
```
# Set "UK" in the 'Code' column based on condition
data.loc[data['Entity'] == 'Northern Ireland', 'Code'] = 'UK'
```

[19]:
```
# Set "OECD" in the 'Code' column based on condition
data.loc[data['Entity'] == 'OECD Countries', 'Code'] = 'OECD'
```

[20]:
```
# Set "Americas" in the 'Code' column based on condition
data.loc[data['Entity'] == 'Region of the Americas (WHO)', 'Code'] = 'Americas'
```

[21]:
```
# Set "UK" in the 'Code' column based on condition
data.loc[data['Entity'] == 'Scotland', 'Code'] = 'UK'
```

[22]:
```
# Set "South Asia" in the 'Code' column based on condition
data.loc[data['Entity'] == 'South Asia (WB)', 'Code'] = 'South Asia'
```

[23]:
```
# Set "South East Asia" in the 'Code' column based on condition
data.loc[data['Entity'] == 'South-East Asia Region (WHO)', 'Code'] = 'South␣
 ↪East Asia'
```

[24]:
```
# Set "Sub Sahara" in the 'Code' column based on condition
data.loc[data['Entity'] == 'Sub-Saharan Africa (WB)', 'Code'] = 'Sub Sahara'
```

[25]:
```
# Set "UK" in the 'Code' column based on condition
data.loc[data['Entity'] == 'Wales', 'Code'] = 'UK'
```

[26]:
```
# Set "Western Pacific" in the 'Code' column based on condition
data.loc[data['Entity'] == 'Western Pacific Region (WHO)', 'Code'] = 'Western␣
 ↪Pacific'
```

[27]:
```
# Set "WB High Income" in the 'Code' column based on condition
data.loc[data['Entity'] == 'World Bank High Income', 'Code'] = 'WB High Income'
```

[28]:
```
# Set "WB Low Income" in the 'Code' column based on condition
data.loc[data['Entity'] == 'World Bank Low Income', 'Code'] = 'WB Low Income'
```

[29]:
```
# Set "WB Lower Middle Income" in the 'Code' column based on condition
data.loc[data['Entity'] == 'World Bank Lower Middle Income', 'Code'] = 'WB␣
 ↪Lower Middle Income'
```

[30]:
```
# Set "WB Upper Middle Income" in the 'Code' column based on condition
data.loc[data['Entity'] == 'World Bank Upper Middle Income', 'Code'] = 'WB␣
 ↪Upper Middle Income'
```

[31]:
```
# Check for null values in the dataset
null_values = data.isnull().sum()
print(null_values)
```

```
Entity
0
Code
30
Year
0
Deaths - Cause: All causes - Risk: Household air pollution from solid fuels -
Sex: Both - Age: Age-standardized (Rate)     0
Deaths - Cause: All causes - Risk: Ambient particulate matter pollution - Sex:
Both - Age: Age-standardized (Rate)          0
Deaths - Cause: All causes - Risk: Air pollution - Sex: Both - Age: Age-
standardized (Rate)                                  0
Deaths - Cause: All causes - Risk: Ambient ozone pollution - Sex: Both - Age:
Age-standardized (Rate)                      0
dtype: int64
```

[32]:
```
# Print rows with null values in the 'Code' column
null_code_rows = data[data['Code'].isnull()]
print(null_code_rows)
```

```
     Entity Code       Year  \
2100    G20  NaN 1990-01-01
2101    G20  NaN 1991-01-01
2102    G20  NaN 1992-01-01
2103    G20  NaN 1993-01-01
2104    G20  NaN 1994-01-01
2105    G20  NaN 1995-01-01
2106    G20  NaN 1996-01-01
2107    G20  NaN 1997-01-01
2108    G20  NaN 1998-01-01
2109    G20  NaN 1999-01-01
2110    G20  NaN 2000-01-01
2111    G20  NaN 2001-01-01
2112    G20  NaN 2002-01-01
2113    G20  NaN 2003-01-01
2114    G20  NaN 2004-01-01
2115    G20  NaN 2005-01-01
2116    G20  NaN 2006-01-01
2117    G20  NaN 2007-01-01
2118    G20  NaN 2008-01-01
2119    G20  NaN 2009-01-01
2120    G20  NaN 2010-01-01
2121    G20  NaN 2011-01-01
2122    G20  NaN 2012-01-01
2123    G20  NaN 2013-01-01
2124    G20  NaN 2014-01-01
2125    G20  NaN 2015-01-01
2126    G20  NaN 2016-01-01
```

```
2127    G20  NaN 2017-01-01
2128    G20  NaN 2018-01-01
2129    G20  NaN 2019-01-01

      Deaths - Cause: All causes - Risk: Household air pollution from solid
fuels - Sex: Both - Age: Age-standardized (Rate)  \
2100                                          86.867471
2101                                          84.490909
2102                                          82.082766
2103                                          79.375291
2104                                          76.233200
2105                                          72.888662
2106                                          70.322468
2107                                          68.250940
2108                                          65.243250
2109                                          62.003755
2110                                          59.729350
2111                                          57.534216
2112                                          55.429575
2113                                          52.822323
2114                                          49.873854
2115                                          47.663579
2116                                          44.951345
2117                                          42.330920
2118                                          40.133535
2119                                          37.305216
2120                                          35.006944
2121                                          32.892990
2122                                          30.607468
2123                                          28.512336
2124                                          26.374525
2125                                          24.612583
2126                                          23.071755
2127                                          21.828596
2128                                          20.563681
2129                                          18.937770

      Deaths - Cause: All causes - Risk: Ambient particulate matter pollution -
Sex: Both - Age: Age-standardized (Rate)  \
2100                                          53.943413
2101                                          53.703609
2102                                          53.714450
2103                                          54.360023
2104                                          54.389229
2105                                          53.867596
2106                                          53.565895
2107                                          53.727254
2108                                          53.845696
```

2109	54.220875
2110	54.543184
2111	54.519199
2112	54.646691
2113	54.396332
2114	53.692545
2115	53.547801
2116	52.645522
2117	52.613051
2118	53.389983
2119	53.643252
2120	54.057170
2121	54.294720
2122	54.470495
2123	54.874330
2124	54.700448
2125	54.001101
2126	52.398520
2127	50.968186
2128	51.010800
2129	51.540085

	Deaths - Cause: All causes - Risk: Air pollution - Sex: Both - Age: Age-standardized (Rate) \
2100	143.979168
2101	141.323242
2102	138.973439
2103	136.997713
2104	134.035869
2105	130.296575
2106	127.495943
2107	125.700906
2108	122.752487
2109	119.999433
2110	117.952002
2111	115.746952
2112	113.735335
2113	110.902191
2114	107.284873
2115	104.982407
2116	101.521374
2117	98.858066
2118	97.404640
2119	94.505062
2120	92.539610
2121	90.561653
2122	88.164017
2123	86.221881

```
2124                                                83.780813
2125                                                81.566529
2126                                                78.630580
2127                                                75.958190
2128                                                74.648374
2129                                                73.531305

      Deaths - Cause: All causes - Risk: Ambient ozone pollution - Sex: Both -
Age: Age-standardized (Rate)
2100                                                 6.693509
2101                                                 6.616428
2102                                                 6.712769
2103                                                 6.800665
2104                                                 6.955164
2105                                                 6.975467
2106                                                 6.970774
2107                                                 7.185669
2108                                                 7.060744
2109                                                 7.251380
2110                                                 6.940738
2111                                                 6.861547
2112                                                 6.720300
2113                                                 6.808742
2114                                                 6.812667
2115                                                 6.756852
2116                                                 6.895459
2117                                                 6.918029
2118                                                 6.921928
2119                                                 6.306378
2120                                                 6.004995
2121                                                 5.790125
2122                                                 5.460978
2123                                                 5.238093
2124                                                 5.026364
2125                                                 5.094502
2126                                                 5.299954
2127                                                 5.331189
2128                                                 5.077153
2129                                                 5.066646
```

[33]:
```
# Set "G20" in the 'Code' column based on condition
data.loc[data['Entity'] == 'G20', 'Code'] = 'G20 Countries'
```

[34]:
```
# Print rows with null values in the 'Code' column
null_code_rows = data[data['Code'].isnull()]
print(null_code_rows)
```

```
Empty DataFrame
```

```
Columns: [Entity, Code, Year, Deaths - Cause: All causes - Risk: Household air
pollution from solid fuels - Sex: Both - Age: Age-standardized (Rate), Deaths -
Cause: All causes - Risk: Ambient particulate matter pollution - Sex: Both -
Age: Age-standardized (Rate), Deaths - Cause: All causes - Risk: Air pollution -
Sex: Both - Age: Age-standardized (Rate), Deaths - Cause: All causes - Risk:
Ambient ozone pollution - Sex: Both - Age: Age-standardized (Rate)]
Index: []
```

[35]:
```
# Add 'Region' to values in 'Code' column containing a space
data.loc[data['Code'].str.contains(' '), 'Code'] = data.loc[data['Code'].str.
 ↪contains(' '), 'Code'] + ' Region'
```

[36]:
```
# Count the number of rows containing 'Region' in the 'Code' column
count_region = data['Code'].str.contains('Region').sum()
print(count_region)
```

```
450
```

[37]:
```
# Print the column names
column_names = data.columns
print(column_names)
```

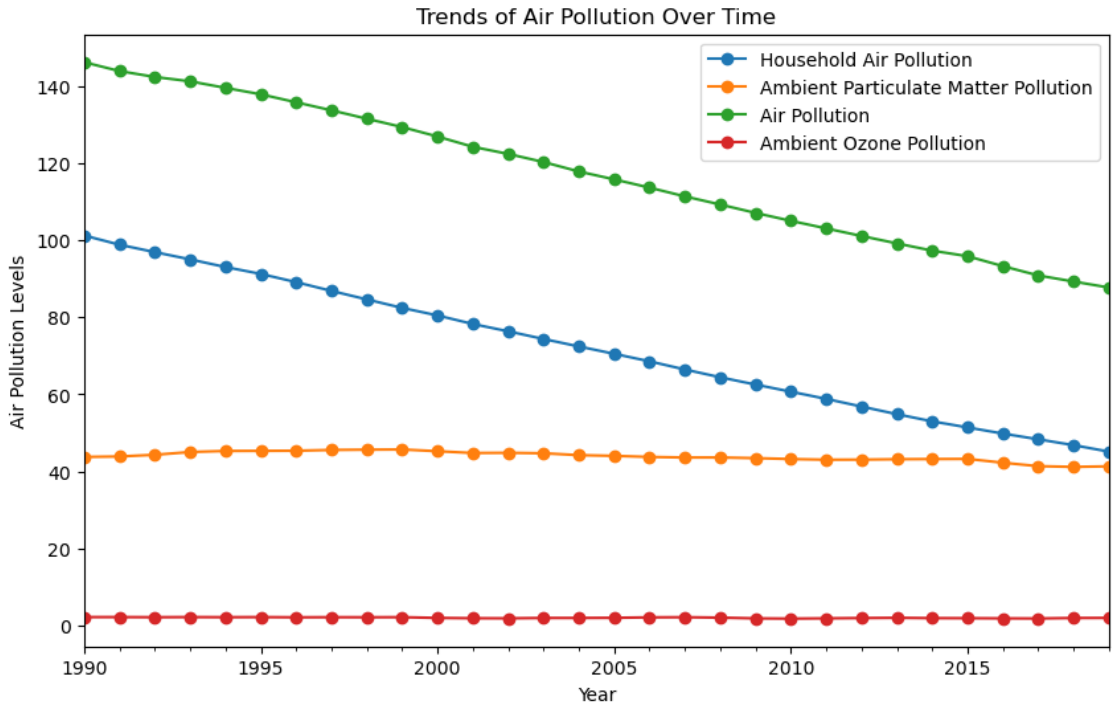
```
Index(['Entity', 'Code', 'Year',
       'Deaths - Cause: All causes - Risk: Household air pollution from solid
fuels - Sex: Both - Age: Age-standardized (Rate)',
       'Deaths - Cause: All causes - Risk: Ambient particulate matter pollution
- Sex: Both - Age: Age-standardized (Rate)',
       'Deaths - Cause: All causes - Risk: Air pollution - Sex: Both - Age: Age-
standardized (Rate)',
       'Deaths - Cause: All causes - Risk: Ambient ozone pollution - Sex: Both -
Age: Age-standardized (Rate)'],
      dtype='object')
```

[38]:
```
# Define the new column names
new_column_names = {
    'Deaths - Cause: All causes - Risk: Household air pollution from solid␣
 ↪fuels - Sex: Both - Age: Age-standardized (Rate)': 'Household Air Pollution',
    'Deaths - Cause: All causes - Risk: Ambient particulate matter pollution -␣
 ↪Sex: Both - Age: Age-standardized (Rate)': 'Ambient Particulate Matter␣
 ↪Pollution',
    'Deaths - Cause: All causes - Risk: Air pollution - Sex: Both - Age:␣
 ↪Age-standardized (Rate)': 'Air Pollution',
    'Deaths - Cause: All causes - Risk: Ambient ozone pollution - Sex: Both -␣
 ↪Age: Age-standardized (Rate)': 'Ambient Ozone Pollution'
}

# Rename the columns
data = data.rename(columns=new_column_names)
```

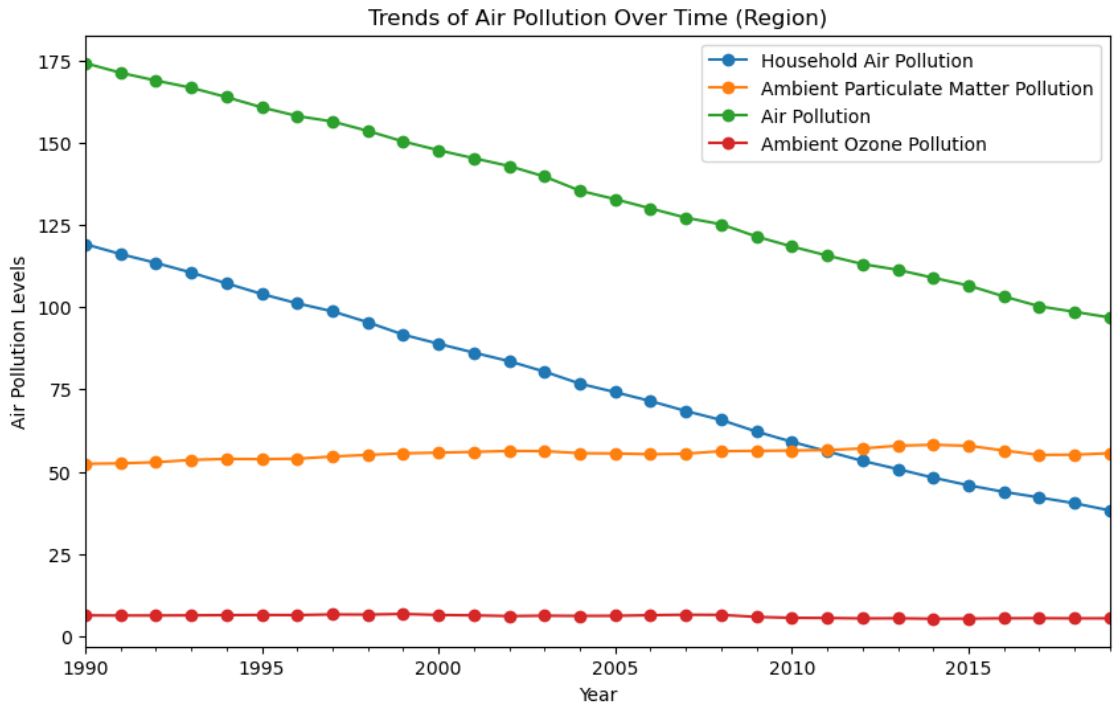
[39]:
```
# Print the column names
column_names = data.columns
print(column_names)
```

```
Index(['Entity', 'Code', 'Year', 'Household Air Pollution',
       'Ambient Particulate Matter Pollution', 'Air Pollution',
       'Ambient Ozone Pollution'],
      dtype='object')
```

[40]:
```
import matplotlib.pyplot as plt

# Filter the data excluding rows with "Region" in the Code column
filtered_data = data[~data['Code'].str.contains('Region')]

# Group the data by Year and calculate the mean for each air pollution variable
grouped_data = filtered_data.groupby('Year')[['Household Air Pollution',␣
 ↪'Ambient Particulate Matter Pollution', 'Air Pollution', 'Ambient Ozone␣
 ↪Pollution']].mean()

# Plot the trends over time
grouped_data.plot(marker='o', linestyle='-', figsize=(10, 6))
plt.xlabel('Year')
plt.ylabel('Air Pollution Levels')
plt.title('Trends of Air Pollution Over Time')
plt.legend()
plt.show()
```

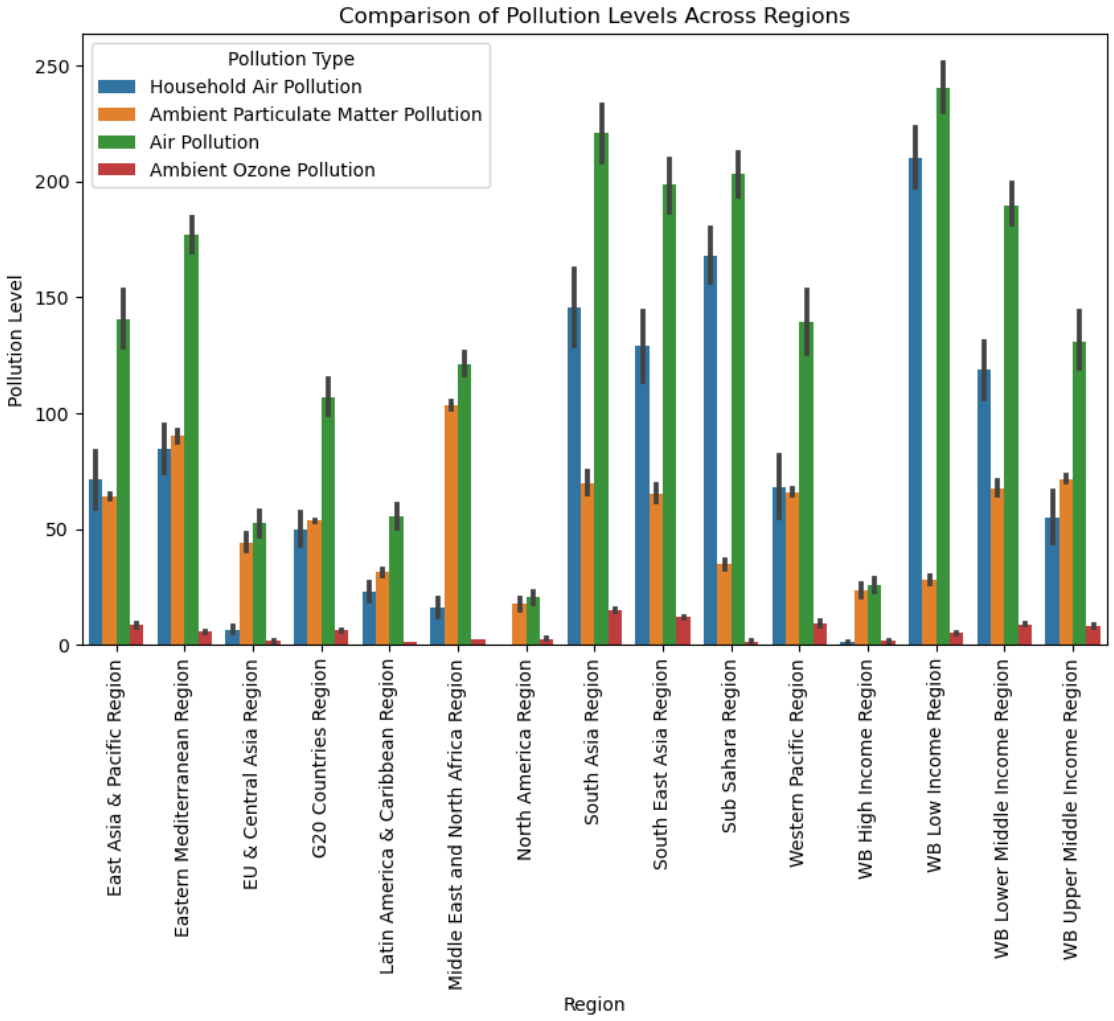


[41]:
```
import matplotlib.pyplot as plt

# Filter the data to include only rows with "Region" in the Code column
filtered_data = data[data['Code'].str.contains('Region')]

# Group the data by Year and calculate the mean for each air pollution variable
grouped_data = filtered_data.groupby('Year')[['Household Air Pollution',␣
 ↪'Ambient Particulate Matter Pollution', 'Air Pollution', 'Ambient Ozone␣
 ↪Pollution']].mean()

# Plot the trends over time
grouped_data.plot(marker='o', linestyle='-', figsize=(10, 6))
plt.xlabel('Year')
plt.ylabel('Air Pollution Levels')
plt.title('Trends of Air Pollution Over Time (Region)')
plt.legend()
plt.show()
```



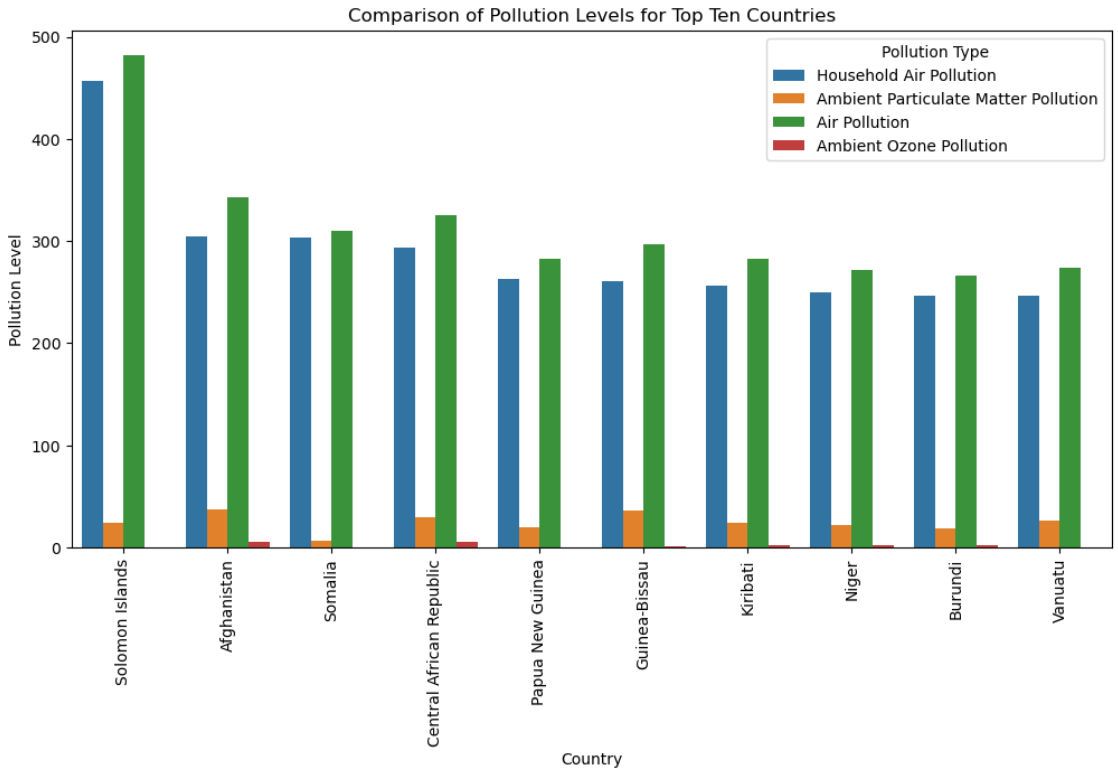
[42]:
```
import seaborn as sns
import matplotlib.pyplot as plt
```

```
# Filter the data to include only rows with "Region" in the Code column
filtered_data = data[data['Code'].str.contains('Region')]

# Select the relevant columns for comparison
pollution_data = filtered_data[['Code', 'Household Air Pollution', 'Ambient␣
 ↪Particulate Matter Pollution', 'Air Pollution', 'Ambient Ozone Pollution']]

# Melt the data to have a single column for pollution variable and its values
melted_data = pollution_data.melt(id_vars='Code', var_name='Pollution Type',␣
 ↪value_name='Pollution Level')

# Create a bar plot to compare pollution levels across regions
plt.figure(figsize=(10, 6))
sns.barplot(x='Code', y='Pollution Level', hue='Pollution Type',␣
 ↪data=melted_data)
plt.xlabel('Region')
plt.ylabel('Pollution Level')
plt.title('Comparison of Pollution Levels Across Regions')
plt.legend(title='Pollution Type')
plt.xticks(rotation=90)
plt.show()
```



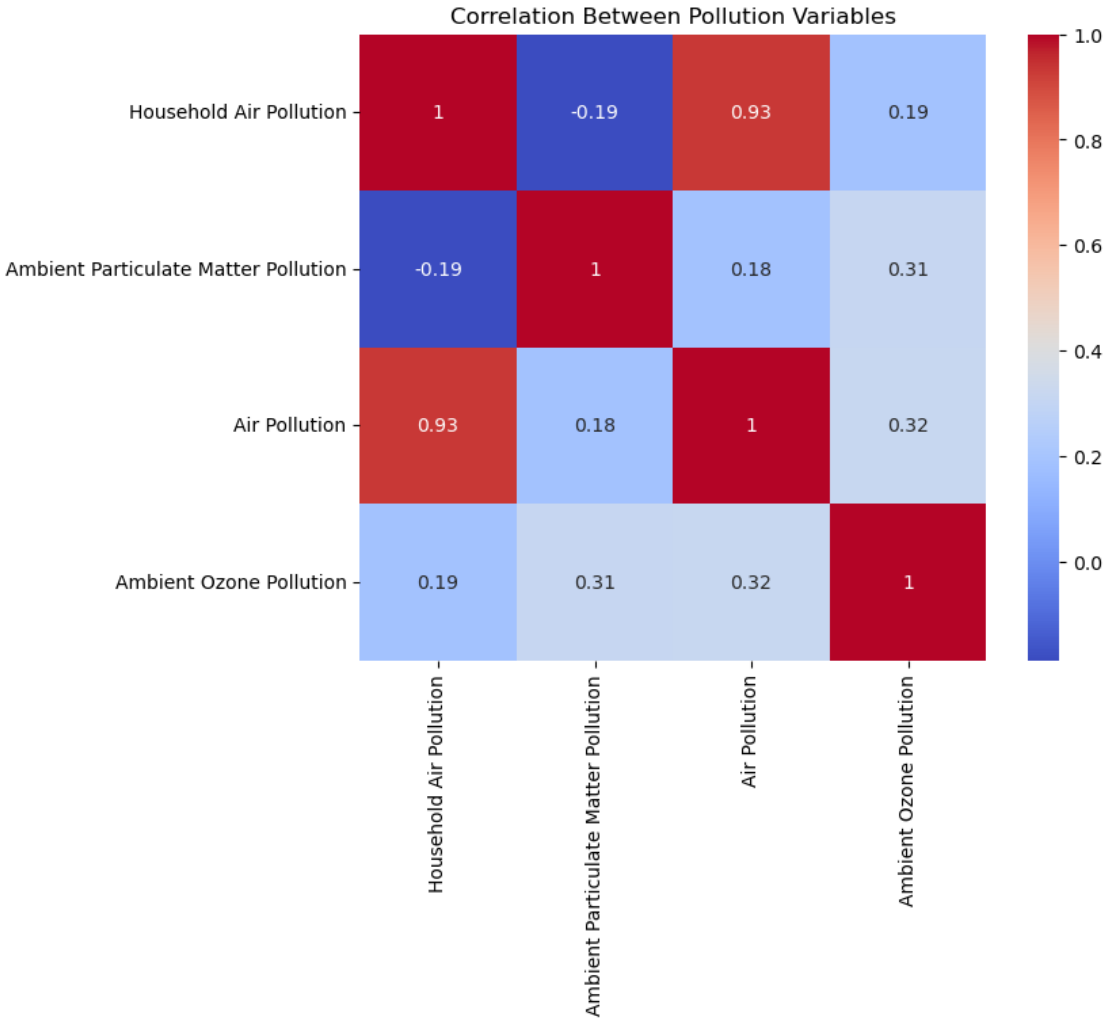
[43]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Filter the data to include only the relevant columns
pollution_data = data[['Entity', 'Household Air Pollution', 'Ambient␣
 ↪Particulate Matter Pollution', 'Air Pollution', 'Ambient Ozone Pollution']]

# Sort the data based on the average pollution levels across all years
sorted_data = pollution_data.groupby('Entity').mean().sort_values(by='Household␣
 ↪Air Pollution', ascending=False)

# Select the top ten countries
top_ten_countries = sorted_data.head(10)

# Reset the index to have 'Entity' as a regular column
```

```
top_ten_countries = top_ten_countries.reset_index()

# Melt the data to have a single column for pollution variable and its values
melted_data = top_ten_countries.melt(id_vars='Entity', var_name='Pollution␣
 ↪Type', value_name='Pollution Level')

# Create a bar plot to compare pollution levels for the top ten countries
plt.figure(figsize=(12, 6))
sns.barplot(x='Entity', y='Pollution Level', hue='Pollution Type',␣
 ↪data=melted_data)
plt.xlabel('Country')
plt.ylabel('Pollution Level')
plt.title('Comparison of Pollution Levels for Top Ten Countries')
plt.legend(title='Pollution Type')
plt.xticks(rotation=90)
plt.show()
```



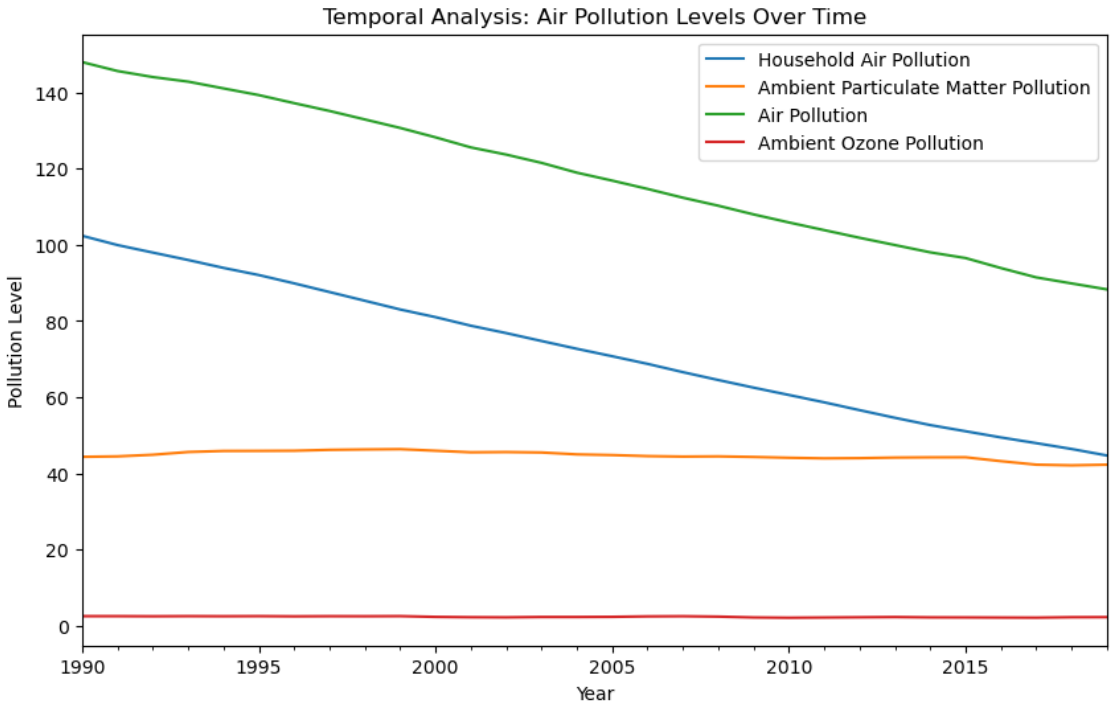
[44]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Select the relevant columns for correlation analysis
```

```
pollution_data = data[['Household Air Pollution', 'Ambient Particulate Matter␣
 ↪Pollution', 'Air Pollution', 'Ambient Ozone Pollution']]

# Compute the correlation matrix
correlation_matrix = pollution_data.corr()

# Generate a heatmap to visualize the correlation matrix
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', square=True)
plt.title('Correlation Between Pollution Variables')
plt.show()
```



[45]:
```
import matplotlib.pyplot as plt

# Select the relevant columns for temporal analysis
```

```
pollution_data = data[['Year', 'Household Air Pollution', 'Ambient Particulate␣
 ↪Matter Pollution', 'Air Pollution', 'Ambient Ozone Pollution']]

# Group the data by year and calculate the mean pollution levels for each year
mean_pollution_by_year = pollution_data.groupby('Year').mean()

# Line plot to visualize the temporal changes in pollution levels
plt.figure(figsize=(10, 6))
mean_pollution_by_year['Household Air Pollution'].plot(label='Household Air␣
 ↪Pollution')
mean_pollution_by_year['Ambient Particulate Matter Pollution'].
 ↪plot(label='Ambient Particulate Matter Pollution')
mean_pollution_by_year['Air Pollution'].plot(label='Air Pollution')
mean_pollution_by_year['Ambient Ozone Pollution'].plot(label='Ambient Ozone␣
 ↪Pollution')
plt.xlabel('Year')
plt.ylabel('Pollution Level')
plt.title('Temporal Analysis: Air Pollution Levels Over Time')
plt.legend()
plt.show()
```



[ ]:

[46]:
```
import pycountry_convert as pc

# Function to get continent name based on country name
def get_continent(country_name):
    try:
        country_alpha2 = pc.country_name_to_country_alpha2(country_name)
        continent_code = pc.country_alpha2_to_continent_code(country_alpha2)
        continent_name = pc.
 ↪convert_continent_code_to_continent_name(continent_code)
        return continent_name
    except:
        return None

# Create a new column 'Continent' using the 'Entity' column
data['Continent'] = data['Entity'].apply(get_continent)
```

[ ]:
```

```

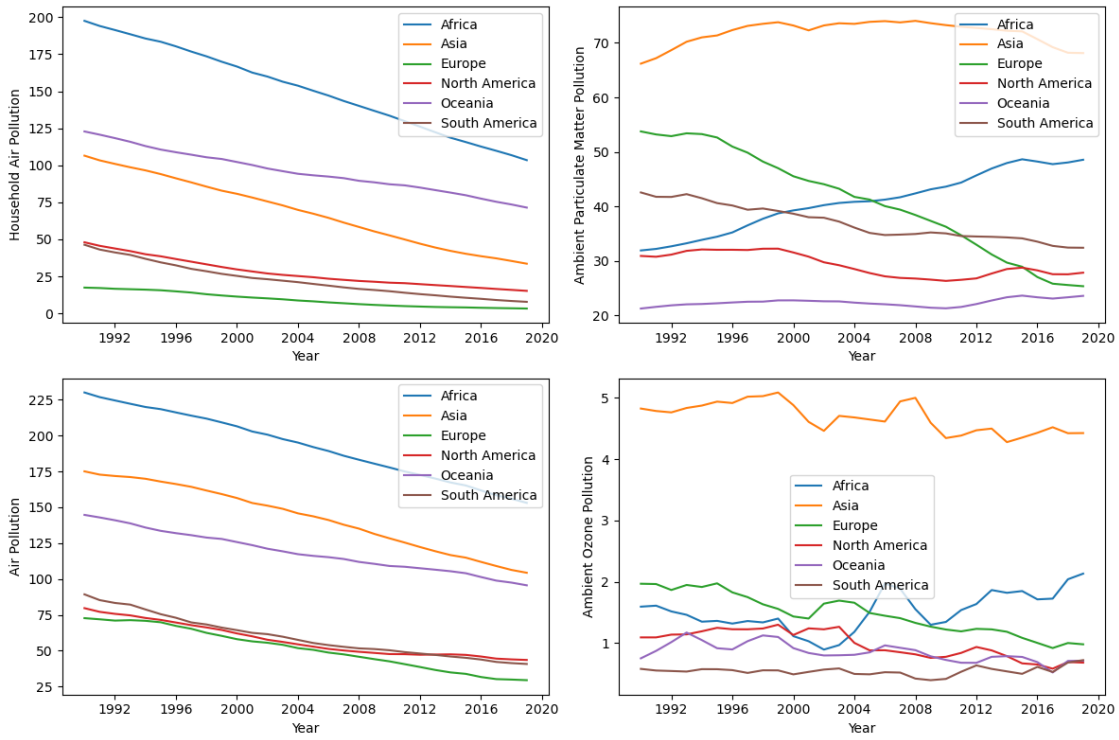
[47]:
```
data.head()
```

[47]:

	Entity	Code	Year	Household Air Pollution	Ambient Particulate Matter Pollution	Air Pollution	Ambient Ozone Pollution	Continent
0	Afghanistan	AFG	1990-01-01	370.050474	30.822693	402.175651	6.581093	Asia
1	Afghanistan	AFG	1991-01-01	358.978418	29.826184	390.085258	6.267613	Asia
2	Afghanistan	AFG	1992-01-01	352.766453	29.202030	383.201196	5.926444	Asia
3	Afghanistan	AFG	1993-01-01	357.055923	29.429702	387.704919	5.860345	Asia
4	Afghanistan	AFG	1994-01-01	362.970439	29.813259	394.022027	6.065343	Asia

[48]:
```
import matplotlib.pyplot as plt

# Group the data by 'Continent' and 'Year' columns and calculate the mean of␣
 ↪air pollution variables
continent_year_data = data.groupby(['Continent', 'Year']).
 ↪mean(numeric_only=True).reset_index()
```

```
# Plot the trends for each air pollution variable based on continents
plt.figure(figsize=(12, 8))

# Household Air Pollution
plt.subplot(2, 2, 1)
for continent in continent_year_data['Continent'].unique():
    subset = continent_year_data[continent_year_data['Continent'] == continent]
    plt.plot(subset['Year'], subset['Household Air Pollution'], label=continent)
plt.xlabel('Year')
plt.ylabel('Household Air Pollution')
plt.legend()

# Ambient Particulate Matter Pollution
plt.subplot(2, 2, 2)
for continent in continent_year_data['Continent'].unique():
    subset = continent_year_data[continent_year_data['Continent'] == continent]
    plt.plot(subset['Year'], subset['Ambient Particulate Matter Pollution'],␣
 ↪label=continent)
plt.xlabel('Year')
plt.ylabel('Ambient Particulate Matter Pollution')
plt.legend()

# Air Pollution
plt.subplot(2, 2, 3)
for continent in continent_year_data['Continent'].unique():
    subset = continent_year_data[continent_year_data['Continent'] == continent]
    plt.plot(subset['Year'], subset['Air Pollution'], label=continent)
plt.xlabel('Year')
plt.ylabel('Air Pollution')
plt.legend()

# Ambient Ozone Pollution
plt.subplot(2, 2, 4)
for continent in continent_year_data['Continent'].unique():
    subset = continent_year_data[continent_year_data['Continent'] == continent]
    plt.plot(subset['Year'], subset['Ambient Ozone Pollution'], label=continent)
plt.xlabel('Year')
plt.ylabel('Ambient Ozone Pollution')
plt.legend()

plt.tight_layout()
plt.show()
```



[49]:
```
data.head()
```

[49]:

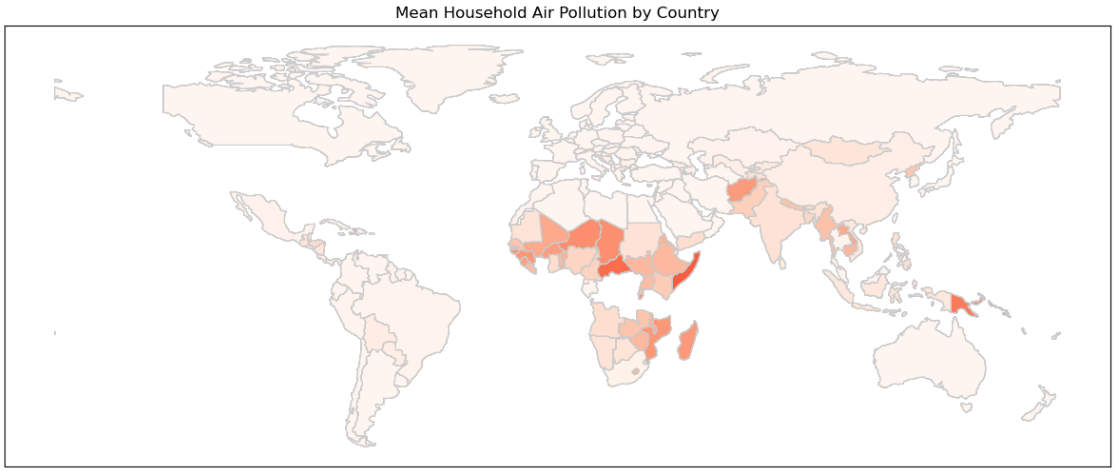
	Entity	Code	Year	Household Air Pollution	Ambient Particulate Matter Pollution	Air Pollution	Ambient Ozone Pollution	Continent
0	Afghanistan	AFG	1990-01-01	370.050474	30.822693	402.175651	6.581093	Asia
1	Afghanistan	AFG	1991-01-01	358.978418	29.826184	390.085258	6.267613	Asia
2	Afghanistan	AFG	1992-01-01	352.766453	29.202030	383.201196	5.926444	Asia
3	Afghanistan	AFG	1993-01-01	357.055923	29.429702	387.704919	5.860345	Asia
4	Afghanistan	AFG	1994-01-01	362.970439	29.813259	394.022027	6.065343	Asia

[50]:
```
# Add "Region" after "Africa" values in the Code column
data.loc[data['Code'].str.contains('Africa', na=False), 'Code'] += ' Region'

# Export the updated data to CSV format
output_file_path = r"C:\Users\user\Desktop\portfolio\air␣
 ↪pollution\airpollution_clean.csv"
data.to_csv(output_file_path, index=False)

print("File exported successfully.")
```

```
File exported successfully.
```

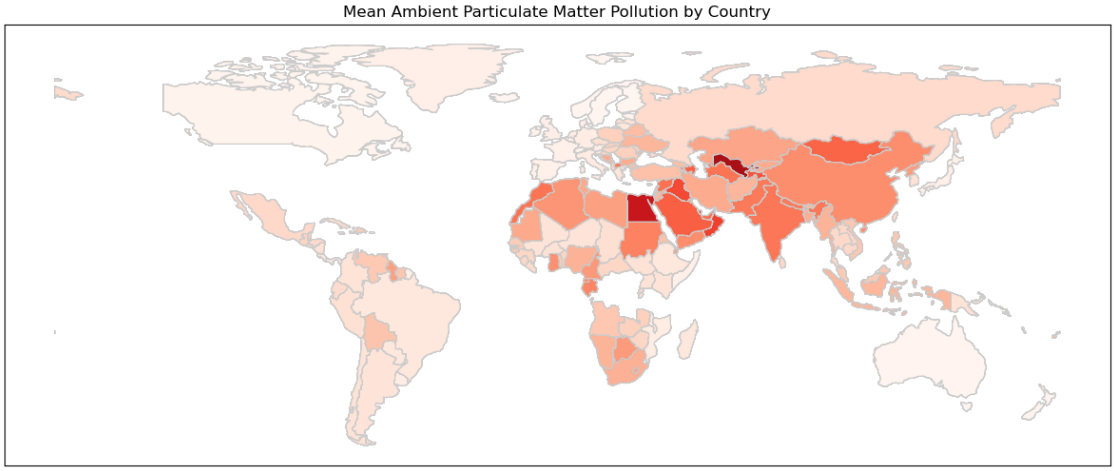
[51]:
```
import pandas as pd
import geopandas as gpd
import matplotlib.pyplot as plt

# Read the data file
data_path = r"C:\Users\user\Desktop\portfolio\air pollution\airpollution_clean.
 ↪csv"
data = pd.read_csv(data_path)

# Filter the data for countries where the Code column does not contain the word␣
 ↪"Region"
filtered_data = data[~data['Code'].str.contains('Region')]

# Read the shapefile data
shapefile_path = r"D:\World_Map_Shapefile\ne_110m_admin_0_countries.shp"
world_map = gpd.read_file(shapefile_path)

# Merge the filtered data with the shapefile data based on the 'Entity' column
merged_data = world_map.merge(filtered_data, left_on='ADMIN', right_on='Entity')

# Create a new figure and axis
fig, ax = plt.subplots(figsize=(15, 10))

# Plot the map with color intensity based on the mean values of "Household Air␣
 ↪Pollution"
merged_data.plot(column='Household Air Pollution', cmap='Reds', linewidth=0.8,␣
 ↪ax=ax, edgecolor='0.8')

# Add a title to the map
ax.set_title('Mean Household Air Pollution by Country')

# Remove the axis ticks and labels
ax.set_xticks([])
ax.set_yticks([])
ax.set_xticklabels([])
```

```
ax.set_yticklabels([])

# Display the map
plt.show()
```



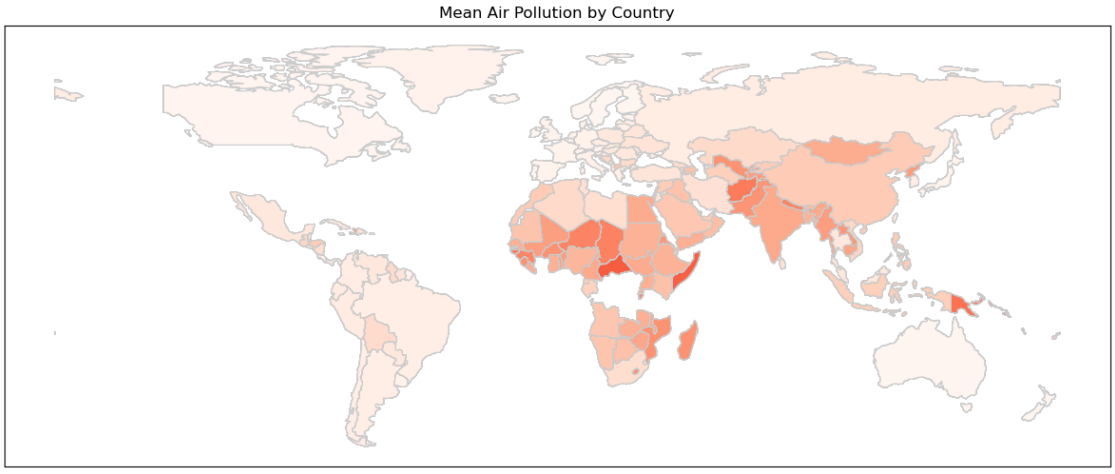
[6]:
```
import pandas as pd
import geopandas as gpd
import matplotlib.pyplot as plt

# Read the data file
data_path = r"C:\Users\user\Desktop\portfolio\air pollution\airpollution_clean.
 ↪csv"
data = pd.read_csv(data_path)

# Filter the data for countries where the Code column does not contain the word␣
 ↪"Region"
filtered_data = data[~data['Code'].str.contains('Region')]

# Read the shapefile data
shapefile_path = r"D:\World_Map_Shapefile\ne_110m_admin_0_countries.shp"
world_map = gpd.read_file(shapefile_path)

# Merge the filtered data with the shapefile data based on the 'Entity' column
merged_data = world_map.merge(filtered_data, left_on='ADMIN', right_on='Entity')

# Create a new figure and axis
fig, ax = plt.subplots(figsize=(15, 10))

# Plot the map with color intensity based on the mean values of "Household Air␣
 ↪Pollution"
```

```
merged_data.plot(column='Ambient Particulate Matter Pollution', cmap='Reds',␣
 ↪linewidth=0.8, ax=ax, edgecolor='0.8')

# Add a title to the map
ax.set_title('Mean Ambient Particulate Matter Pollution by Country')

# Remove the axis ticks and labels
ax.set_xticks([])
ax.set_yticks([])
ax.set_xticklabels([])
ax.set_yticklabels([])

# Display the map
plt.show()
```



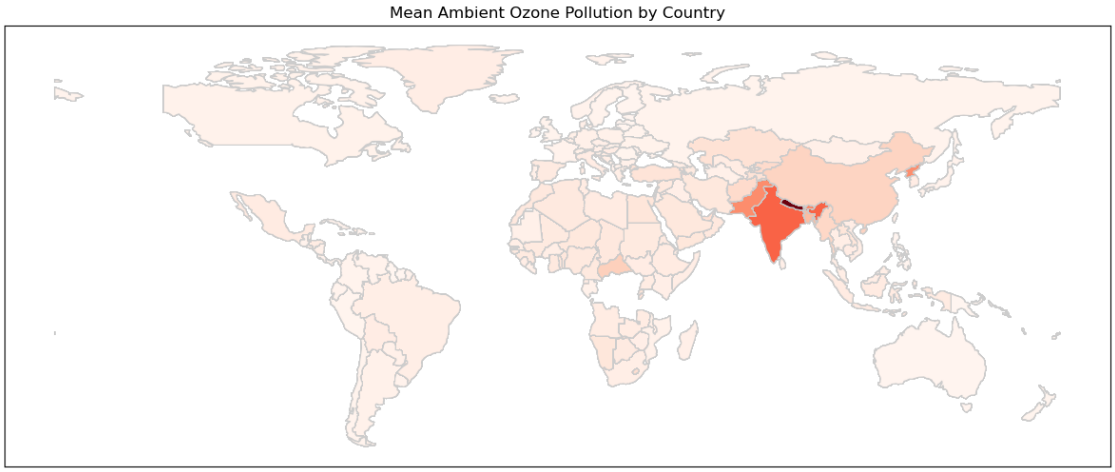
[7]:
```
import pandas as pd
import geopandas as gpd
import matplotlib.pyplot as plt

# Read the data file
data_path = r"C:\Users\user\Desktop\portfolio\air pollution\airpollution_clean.
 ↪csv"
data = pd.read_csv(data_path)

# Filter the data for countries where the Code column does not contain the word␣
 ↪"Region"
filtered_data = data[~data['Code'].str.contains('Region')]

# Read the shapefile data
shapefile_path = r"D:\World_Map_Shapefile\ne_110m_admin_0_countries.shp"
```

```
world_map = gpd.read_file(shapefile_path)

# Merge the filtered data with the shapefile data based on the 'Entity' column
merged_data = world_map.merge(filtered_data, left_on='ADMIN', right_on='Entity')

# Create a new figure and axis
fig, ax = plt.subplots(figsize=(15, 10))

# Plot the map with color intensity based on the mean values of "Household Air␣
 ↪Pollution"
merged_data.plot(column='Air Pollution', cmap='Reds', linewidth=0.8, ax=ax,␣
 ↪edgecolor='0.8')

# Add a title to the map
ax.set_title('Mean Air Pollution by Country')

# Remove the axis ticks and labels
ax.set_xticks([])
ax.set_yticks([])
ax.set_xticklabels([])
ax.set_yticklabels([])

# Display the map
plt.show()
```



[8]:
```
import pandas as pd
import geopandas as gpd
import matplotlib.pyplot as plt

# Read the data file
```

```
data_path = r"C:\Users\user\Desktop\portfolio\air pollution\airpollution_clean.
 ↪csv"
data = pd.read_csv(data_path)

# Filter the data for countries where the Code column does not contain the word␣
 ↪"Region"
filtered_data = data[~data['Code'].str.contains('Region')]

# Read the shapefile data
shapefile_path = r"D:\World_Map_Shapefile\ne_110m_admin_0_countries.shp"
world_map = gpd.read_file(shapefile_path)

# Merge the filtered data with the shapefile data based on the 'Entity' column
merged_data = world_map.merge(filtered_data, left_on='ADMIN', right_on='Entity')

# Create a new figure and axis
fig, ax = plt.subplots(figsize=(15, 10))

# Plot the map with color intensity based on the mean values of "Household Air␣
 ↪Pollution"
merged_data.plot(column='Ambient Ozone Pollution', cmap='Reds', linewidth=0.8,␣
 ↪ax=ax, edgecolor='0.8')

# Add a title to the map
ax.set_title('Mean Ambient Ozone Pollution by Country')

# Remove the axis ticks and labels
ax.set_xticks([])
ax.set_yticks([])
ax.set_xticklabels([])
ax.set_yticklabels([])

# Display the map
plt.show()
```


Mean Ambient Ozone Pollution by Country

[ ]: