BACSE101 Problem Solving using Python

PROJECT REPORT

on

Global CO2 Emissions Explorer

Prepared by

Arav Kilak - 25BCE2015
Ishan Tayal - 25BCE2031

Under the supervision of

Professor Thirumoorthy Krishnan

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

School of Computer Science and Engineering
Vellore Institute of Technology, Vellore.

October 27, 2025

Table of Contents

Abstract

The "Global CO2 Emissions Explorer" is a command-line interface (CLI) application developed in Python that serves as a tool for data journalism and environmental data analysis. This project leverages the Pandas and Numpy libraries to parse, clean, and analyze a large, real-world dataset of global CO2 emissions from "Our World in Data." It provides a user-friendly, menu-driven interface that allows non-technical users to extract meaningful insights, such as identifying top-emitting countries, tracking historical trends, and comparing national emissions data. The tool successfully fulfills the two-part requirement of the Assessment 6 project by combining Python fundamentals for the menu operations with robust data analytics using Pandas and Numpy.

1. Introduction

This project is a command-line tool designed to make complex environmental data accessible and understandable. In a world where climate change is a pressing issue, large datasets on CO2 emissions are publicly available but often remain in formats (like large CSV files) that are inaccessible to the general public. Our project bridges this gap by providing a simple menu that allows users to ask specific questions and receive immediate, analyzed answers directly in their terminal. The project is split into two main Python files: main.py, which controls the user-facing menu, and analysis.py, which houses all the data processing logic.

1.1 Domain Information

The project operates in the environmental science and data journalism domain. It focuses on the analysis of greenhouse gas emissions, specifically Carbon Dioxide (CO2). This data is critical for researchers, policymakers, journalists, and the public to understand emission trends, identify major contributors, and evaluate the effectiveness of climate policies. The dataset used (owid-co2-data.csv) is a well-known, public dataset from "Our World in Data", providing comprehensive emissions data by country and year.

1.2 Software Libraries Used

Software Libraries Used

The project was built using Python 3 and relies on two primary data science libraries:

- **Pandas:** Used for the core data analysis tasks. This includes loading the CSV data into a DataFrame, cleaning missing values (fillna), filtering data (e.g., by country or year), sorting values (sort_values), and grouping data (groupby).

- **Numpy:** Used for numerical operations and statistical calculations. Specifically, it is used to calculate the mean (np.mean) and median (np.median) emissions for the global statistics report.

1.3 Contributions by Team Members

Contributions by Team Members

- **Arav Kilak (25BCE2015):** Focused on the "backend" logic. Wrote the analysis.py script, including all Pandas/Numpy functions for data loading, filtering, and statistical calculation. Responsible for debugging data-related errors. Managed the GitHub repository for submission.

- **Ishan Tayal (25BCE2031):** Focused on the "frontend" development. Wrote the main.py script, implemented the menu-driven operations, and handled all user input and program flow logic. Led project management and documentation. Coordinated the integration of the two scripts. Prepared the project report.

1.4 Challenges Faced

During development, our team faced two significant challenges that went beyond simple syntax errors: one related to data integrity and another to team workflow.

- **1. Handling Aggregate vs. Granular Data:** Our first major issue was with the dataset itself. When we implemented the "Top 10 Emitters" feature, the results were incorrect. The list showed aggregate regions like 'World', 'Asia', 'Europe', and 'High-income countries' instead of individual countries. We realized the owid-co2-data.csv file contains rows for continents and income groups, not just countries. To solve this, we had to create an "exclusion list" (non_country_list) and use it to filter the DataFrame *before* performing any rankings. This was a critical data-cleaning step to ensure our analysis was accurate and only showed actual countries.

- **2. Integrating the Menu and Analysis Modules:** Since we split the project into main.py (for the menu) and analysis.py (for the logic), we initially faced frequent AttributeError and TypeError issues. The menu code in main.py would call a function that had a different name in analysis.py, or it would expect a different return value (e.g., expecting a printed string but receiving a full DataFrame object). To resolve this, we had to establish a clear "contract" between the two files. We defined the *exact* function names (e.g., show_top_emitters), the parameters each would accept (df), and what each function would be responsible for (e.g., show_top_emitters is responsible for *printing* its own output, not *returning* it). This made integrating our separate parts much smoother.

1.5 GitHub Repository

The complete source code and project files for this explorer are publicly available on GitHub at: https://github.com/kenthusian/co2-emissions-explorer

2. Problem Statement and Objectives

Problem Statement: Publicly available environmental datasets, such as the comprehensive CO2 emissions data from "Our World in Data," are often large and complex. This makes it difficult for individuals without technical data analysis skills (like SQL or Pandas) to access, query, or derive meaningful insights from the data.

Objectives: The primary objectives of this project are:

1. To develop a user-friendly, menu-driven command-line tool using Python fundamentals.

2. To use the Pandas and Numpy libraries to load, clean, and analyze a large, real-world CO2 emissions dataset.

3. To implement a set of clear, distinct features that allow users to ask specific questions about the data (e.g., Top 10 emitters, country-specific trends).

4. To create a modular, two-part project structure (menu + analysis) that satisfies the requirements for Assessment 6.

3. Implementation

The implementation is split into features, where each feature corresponds to a menu option. The code for each feature resides in analysis.py and is called from main.py.

3.1 Feature: Show Top 10 Total CO2 Emitters

This feature finds the most recent year in the dataset, filters for all countries in that year, and displays the 10 countries with the highest total co2 emissions.

```
def show_top_emitters(df):
    latest_df, year = get_latest_year_df(df)

    top_10 = latest_df.sort_values(by='co2', ascending=False).head(10)

    print(f"\n--- Top 10 Total CO2 Emitters ({year}) ---")
    print(top_10[['country', 'co2']].to_string(index=False))
```

3.2. Feature: Show Top 10 Per Capita CO2 Emitters

This feature is similar to 3.1 but sorts the data by the co2_per_capita column to show which countries have the highest emissions relative to their population.

```
def show_top_per_capita(df):
    latest_df, year = get_latest_year_df(df)

    top_10 = latest_df.sort_values(by='co2_per_capita',
ascending=False).head(10)

    print(f"\n--- Top 10 Per Capita CO2 Emitters ({year}) ---")
    print(top_10[['country', 'co2_per_capita']].to_string(index=False))
```

3.3. Feature: Analyze Emissions for a Specific Country

This feature prompts the user for a country name. It then filters the entire dataset for that country and displays its historical emissions data (total and per capita) at 5-year intervals to show its trend over time.

```
def analyze_country(df):
    country = input("Enter the Country name (e.g., 'India'):
").strip().title()

    country_df = df[df['country'] == country]

    if country_df.empty:
        print(f"No data found for: {country}")
    else:
        print(f"\n--- Emissions Trend for {country} (every 5 years) ---")
        print(country_df[country_df['year'] % 5 == 0][['year', 'co2',
'co2_per_capita']].to_string(index=False))
```

3.4. Feature: Compare Emissions Between Two Countries

This feature asks the user for two country names. It then finds the data for both countries in the most recent year and displays their total and per capita emissions side-by-side for easy comparison.

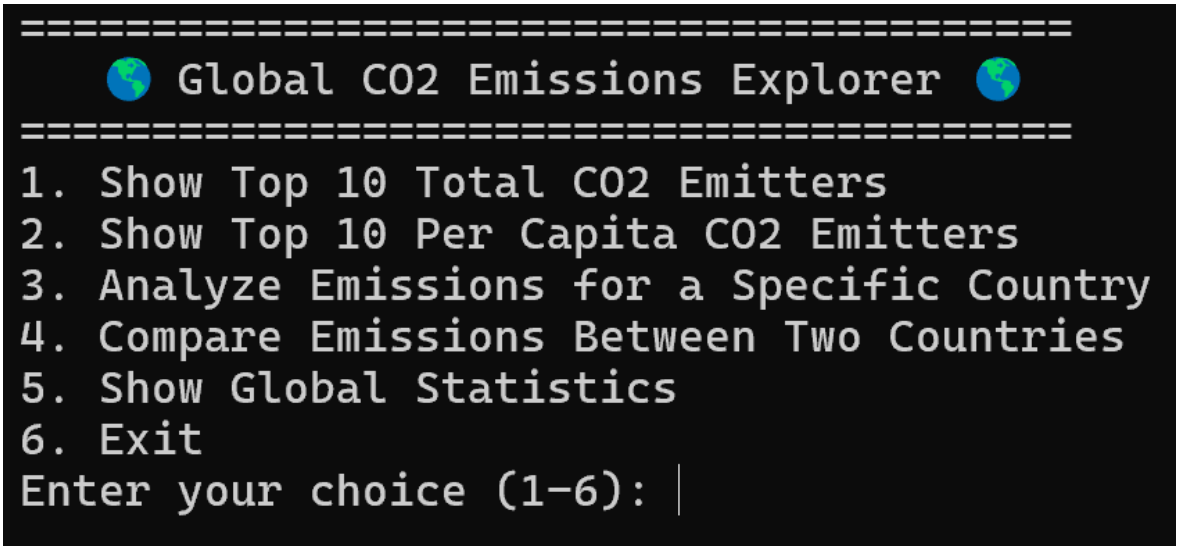
```
def compare_countries(df):
    country1 = input("Enter first country: ").strip().title()
    country2 = input("Enter second country: ").strip().title()

    latest_df, year = get_latest_year_df(df)

    compare_df = latest_df[latest_df['country'].isin([country1,
country2])]

    if compare_df.empty:
        print(f"One or both countries not found in {year} data.")
    else:
        print(f"\n--- Comparison for {year} ---")
        print(compare_df[['country', 'co2',
'co2_per_capita']].to_string(index=False))
```

3.5. Feature: Show Global Statistics

This feature uses the specific "World" entry in the dataset to show the total global emissions. It also uses Numpy to calculate the average (np.mean) and median (np.median) emissions across all individual countries.

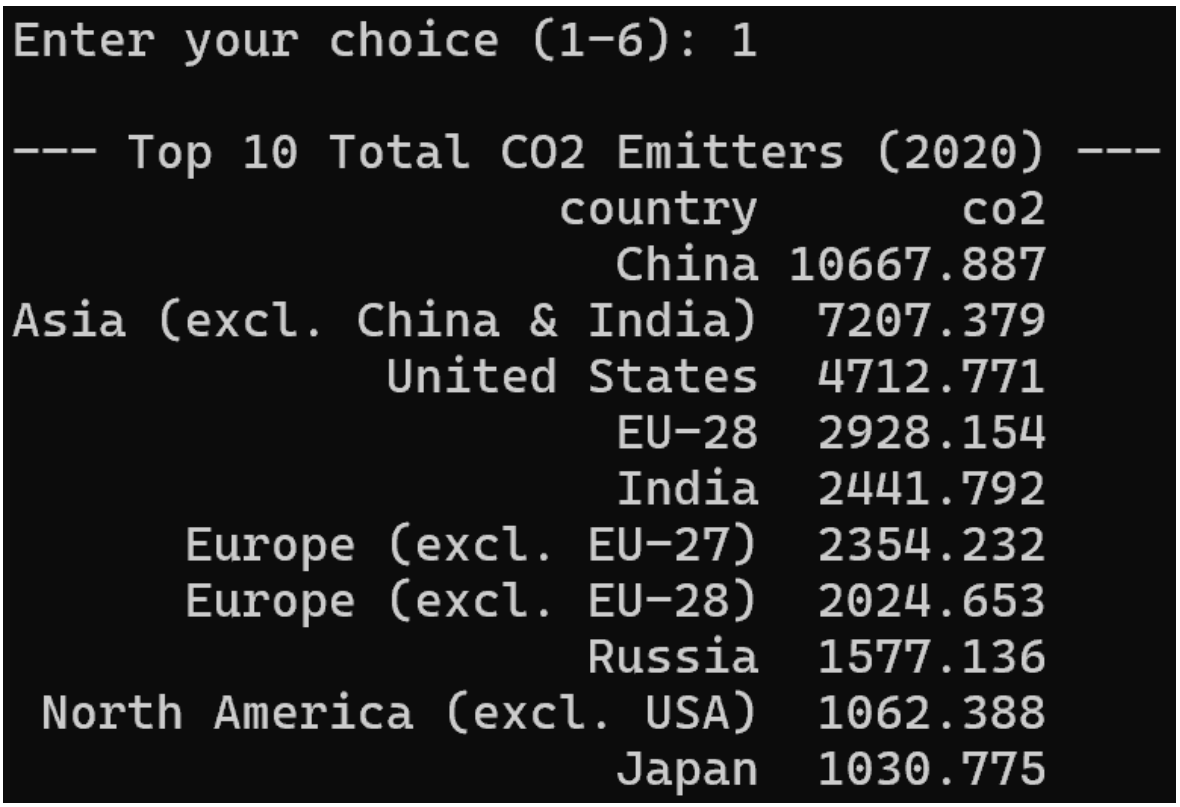
```
def show_global_stats(df):
    latest_year = df['year'].max()
    world_data = df[(df['year'] == latest_year) & (df['country'] ==
'World')]

    if world_data.empty:
        print("Could not find 'World' data for global stats.")
        return

    world_stats = world_data.iloc[0]

    latest_countries_df, _ = get_latest_year_df(df)

    valid_emitters = latest_countries_df[latest_countries_df['co2'] > 0]

    avg_emission = np.mean(valid_emitters['co2'])
    median_emission = np.median(valid_emitters['co2'])

    print(f"\n--- Global Stats for {latest_year} ---")
    print(f"Total Global CO2 Emissions: {world_stats['co2']:.2f} (million
tonnes)")
    print(f"Average Global CO2 Per Capita:
{world_stats['co2_per_capita']:.2f} (tonnes per person)")
    print(f"--- Per-Country Stats ---")
    print(f"Average Emission per Country: {avg_emission:.2f} (million
tonnes)")
    print(f"Median Emission per Country: {median_emission:.2f} (million
tonnes)")
```

5.Demo Screenshots



Main Menu

Data Loading



Feature 1 Output

```
Enter the Country name (e.g., 'India'): India

--- Emissions Trend for India (every 5 years) ---
 year      co2  co2_per_capita
 1860    0.644           0.003
 1865    0.568           0.002
 1870    0.000           0.000
 1875    0.000           0.000
 1880    1.889           0.007
 1885    2.463           0.009
 1890    4.437           0.016
 1895    7.043           0.024
 1900   11.945           0.041
 1905   16.952           0.056
 1910   23.517           0.076
 1915   33.184           0.106
 1920   34.913           0.110
 1925   40.137           0.121
 1930   42.459           0.122
 1935   41.450           0.111
 1940   52.382           0.133
 1945   52.946           0.138
 1950   61.177           0.163
 1955   78.879           0.192
 1960  111.450           0.247
 1965  153.868           0.308
 1970  181.899           0.328
 1975  234.439           0.376
 1980  291.992           0.418
 1985  397.953           0.507
 1990  578.518           0.662
 1995  762.121           0.791
 2000  978.919           0.926
 2005 1185.953           1.033
 2010 1677.888           1.359
 2015 2268.567           1.732
 2020 2441.792           1.769
```

Feature 3 Output

```
Enter your choice (1-6): 4
Enter first country: India
Enter second country: China

--- Comparison for 2020 ---
country        co2  co2_per_capita
  China 10667.887           7.412
  India  2441.792           1.769
```

Feature 4 Output

```
Enter your choice (1-6): 5

--- Global Stats for 2020 ---
Total Global CO2 Emissions: 34807.26 (million tonnes)
Average Global CO2 Per Capita: 4.46 (tonnes per person)
--- Per-Country Stats ---
Average Emission per Country: 220.45 (million tonnes)
Median Emission per Country: 10.02 (million tonnes)
```

Feature 5 Output

5. Conclusion

The "Global CO2 Emissions Explorer" project successfully meets all the objectives set forth in the project guidelines. We created a functional, two-part application that effectively separates user-facing logic (menu) from data analysis logic (Pandas/Numpy). The tool demonstrates a practical application of data science libraries to solve a real-world problem: making complex data accessible. Through this project, we gained hands-on experience in data cleaning, filtering, and aggregation, and successfully overcame debugging challenges like the initial KeyError. The final application is a useful and robust tool for anyone interested in exploring global CO2 emissions data.