Data Report, Akash Yadav (23304362)

1. Data Report: Intro

In recent years, problems like drought, severe heat waves, ever-increasing temperatures, heavy rainfall causing floods, seasonal cycle inconsistency, and hunger crises have been escalating, highlighting the urgent need to understand the main causes of environmental damage. With the global population continuing to rise at an unpredictable rate, the increase in CO2 emissions from forest fires, crop cultivation, pesticide manufacturing, and agrifood waste disposal presents a significant challenge to sustainable development and climate stability.To better understand the correlation between population growth and CO2 emissions on climate change and these rising issues, we will utilize a robust dataset sourced from Kaggle. The objective of this deep dive and analysis is to provide insights that are relevant to various fields, including policy-making, resource allocation, and urban planning.

Main Question:

How does population distribution and density influence the climate impact of CO2 emissions across different regions?

2. Data Sources

2.1 Data Source 1: Agri-food CO2 emission

I have selected CO2 emissions as my data source because it provides country and year-wise CO2 emissions from various sources. This dataset has received the highest number of upvotes on Kaggle and provides open data license.

Data Source: Kaggle | Data URL: Agri-food CO2 emission | Data Type: CSV | License: CC BY Creative Commons Attribution

Data Description:

The agricultural sector is a significant contributor to climate change. This dataset plays a crucial role in understanding and monitoring the impact of agricultural activities on CO2 emissions.The dataset used for the study on CO2 emissions and temperature change for each country from 1990 to 2020. It has 7k rows and 31 columns.

2.2 Data Source 2: World Population

For population data, I found this data source. This has data about growth rate, world percentage change, density, and decade-wise data.

Data Source: Kaggle | Data URL: 2023 World Population | Data Type: CSV | License: CC0: Public Domain

Data Description:

The US Census Bureau's world population clock estimated that the global population as of September 2022 was 7,922,312,800 people and was expected to reach 8 billion by mid-November of 2022. This total far exceeds the 2015 world population of 7.2 billion. The world's population continues to increase by roughly 140 people per minute, with births outweighing deaths in most countries. However, The rate of

population growth has been slowing for several decades. This slowdown is expected to continue until the rate of population growth reaches zero (an equal number of births and deaths) around 2080-2100.

Data Structure and Quality:

Both the datasets are structured data organized in a table-like format. Some inconsistencies were present in the data, which were addressed through missing data imputation. It adheres to all dimensions of data quality: **Accuracy, Completeness, Consistency, Timeliness, and Relevancy**.

2.3 License Description:

CC0 1.0 DEED: Both the data sources are *open licensed data*:

The person who associated a work with this deed has dedicated the work to the public domain by waiving all of his or her rights to the work worldwide under copyright law, including all related and neighboring rights, to the extent allowed by law. You can **copy, modify, distribute and perform the work, even for commercial purposes, all without asking permission**.for quick review of License, click on aforementioned hyperlinks.

3. Data Pipeline: -> Kaggle Authentication -> Extract -> Clean&Transform -> Saved to DB

- step 1. Initialization of Pipeline, here you will have to put Kaggle user and token in .kaggle/kaggle.json.
- step 2. Authentication and connection establishment.
- step 3. Extract Data class will call load dataset.
- step 4. Clean and Transform method will take care of missing data and correct datatypes.
- step 5. Lastly save trasformed data to sqlite database.
- **Technologies**: Python, Shell script, SQLite Db, Pandas, Numpy, KaggleApi

Transformation and Cleaning Technique:

- Data Imputation: There was some missing numerical data, which was imputed using the mean technique to maintain the integrity of the dataset and enable meaningful analysis.
- No Normalization and Standardization: There was no need to normalize or standardize data, as we wanted to analyze the data without losing its original shape and outliers, which can be significant in finding relevant insights.

Problems encountered and solution

- Kaggle connectivity issues, to solve these problems created .kaggle/kaggle.json which will hold user credentials to connect and authenticate user.
- Extract call was downloading and unzipping all files in data dir which was not required, to remove unnecessary files clean-up method implemented which will remove them all.

How pipeline deal with errors and changing input

- Current data pipeline deals with missing data imputation, remove temporary files from directory, work for any input data as long as data source is Kaggle.

4. Result and Limitations

- The output data format is a relational database (SQLite) because it can be easily maintained and analyzed. Another reason is that both datasets are structured data types, so saving them in any other format does not make sense.

- If the data source is other than Kaggle, then the extract method won't work, which is a limitation. However, other blocks of the pipeline are unaffected and perform their jobs flawlessly.
- The pipeline is automated, so if the input dataset has more missing data, the imputation method would require a more emphasized and precise approach to replace missing values. One possible solution for this would be to set some kind of threshold (in %) to choose from missing value computation methods such as KNN, mean, median, mode, and delete.
- The ETL pipeline has basic transformation operations. After EDA, the significance of features can be considered to drop and keep columns, which cannot be done at the initial stage.

In [234…

```
from IPython.display import Image, display
display(Image(filename='/images/work_flow.png', width=500, height=600))
```