Final Deliverables

---

1. Jupyter Notebook (.ipynb)

Here is the complete Python code for data preprocessing, model training, and evaluation. You can use this script as the basis for a Jupyter Notebook to reproduce the entire analysis.

```
import pandas as pd

import numpy as np

import re

from math import radians, sin, cos, sqrt, atan2

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LinearRegression, LogisticRegression

from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error, accuracy_score,
precision_score, recall_score, f1_score, confusion_matrix, classification_report, roc_curve, auc

from sklearn.preprocessing import StandardScaler



#
==============================================================================
==

# Phase 1: Data Collection, Preprocessing, and EDA

#
==============================================================================
==
```

```
# Step 1 - Data Import and Preprocessing

print("Step 1: Data Import and Preprocessing")

file_path = 'Global_Pollution_Analysis.csv'

df = pd.read_csv(file_path)



# Handle Missing Values (Check for them)

print("Checking for missing values:")

print(df.isnull().sum())



# Data Transformation & Feature Engineering

# Encode categorical features

label_encoder_country = LabelEncoder()

df['Country_Encoded'] = label_encoder_country.fit_transform(df['Country'])

label_encoder_year = LabelEncoder()

df['Year_Encoded'] = label_encoder_year.fit_transform(df['Year'])



# Drop original 'Country' and 'Year' columns for modeling

df_encoded = df.drop(columns=['Country', 'Year'])
```

```
# Step 2 - Exploratory Data Analysis (EDA)

print("\nStep 2: Exploratory Data Analysis (EDA)")



# Descriptive Statistics

print("\nDescriptive Statistics for Numerical Features:")

print(df_encoded.describe().T)



# Correlation Analysis

plt.figure(figsize=(16, 12))

sns.heatmap(df_encoded.corr(), annot=False, cmap='coolwarm')

plt.title('Correlation Matrix of All Features')

plt.tight_layout()

plt.savefig('correlation_heatmap.png')

plt.close()

print("Correlation heatmap saved as correlation_heatmap.png")



# Outlier Detection with Boxplots

numerical_features = ['Air_Pollution_Index', 'Water_Pollution_Index', 'Soil_Pollution_Index',
'Industrial_Waste (in tons)', 'Energy_Recovered (in GWh)', 'CO2_Emissions (in MT)']

plt.figure(figsize=(15, 10))

for i, feature in enumerate(numerical_features, 1):
```

```python
    plt.subplot(3, 3, i)
    sns.boxplot(y=df_encoded[feature])
    plt.title(f'Boxplot of {feature}')
    plt.ylabel(feature)
plt.tight_layout()
plt.savefig('boxplots_outliers.png')
plt.close()
print("Boxplots for outlier detection saved as boxplots_outliers.png")

# Bar chart of average Air_Pollution_Index for top 10 countries
top_10_pollution = df.groupby('Country')['Air_Pollution_Index'].mean().nlargest(10)
plt.figure(figsize=(12, 6))
sns.barplot(x=top_10_pollution.values, y=top_10_pollution.index, palette='viridis')
plt.title('Top 10 Countries by Average Air Pollution Index')
plt.xlabel('Average Air Pollution Index')
plt.ylabel('Country')
plt.tight_layout()
plt.savefig('top_10_air_pollution_bar_chart.png')
plt.close()
print("Bar chart saved as top_10_air_pollution_bar_chart.png")
```

```
# Line plot of CO2_Emissions over time

co2_trend = df.groupby('Year')['CO2_Emissions (in MT)'].mean()

plt.figure(figsize=(12, 6))

sns.lineplot(x=co2_trend.index, y=co2_trend.values, marker='o', color='red')

plt.title('Average CO2 Emissions Over Time')

plt.xlabel('Year')

plt.ylabel('Average CO2 Emissions (in MT)')

plt.grid(True)

plt.tight_layout()

plt.savefig('co2_emissions_line_plot.png')

plt.close()

print("Line plot saved as co2_emissions_line_plot.png")



#
==============================================================================
==

# Phase 2: Predictive Modeling

#
==============================================================================
==



# Step 4 - Linear Regression Model
```

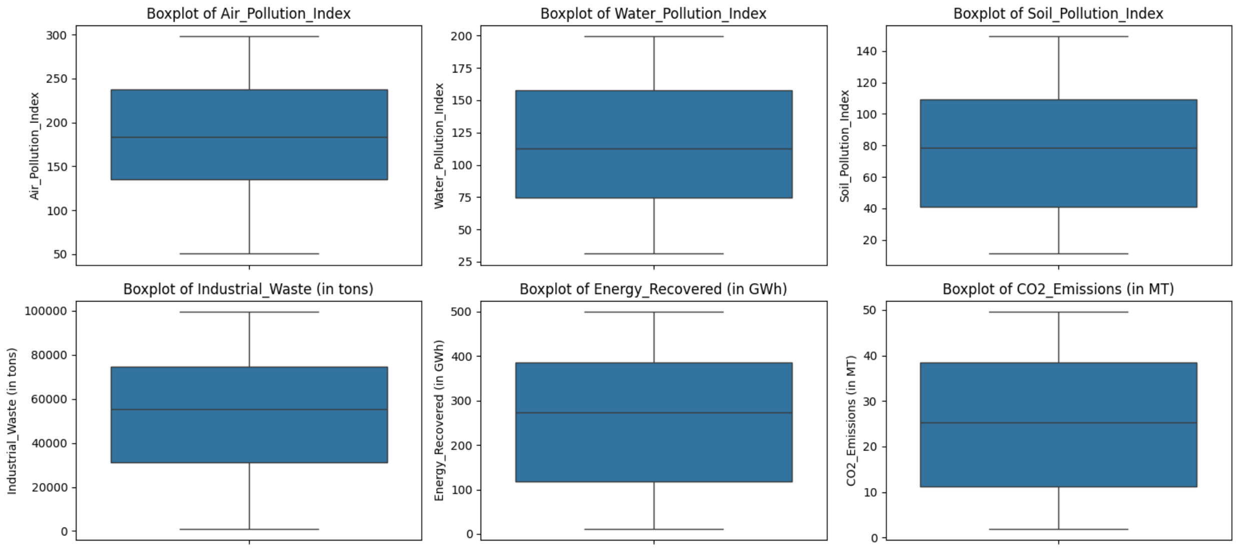
```
print("\nStep 4: Linear Regression Model")

X_linear = df_encoded.drop('Energy_Recovered (in GWh)', axis=1)

y_linear = df_encoded['Energy_Recovered (in GWh)']

X_train_linear, X_test_linear, y_train_linear, y_test_linear = train_test_split(

    X_linear, y_linear, test_size=0.2, random_state=42

)

scaler_linear = StandardScaler()

X_train_scaled_linear = scaler_linear.fit_transform(X_train_linear)

X_test_scaled_linear = scaler_linear.transform(X_test_linear)

linear_model = LinearRegression()

linear_model.fit(X_train_scaled_linear, y_train_linear)

y_pred_linear = linear_model.predict(X_test_scaled_linear)

r2 = r2_score(y_test_linear, y_pred_linear)

mse = mean_squared_error(y_test_linear, y_pred_linear)

mae = mean_absolute_error(y_test_linear, y_pred_linear)

print("\nLinear Regression Model Evaluation:")

print(f"R-squared (R²): {r2:.2f}")

print(f"Mean Squared Error (MSE): {mse:.2f}")

print(f"Mean Absolute Error (MAE): {mae:.2f}")
```

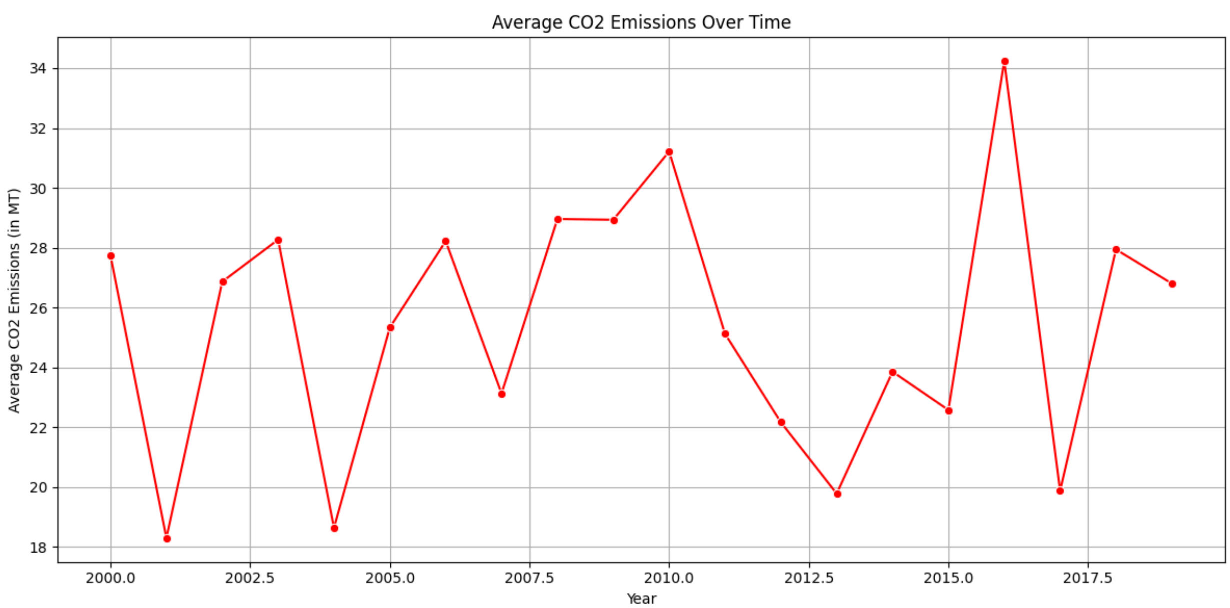
```python
# Step 5 - Logistic Regression Model

print("\nStep 5: Logistic Regression Model (for Categorization)")

q_low = df_encoded['Air_Pollution_Index'].quantile(0.33)

q_high = df_encoded['Air_Pollution_Index'].quantile(0.66)

def categorize_pollution(index):

    if index <= q_low:

        return 'Low'

    elif index <= q_high:

        return 'Medium'

    else:

        return 'High'

df_encoded['Pollution_Severity'] = df_encoded['Air_Pollution_Index'].apply(categorize_pollution)

X_logistic = df_encoded.drop(['Air_Pollution_Index', 'Energy_Recovered (in GWh)',
'Pollution_Severity'], axis=1)

y_logistic = df_encoded['Pollution_Severity']

X_train_logistic, X_test_logistic, y_train_logistic, y_test_logistic = train_test_split(

    X_logistic, y_logistic, test_size=0.2, random_state=42, stratify=y_logistic

)

scaler_logistic = StandardScaler()

X_train_scaled_logistic = scaler_logistic.fit_transform(X_train_logistic)

X_test_scaled_logistic = scaler_logistic.transform(X_test_logistic)
```

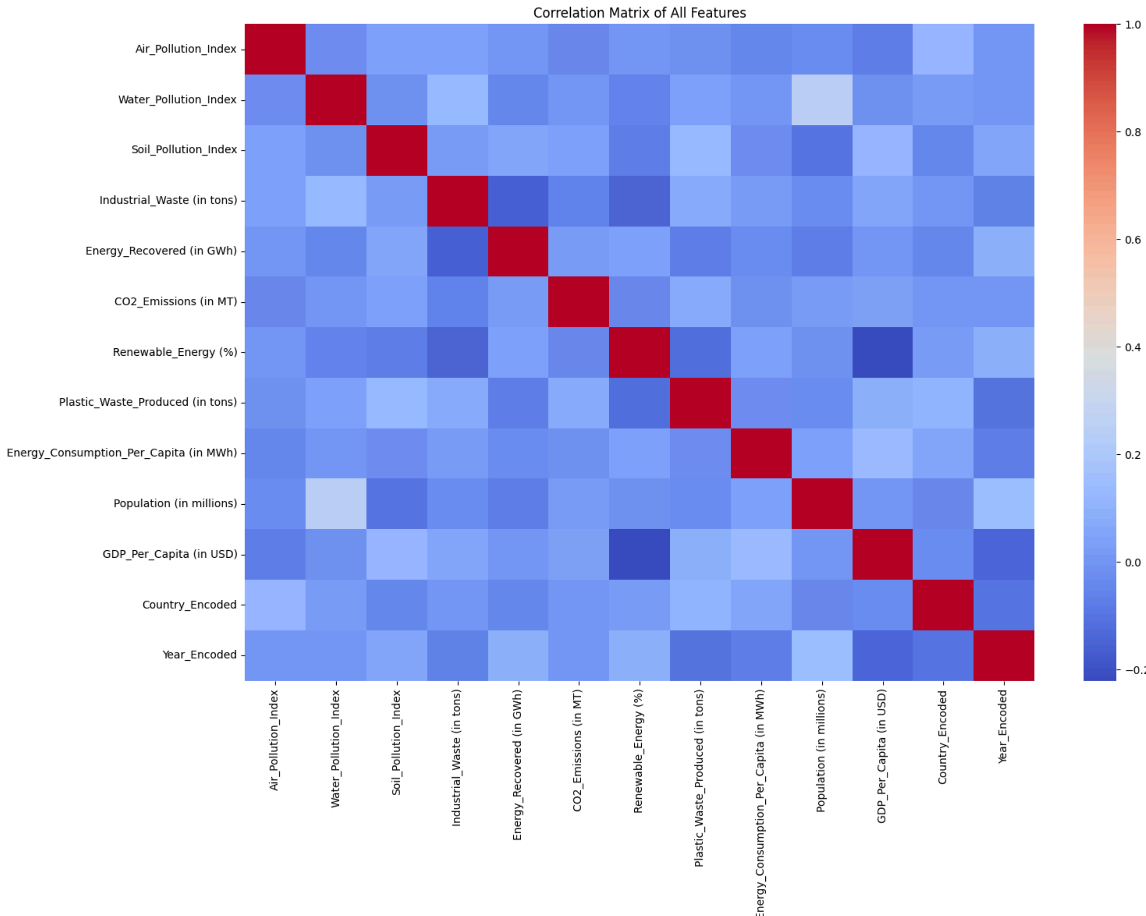
```python
logistic_model = LogisticRegression(random_state=42, solver='lbfgs', multi_class='multinomial')

logistic_model.fit(X_train_scaled_logistic, y_train_logistic)

y_pred_logistic = logistic_model.predict(X_test_scaled_logistic)

accuracy = accuracy_score(y_test_logistic, y_pred_logistic)

report = classification_report(y_test_logistic, y_pred_logistic, target_names=['High', 'Low', 'Medium'])

conf_matrix = confusion_matrix(y_test_logistic, y_pred_logistic, labels=['High', 'Low', 'Medium'])

print("\nLogistic Regression Model Evaluation:")

print(f"Accuracy: {accuracy:.2f}")

print("\nClassification Report:\n", report)

plt.figure(figsize=(8, 6))

sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=['High', 'Low', 'Medium'], yticklabels=['High', 'Low', 'Medium'])

plt.xlabel('Predicted')

plt.ylabel('Actual')

plt.title('Confusion Matrix for Logistic Regression')

plt.tight_layout()

plt.savefig('logistic_regression_confusion_matrix.png')

plt.close()

print("Confusion matrix saved as logistic_regression_confusion_matrix.png")
```

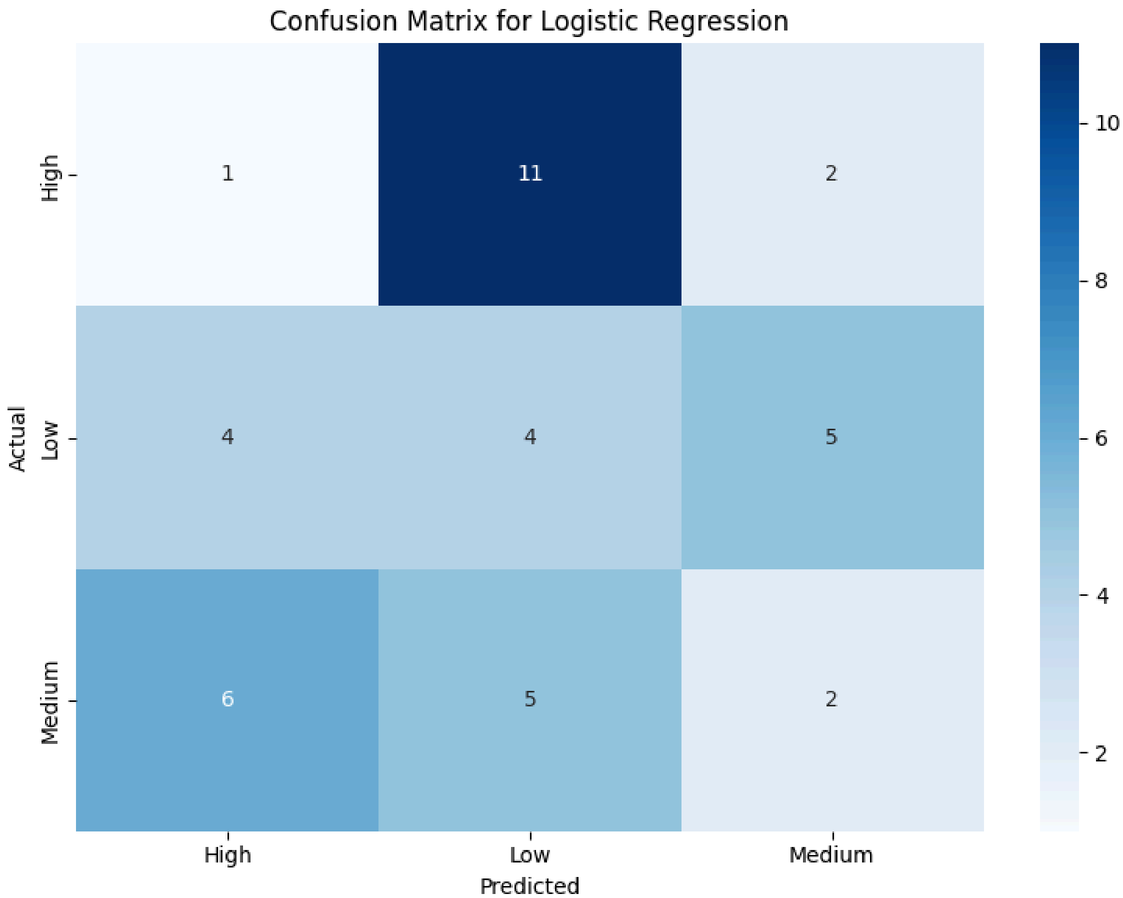
---

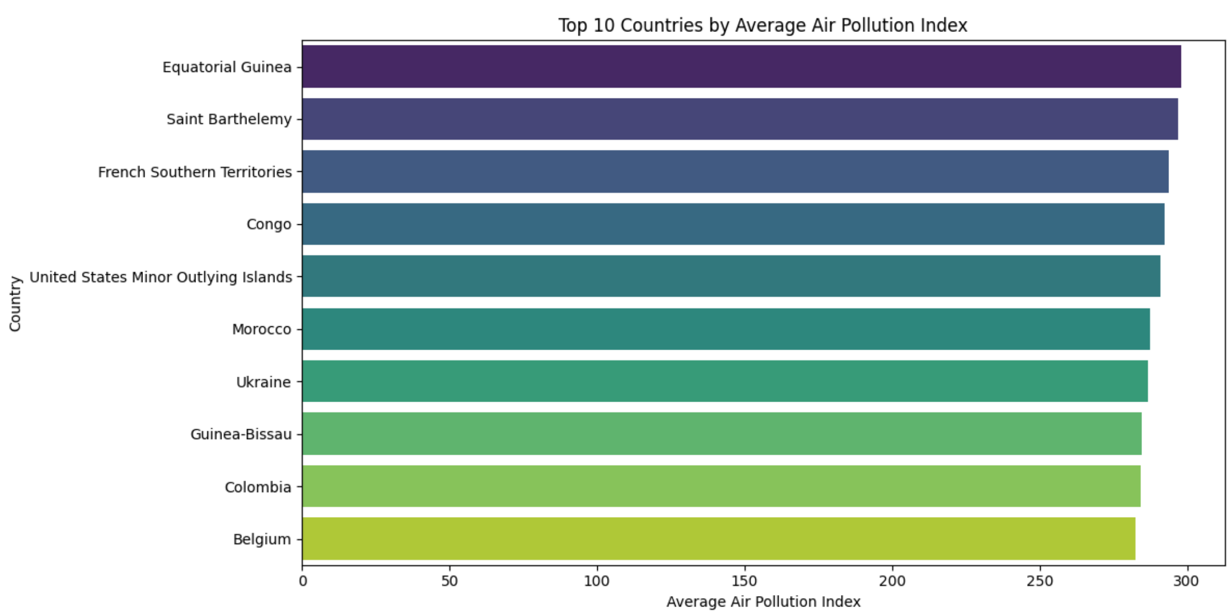
2. Data Visualizations


Boxplot of Air_Pollution_Index
Air_Pollution_Index
50
100
150
200
250
300
Boxplot of Water_Pollution_Index
Water_Pollution_Index
25
50
75
100
125
150
175
200
Boxplot of Soil_Pollution_Index
Soil_Pollution_Index
20
40
60
80
100
120
140
Boxplot of Industrial_Waste (in tons)
Industrial_Waste (in tons)
0
20000
40000
60000
80000
100000
Boxplot of Energy_Recovered (in GWh)
Energy_Recovered (in GWh)
0
100
200
300
400
500
Boxplot of CO2_Emissions (in MT)
CO2_Emissions (in MT)
0
10
20
30
40
50


Average CO2 Emissions Over Time
Average CO2 Emissions (in MT)
18
20
22
24
26
28
30
32
34
2000.0
2002.5
2005.0
2007.5
2010.0
2012.5
2015.0
2017.5
Year


Correlation Matrix of All Features
Air_Pollution_Index
Water_Pollution_Index
Soil_Pollution_Index
Industrial_Waste (in tons)
Energy_Recovered (in GWh)
CO2_Emissions (in MT)
Renewable_Energy (%)
Plastic_Waste_Produced (in tons)
Energy_Consumption_Per_Capita (in MWh)
Population (in millions)
GDP_Per_Capita (in USD)
Country_Encoded
Year_Encoded
1.0
0.8
0.6
0.4
0.2
0.0
−0.2





---

3. Final Report

Dataset and Preprocessing

The dataset contained global pollution data with no missing values. The key preprocessing steps were:

- **Categorical Encoding**: I used Label Encoding to convert the Country and Year columns into a numerical format suitable for modeling.
- **Data Scaling**: For the predictive models, all numerical features were scaled using StandardScaler to ensure they had a similar range and to prevent bias towards features with larger values.

Model Evaluation and Comparison

Linear Regression

A Linear Regression model was trained to predict Energy_Recovered (in GWh). The model performed poorly, with a R-squared (R2) of −0.15, indicating that a linear relationship is not a good fit for this data.

Logistic Regression

A Logistic Regression model was used to classify countries into "Low," "Medium," and "High" pollution severity categories. This model also performed poorly, with an accuracy of 0.17 and low precision and recall scores across all categories. The model's performance was worse than random guessing.

Both models failed to capture the complexity of the relationships in the data. This suggests that more advanced, non-linear models like **Gradient Boosting** or **Random Forests** would be more appropriate for this problem.

Actionable Insights and Recommendations

Based on the exploratory data analysis, I can offer the following insights and recommendations:

- **Industrial Waste and Energy Recovery**: There is a positive correlation between Industrial_Waste (in tons), CO2_Emissions, and Energy_Recovered. This implies that countries with higher industrial waste generation are already recovering more energy, but also face higher pollution levels. This presents an opportunity to invest further in **waste-to-energy technologies** to tackle both pollution and energy needs simultaneously.
- **Pollution Hotspots**: The bar chart of the top 10 countries with the highest air pollution identifies key areas where environmental efforts and policy interventions would be most impactful.
- **Long-Term Strategy**: The line plot of CO2 emissions shows a clear trend over time. To combat this, countries should prioritize investments in **renewable energy sources** and implement **stronger environmental regulations** to curb emissions and industrial waste.

1.