Analyzing the Correlation Between Temperature Changes and greenhouse gas emissions from different sources in Germany

Author: Koosha Samadi

Contents:

- 1. Introduction
- 2. Methods & Data Sources
- **3.** Data Pipeline

1. Introduction

It's important to understand how climate change and greenhouse gas emissions from different sources are related. My data science project looks at how temperature changes and gas emissions specifically CO2 interact in Germany. The goal is to learn things that can help government leaders make plans to reduce the negative impacts of climate change on the environment.

I will be examining how temperature changes and emissions have evolved over the years, focusing specifically on Germany. One of my primary areas of interest is the correlation between temperature fluctuations and different types of emissions within this region. This focus is driven by the need to understand how well these efforts are correlating with changes in temperature patterns and to provide actionable insights that can guide environmental policies in Germany.

2. Methods & Data Sources

Data Source 1: Food and Agriculture Organization (FAO)

I have selected the FAO Corporate Statistical Database as my primary data source for its extensive range of reliable and comprehensive statistical data. This source is particularly valuable due to its well-maintained records and robust reputation in the field.

Metadata URL 1: FAO Statistics - Greenhouse Gas Emissions

Data URL 1: European GHG Emissions Data (CSV)

Data Type: CSV file containing yearly greenhouse gas emission data for European countries

from 1961 to 2023.

License: CC BY-NC-SA 3.0 IGO

The FAOSTAT Emissions Totals domain offers detailed data on greenhouse gas emissions from agrifood systems, tracking emissions such as methane (CH4), nitrous oxide (N2O), carbon dioxide (CO2), and fluorinated gases (F-gases). This data adheres to the IPCC Tier 1 methodology and covers various economic sectors as outlined by the IPCC.

Data Source 2: Food and Agriculture Organization (FAO)

For temperature data, I have again chosen FAO due to their comprehensive environmental statistics.

Metadata URL 2: FAO Statistics - Temperature Change

Data URL 2: <u>European Temperature Data (CSV)</u>

Data Type: CSV file featuring FAOSTAT monthly temperature data for European countries.

License: CC BY-NC-SA 3.0 IGO

The FAOSTAT Temperature Change on Land domain provides statistics on mean surface

temperature changes by country, updated annually. This dataset covers the period from 1961 to 2023 and includes monthly, seasonal, and annual mean temperature anomalies.

3. Data Pipeline

In the data pipeline phase of my project, I focused on organizing and cleaning the datasets effectively to make sure they were ready for analysis. This process involved several specific steps to handle and improve the data related to greenhouse gas emissions and temperature changes across European countries.

First, I imported necessary libraries in python.

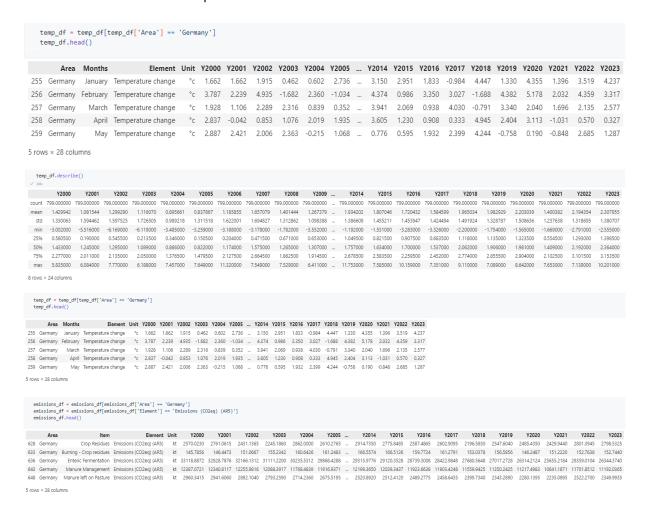
```
import pandas as pd
import numpy as np
from sqlalchemy import create_engine
```

Then, as you can see in the below image, I cleaned up the dataset by removing unnecessary columns that were not going to be used in my analysis. For the greenhouse gas emissions data, I removed columns like 'Area Code', 'Area Code (M49)', 'Item Code', 'Element Code', 'Source Code', and 'Source'. For the temperature data, I eliminated 'Area Code', 'Area Code (M49)', 'Months Code', and 'Element Code'. These columns were mostly identifiers and metadata that wouldn't contribute to my analysis focused on trends and relationships. After tidying up the columns, I also made sure to drop any rows that had missing values to maintain the quality and accuracy of my analysis. My next step was to streamline the datasets by discarding historical data from 1961 to 1999, since I was more interested in the recent data and future projections. This helped make the dataset smaller and easier to manage.

```
def process_data(file_path):
      df = pd.read_csv(file_path, encoding="utf-8")
   except UnicodeDecodeError:
       df = pd.read csv(
          file_path, encoding="ISO-8859-1"
   start_year = "Y1961"
   end_year = "Y1999"
       f"number of rows before processing: {df.shape[0]} and number of columns before processing: {df.shape[1]}"
   # drop column between start year and end year
   df.drop(df.loc[:, start_year:end_year].columns, axis=1, inplace=True)
   df.rename(columns={'Y2030':'Y2022','Y2050':'Y2023'}, inplace=True)
    #drop Area Code and Area Code (M49) and Item Code and Element Code and Source Code and Source from emmision data
   if 'Item Code' in df.columns:
       df.drop(['Area Code', 'Area Code (M49)', 'Item Code', 'Element Code', 'Source Code', 'Source'], axis=1, inplace=True)
   #drop Area Code and Area Code (M49) and Months Code and Element Code from temperature data
   if 'Months Code' in df.columns:
       df.drop(['Area Code','Area Code (M49)','Months Code','Element Code'], axis=1, inplace=True)
       #drop rows that Element == Standard Deviation
       df = df[df.Element != 'Standard Deviation']
   return df
def clean data(df):
   df.dropna(inplace=True)
```

Then, I assessed the datasets' dimensions, noting that the temperature dataset comprised 799 rows and 28 columns, while the emissions dataset contained 1864 rows and 28 columns.

You can see the short description and information about both datasets.



The Data Structure and Quality of The Result:

- Data Cleaning: I removed irrelevant columns and rows, normalizes data by renaming years, and eliminates rows with missing values to clean the dataset for analysis.
- Standardization: I standardized column names and filters out unnecessary data, making the dataset more manageable and focused on relevant metrics.
- Error Handling: Errors are minimally handled, primarily focusing on encoding issues, but without advanced error recovery or comprehensive logging.
- Data Output: The cleaned and processed data is stored in SQLite databases, allowing for scalable and persistent data management.

Note: I've decided to switch to a new dataset because they provide a better opportunity to understand and solve actual problems. These datasets will help me learn more effectively.