Data Analysis

August 7, 2023

1 Data Analysis: World Data 2023

1.1 Objective

Purpose of this prject is to gather some interesting insights out of this data.

1.2 Tech Stack

Programming Language: Python

Libraries: Numpy, Pandas, Seaborn

Algorithms:

1.3 Code

[1]:

```
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import seaborn as sns # data visualization

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory

# For Kaggle environment
"""
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
"""
```

```
# You can write up to 20GB to the current directory (/kaggle/working/) that␣
 ↪gets preserved as output when you create a version using "Save & Run All"
# You can also write temporary files to /kaggle/temp/, but they won't be saved␣
 ↪outside of the current session
```

```
[1]: "\nimport os\nfor dirname, _, filenames in os.walk('/kaggle/input'):\n    for
     filename in filenames:\n        print(os.path.join(dirname, filename))\n"
```

1.3.1 Data Pre-processing

In this section we will perform all the necessary task to align the data as per our needs. The process will include multiple steps such as Data profiling, Data cleansin, Data reducti, . Data transformat, .. Data enrich, … Data valid etc. All these steps will be used per our need.tion.

```
[2]: #  having the first look at the dataset

     dataset = pd.read_csv("./world-data-2023.csv")
     print(dataset.head(5))
```

```
       Country Density\n(P/Km2) Abbreviation Agricultural Land( %)
0  Afghanistan              60           AF                58.10%  \
1      Albania             105           AL                43.10%
2      Algeria              18           DZ                17.40%
3      Andorra             164           AD                40.00%
4       Angola              26           AO                47.50%

  Land Area(Km2) Armed Forces size  Birth Rate  Calling Code
0        652,230           323,000       32.49          93.0  \
1         28,748             9,000       11.78         355.0
2      2,381,741           317,000       24.28         213.0
3            468               NaN        7.20         376.0
4      1,246,700           117,000       40.73         244.0

  Capital/Major City Co2-Emissions  … Out of pocket health expenditure
0              Kabul         8,672  …                           78.40%  \
1             Tirana         4,536  …                           56.90%
2            Algiers       150,006  …                           28.10%
3   Andorra la Vella           469  …                           36.40%
4             Luanda        34,693  …                           33.40%

  Physicians per thousand  Population
0                    0.28  38,041,754  \
1                    1.20   2,854,191
2                    1.72  43,053,054
3                    3.33      77,142
4                    0.21  31,825,295

   Population: Labor force participation (%) Tax revenue (%) Total tax rate
```

```
0                                         48.90%           9.30%          71.40%  \
1                                         55.70%          18.60%          36.60%
2                                         41.20%          37.20%          66.10%
3                                            NaN             NaN             NaN
4                                         77.50%           9.20%          49.10%

  Unemployment rate Urban_population   Latitude  Longitude
0            11.12%        9,797,273  33.939110  67.709953
1            12.33%        1,747,593  41.153332  20.168331
2            11.70%       31,510,100  28.033886   1.659626
3               NaN           67,873  42.506285   1.521801
4             6.89%       21,061,025 -11.202692  17.873887

[5 rows x 35 columns]
```

[3]:

```python
# Let's see what columns are we working with
columns = dataset.columns

print(columns)
```

```
Index(['Country', 'Density\n(P/Km2)', 'Abbreviation', 'Agricultural Land( %)',
       'Land Area(Km2)', 'Armed Forces size', 'Birth Rate', 'Calling Code',
       'Capital/Major City', 'Co2-Emissions', 'CPI', 'CPI Change (%)',
       'Currency-Code', 'Fertility Rate', 'Forested Area (%)',
       'Gasoline Price', 'GDP', 'Gross primary education enrollment (%)',
       'Gross tertiary education enrollment (%)', 'Infant mortality',
       'Largest city', 'Life expectancy', 'Maternal mortality ratio',
       'Minimum wage', 'Official language', 'Out of pocket health expenditure',
       'Physicians per thousand', 'Population',
       'Population: Labor force participation (%)', 'Tax revenue (%)',
       'Total tax rate', 'Unemployment rate', 'Urban_population', 'Latitude',
       'Longitude'],
      dtype='object')
```

Features guide **Country:** Name of the country

Density (P/Km2): Population density measured in persons per square kilometer

Abbreviation: Abbreviation or code representing the country

Agricultural Land (%): Percentage of land area used for agricultural purposes

Land Area (Km2): Total land area of the country in square kilometer

Armed Forces Size: Size of the armed forces in the country

Birth Rate: Number of births per 1,000 population per year

Calling Code: International calling code for the country

Capital/Major City: Name of the capital or major city

CO2 Emissions: Carbon dioxide emissions in tons

CPI: Consumer Price Index, a measure of inflation and purchasing power

CPI Change (%): Percentage change in the Consumer Price Index compared to the previous year

Currency_Code: Currency code used in the country

Fertility Rate: Average number of children born to a woman during her lifetime

Forested Area (%): Percentage of land area covered by forests

Gasoline_Price: Price of gasoline per liter in local currency

GDP: Gross Domestic Product, the total value of goods and services produced in the country

Gross Primary Education Enrollment (%): Gross enrollment ratio for primary education

Gross Tertiary Education Enrollment (%): Gross enrollment ratio for tertiary education

Infant Mortality: Number of deaths per 1,000 live births before reaching one year of age

Largest City: Name of the country's largest city

Life Expectancy: Average number of years a newborn is expected to live

Maternal Mortality Ratio: Number of maternal deaths per 100,000 live births

Minimum Wage: Minimum wage level in local currency

Official Language: Official language(s) spoken in the country

Out of Pocket Health Expenditure (%): Percentage of total health expenditure paid out-of-pocket by individuals

Physicians per Thousand: Number of physicians per thousand people

Population: Total population of the country

Population: Labor Force Participation (%): Percentage of the population that is part of the labor force

Tax Revenue (%): Tax revenue as a percentage of GDP

Total Tax Rate: Overall tax burden as a percentage of commercial profits

Unemployment Rate: Percentage of the labor force that is unemployed

Urban Population: Percentage of the population living in urban areas

Latitude: Latitude coordinate of the country's location

Longitude: Longitude coordinate of the country's location

[4]:

```
# Let's see what datatype these features are

print(dataset.dtypes)
```

```
Country                                       object
Density\n(P/Km2)                              object
Abbreviation                                  object
Agricultural Land( %)                         object
Land Area(Km2)                                object
Armed Forces size                             object
Birth Rate                                   float64
Calling Code                                 float64
Capital/Major City                            object
Co2-Emissions                                 object
CPI                                           object
CPI Change (%)                                object
Currency-Code                                 object
Fertility Rate                               float64
Forested Area (%)                             object
Gasoline Price                                object
GDP                                           object
Gross primary education enrollment (%)        object
Gross tertiary education enrollment (%)       object
Infant mortality                             float64
Largest city                                  object
Life expectancy                              float64
Maternal mortality ratio                     float64
Minimum wage                                  object
Official language                             object
Out of pocket health expenditure              object
Physicians per thousand                      float64
Population                                    object
Population: Labor force participation (%)     object
Tax revenue (%)                               object
Total tax rate                                object
Unemployment rate                             object
Urban_population                              object
Latitude                                     float64
Longitude                                    float64
dtype: object
```

As there are so many features with object datatype, which should have been int or float, I suspect that there are null values present in the dataset.

[5]:

```
# Let's have a look at the columns having null values in the dataset

print(dataset.isnull().sum())
```

```
Country                                       0
Density\n(P/Km2)                              0
Abbreviation                                  7
Agricultural Land( %)                         7
Land Area(Km2)                                1
```

```
Armed Forces size                            24
Birth Rate                                    6
Calling Code                                  1
Capital/Major City                            3
Co2-Emissions                                 7
CPI                                          17
CPI Change (%)                               16
Currency-Code                                15
Fertility Rate                                7
Forested Area (%)                             7
Gasoline Price                               20
GDP                                           2
Gross primary education enrollment (%)        7
Gross tertiary education enrollment (%)      12
Infant mortality                              6
Largest city                                  6
Life expectancy                               8
Maternal mortality ratio                     14
Minimum wage                                 45
Official language                             5
Out of pocket health expenditure              7
Physicians per thousand                       7
Population                                    1
Population: Labor force participation (%)    19
Tax revenue (%)                              26
Total tax rate                               12
Unemployment rate                            19
Urban_population                              5
Latitude                                      1
Longitude                                     1
dtype: int64
```

There are numerous columns containing null values. The initial impulse might be to drop the rows with missing data, but since this dataset is not intended for machine learning, doing so would lead to the exclusion of entire countries, introducing potential biases or skewed outcomes. Moreover, valuable insights from these countries could be lost.

To address these concernsIwe propose marking numerical null values as "-1" and string-based null values as "NULL". This approach effectively indicates that the data was initially empty, ensuring that we retain all potential important insights without compromising the integrity of our results.

Upon examining the dataset, we observe two columns, namely "Longitude" and "Latitude," which cannot be marked as -1 since their valid range is from -180 to 180. To handle null values in these columns, we will fill them with a default value of 1000.

[6]:

```
# Let's fill all the numeric columns with -1

dataset[columns[3]].fillna(-1, inplace=True)       # Agricultural Land( %)
dataset[columns[4]].fillna(-1, inplace=True)       # Land Area(Km2)
```

```
dataset[columns[5]].fillna(-1, inplace=True)       # Armed Forces size
dataset[columns[6]].fillna(-1, inplace=True)       # Birth Rate
dataset[columns[7]].fillna(-1, inplace=True)       # Calling Code
dataset[columns[9]].fillna(-1, inplace=True)       # Co2-Emissions
dataset[columns[10]].fillna(-1, inplace=True)      # CPI
dataset[columns[11]].fillna(-1, inplace=True)      # CPI Change (%)
dataset[columns[13]].fillna(-1, inplace=True)      # Fertility Rate
dataset[columns[14]].fillna(-1, inplace=True)      # Forested Area (%)
dataset[columns[15]].fillna(-1, inplace=True)      # Gasoline Price
dataset[columns[16]].fillna(-1, inplace=True)      # GDP
dataset[columns[17]].fillna(-1, inplace=True)      # Gross primary education␣
 ↪enrollment (%)
dataset[columns[18]].fillna(-1, inplace=True)      # Gross tertiary education␣
 ↪enrollment (%)
dataset[columns[19]].fillna(-1, inplace=True)      # Infant mortality
dataset[columns[21]].fillna(-1, inplace=True)      # Life expectancy
dataset[columns[22]].fillna(-1, inplace=True)      # Maternal mortality ratio
dataset[columns[23]].fillna(-1, inplace=True)      # Minimum wage
dataset[columns[25]].fillna(-1, inplace=True)      # Out of pocket health␣
 ↪expenditure
dataset[columns[26]].fillna(-1, inplace=True)      # Physicians per thousand
dataset[columns[27]].fillna(-1, inplace=True)      # Population
dataset[columns[28]].fillna(-1, inplace=True)      # Population: Labor force␣
 ↪participation (%)
dataset[columns[29]].fillna(-1, inplace=True)      # Tax revenue (%)
dataset[columns[30]].fillna(-1, inplace=True)      # Total tax rate
dataset[columns[31]].fillna(-1, inplace=True)      # Unemployment rate
dataset[columns[32]].fillna(-1, inplace=True)      # Urban_population

# Let's fill all the textual columns with the string "NULL"
dataset[columns[2]].fillna("NULL", inplace=True)   # Abbreviation
dataset[columns[8]].fillna("NULL", inplace=True)   # Capital/Major City
dataset[columns[12]].fillna("NULL", inplace=True)  # Currency-Code
dataset[columns[20]].fillna("NULL", inplace=True)  # Largest city
dataset[columns[24]].fillna("NULL", inplace=True)  # Official language

# Let's fill the "Longitude" and "Latitude" with 1000
dataset[columns[33]].fillna(1000, inplace=True)    # Latitude
dataset[columns[34]].fillna(1000, inplace=True)    # Longitude
```

[7]:

```
# Now Let's see if we have any null values left in our dataset or not...

print(dataset.isnull().sum())
```

```
Country                                      0
Density\n(P/Km2)                             0
Abbreviation                                 0
```

```
Agricultural Land( %)                        0
Land Area(Km2)                               0
Armed Forces size                            0
Birth Rate                                   0
Calling Code                                 0
Capital/Major City                           0
Co2-Emissions                                0
CPI                                          0
CPI Change (%)                               0
Currency-Code                                0
Fertility Rate                               0
Forested Area (%)                            0
Gasoline Price                               0
GDP                                          0
Gross primary education enrollment (%)       0
Gross tertiary education enrollment (%)      0
Infant mortality                             0
Largest city                                 0
Life expectancy                              0
Maternal mortality ratio                     0
Minimum wage                                 0
Official language                            0
Out of pocket health expenditure             0
Physicians per thousand                      0
Population                                   0
Population: Labor force participation (%)    0
Tax revenue (%)                              0
Total tax rate                               0
Unemployment rate                            0
Urban_population                             0
Latitude                                     0
Longitude                                    0
dtype: int64
```

[8]:

```
# Now Let's save this dataset as a temporary file.

dataset.to_csv('./temp/null_filled_dataset.csv')

dataset = pd.read_csv("./temp/null_filled_dataset.csv")
```

Now, we need to convert the columns with object type to float, string or int by removing the string characters, such as “,” and “%”.

[9]:

```
# Let's convert these datatypes to their appropriate ones

dataset[columns[1]]  = dataset[columns[1]].str.replace(',', '').astype(int)     ␣
 ↪                      # Density (P/Km2)
```

```
dataset[columns[3]]  = dataset[columns[3]].str.replace('%', '').astype(float)  ␣
 ↪                       # Agricultural Land( %)
dataset[columns[4]]  = dataset[columns[4]].str.replace(',', '').astype(int)    ␣
 ↪                       # Land Area(Km2)
dataset[columns[5]]  = dataset[columns[5]].str.replace(',', '').astype(int)    ␣
 ↪                       # Armed Forces size
dataset[columns[7]]  = dataset[columns[7]].astype(int)                         ␣
 ↪                       # Calling Code
dataset[columns[9]]  = dataset[columns[9]].str.replace(',', '').astype(int)    ␣
 ↪                       # Co2-Emissions
dataset[columns[10]] = dataset[columns[10]].str.replace(',', '').astype(float) ␣
 ↪                       # CPI
dataset[columns[11]] = dataset[columns[11]].str.replace('%', '').astype(float) ␣
 ↪                       # CPI Change (%)
dataset[columns[14]] = dataset[columns[14]].str.replace('%', '').astype(float) ␣
 ↪                       # Forested Area (%)
dataset[columns[15]] = dataset[columns[15]].str.replace('$', '').astype(float) ␣
 ↪                       # Gasoline Price
dataset[columns[16]] = dataset[columns[16]].str.replace('$', '').str.
 ↪replace(',', '').astype(float)  # GDP
dataset[columns[17]] = dataset[columns[17]].str.replace('%', '').astype(float) ␣
 ↪                       # Gross primary education enrollment (%)
dataset[columns[18]] = dataset[columns[18]].str.replace('%', '').astype(float) ␣
 ↪                       # Gross tertiary education enrollment (%)
dataset[columns[22]] = dataset[columns[22]].astype(int)                        ␣
 ↪                       # Maternal mortality ratio
dataset[columns[23]] = dataset[columns[23]].str.replace('$', '').astype(float) ␣
 ↪                       # Minimum wage
dataset[columns[25]] = dataset[columns[25]].str.replace('%', '').astype(float) ␣
 ↪                       # Out of pocket health expenditure
dataset[columns[27]] = dataset[columns[27]].str.replace(',', '').astype(int)   ␣
 ↪                       # Population
dataset[columns[28]] = dataset[columns[28]].str.replace('%', '').astype(float) ␣
 ↪                       # Population: Labor force participation (%)
dataset[columns[29]] = dataset[columns[29]].str.replace('%', '').astype(float) ␣
 ↪                       # Tax revenue (%)
dataset[columns[30]] = dataset[columns[30]].str.replace('%', '').astype(float) ␣
 ↪                       # Total tax rate
dataset[columns[31]] = dataset[columns[31]].str.replace('%', '').astype(float) ␣
 ↪                       # Unemployment rate
dataset[columns[32]] = dataset[columns[32]].str.replace(',', '').astype(int)   ␣
 ↪                       # Urban_population
```

[10]:

```
# Let's see the datatypes of these features now
```

```
print(dataset.dtypes)
```

```
Unnamed: 0                                     int64
Country                                       object
Density\n(P/Km2)                               int32
Abbreviation                                  object
Agricultural Land( %)                        float64
Land Area(Km2)                                 int32
Armed Forces size                              int32
Birth Rate                                   float64
Calling Code                                   int32
Capital/Major City                            object
Co2-Emissions                                  int32
CPI                                          float64
CPI Change (%)                               float64
Currency-Code                                 object
Fertility Rate                               float64
Forested Area (%)                            float64
Gasoline Price                               float64
GDP                                            int64
Gross primary education enrollment (%)       float64
Gross tertiary education enrollment (%)      float64
Infant mortality                             float64
Largest city                                  object
Life expectancy                              float64
Maternal mortality ratio                       int32
Minimum wage                                 float64
Official language                             object
Out of pocket health expenditure             float64
Physicians per thousand                      float64
Population                                     int32
Population: Labor force participation (%)    float64
Tax revenue (%)                              float64
Total tax rate                               float64
Unemployment rate                            float64
Urban_population                               int32
Latitude                                     float64
Longitude                                    float64
dtype: object
```

[11]:

```
# Now Let's save this dataset as a temporary file.

dataset.to_csv('./temp/correct_dtype_dataset.csv')

dataset = pd.read_csv("./temp/correct_dtype_dataset.csv")
```

1.4 Work in progress

Please note that the analysis is not yet complete, and there may be areas that require further attention and refinement. Your feedback on the current work done would be invaluable in helping me identify potential improvements and areas where I can delve deeper for more insights. As I continue to work on this project, I aim to provide a comprehensive and accurate analysis. Your inputs as a fresh pair of eyes will be highly appreciated in making this data analysis report more robust and informative. Thank you for your time and support in this endeavor.

1.5 Let's Collaborate!

I'm always looking for exciting projects and collaborations. If you have suggestions, improvements, or would like to contribute your analysis on a Kaggle dataset, I encourage you to open an issue or create a pull request. Let's connect and create something awesome together!

```
<a href="https://twitter.com/avgeekgupta" target="_blank">
    <img src="https://abs.twimg.com/responsive-web/client-web/icon-ios.b1fc727a.png" width="35"
</a>
<a href="https://www.linkedin.com/in/avgeekgupta" target="_blank">
    <img src="https://static.licdn.com/sc/h/eahiplrwoq61f4uan012ia17i" width="35"> &nbsp; &nbs
</a>
<a href="https://www.kaggle.com/avgeekgupta" target="_blank">
    <img src="https://www.kaggle.com/static/images/favicon.ico" width="35"> &nbsp; &nbsp;
</a>
<a href="https://avgeekgupta.me" target="_blank">
    <img src="https://cdn-icons-png.flaticon.com/512/5602/5602732.png" width="35"> &nbsp; &nbs
</a>
<a href="mailto:u8karshgupta@gmail.com">
    <img src="https://ssl.gstatic.com/ui/v1/icons/mail/rfr/gmail.ico" width="35"> &nbsp; &nbsp
</a>
<a href="tel:+918938914511">
    <img src="https://raw.githubusercontent.com/AvGeekGupta/AvGeekGupta/master/public/assets/p
</a>
```

Feel free to clone, fork, download or use any part of the code from the reports to learn, share, and enrich your understanding of data analysis. Happy analysing!

```
[ ]:
```