ANAND INSTITUTE OF HIGHER TECHNOLOGY
OLD MAHABALIPURAM ROAD,
KALASALINGAM NAGAR,
KAZHIPATTUR - 603103

IBM - NAAN MUDHALVAN
DATA ANALYTICS WITH COGNOS
COVID VACCINES ANALYSIS

PHASE – 3

NAME : SHARU DHARSHINI S

REG No. : 310121104098

BRANCH : COMPUTER SCIENCE & ENGINEERING

YEAR/SEM : III / V

COVID VACCINES ANALYSIS

INTRODUCTION

- The COVID-19 pandemic, caused by the novel coronavirus SARS-CoV-2, has taken the world by storm, leading to widespread illness, loss of life, and societal disruption.
- In response to this global health crisis, the development and deployment of COVID-19 Vaccines have played a pivotal role in controlling the spread of the virus.
- Analyzing COVID-19 Vaccines is a critical aspect of managing and mitigating the impact of the global pandemic. Such analysis encompasses various areas including: Vaccine Development, Efficacy and Effectiveness, Safety and Side Effects, Vaccine Distribution and Access, Variants and Vaccine Adaptations, Vaccine Hesitancy, Global Vaccination Strategies, Economic Impact and Future Preparedness.
- These analyses are crucial for optimizing vaccination strategies ensuring public health, and advancing our understanding of vaccine development and deployment in the face of global health crisis.
- The COVID-19 pandemic has spurred unprecedented efforts in vaccine development and distribution. As vaccines are administered to millions of people worldwide, it is crucial to monitor and optimize the distribution process while closely monitoring adverse effects. Advanced machine learning techniques can play a pivotal role in achieving these goals.

PHASE - 3 : { DEVELOPMENT PART 1 }

- Start building & begin conducting the COVID VACCINES ANALYSIS by collecting and preprocessing the dataset and performing exploratory data analysis.
- In this Phase3, Data preprocessing is a critical step in the analysis of COVID-19 Vaccine data, as it lays the foundation for extracting meaningful insights and patterns from the vast and diverse sources of information related to the pandemic.
- This process involves collecting, cleaning, transforming, reduction of null values, visualization, scalability, efficiency and structuring raw data to make it suitable for analysis.
- The goal of COVID-19 Vaccine Analysis in this Phase3 is to prepare the raw data for analysis, modelling, and decision making.

DATA COLLECTION :

COVID VACCINES ANALYSIS is done by using the Dataset of **"COVID-19 World Vaccination Progress"** provided by the dataset site www.Kaggle.com

DATASET:

https://www.kaggle.com/datasets/gpreda/covid-world-vaccination-progress

DATASET AND ITS DETAILS :

The dataset "COVID-19 World Vaccination Progress" on Kaggle is a collection of data related to the COVID-19 Vaccination efforts worldwide. It provides information about the progress of COVID-19 Vaccinations in various countries and regions. This dataset is designed to help researchers, data scientists, and analysts understand and analyze the progress of COVID-19 Vaccination campaigns across different countries. A second file, with manufacturers information is included. Below is a detailed overview of the dataset:

TITLE: COVID-19 World Vaccination Progress

DATASET ID: gpreda/covid-world-vaccination-progress

SOURCE: The dataset was created by a Kaggle user named Gabriel Preda, collected from various sources, including government health agencies, international organizations, and research institutions.

DESCRIPTION:

1. The dataset provides information about the COVID-19 Vaccination progress from various countries around the world.

2. It includes data on vaccine distribution, vaccination coverage, and other related statistics.
3. The dataset may include information about the types of vaccines used, vaccination rates over time, and population demographics.

COLUMNS/ATTRIBUTES:

1.The dataset typically contains columns such as country, iso_code, date, total_vaccinations, people_vaccinated, people_fully_vaccinated, daily_vaccinations_raw, daily_vaccinations, and more.

2.These columns provide information about the total number of vaccinations, daily vaccination rates, and other vaccination-related metrics for each country.

USAGE:

1.Analyzing vaccination progress over time for different countries.
2.Identifying countries with high vaccination rates or disparities.
3.Forecasting future vaccination trends.
4.Studying the impact of different vaccines on vaccination rates.
5.Correlating vaccination progress with COVID-19 infection and mortality rates.

DATA FORMAT:

The data is usually structured as a CSV (Comma-Separated Values) file, with rows representing different countries or regions and columns representing various attributes related to vaccination progress and population.

UPDATES:

The dataset may be updated regularly to reflect the latest vaccination data, making it useful for tracking changes and trends over time.

COLUMNS:

- Country- this is the country for which the vaccination information is provided.
- Country ISO Code - ISO code for the country.
- Date - date for the data entry; for some of the dates we have only the daily vaccinations, for others, only the (cumulative) total.
- Total number of vaccinations - this is the absolute number of total immunizations in the country. Total number of people vaccinated - a person, depending on the immunization scheme, will receive one or more (typically 2)

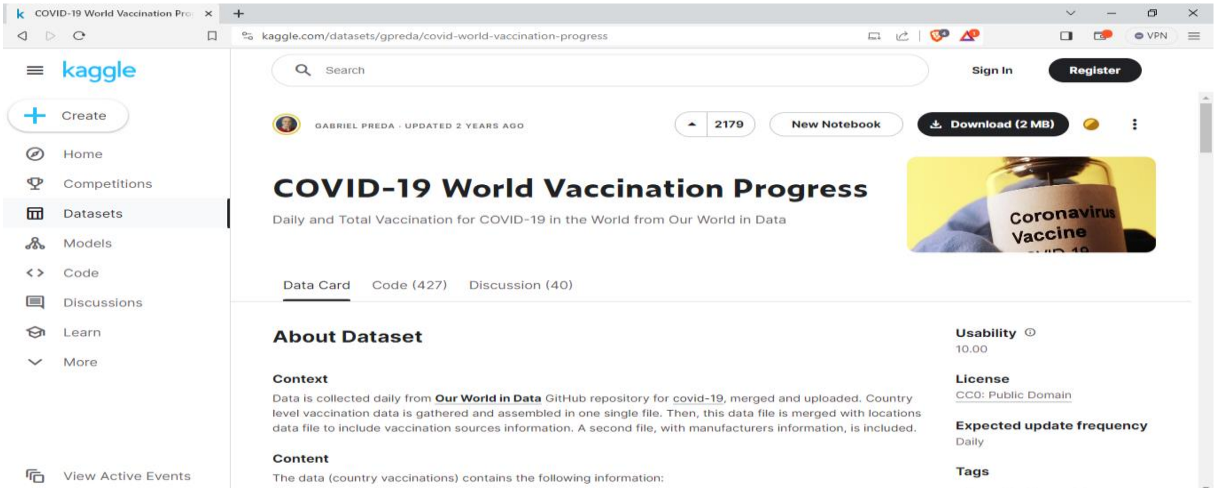
vaccines; at a certain moment, the number of vaccinations might be larger than the number of people.

- Total number of people fully vaccinated - this is the number of people that received the entire set of immunization according to the immunization scheme (typically 2); at a certain moment in time, there might be a certain number of people that received one vaccine and another number (smaller) of people that received all vaccines in the scheme.
- Daily vaccinations (raw) - for a certain data entry, the number of vaccinations for that date/country.
- Daily vaccinations - for a certain data entry, the number of vaccinations for that date/country.
- Total vaccinations per hundred - ratio (in percent) between vaccination number and total population up to the date in the country.
- Total number of people vaccinated per hour- ratio (in percent) between population immunized and total population up to the date in the country.
- Total number of people fully vaccinated per hundred - ratio (in percent) between population fully immunized and total population up to the date in the country.
- Number of vaccinations per day - number of daily vaccinations for that day and country.
- Daily vaccinations per million - ratio (in ppm) between vaccination number and total population for the current date in the country.
- Vaccines used in the country - total number of vaccines used in the country (up to date).
- Source name - source of the information (national authority, international organization, local organization etc.).
- Source website - website of the source of information.

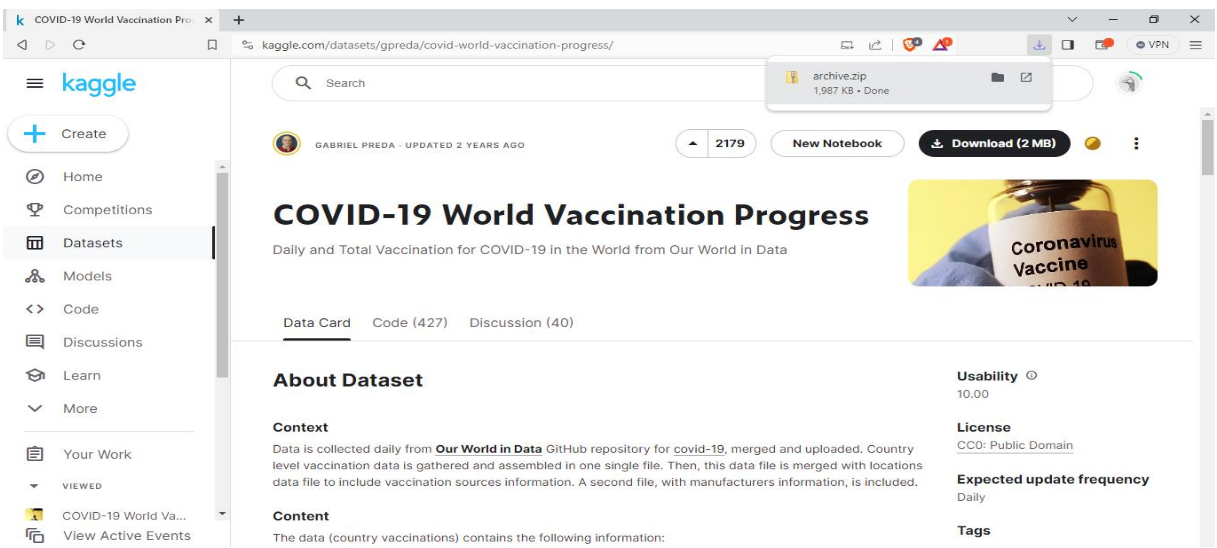
There is a second file added (country vaccinations by manufacturer), with the following columns:

- Location - country.
- Date - date.
- Vaccine - vaccine type.
- Total number of vaccinations - total number of vaccinations / current time and vaccine type.

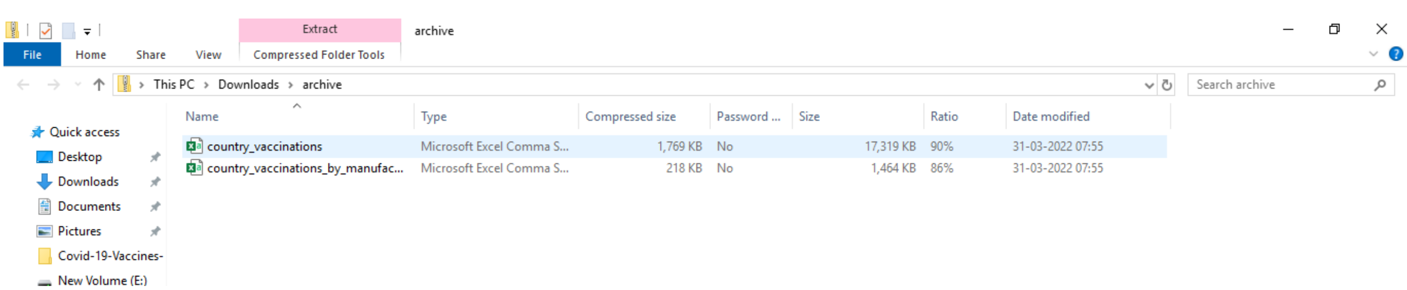
1. VISIT THE KAGGLE WEBSITE FOR COVID-19 WORLD VACCINATION POGRESS DATASET :

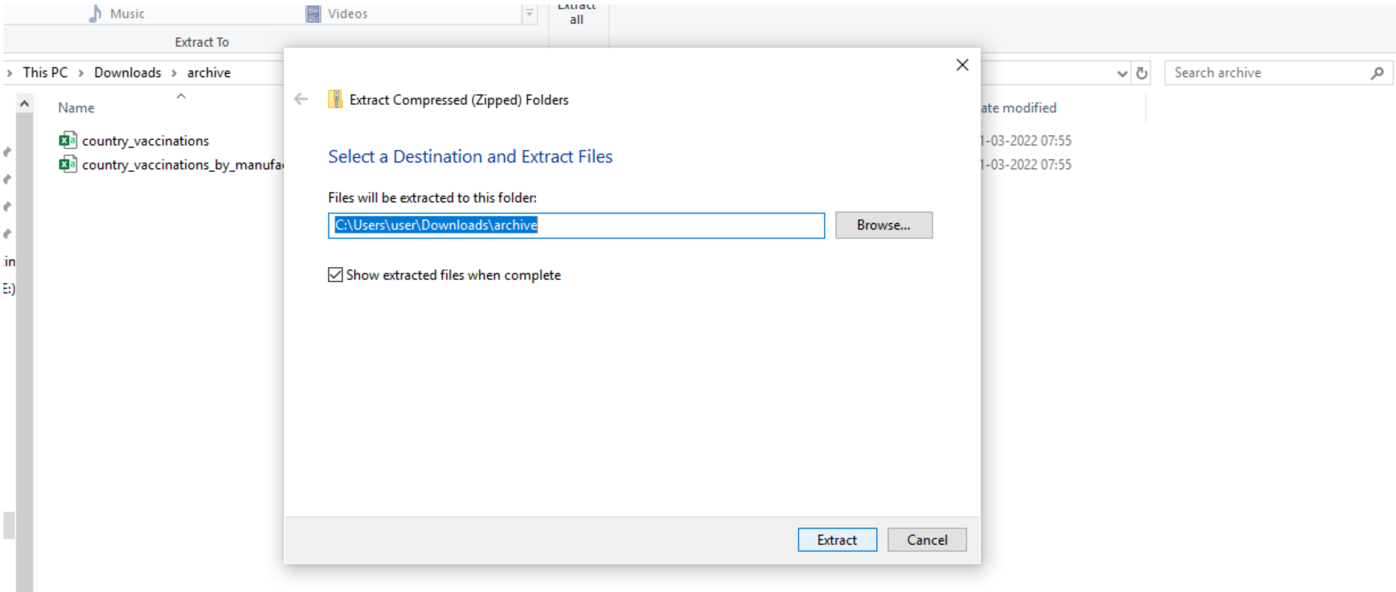


2. DOWNLOAD THE DATASET FROM THE KAGGLE WEBSITE :

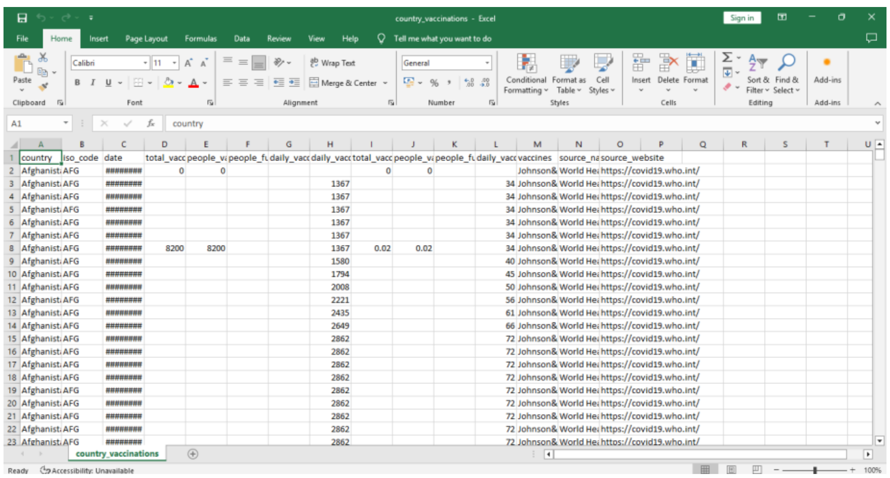


3. EXTRACT THE TWO CSV FILES FROM THE ARCHIVE ZIP FOLDER :

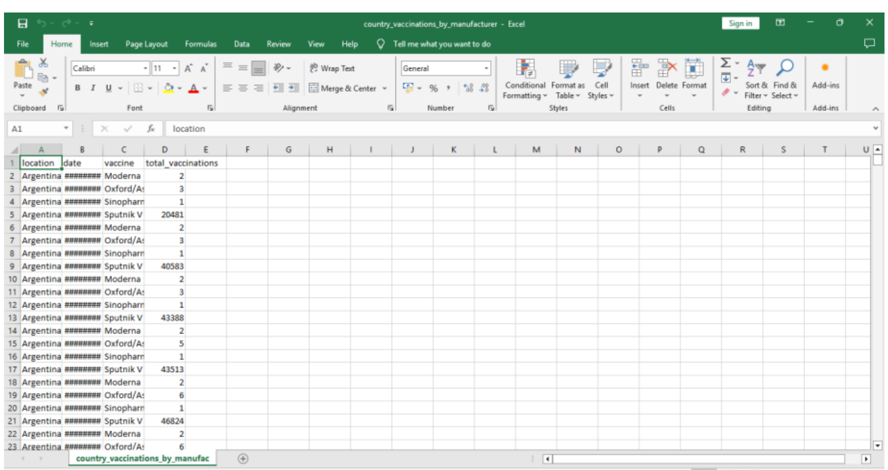




4. DATASET : country_vaccinations.csv



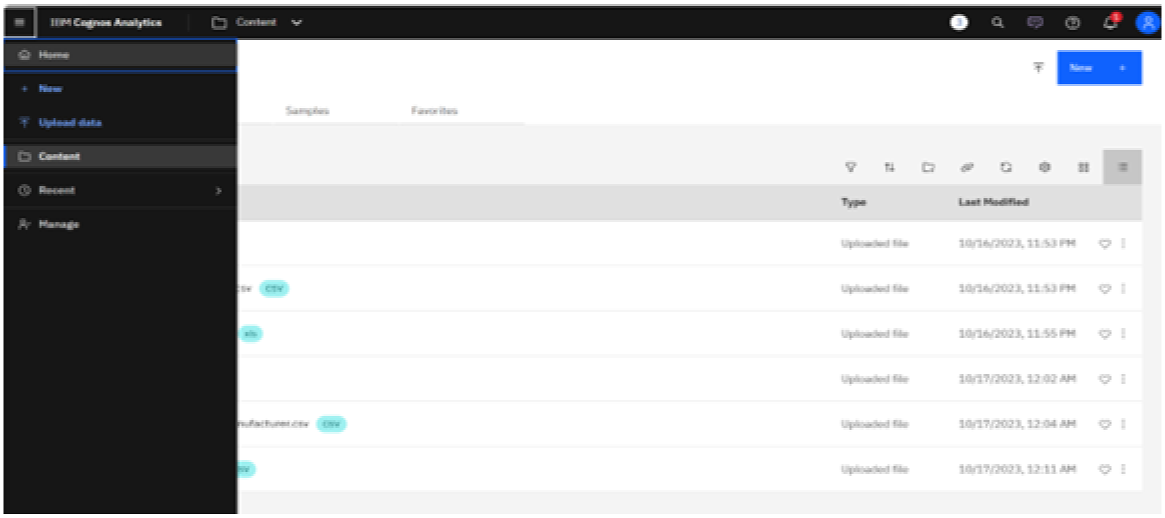
5. DATASET: country_vaccinations_by_manufacturer.csv



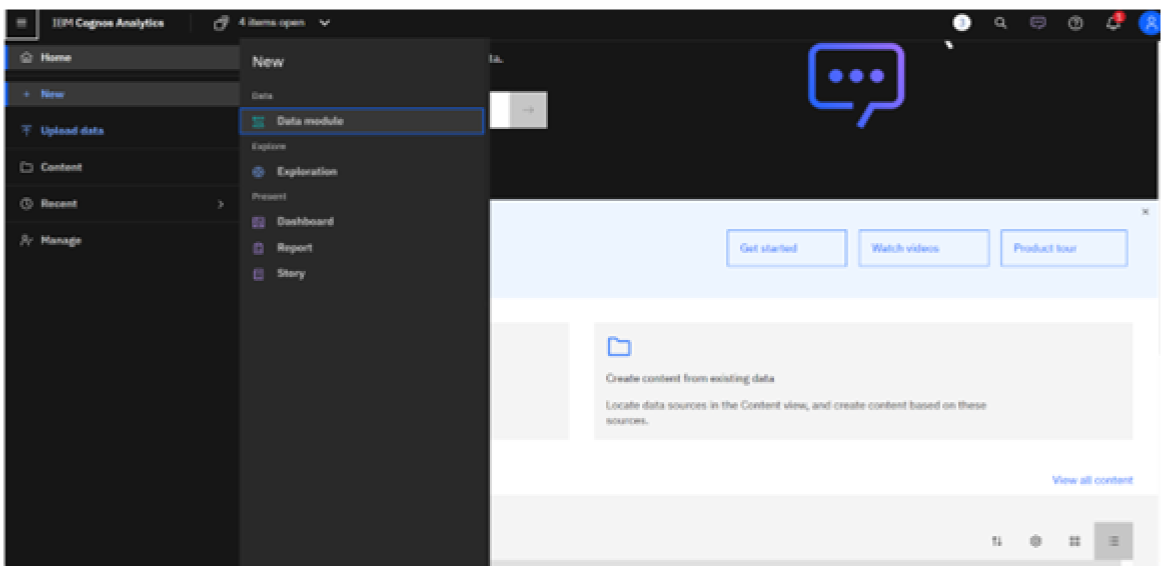
6. DATA LOADING :

Steps Involved in data loading on IBM cognos.

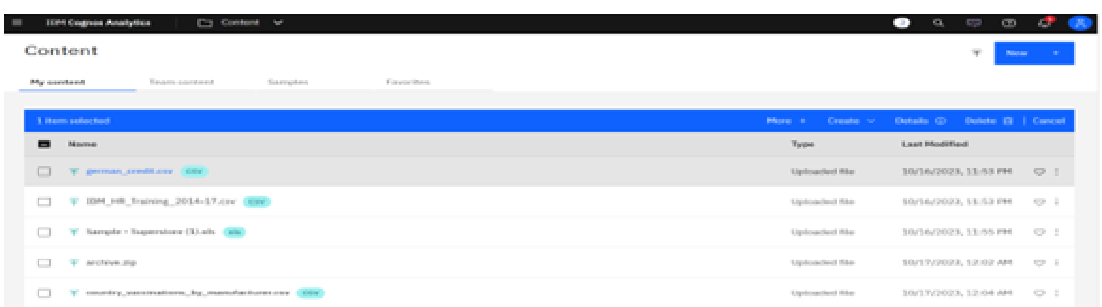
- Login to your IBM cognos.
- Click more menu from the left side.
- Select new tab.



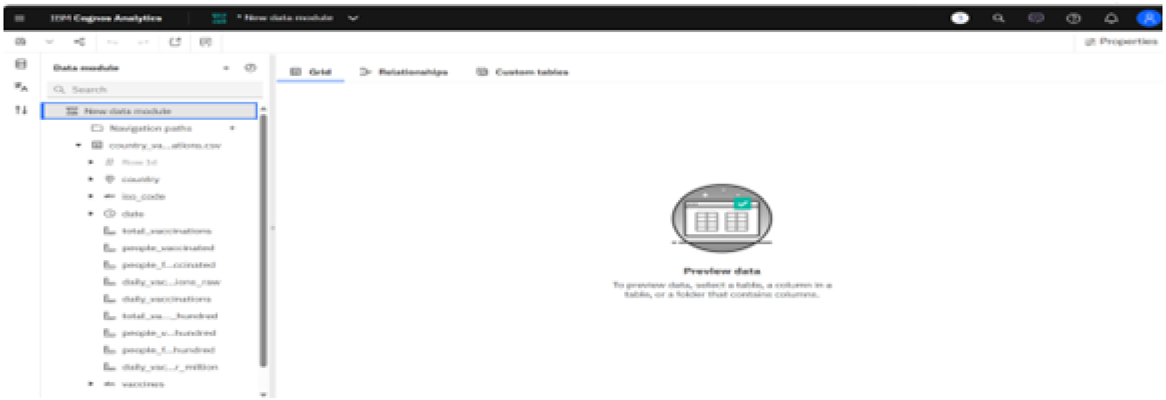
7. CLICK DATA MODULE TAB :



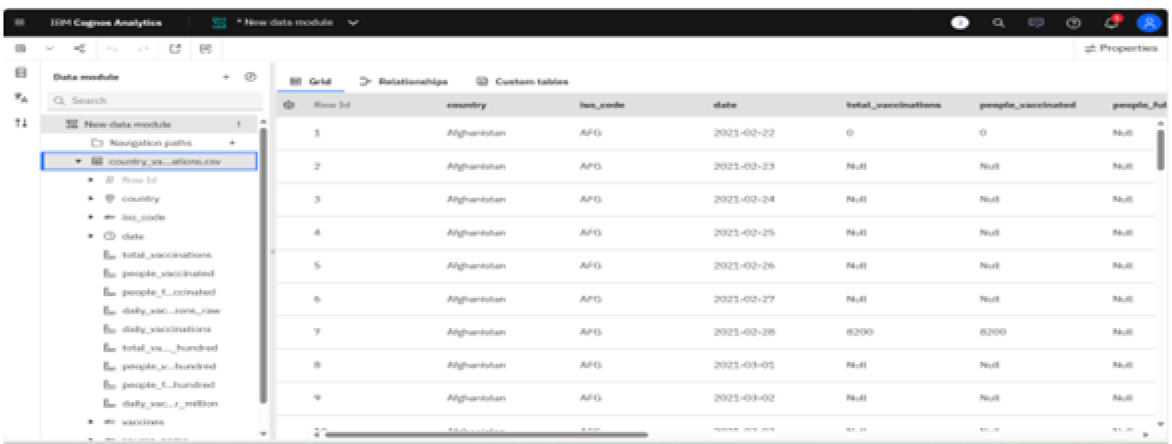
8. UPLOAD THE DATASET FOR YOUR PROJECT AND SELECT THE CORRESPONDING FILE :



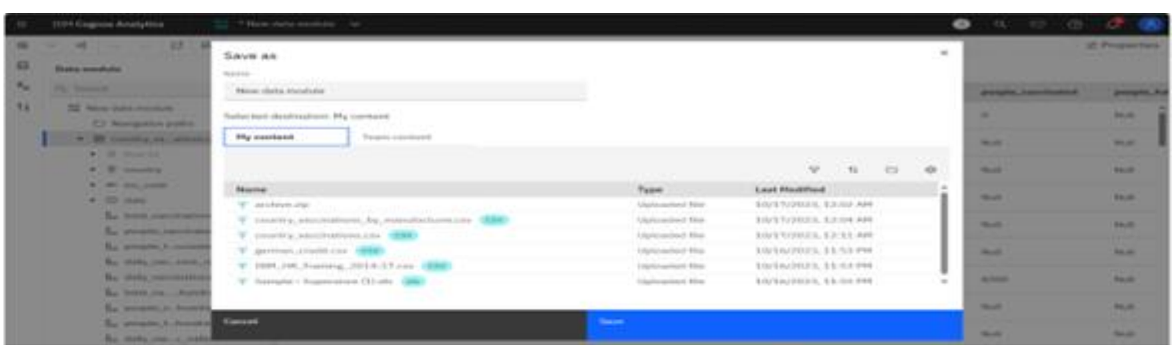
9. PREVIEW THE DATA :



10. EXPLORE THE DATA :

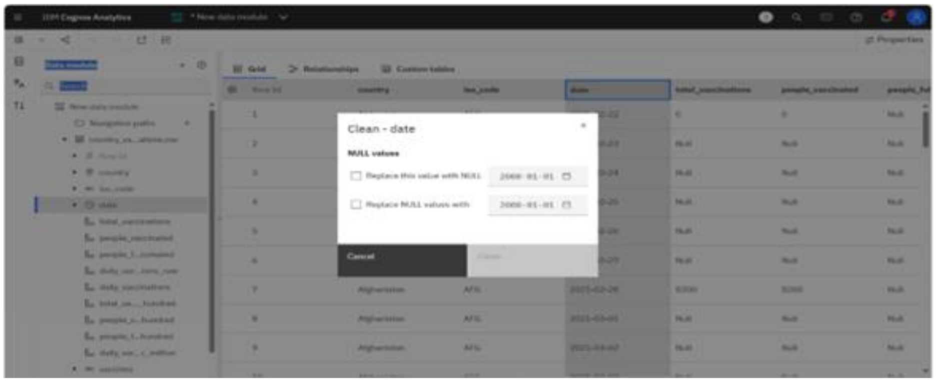


11. SAVE THE DATA MODULE :



12. DATA PREPROCESSING AND CLEANING :

- Handling missing data.
- Data Transformation.
- Data Type Conversion.
- Removing Duplicates.
- Dealing with Outliers Once you saved the data module.
- Click the corresponding dataset on IBM cognos and Preview the module Right.
- Click the row where you want to clean the data . It provides the UI to Clean the data and makes the task easy one, Now Updating and Replacing the Null values are simple.

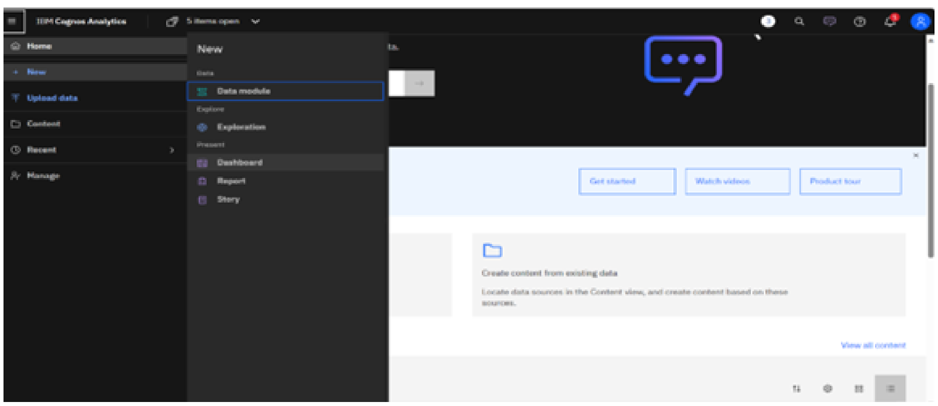


Data module will be updated by doing the above process
after the completion of process, start creating the dashboard for Visualization.

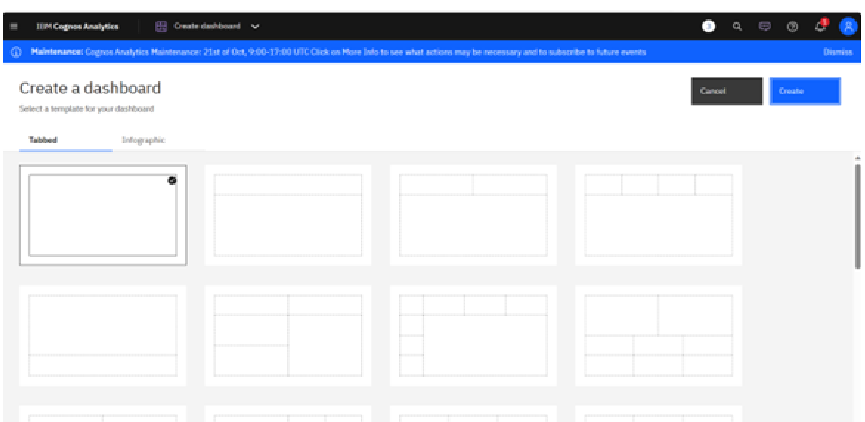
13. DASHBOARD CREATION :

Dashboard creation are helpful for visualizing the data

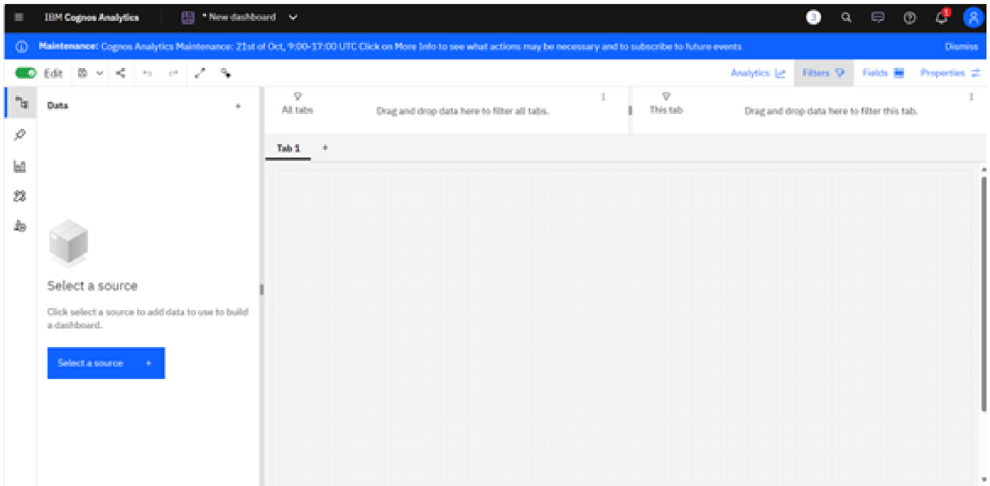
- Goto Home menu
- Select the new tab
- Click dashboard



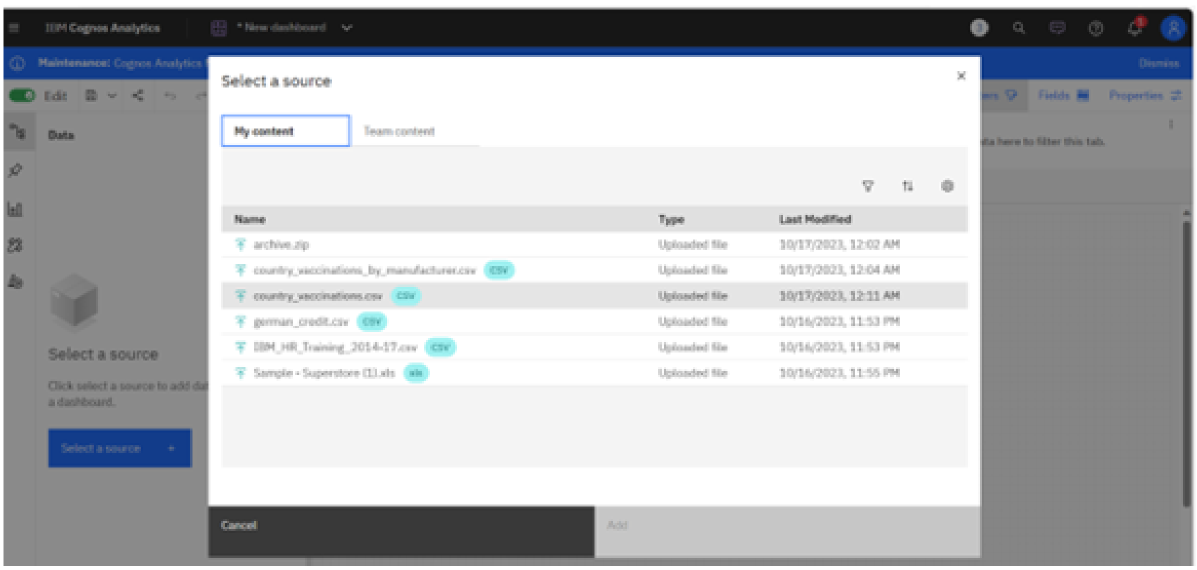
14. CHOOSE THE TEMPLATE FOR YOUR PROJECT :



15. NOW THE DASHBOARD IS CREATED :



16. SELECT THE DATA SOURCE :

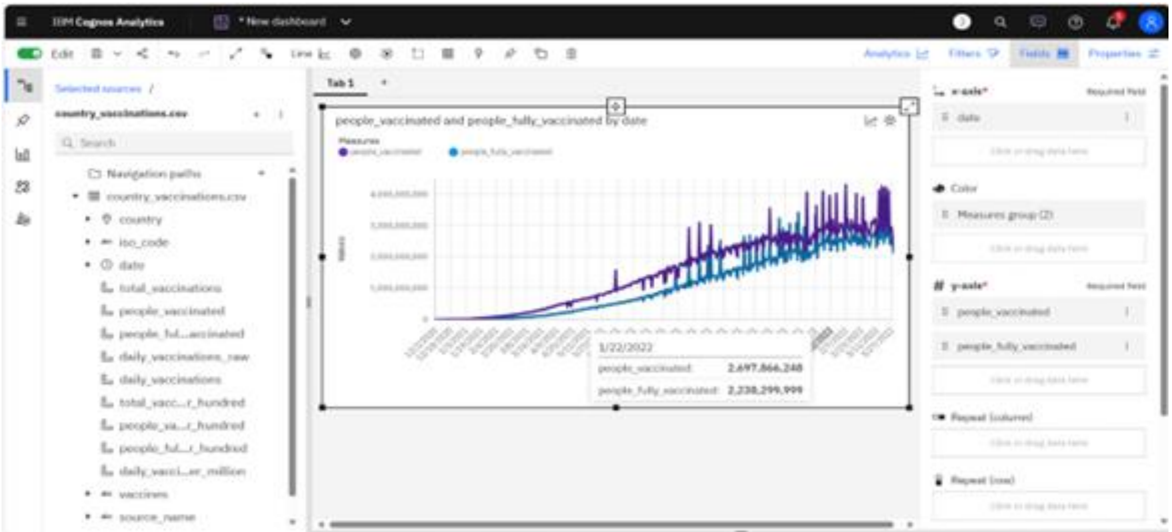


17. VISUALIZATION :

After creating the dashboard, the next step is to visualize the data in IBM Cognos

- Goes to the Corresponding Dashboard
- Select the visualizations tab in the left side of title bar

Choose the system as you want and put the data source for the required columns



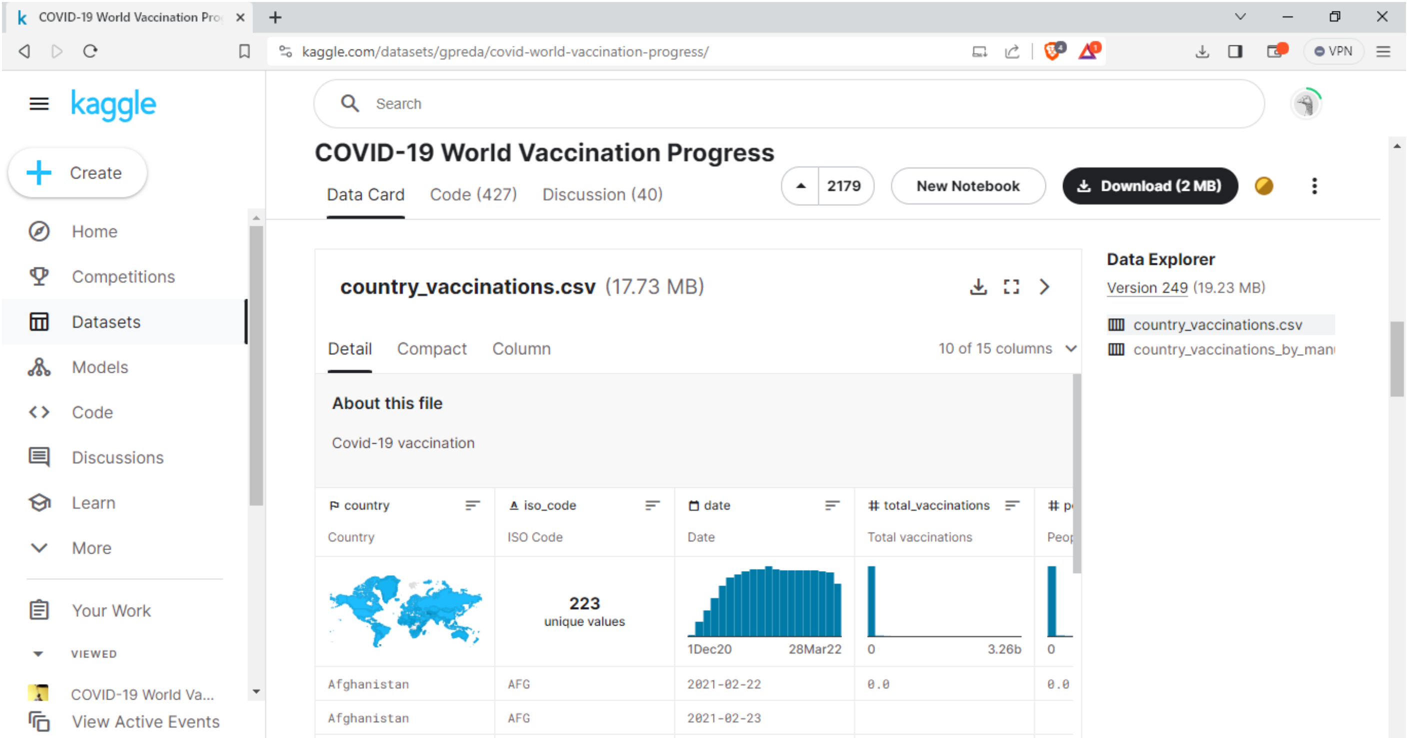
In the above screen shot displays the Line graph and model compares the "**people _vaccinated**" and "**people _fully _vaccinated**" from the time period of 2020 to 2022

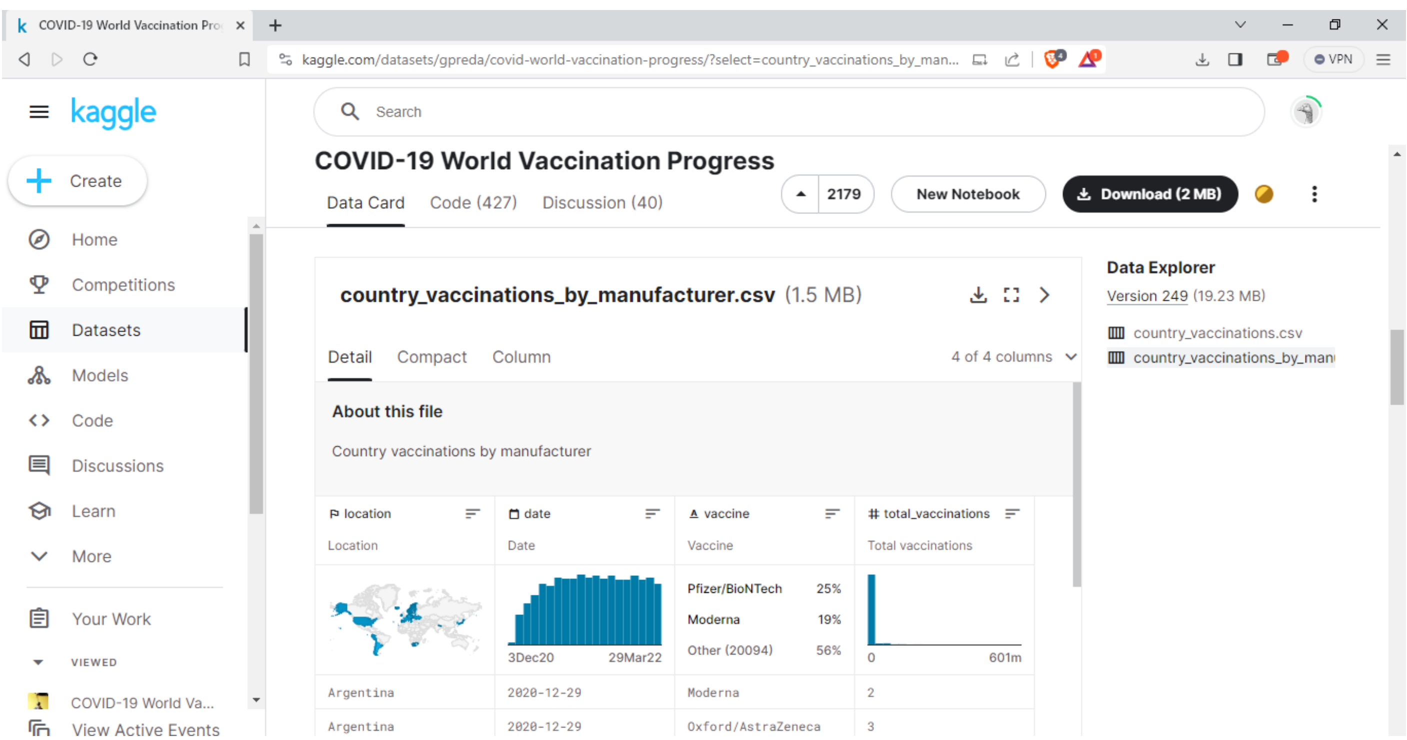
X-axis =**Dates**
Y-axis = **people _vaccinated, people _fully _vaccinated**

After performing these activities a comprehensive document will be created to demonstrate the ability to Communicate and share finding.

DATA OBSERVATION :

- The country_vaccinations_by_manufacturer.csv and country_vaccinations.csv file contains metrics with rows representing different countries or regions and columns representing various attributes related to vaccination progress and population.
- It contains Covid Vaccines Analysis Parameters such as Location-country, Date,Vaccine- vaccine type,Total number of vaccinations, iso_code, total_vaccinations, people_vaccinated, people_fully_vaccinated, daily_vaccinations_raw, daily_vaccinations, and more.


COVID-19 World Vaccination Pro
kaggle.com/datasets/gpreda/covid-world-vaccination-progress/
kaggle
Search
Create
Home
Competitions
Datasets
Models
Code
Discussions
Learn
More
Your Work
VIEWED
COVID-19 World Va...
View Active Events
COVID-19 World Vaccination Progress
Data Card
Code (427)
Discussion (40)
2179
New Notebook
Download (2 MB)
country_vaccinations.csv (17.73 MB)
Detail
Compact
Column
10 of 15 columns
About this file
Covid-19 vaccination
country
iso_code
date
total_vaccinations
Country
ISO Code
Date
Total vaccinations
223
unique values
1Dec20
28Mar22
0
3.26b
0
Afghanistan
AFG
2021-02-22
0.0
0.0
Afghanistan
AFG
2021-02-23
Data Explorer
Version 249 (19.23 MB)
country_vaccinations.csv
country_vaccinations_by_man


COVID-19 World Vaccination Pro
kaggle.com/datasets/gpreda/covid-world-vaccination-progress/?select=country_vaccinations_by_man...
kaggle
Search
Create
Home
Competitions
Datasets
Models
Code
Discussions
Learn
More
Your Work
VIEWED
COVID-19 World Va...
View Active Events
COVID-19 World Vaccination Progress
Data Card
Code (427)
Discussion (40)
2179
New Notebook
Download (2 MB)
country_vaccinations_by_manufacturer.csv (1.5 MB)
Detail
Compact
Column
4 of 4 columns
About this file
Country vaccinations by manufacturer
location
date
vaccine
total_vaccinations
Location
Date
Vaccine
Total vaccinations
3Dec20
29Mar22
Pfizer/BioNTech 25%
Moderna 19%
Other (20094) 56%
0
601m
Argentina
2020-12-29
Moderna
2
Argentina
2020-12-29
Oxford/AstraZeneca
3
Data Explorer
Version 249 (19.23 MB)
country_vaccinations.csv
country_vaccinations_by_man

IMPORTANCE OF LOADING AND PREPROCESSING DATASET :

Loading and preprocessing the dataset is an important first step in building any machine learning model. However, it is especially important for advanced machine learning models, as vaccines datasets are often complex. By loading and preprocessing the dataset, we can ensure that the machine learning algorithm is able to learn from the data effectively and accurately.

CHALLENGES INVOLVED IN LOADING AND PREPROCESSING A COVID VACCINES DATASET :

There are number of challenges involved in loading and preprocessing a Covid Vaccines dataset that we use for Covid Vaccines Analysis includes :

Handling Missing Values :

Covid Vaccines dataset often contain missing values, which can be due to a variety of factors, such as human error or incomplete data collection. Common methods for handling missing values include dropping the rows with missing values, imputing the missing values with the mean or median of the feature, or using a more sophisticated method such as multiple imputation.

Scaling The Features :

It is often helpful to scale the features before training a machine learning model. This can help to improve the performance of the model and make it more robust to outliers. There are a variety of ways to scale the features, such as min-max scaling and standard scaling.

Splitting the Dataset into Training and Testing Sets:

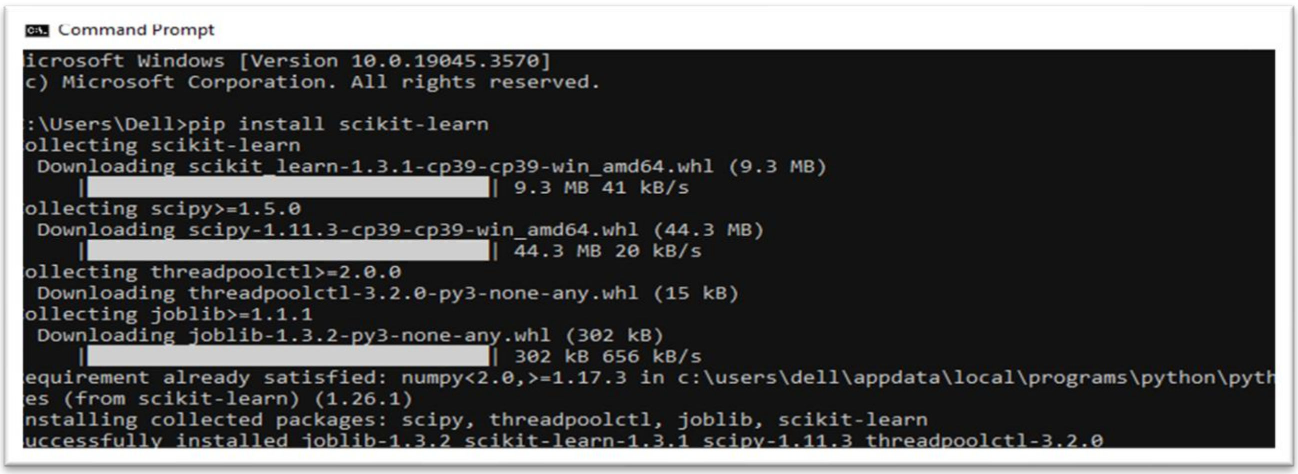
Once the data has been pre-processed, we need to split the dataset into training and testing sets. The training set will be used to train the model, and the testing set will be used to evaluate the performance of the model on unseen data. It is important to split the dataset in a way that is representative of the real world distribution of the data.

HOW TO OVERCOME THE CHALLENGES OF LOADING AND PREPROCESSING COVID VACCINES DATASET :

To overcome the challenges of loading and preprocessing a Covid Vaccines dataset, can include the following factors and features:

Use a Data Preprocessing Library:

There are a number of libraries available that can help with data preprocessing tasks, such as handling missing values, encoding categorical variables, and scaling the features.



```
Command Prompt
icrosoft Windows [Version 10.0.19045.3570]
c) Microsoft Corporation. All rights reserved.

:\Users\Dell>pip install scikit-learn
ollecting scikit-learn
 Downloading scikit_learn-1.3.1-cp39-cp39-win_amd64.whl (9.3 MB)
    |████████████████████████████████| 9.3 MB 41 kB/s
ollecting scipy>=1.5.0
 Downloading scipy-1.11.3-cp39-cp39-win_amd64.whl (44.3 MB)
    |████████████████████████████████| 44.3 MB 20 kB/s
ollecting threadpoolctl>=2.0.0
 Downloading threadpoolctl-3.2.0-py3-none-any.whl (15 kB)
ollecting joblib>=1.1.1
 Downloading joblib-1.3.2-py3-none-any.whl (302 kB)
    |████████████████████████████████| 302 kB 656 kB/s
equirement already satisfied: numpy<2.0,>=1.17.3 in c:\users\dell\appdata\local\programs\python\pyth
es (from scikit-learn) (1.26.1)
nstalling collected packages: scipy, threadpoolctl, joblib, scikit-learn
uccessfully installed joblib-1.3.2 scikit-learn-1.3.1 scipy-1.11.3 threadpoolctl-3.2.0
```

Carefully Consider the Specific needs of your Model:

The best way to preprocess the data will depend on the specific machine learning algorithm that you are using. It is important to carefully consider the requirements of the algorithm and to preprocess the data in a way that is compatible with the algorithm.

Validate the Preprocessed Data:

It is important to validate the preprocessed data to ensure that it is in a format that can be used by the machine learning algorithm and that it is of high quality. This can be done by inspecting the data visually or by using statistical methods.

LOADING THE DATASET:

- Loading the dataset using machine learning is the process of bringing the data into the machine learning environment so that it can be used to train and evaluate a model.
- The specific steps involved in loading the dataset will vary depending on the machine learning library or framework that is being used. However, there are some general steps that are common to most machine learning frameworks:

Identify the Dataset:

The first step is to identify the dataset that you want to load. This dataset may be stored in a local file, in a database, or in a cloud storage service.

- country_vaccinations.csv
- country_vaccinations_by_manufacturer.csv

Load the Dataset:

Once you have identified the dataset, you need to load it into the machine learning environment. This may involve using a built-in function in the machine learning library, or it may involve writing your own code.

```
df = pd.read_csv(“country_vaccinations.csv")
print(df)
```

Preprocess the Dataset:

Once the dataset is loaded into the machine learning environment, you may need to preprocess it before you can start training and evaluating your model. This may involve cleaning the data, transforming the data into a suitable format, and splitting the data into training and test sets.

Let’s see, How the Covid Vaccines Datset is Loaded and Accessed with the help of using the Python Jupyter Notebook.

IMPORT THE REQUIRED LIBRARIES:

To perform the data preprocessing, splitting, scaling, and other tasks as described, several libraries in Python are needed to be imported. Here are the required libraries for the code:

1. **For loading and preprocessing the dataset:**
   ```
   import pandas as pd
   import numpy as np
   import matplotlib.pyplot as plt
   import seaborn as sns
   ```

2. **For handling missing data:**
   ```
   from sklearn.impute import SimpleImputer
   ```

3. **For splitting the dataset into training and test sets:**
   ```
   from sklearn.model_selection import train_test_split
   ```

4. **For feature scaling:**
   ```
   from sklearn.preprocessing import StandardScaler
   ```

5. **To Install Python Libraries :**
   To Install the necessary libraries, run the given command in the command prompt.

Library		**command**
Pandas	-	pip install pandas
Numpy	-	pip install numpy
Matplotlib.pyplot	-	pip install matplotlib
Seaborn	-	pip install seaborn
Sklearn.model_selection	-	pip install scikit-learn
Sklearn.preprocessing	-	pip install scikit-learn

IMPORT AND LOAD THE DATASET :

Use Pandas to read the dataset file you downloaded and into a DataFrame:

Code:

```python
import pandas as pd
import numpy as np

dataset = pd.read_csv("country_vaccinations.csv")
# Creating matrix
# Create a pivot table
vaccine_matrix = dataset.pivot(index='country',columns='date',values='total_vaccinations')
# Fill missing values with 0 or any other appropriate value
vaccine_matrix = vaccine_matrix.fillna(0)
# Convert the matrix to a NumPy array
vaccine_matrix_array = vaccine_matrix.to_numpy()
# Display the matrix
print(vaccine_matrix)
```

Output:

```
date               2020-12-02  2020-12-03  2020-12-04  2020-12-05  2020-12-06  \
country
Afghanistan               0.0         0.0         0.0         0.0         0.0
Albania                   0.0         0.0         0.0         0.0         0.0
Algeria                   0.0         0.0         0.0         0.0         0.0
Andorra                   0.0         0.0         0.0         0.0         0.0
Angola                    0.0         0.0         0.0         0.0         0.0
...                       ...         ...         ...         ...         ...
Wales                     0.0         0.0         0.0         0.0         0.0
Wallis and Futuna         0.0         0.0         0.0         0.0         0.0
Yemen                     0.0         0.0         0.0         0.0         0.0
Zambia                    0.0         0.0         0.0         0.0         0.0
Zimbabwe                  0.0         0.0         0.0         0.0         0.0

date               2020-12-07  2020-12-08  2020-12-09  2020-12-10  2020-12-11  \
country
Afghanistan               0.0         0.0         0.0         0.0         0.0
Albania                   0.0         0.0         0.0         0.0         0.0
Algeria                   0.0         0.0         0.0         0.0         0.0
Andorra                   0.0         0.0         0.0         0.0         0.0
Angola                    0.0         0.0         0.0         0.0         0.0
...                       ...         ...         ...         ...         ...
Wales                     0.0         0.0         0.0         0.0         0.0
Wallis and Futuna         0.0         0.0         0.0         0.0         0.0
Yemen                     0.0         0.0         0.0         0.0         0.0
Zambia                    0.0         0.0         0.0         0.0         0.0
Zimbabwe                  0.0         0.0         0.0         0.0         0.0

date               ...  2022-03-20  2022-03-21  2022-03-22  2022-03-23  \
country            ...
Afghanistan        ...         0.0         0.0   5751015.0         0.0
Albania            ...         0.0         0.0         0.0         0.0
Algeria            ...         0.0         0.0         0.0         0.0
Andorra            ...         0.0         0.0         0.0         0.0
Angola             ...         0.0         0.0         0.0         0.0
...                ...         ...         ...         ...         ...
Wales              ...   6914650.0   6916175.0   6917707.0   6919439.0
Wallis and Futuna  ...         0.0     12940.0         0.0         0.0
Yemen              ...         0.0         0.0         0.0         0.0
Zambia             ...         0.0   3288541.0         0.0   3325582.0
Zimbabwe           ...   8210637.0   8230061.0   8313471.0   8414477.0
```

```
date                2022-03-24  2022-03-25  2022-03-26  2022-03-27  2022-03-28  \
country
Afghanistan                0.0         0.0         0.0         0.0         0.0
Albania              2754244.0         0.0         0.0         0.0         0.0
Algeria                    0.0         0.0         0.0         0.0         0.0
Andorra                    0.0         0.0         0.0         0.0         0.0
Angola                     0.0  17535411.0         0.0         0.0         0.0
...                        ...         ...         ...         ...         ...
Wales                6921195.0   6923298.0   6923706.0   6925183.0   6927437.0
Wallis and Futuna          0.0         0.0         0.0         0.0     13073.0
Yemen                      0.0         0.0         0.0         0.0         0.0
Zambia               3345769.0         0.0         0.0         0.0   3390539.0
Zimbabwe             8552429.0   8691642.0   8791728.0   8845039.0   8934360.0

date                2022-03-29
country
Afghanistan                0.0
Albania                    0.0
Algeria                    0.0
Andorra                    0.0
Angola                     0.0
...                        ...
Wales                      0.0
Wallis and Futuna          0.0
Yemen                      0.0
Zambia               3402612.0
Zimbabwe             9039729.0

[223 rows x 483 columns]
```

In this code, we first pivot the DataFrame to transform it into a matrix where rows represent countries, columns represent dates, and the values are the total vaccination counts. We fill any missing values with 0.

In [6]:
```
#import the required libraries
#import the required dataset
#view the dataset

import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
%matplotlib inline
df=pd.read_csv('Documents/country_vaccinations.csv')
df.head()
```

Out[6]:

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_va
0	Afghanistan	AFG	2021-02-22	0.0	0.0	NaN	
1	Afghanistan	AFG	2021-02-23	NaN	NaN	NaN	
2	Afghanistan	AFG	2021-02-24	NaN	NaN	NaN	
3	Afghanistan	AFG	2021-02-25	NaN	NaN	NaN	
4	Afghanistan	AFG	2021-02-26	NaN	NaN	NaN	

<u>PREPROCESSING THE DATASET</u> :

- Data preprocessing is the process of
    1. Cleaning
    2. Transforming
    3. Integrating Data

  in order to make it ready for analysis.

- This may involve removing errors and inconsistencies, handling missing values, transforming the data into a consistent format, and scaling the data to a suitable range.

Some common data preprocessing tasks include:
<u>Data Cleaning:</u> This involves identifying and correcting errors and inconsistencies in the data. For example, this may involve removing duplicate records, correcting typos, and filling in missing values.

<u>Data Transformation</u>: This involves converting the data into a format that is suitable for the analysis task. For example, this may involve converting categorical data to numerical data, or scaling the data to a suitable range.

<u>Feature Engineering</u>: This involves creating new features from the existing data. For example, this may involve creating features that represent interactions between variables, or features that represent summary statistics of the data

<u>Data Integration</u>: This involves combining data from multiple sources into a single dataset. This may involve resolving inconsistencies in the data, such as different data formats or different variable names.

Data preprocessing is an essential step in many data science projects. By carefully preprocessing the data, data scientists can improve the accuracy and reliability.

[3]:
```python
#DATA PREPROCESSING:

#1.Handling Missing Values:

# Check for missing values
df.isnull().sum()
```

[3]:
```
country                                0
iso_code                               0
date                                   0
total_vaccinations                 42905
people_vaccinated                  45218
```

```
people_fully_vaccinated            47710
daily_vaccinations_raw             51150
daily_vaccinations                   299
total_vaccinations_per_hundred     42905
people_vaccinated_per_hundred      45218
people_fully_vaccinated_per_hundred 47710
daily_vaccinations_per_million       299
vaccines                               0
source_name                            0
source_website                         0
dtype: int64
```

[4]:
```python
# Fill missing values with appropriate values (e.g., mean, median, or a␣
↪specific value)
df.fillna({'total_vaccinations': 0,
           'people_vaccinated': 0,
           'people_fully_vaccinated':0,
           'daily_vaccinations_raw':0,
           'daily_vaccinations':0,
           'total_vaccinations_per_hundred': 0,
           'people_vaccinated_per_hundred': 0,
           'people_fully_vaccinated_per_hundred':0,
           'daily_vaccinations_per_million':0}, inplace=True)

df.isnull().sum()
```

[4]:
```
country                                0
iso_code                               0
date                                   0
total_vaccinations                     0
people_vaccinated                      0
people_fully_vaccinated                0
daily_vaccinations_raw                 0
daily_vaccinations                     0
total_vaccinations_per_hundred         0
people_vaccinated_per_hundred          0
people_fully_vaccinated_per_hundred    0
daily_vaccinations_per_million         0
vaccines                               0
source_name                            0
source_website                         0
dtype: int64
```

[5]:
```
#2.Data Type Conversion:

df['date'] = pd.to_datetime(df['date'])
```

```
df
```

[5]:
```
           country iso_code       date  total_vaccinations  people_vaccinated  \
0      Afghanistan      AFG 2021-02-22                 0.0                0.0
1      Afghanistan      AFG 2021-02-23                 0.0                0.0
2      Afghanistan      AFG 2021-02-24                 0.0                0.0
3      Afghanistan      AFG 2021-02-25                 0.0                0.0
4      Afghanistan      AFG 2021-02-26                 0.0                0.0
...            ...      ...        ...                 ...                ...
86507     Zimbabwe      ZWE 2022-03-25           8691642.0          4814582.0
86508     Zimbabwe      ZWE 2022-03-26           8791728.0          4886242.0
86509     Zimbabwe      ZWE 2022-03-27           8845039.0          4918147.0
86510     Zimbabwe      ZWE 2022-03-28           8934360.0          4975433.0
86511     Zimbabwe      ZWE 2022-03-29           9039729.0          5053114.0

       people_fully_vaccinated  daily_vaccinations_raw  daily_vaccinations  \
0                          0.0                     0.0                 0.0
1                          0.0                     0.0              1367.0
2                          0.0                     0.0              1367.0
3                          0.0                     0.0              1367.0
4                          0.0                     0.0              1367.0
...                        ...                     ...                 ...
86507                3473523.0                139213.0             69579.0
86508                3487962.0                100086.0             83429.0
86509                3493763.0                 53311.0             90629.0
86510                3501493.0                 89321.0            100614.0
86511                3510256.0                105369.0            103751.0

       total_vaccinations_per_hundred  people_vaccinated_per_hundred  \
0                                0.00                           0.00
1                                0.00                           0.00
2                                0.00                           0.00
3                                0.00                           0.00
4                                0.00                           0.00
...                               ...                            ...
86507                           57.59                          31.90
86508                           58.25                          32.38
86509                           58.61                          32.59
86510                           59.20                          32.97
86511                           59.90                          33.48

       people_fully_vaccinated_per_hundred  daily_vaccinations_per_million  \
0                                     0.00                             0.0
1                                     0.00                            34.0
2                                     0.00                            34.0
3                                     0.00                            34.0
4                                     0.00                            34.0
```

```
...                              ...                    ...
86507                          23.02                 4610.0
86508                          23.11                 5528.0
86509                          23.15                 6005.0
86510                          23.20                 6667.0
86511                          23.26                 6874.0

                                                vaccines  \
0      Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
1      Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
2      Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
3      Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
4      Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
...                                                  ...
86507  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac...
86508  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac...
86509  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac...
86510  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac...
86511  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac...

                     source_name  \
0      World Health Organization
1      World Health Organization
2      World Health Organization
3      World Health Organization
4      World Health Organization
...                          ...
86507         Ministry of Health
86508         Ministry of Health
86509         Ministry of Health
86510         Ministry of Health
86511         Ministry of Health

                                          source_website
0                                https://covid19.who.int/
1                                https://covid19.who.int/
2                                https://covid19.who.int/
3                                https://covid19.who.int/
4                                https://covid19.who.int/
...                                                  ...
86507  https://www.arcgis.com/home/webmap/viewer.html...
86508  https://www.arcgis.com/home/webmap/viewer.html...
86509  https://www.arcgis.com/home/webmap/viewer.html...
86510  https://www.arcgis.com/home/webmap/viewer.html...
86511  https://www.arcgis.com/home/webmap/viewer.html...

[86512 rows x 15 columns]
```

HANDLING THE MISSING DATA :

Scikit-learn library provides the SimpleImputer class, which is a handy tool for handling missing data.

Code:

```
imputer = SimpleImputer(strategy='mean')
# Fit and transform the imputer on your matrix
vaccine_matrix_imputed = imputer.fit_transform(vaccine_matrix)
# Convert the imputed array back to a Pandas DataFrame
vaccine_matrix_imputed_df = pd.DataFrame(vaccine_matrix_imputed, columns=vaccine_matrix.columns, index=vaccine_matrix.index)
# Display the matrix with missing values handled
print(vaccine_matrix_imputed_df)
```

Output:

```
date              2020-12-02  2020-12-03  2020-12-04  2020-12-05  2020-12-06  \
country
Afghanistan              0.0         0.0         0.0         0.0         0.0
Albania                  0.0         0.0         0.0         0.0         0.0
Algeria                  0.0         0.0         0.0         0.0         0.0
Andorra                  0.0         0.0         0.0         0.0         0.0
Angola                   0.0         0.0         0.0         0.0         0.0
...                      ...         ...         ...         ...         ...
Wales                    0.0         0.0         0.0         0.0         0.0
Wallis and Futuna        0.0         0.0         0.0         0.0         0.0
Yemen                    0.0         0.0         0.0         0.0         0.0
Zambia                   0.0         0.0         0.0         0.0         0.0
Zimbabwe                 0.0         0.0         0.0         0.0         0.0

date              2020-12-07  2020-12-08  2020-12-09  2020-12-10  2020-12-11  \
```

```
country
Afghanistan          0.0        0.0        0.0        0.0        0.0
Albania              0.0        0.0        0.0        0.0        0.0
Algeria              0.0        0.0        0.0        0.0        0.0
Andorra              0.0        0.0        0.0        0.0        0.0
Angola               0.0        0.0        0.0        0.0        0.0
...                  ...        ...        ...        ...        ...
Wales                0.0        0.0        0.0        0.0        0.0
Wallis and Futuna    0.0        0.0        0.0        0.0        0.0
Yemen                0.0        0.0        0.0        0.0        0.0
Zambia               0.0        0.0        0.0        0.0        0.0
Zimbabwe             0.0        0.0        0.0        0.0        0.0

date               ...  2022-03-20  2022-03-21  2022-03-22  2022-03-23  \
country            ...
Afghanistan        ...         0.0         0.0   5751015.0         0.0
Albania            ...         0.0         0.0         0.0         0.0
Algeria            ...         0.0         0.0         0.0         0.0
Andorra            ...         0.0         0.0         0.0         0.0
Angola             ...         0.0         0.0         0.0         0.0
...                ...         ...         ...         ...         ...
Wales              ...   6914650.0   6916175.0   6917707.0   6919439.0
Wallis and Futuna  ...         0.0     12940.0         0.0         0.0
Yemen              ...         0.0         0.0         0.0         0.0
Zambia             ...         0.0   3288541.0         0.0   3325582.0
Zimbabwe           ...   8210637.0   8230061.0   8313471.0   8414477.0

date               2022-03-24  2022-03-25  2022-03-26  2022-03-27  2022-03-28  \
country
```

```
Afghanistan           0.0        0.0        0.0        0.0        0.0
Albania         2754244.0        0.0        0.0        0.0        0.0
Algeria               0.0        0.0        0.0        0.0        0.0
Andorra               0.0        0.0        0.0        0.0        0.0
Angola                0.0  17535411.0       0.0        0.0        0.0
...                   ...        ...        ...        ...        ...
Wales           6921195.0  6923298.0  6923706.0  6925183.0  6927437.0
Wallis and Futuna     0.0        0.0        0.0        0.0    13073.0
Yemen                 0.0        0.0        0.0        0.0        0.0
Zambia          3345769.0        0.0        0.0        0.0  3390539.0
Zimbabwe        8552429.0  8691642.0  8791728.0  8845039.0  8934360.0

date            2022-03-29
country
Afghanistan           0.0
Albania               0.0
Algeria               0.0
Andorra               0.0
Angola                0.0
 ...                  ...
Wales                 0.0
Wallis and Futuna     0.0
Yemen                 0.0
Zambia          3402612.0
Zimbabwe        9039729.0
[223 rows x 483 columns]
```

The SimpleImputer is used to replace missing values in the vaccine_matrix with the mean of the non-missing values.

[6]:
```
#3.Handling Duplicates:

df.drop_duplicates(inplace=True)

df
```

[6]:
```
           country iso_code        date  total_vaccinations  people_vaccinated  \
0      Afghanistan      AFG  2021-02-22                 0.0                0.0
1      Afghanistan      AFG  2021-02-23                 0.0                0.0
2      Afghanistan      AFG  2021-02-24                 0.0                0.0
3      Afghanistan      AFG  2021-02-25                 0.0                0.0
4      Afghanistan      AFG  2021-02-26                 0.0                0.0
…              …      …         …                 …                …
86507     Zimbabwe      ZWE  2022-03-25           8691642.0          4814582.0
86508     Zimbabwe      ZWE  2022-03-26           8791728.0          4886242.0
86509     Zimbabwe      ZWE  2022-03-27           8845039.0          4918147.0
86510     Zimbabwe      ZWE  2022-03-28           8934360.0          4975433.0
86511     Zimbabwe      ZWE  2022-03-29           9039729.0          5053114.0

       people_fully_vaccinated  daily_vaccinations_raw  daily_vaccinations  \
0                          0.0                     0.0                 0.0
1                          0.0                     0.0              1367.0
2                          0.0                     0.0              1367.0
3                          0.0                     0.0              1367.0
4                          0.0                     0.0              1367.0
…                          …                     …                 …
86507                3473523.0                139213.0             69579.0
86508                3487962.0                100086.0             83429.0
86509                3493763.0                 53311.0             90629.0
86510                3501493.0                 89321.0            100614.0
86511                3510256.0                105369.0            103751.0

       total_vaccinations_per_hundred  people_vaccinated_per_hundred  \
0                                0.00                           0.00
1                                0.00                           0.00
2                                0.00                           0.00
3                                0.00                           0.00
4                                0.00                           0.00
…                                …                           …
86507                           57.59                          31.90
86508                           58.25                          32.38
86509                           58.61                          32.59
86510                           59.20                          32.97
86511                           59.90                          33.48

       people_fully_vaccinated_per_hundred  daily_vaccinations_per_million  \
0                                     0.00                             0.0


1                                     0.00                            34.0
2                                     0.00                            34.0
3                                     0.00                            34.0
4                                     0.00                            34.0
…                                     …                            …
86507                                23.02                          4610.0
86508                                23.11                          5528.0
86509                                23.15                          6005.0
86510                                23.20                          6667.0
86511                                23.26                          6874.0

                                                vaccines  \
0      Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi…
1      Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi…
2      Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi…
3      Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi…
4      Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi…
…                                                   …
86507  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac…
86508  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac…
86509  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac…
86510  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac…
86511  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac…

                     source_name  \
0      World Health Organization
1      World Health Organization
2      World Health Organization
3      World Health Organization
4      World Health Organization
…                            …
86507          Ministry of Health
86508          Ministry of Health
86509          Ministry of Health
86510          Ministry of Health
86511          Ministry of Health

                                          source_website
0                                  https://covid19.who.int/
1                                  https://covid19.who.int/
2                                  https://covid19.who.int/
3                                  https://covid19.who.int/
4                                  https://covid19.who.int/
…                                                   …
86507  https://www.arcgis.com/home/webmap/viewer.html…
86508  https://www.arcgis.com/home/webmap/viewer.html…
86509  https://www.arcgis.com/home/webmap/viewer.html…


86510  https://www.arcgis.com/home/webmap/viewer.html…
86511  https://www.arcgis.com/home/webmap/viewer.html…

[86512 rows x 15 columns]
```

In [7]:
```
#drop the null values in the datsets using drop()
df1=df.dropna()
print(df1)
```

```
           country iso_code        date  total_vaccinations  \
94     Afghanistan      AFG  2021-05-27            593313.0
101    Afghanistan      AFG  2021-06-03            630305.0
339    Afghanistan      AFG  2022-01-27           5081064.0
433        Albania      ALB  2021-02-18              3049.0
515        Albania      ALB  2021-05-11            622507.0
...            ...      ...         ...                 ...
86507     Zimbabwe      ZWE  2022-03-25           8691642.0
86508     Zimbabwe      ZWE  2022-03-26           8791728.0
86509     Zimbabwe      ZWE  2022-03-27           8845039.0
86510     Zimbabwe      ZWE  2022-03-28           8934360.0
86511     Zimbabwe      ZWE  2022-03-29           9039729.0

       people_vaccinated  people_fully_vaccinated  daily_vaccinations_raw  \
94              479574.0                 113739.0                  2859.0
101             481800.0                 148505.0                  4015.0
339            4517380.0                3868832.0                  6868.0
433               2438.0                    611.0                  1348.0
515             440921.0                 181586.0                  9548.0
...                  ...                      ...                     ...
86507          4814582.0                3473523.0                139213.0
86508          4886242.0                3487962.0                100086.0
86509          4918147.0                3493763.0                 53311.0
86510          4975433.0                3501493.0                 89321.0
86511          5053114.0                3510256.0                105369.0

       daily_vaccinations  total_vaccinations_per_hundred  \
94                 6487.0                            1.49
101                5285.0                            1.58
339                9802.0                           12.76
433                 254.0                            0.11
515               12160.0                           21.67
...                   ...                             ...
86507             69579.0                           57.59
86508             83429.0                           58.25
86509             90629.0                           58.61
86510            100614.0                           59.20
86511            103751.0                           59.90

       people_vaccinated_per_hundred  people_fully_vaccinated_per_hundred  \
94                              1.20                                 0.29
101                             1.21                                 0.37
339                            11.34                                 9.71
433                             0.08                                 0.02
515                            15.35                                 6.32
...                              ...                                  ...

...                              ...                                  ...
86507                          31.90                                23.02
86508                          32.38                                23.11
86509                          32.59                                23.15
86510                          32.97                                23.20
86511                          33.48                                23.26

       daily_vaccinations_per_million  \
94                              163.0
101                             133.0
339                             246.0
433                              88.0
515                            4233.0
...                               ...
86507                          4610.0
```

```
86508                        5528.0
86509                        6005.0
86510                        6667.0
86511                        6874.0

                                           vaccines  \
94     Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
101    Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
339    Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
433    Oxford/AstraZeneca, Pfizer/BioNTech, Sinovac, ...
515    Oxford/AstraZeneca, Pfizer/BioNTech, Sinovac, ...
...                                                ...
86507  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac...
86508  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac...
86509  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac...
86510  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac...
86511  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac...

                     source_name  \
94     World Health Organization
101    World Health Organization
339    World Health Organization
433           Ministry of Health
515           Ministry of Health
...                          ...
86507         Ministry of Health
86508         Ministry of Health
86509         Ministry of Health
86510         Ministry of Health
86511         Ministry of Health

                                          source_website
94                            https://covid19.who.int/
101                           https://covid19.who.int/
339                           https://covid19.who.int/
433    https://shendetesia.gov.al/vaksinimi-anticovid...
515    https://shendetesia.gov.al/vaksinimi-anticovid...
...                                                ...
86507  https://www.arcgis.com/home/webmap/viewer.html...
86508  https://www.arcgis.com/home/webmap/viewer.html...
86509  https://www.arcgis.com/home/webmap/viewer.html...
86510  https://www.arcgis.com/home/webmap/viewer.html...
86511  https://www.arcgis.com/home/webmap/viewer.html...

[30847 rows x 15 columns]
```

In [8]:
```
#view the information of the dataset
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 30847 entries, 94 to 86511
Data columns (total 15 columns):
 #   Column                               Non-Null Count  Dtype
---  ------                               --------------  -----
 0   country                              30847 non-null  object
 1   iso_code                             30847 non-null  object
 2   date                                 30847 non-null  object
 3   total_vaccinations                   30847 non-null  float64
 4   people_vaccinated                    30847 non-null  float64
 5   people_fully_vaccinated              30847 non-null  float64
 6   daily_vaccinations_raw               30847 non-null  float64
 7   daily_vaccinations                   30847 non-null  float64
 8   total_vaccinations_per_hundred       30847 non-null  float64
 9   people_vaccinated_per_hundred        30847 non-null  float64
 10  people_fully_vaccinated_per_hundred  30847 non-null  float64
 11  daily_vaccinations_per_million       30847 non-null  float64
 12  vaccines                             30847 non-null  object
 13  source_name                          30847 non-null  object
 14  source_website                       30847 non-null  object
dtypes: float64(9), object(6)
memory usage: 3.8+ MB
```

In [9]:
```
#view the statistical analysis the dataset
df1.describe()
```

Out[9]:

	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vacc
count	3.084700e+04	3.084700e+04	3.084700e+04	3.084700e+04	3.084
mean	3.980375e+07	2.177533e+07	1.579596e+07	2.021875e+05	1.975
std	1.451667e+08	8.053173e+07	5.898165e+07	7.041931e+05	6.400
min	3.000000e+00	3.000000e+00	1.000000e+00	0.000000e+00	0.000
25%	1.153332e+06	7.339795e+05	3.704450e+05	5.498000e+03	7.329
50%	6.335305e+06	3.688092e+06	2.211035e+06	2.908100e+04	3.247
75%	2.520629e+07	1.440668e+07	9.121526e+06	1.344580e+05	1.402
max	3.243599e+09	1.275541e+09	1.240777e+09	1.862727e+07	1.307

In [10]:
```
#view the columns count
df.isnull().sum()
```

Out[10]:
```
country                                  0
iso_code                                 0
date                                     0
total_vaccinations                   42905
people_vaccinated                    45218
people_fully_vaccinated              47710
daily_vaccinations_raw               51150
daily_vaccinations                     299
total_vaccinations_per_hundred       42905
people_vaccinated_per_hundred        45218
people_fully_vaccinated_per_hundred  47710
daily_vaccinations_per_million         299
vaccines                                 0
source_name                              0
source_website                           0
dtype: int64
```

In [11]:
```
#view the columns in the dataset
df.columns
```

Out[11]:
```
Index(['country', 'iso_code', 'date', 'total_vaccinations',
       'people_vaccinated', 'people_fully_vaccinated',
       'daily_vaccinations_raw', 'daily_vaccinations',
       'total_vaccinations_per_hundred', 'people_vaccinated_per_hundred',
       'people_fully_vaccinated_per_hundred', 'daily_vaccinations_per_million',
       'vaccines', 'source_name', 'source_website'],
      dtype='object')
```

In [12]:
```
#convert the float column into integer column

df1['people_vaccinated'] = df1['people_vaccinated'].astype(int)

df1['people_fully_vaccinated'] = df1['people_fully_vaccinated'].astype(int)

df1['daily_vaccinations_raw'] = df1['daily_vaccinations_raw'].astype(int)

df1['total_vaccinations_per_hundred'] = df1['total_vaccinations_per_hundred'].astype(i

df1['people_vaccinated_per_hundred'] = df1['people_vaccinated_per_hundred'].astype(int

df1['people_fully_vaccinated_per_hundred'] = df1['people_fully_vaccinated_per_hundred'

df1['daily_vaccinations_per_million'] = df1['daily_vaccinations_per_million'].astype(i

df1.head()
```

Out[12]:

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_
94	Afghanistan	AFG	2021-05-27	593313.0	479574	113739	
101	Afghanistan	AFG	2021-06-03	630305.0	481800	148505	
339	Afghanistan	AFG	2022-01-27	5081064.0	4517380	3868832	
433	Albania	ALB	2021-02-18	3049.0	2438	611	
515	Albania	ALB	2021-05-11	622507.0	440921	181586	

In [13]:
```
#again check the information of dataset
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 30847 entries, 94 to 86511
Data columns (total 15 columns):
 #   Column                               Non-Null Count  Dtype
---  ------                               --------------  -----
 0   country                              30847 non-null  object
 1   iso_code                             30847 non-null  object
 2   date                                 30847 non-null  object
 3   total_vaccinations                   30847 non-null  float64
 4   people_vaccinated                    30847 non-null  int32
 5   people_fully_vaccinated              30847 non-null  int32
 6   daily_vaccinations_raw               30847 non-null  int32
 7   daily_vaccinations                   30847 non-null  float64
 8   total_vaccinations_per_hundred       30847 non-null  int32
 9   people_vaccinated_per_hundred        30847 non-null  int32
 10  people_fully_vaccinated_per_hundred  30847 non-null  int32
 11  daily_vaccinations_per_million       30847 non-null  int32
 12  vaccines                             30847 non-null  object
 13  source_name                          30847 non-null  object
 14  source_website                       30847 non-null  object
dtypes: float64(2), int32(7), object(6)
memory usage: 2.9+ MB
```

In [14]:
```
#drop the unwanted column in dataset

df1=df1.drop(['vaccines','source_name','source_website'],axis=1)

df1
```

Out[14]:

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	da
94	Afghanistan	AFG	2021-05-27	593313.0	479574	113739	
101	Afghanistan	AFG	2021-06-03	630305.0	481800	148505	
339	Afghanistan	AFG	2022-01-27	5081064.0	4517380	3868832	
433	Albania	ALB	2021-02-18	3049.0	2438	611	
515	Albania	ALB	2021-05-11	622507.0	440921	181586	
...	...	...	...	...	...	...	
86507	Zimbabwe	ZWE	2022-03-25	8691642.0	4814582	3473523	
86508	Zimbabwe	ZWE	2022-03-26	8791728.0	4886242	3487962	
86509	Zimbabwe	ZWE	2022-03-27	8845039.0	4918147	3493763	
86510	Zimbabwe	ZWE	2022-03-28	8934360.0	4975433	3501493	
86511	Zimbabwe	ZWE	2022-03-29	9039729.0	5053114	3510256	

30847 rows × 12 columns

In [39]:
```
#The date is in the 'object' format. Let us change it to Datetime format for easy hand
df1['date'] =pd.to_datetime(df['date'], format='%Y-%m-%d')
```

In [15]:
```
df1
```

Out[15]:

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	da
94	Afghanistan	AFG	2021-05-27	593313.0	479574	113739	
101	Afghanistan	AFG	2021-06-03	630305.0	481800	148505	
339	Afghanistan	AFG	2022-01-27	5081064.0	4517380	3868832	
433	Albania	ALB	2021-02-18	3049.0	2438	611	
515	Albania	ALB	2021-05-11	622507.0	440921	181586	
...	...	...	...	...	...	...	
86507	Zimbabwe	ZWE	2022-03-25	8691642.0	4814582	3473523	
86508	Zimbabwe	ZWE	2022-03-26	8791728.0	4886242	3487962	
86509	Zimbabwe	ZWE	2022-03-27	8845039.0	4918147	3493763	
86510	Zimbabwe	ZWE	2022-03-28	8934360.0	4975433	3501493	
86511	Zimbabwe	ZWE	2022-03-29	9039729.0	5053114	3510256	

30847 rows × 12 columns

In [46]:

```
#Group by total vaccinations given by country and sort descending to identify the top
vacc_by_country = df.groupby('country').max().sort_values('total_vaccinations', ascend
vacc_by_country = vacc_by_country.iloc[:10]
vacc_by_country
```

Out[46]:

country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vacc
Afghanistan	AFG	2022-03-22	nan	5082824.0	4420127.0	
Russia	RUS	2022-03-29	nan	79954746.0	72841232.0	
Nauru	NRU	2022-03-21	nan	9150.0	7674.0	
Nepal	NPL	2022-03-29	nan	21994736.0	19014212.0	
Netherlands	NLD	2022-03-19	nan	13455761.0	12366525.0	
New Caledonia	NCL	2022-03-28	nan	188003.0	179880.0	
New Zealand	NZL	2022-03-29	nan	4284293.0	4051832.0	
Nicaragua	NIC	2022-03-25	nan	5498389.0	4113547.0	
Niger	NER	2022-03-24	nan	2180972.0	1545630.0	
Nigeria	NGA	2022-03-27	nan	21049754.0	9565143.0	

In [47]:

```
#Now sort by total vaccinations per 100
vacc_by_country = vacc_by_country.sort_values('total_vaccinations_per_hundred', ascend
vacc_by_country
```

Out[47]:

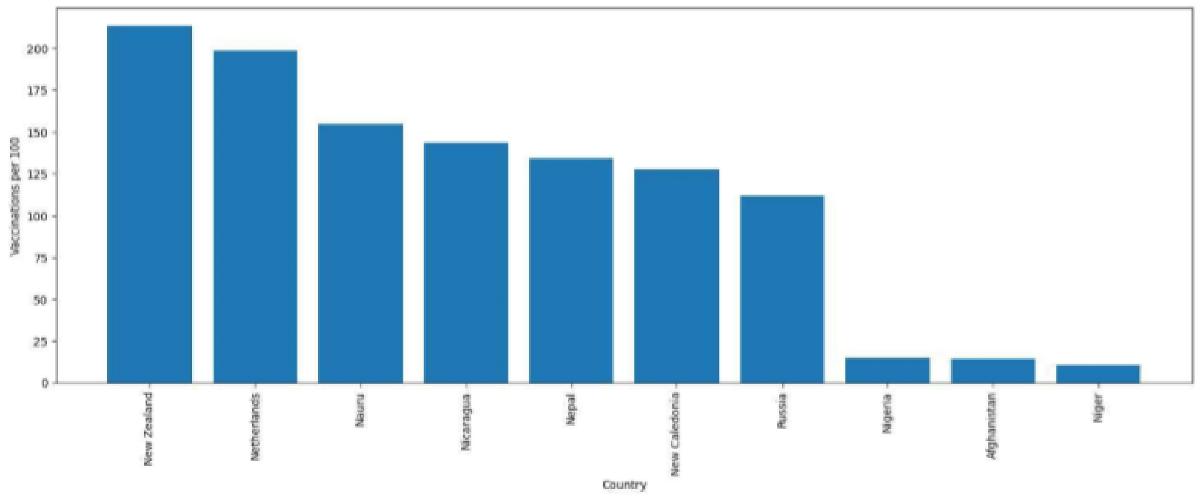
country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vacc
New Zealand	NZL	2022-03-29	nan	4284293.0	4051832.0	
Netherlands	NLD	2022-03-19	nan	13455761.0	12366525.0	
Nauru	NRU	2022-03-21	nan	9150.0	7674.0	
Nicaragua	NIC	2022-03-25	nan	5498389.0	4113547.0	
Nepal	NPL	2022-03-29	nan	21994736.0	19014212.0	
New Caledonia	NCL	2022-03-28	nan	188003.0	179880.0	
Russia	RUS	2022-03-29	nan	79954746.0	72841232.0	
Nigeria	NGA	2022-03-27	nan	21049754.0	9565143.0	
Afghanistan	AFG	2022-03-22	nan	5082824.0	4420127.0	
Niger	NER	2022-03-24	nan	2180972.0	1545630.0	

In [48]:

```
#for data preprocessing step this visualization is take to further analysis

plt.figure(figsize=(18, 6))
plt.bar(vacc_by_country.index, vacc_by_country.total_vaccinations_per_hundred)

plt.xticks(rotation = 90)
plt.ylabel('Vaccinations per 100')
plt.xlabel('Country')
plt.show()
```



In [16]:
```
#this dataset is ready for further analysis
print(df1.head())
```

```
         country iso_code        date  total_vaccinations  people_vaccinated  \
94   Afghanistan      AFG  2021-05-27            593313.0             479574
101  Afghanistan      AFG  2021-06-03            630305.0             481800
339  Afghanistan      AFG  2022-01-27           5081064.0            4517380
433      Albania      ALB  2021-02-18              3049.0               2438
515      Albania      ALB  2021-05-11            622507.0             440921

     people_fully_vaccinated  daily_vaccinations_raw  daily_vaccinations  \
94                    113739                    2859              6487.0
101                   148505                    4015              5285.0
339                  3868832                    6868              9802.0
433                      611                    1348               254.0
515                   181586                    9548             12160.0

     total_vaccinations_per_hundred  people_vaccinated_per_hundred  \
94                                1                              1
101                               1                              1
339                              12                             11
433                               0                              0
515                              21                             15

     people_fully_vaccinated_per_hundred  daily_vaccinations_per_million
94                                     0                             163
101                                    0                             133
339                                    9                             246
433                                    0                              88
515                                    6                            4233
```

ENCODING CATEGORICAL DATA :

To encode categorical data using one-hot encoding in Python, you can use the pd.get_dummies function in the Pandas library. One-hot encoding converts categorical variables into binary (0/1) format, making them suitable for machine learning algorithms.

Code:

[14]:
```
# Use get_dummies to perform one-hot encoding
dataset_encoded = pd.get_dummies(dataset, columns=['country'])
# Display the DataFrame with one-hot encoding
print(dataset_encoded.head())
```

Output:

```
  iso_code        date  total_vaccinations  people_vaccinated  \
0      AFG  2021-02-22                 0.0                0.0
1      AFG  2021-02-23                 NaN                NaN
2      AFG  2021-02-24                 NaN                NaN
3      AFG  2021-02-25                 NaN                NaN
4      AFG  2021-02-26                 NaN                NaN

   people_fully_vaccinated  daily_vaccinations_raw  daily_vaccinations  \
0                      NaN                     NaN                 NaN
1                      NaN                     NaN              1367.0
2                      NaN                     NaN              1367.0
3                      NaN                     NaN              1367.0
4                      NaN                     NaN              1367.0

   total_vaccinations_per_hundred  people_vaccinated_per_hundred  \
0                             0.0                            0.0
1                             NaN                            NaN
2                             NaN                            NaN
3                             NaN                            NaN
4                             NaN                            NaN

   people_fully_vaccinated_per_hundred  ...  country_Uruguay  \
0                                  NaN  ...            False
1                                  NaN  ...            False
2                                  NaN  ...            False
3                                  NaN  ...            False
4                                  NaN  ...            False
```

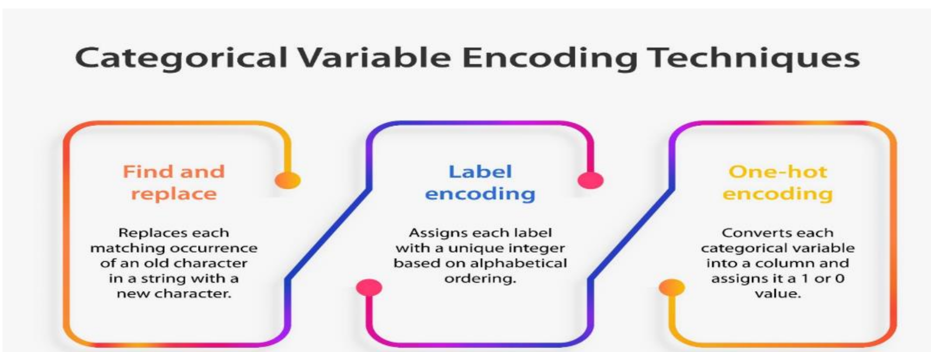
```
   country_Uzbekistan  country_Vanuatu  country_Venezuela  country_Vietnam  \
0               False            False              False            False
1               False            False              False            False
2               False            False              False            False
3               False            False              False            False
4               False            False              False            False

   country_Wales  country_Wallis and Futuna  country_Yemen  country_Zambia  \
0          False                      False          False           False
1          False                      False          False           False
2          False                      False          False           False
3          False                      False          False           False
4          False                      False          False           False

   country_Zimbabwe
0             False
1             False
2             False
3             False
4             False

[5 rows x 237 columns]
```

The get_dummies function will create binary (0/1) columns for each unique category in the ' country' column. This process effectively converts the categorical data into a numerical form at suitable for analysis or machine learning.



[8]:
```python
#5.Encoding Categorical Variables:

df = pd.get_dummies(df, columns=['country', 'vaccines'], drop_first=True)


df
```

[8]:
```
      iso_code        date  total_vaccinations  people_vaccinated  \
0          AFG  2021-02-22           -0.143704          -0.170046
1          AFG  2021-02-23           -0.143704          -0.170046
2          AFG  2021-02-24           -0.143704          -0.170046
3          AFG  2021-02-25           -0.143704          -0.170046
4          AFG  2021-02-26           -0.143704          -0.170046
...        ...         ...                 ...                ...
86507      ZWE  2022-03-25           -0.089753          -0.073170
86508      ZWE  2022-03-26           -0.089132          -0.071728
86509      ZWE  2022-03-27           -0.088801          -0.071086
86510      ZWE  2022-03-28           -0.088247          -0.069933
86511      ZWE  2022-03-29           -0.087593          -0.068370

       people_fully_vaccinated  daily_vaccinations_raw  daily_vaccinations  \
0                          0.0                     0.0                 0.0
1                          0.0                     0.0              1367.0
2                          0.0                     0.0              1367.0
3                          0.0                     0.0              1367.0
4                          0.0                     0.0              1367.0
...                        ...                     ...                 ...
86507                3473523.0                139213.0             69579.0
86508                3487962.0                100086.0             83429.0
86509                3493763.0                 53311.0             90629.0
86510                3501493.0                 89321.0            100614.0
86511                3510256.0                105369.0            103751.0

       total_vaccinations_per_hundred  people_vaccinated_per_hundred  \
```

```
0                                0.00                           0.00
1                                0.00                           0.00
2                                0.00                           0.00
3                                0.00                           0.00
4                                0.00                           0.00
...                              ...                            ...
86507                           57.59                          31.90
86508                           58.25                          32.38
86509                           58.61                          32.59
86510                           59.20                          32.97
86511                           59.90                          33.48

       people_fully_vaccinated_per_hundred  ...  \
0                                     0.00  ...
1                                     0.00  ...
2                                     0.00  ...
3                                     0.00  ...
4                                     0.00  ...
...                                    ...  ...
86507                                23.02  ...
86508                                23.11  ...
86509                                23.15  ...
86510                                23.20  ...
86511                                23.26  ...

       vaccines_Oxford/AstraZeneca, Sputnik V vaccines_Pfizer/BioNTech  \
0                                           0                        0
1                                           0                        0
2                                           0                        0
3                                           0                        0
4                                           0                        0
...                                       ...                      ...
86507                                       0                        0
86508                                       0                        0
86509                                       0                        0
86510                                       0                        0
86511                                       0                        0

      vaccines_Pfizer/BioNTech, Sinopharm/Beijing  \
0                                               0
1                                               0
2                                               0
3                                               0
4                                               0
...                                           ...
86507                                           0
86508                                           0


86509                                           0
86510                                           0
86511                                           0

       vaccines_Pfizer/BioNTech, Sinopharm/Beijing, Sputnik V  \
0                                                      0
1                                                      0
2                                                      0
3                                                      0
4                                                      0
...                                                  ...
86507                                                  0
86508                                                  0
86509                                                  0
86510                                                  0
86511                                                  0
```

```
       vaccines_Pfizer/BioNTech, Sinovac  \
0                                      0
1                                      0
2                                      0
3                                      0
4                                      0
...                                  ...
86507                                  0
86508                                  0
86509                                  0
86510                                  0
86511                                  0

       vaccines_Pfizer/BioNTech, Sinovac, Turkovac  \
0                                                0
1                                                0
2                                                0
3                                                0
4                                                0
...                                            ...
86507                                            0
86508                                            0
86509                                            0
86510                                            0
86511                                            0

       vaccines_Pfizer/BioNTech, Sputnik V  \
0                                        0
1                                        0
2                                        0
3                                        0
4                                        0
...                                    ...
86507                                    0
86508                                    0
86509                                    0
86510                                    0
86511                                    0

       vaccines_QazVac, Sinopharm/Beijing, Sputnik V  \
0                                                  0
1                                                  0
2                                                  0
3                                                  0
4                                                  0
...                                              ...
86507                                              0
86508                                              0
86509                                              0
86510                                              0
86511                                              0

       vaccines_Sinopharm/Beijing  vaccines_Sinopharm/Beijing, Sputnik V
0                               0                                      0
1                               0                                      0
2                               0                                      0
3                               0                                      0
4                               0                                      0
...                           ...                                    ...
86507                           0                                      0
86508                           0                                      0
86509                           0                                      0
86510                           0                                      0
86511                           0                                      0

[86512 rows x 318 columns]
```

SPLITTING THE DATASET INTO TEST SET AND TRAINING SET :

To split dataset into training and test sets using the train_test_split function from scikitlearn,input features (X) and target variable (Y) needed to be specified first.

Code:

```
# Specify your features (X) and target variable (Y)
X = dataset_encoded.drop(columns=['total_vaccinations'])
# X contains all columns except 'total_vaccinations'
Y = dataset_encoded['total_vaccinations']
# Y is the 'total_vaccinations' column
# Split the data into training and test sets (adjust the test_size and random_state as needed)
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=42)
# Display the shapes of the resulting sets to verify the split
print("X_train shape:", X_train.shape)
print("X_test shape:", X_test.shape)
print("Y_train shape:", Y_train.shape)
print("Y_test shape:", Y_test.shape)
```

Output:

```
X_train shape: (69209, 236)
X_test shape: (17303, 236)
Y_train shape: (69209,)
Y_test shape: (17303,)
```

In this code, we first separate the features (X) and the target variable (Y) from the dataset. Then, we use train_test_split to split the data into training and test sets. The test_size parameter determines the proportion of the data that will be allocated to the test set, and random_state is set to a specific value (e.g., 42) to ensure reproducibility.

[11]:
```python
#9.Data Splitting:

from sklearn.model_selection import train_test_split

# Replace 'actual_target_column_name' with the correct column name
X = df.drop('total_vaccinations', axis=1)  # Features
y = df['people_vaccinated']  # Target variable

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
 ↪random_state=42)


#10.Save Preprocessed Data:

df.to_csv('preprocessed_data.csv', index=False)
```

```python
df
```

[11]:
```
      iso_code        date  total_vaccinations  people_vaccinated  month  \
0          AFG  2021-02-22           -0.143704          -0.170046      2
1          AFG  2021-02-23           -0.143704          -0.170046      2
2          AFG  2021-02-24           -0.143704          -0.170046      2
3          AFG  2021-02-25           -0.143704          -0.170046      2
4          AFG  2021-02-26           -0.143704          -0.170046      2
…          …         …                   …                  …      …
86507      ZWE  2022-03-25           -0.089753          -0.073170      3
86508      ZWE  2022-03-26           -0.089132          -0.071728      3
86509      ZWE  2022-03-27           -0.088801          -0.071086      3
86510      ZWE  2022-03-28           -0.088247          -0.069933      3
86511      ZWE  2022-03-29           -0.087593          -0.068370      3

       day_of_week
0                0
1                1
2                2
3                3
4                4
…              …
86507            4
86508            5
86509            6
86510            0
86511            1

[70909 rows x 6 columns]
```

FEATURE SCALING :

Feature scaling is an important preprocessing step in many machine learning algorithms. You can use the StandardScaler from scikit-learn to scale your features so that they have a mean of 0 and a standard deviation of 1.

Code:

```
from sklearn.preprocessing import StandardScaler
# Assuming you have your training and test data (X_train and X_test) defined
# Create a StandardScaler instance
scaler = StandardScaler()
# Fit the scaler on the training data and transform both training and test data
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
# Display the scaled features
print("Scaled X_train:")
print(X_train_scaled)
print("Scaled X_test:")
print(X_test_scaled)
```

In this code, we first create a StandardScaler instance. We then fit the scaler on the training data using the fit_transform method, and apply the same transformation to both the training and test data using the transform method. This ensures that the scaling is consistent between the two sets.

[7]:
```python
#4.Data Scaling and Normalization:

from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
df[['total_vaccinations', 'people_vaccinated']] =scaler.
 ↪fit_transform(df[['total_vaccinations', 'people_vaccinated']])

df
```

[7]:
```
           country iso_code        date  total_vaccinations  people_vaccinated  \
0      Afghanistan      AFG  2021-02-22           -0.143704          -0.170046
1      Afghanistan      AFG  2021-02-23           -0.143704          -0.170046
2      Afghanistan      AFG  2021-02-24           -0.143704          -0.170046
3      Afghanistan      AFG  2021-02-25           -0.143704          -0.170046
4      Afghanistan      AFG  2021-02-26           -0.143704          -0.170046
…              …      …         …                 …                 …
86507     Zimbabwe      ZWE  2022-03-25           -0.089753          -0.073170
86508     Zimbabwe      ZWE  2022-03-26           -0.089132          -0.071728
86509     Zimbabwe      ZWE  2022-03-27           -0.088801          -0.071086
86510     Zimbabwe      ZWE  2022-03-28           -0.088247          -0.069933
86511     Zimbabwe      ZWE  2022-03-29           -0.087593          -0.068370

       people_fully_vaccinated  daily_vaccinations_raw  daily_vaccinations  \
0                          0.0                     0.0                 0.0
1                          0.0                     0.0              1367.0
2                          0.0                     0.0              1367.0
3                          0.0                     0.0              1367.0
4                          0.0                     0.0              1367.0
…                            …                       …                   …
86507                3473523.0                139213.0             69579.0
86508                3487962.0                100086.0             83429.0
86509                3493763.0                 53311.0             90629.0
86510                3501493.0                 89321.0            100614.0
86511                3510256.0                105369.0            103751.0

       total_vaccinations_per_hundred  people_vaccinated_per_hundred  \
0                                0.00                           0.00
1                                0.00                           0.00
2                                0.00                           0.00
3                                0.00                           0.00
4                                0.00                           0.00
…                                   …                              …
86507                           57.59                          31.90
86508                           58.25                          32.38
86509                           58.61                          32.59
86510                           59.20                          32.97
86511                           59.90                          33.48

       people_fully_vaccinated_per_hundred  daily_vaccinations_per_million  \
0                                     0.00                             0.0
1                                     0.00                            34.0
2                                     0.00                            34.0
3                                     0.00                            34.0
4                                     0.00                            34.0
…                                        …                               …
86507                                23.02                          4610.0
86508                                23.11                          5528.0
86509                                23.15                          6005.0
86510                                23.20                          6667.0
86511                                23.26                          6874.0

                                                vaccines  \
0      Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi…
1      Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi…
2      Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi…
3      Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi…
4      Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi…
…                                                    …
86507  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac…
86508  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac…
86509  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac…
86510  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac…
86511  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac…

                     source_name  \
0      World Health Organization
1      World Health Organization
2      World Health Organization
3      World Health Organization
4      World Health Organization
…                              …
86507         Ministry of Health
86508         Ministry of Health
86509         Ministry of Health
86510         Ministry of Health
86511         Ministry of Health

                                          source_website
0                                https://covid19.who.int/
```

```
1                                https://covid19.who.int/
2                                https://covid19.who.int/
3                                https://covid19.who.int/
4                                https://covid19.who.int/
…                                                  …
86507  https://www.arcgis.com/home/webmap/viewer.html…
86508  https://www.arcgis.com/home/webmap/viewer.html…
86509  https://www.arcgis.com/home/webmap/viewer.html…
86510  https://www.arcgis.com/home/webmap/viewer.html…
86511  https://www.arcgis.com/home/webmap/viewer.html…

[86512 rows x 15 columns]
```

[9]:
```
#6.Feature Selection:

df = df[['iso_code', 'date', 'total_vaccinations', 'people_vaccinated']]

df
```

[9]:

	iso_code	date	total_vaccinations	people_vaccinated
0	AFG	2021-02-22	-0.143704	-0.170046
1	AFG	2021-02-23	-0.143704	-0.170046
2	AFG	2021-02-24	-0.143704	-0.170046
3	AFG	2021-02-25	-0.143704	-0.170046
4	AFG	2021-02-26	-0.143704	-0.170046
…	…	…	…	…
86507	ZWE	2022-03-25	-0.089753	-0.073170
86508	ZWE	2022-03-26	-0.089132	-0.071728
86509	ZWE	2022-03-27	-0.088801	-0.071086
86510	ZWE	2022-03-28	-0.088247	-0.069933
86511	ZWE	2022-03-29	-0.087593	-0.068370

[86512 rows x 4 columns]

[10]:
```
#7.Date-Based Features:

df['month'] = df['date'].dt.month
df['day_of_week'] = df['date'].dt.dayofweek

#8.Outlier Detection and Handling:

import numpy as np
from scipy import stats

# Detect outliers using the IQR method
Q1 = df['total_vaccinations'].quantile(0.25)
Q3 = df['total_vaccinations'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Remove outliers
df = df[(df['total_vaccinations'] >= lower_bound) & (df['total_vaccinations']
 <= upper_bound)]

df
```

[10]:

```
      iso_code        date  total_vaccinations  people_vaccinated  month  \
0          AFG  2021-02-22           -0.143704          -0.170046      2
1          AFG  2021-02-23           -0.143704          -0.170046      2
2          AFG  2021-02-24           -0.143704          -0.170046      2
3          AFG  2021-02-25           -0.143704          -0.170046      2
4          AFG  2021-02-26           -0.143704          -0.170046      2
...        ...         ...                 ...                ...    ...
86507      ZWE  2022-03-25           -0.089753          -0.073170      3
86508      ZWE  2022-03-26           -0.089132          -0.071728      3
86509      ZWE  2022-03-27           -0.088801          -0.071086      3
86510      ZWE  2022-03-28           -0.088247          -0.069933      3
86511      ZWE  2022-03-29           -0.087593          -0.068370      3

       day_of_week
0                0
1                1
2                2
3                3
4                4
...            ...
86507            4
86508            5
86509            6
86510            0
86511            1

[70909 rows x 6 columns]
```

CONCLUSION :

- In the quest to build a Covid Vaccines Analysis Model, we have embarked on a critical journey that begins with loading and preprocessing the Covid Vaccines dataset. We have traversed through essential steps, starting with importing the necessary libraries to facilitate data manipulation and analysis.

- Understanding the data's structure, characteristics, and any potential issues through exploratory data analysis (EDA) is essential for informed decision-making.

- Data preprocessing emerged as a pivotal aspect of this process. It involves cleaning, transforming, and refining the dataset to ensure that it aligns with the requirements of machine learning algorithms.

- With these foundational steps completed, our dataset is now primed for the subsequent stages of building and training a Covid Vaccines Analysis Model.

- Covid Vaccines Analysis, as exemplified by this dataset, is a vital component responsible for Vaccination management in the face of managing and mitigating the impact of the global pandemic thereby optimizing vaccination strategies.

IBM DATA ANALYTICS WITH COGNOS

TEAM NAME : Proj_229800_Team_1

PROJECT : 3101-COVID Vaccines Analysis

TEAM MEMBERS :

- **PAVITHRA V**
- **SHARU DHARSHINI S**
- **SUBATHRA E**
- **SHALINI S**

TEAM LEADER : PAVITHRA V