FINAL DOCUMENTATION

COVID VACCINE ANALYSIS

Final Documentation

Table Of Content:

S.No	Title	Page no
1.	Problem outline	3
2.	Design Thinking process	3-4
3.	Development phase	4-37
4.	Data source	38-39
5.	Data Pre-processing	39-46
6.	Analysis Technique	46-48
7.	Key insights and recommendation	48-50

Challenge Taken : COVID Vaccine Analysis

Problem Outline

This project aims to comprehensively analyze Covid-19 vaccine data, specifically concentrating on vaccine effectiveness, distribution patterns, and adverse reactions. The primary objective is to generate actionable insights to assist policymakers and healthcare entities in enhancing their vaccination deployment strategies. Key tasks encompass data collection, data preprocessing, exploratory data analysis, statistical examination, and data visualization.

Design Thinking Process

Introduction: It is a systematic approach to analyzing COVID-19 vaccine data using design thinking principles, offering valuable insights and recommendations for policymakers and healthcare professionals to optimize vaccine deployment and combat the pandemic effectively.

Data Collection:

Data is collected daily from the Our World in Data GitHub repository for COVID-19, and country-level vaccination data is compiled. This data includes information on the number of people vaccinated and fully vaccinated, daily vaccinations, ratios per hundred and per million, and details about the vaccines used in each country. A second file contains data on vaccinations by manufacturer, including the location, date, vaccine type, and the total number of vaccinations. This comprehensive dataset is crucial for analyzing and monitoring the progress of COVID-19 vaccination efforts worldwide.

Data Processing:

In the data preprocessing stage, COVID-19 vaccine data is collected from reliable sources and imported into data analysis tools like Python or R. Missing values are identified and managed through methods like removal or imputation. Numerical features may be normalized for consistent scaling, while categorical features are transformed into numerical representations using techniques like one-hot encoding to prepare the data for analysis.

Exploratory Data Analysis (EDA):

Exploratory Data Analysis (EDA) for COVID-19 vaccine data is a comprehensive process. It involves initial dataset structure review and anomaly detection. Subsequently, it delves into vaccine efficacy analysis, distribution assessment, adverse effect examination, and time series analysis. Correlation and regression are explored, along with cluster analysis for regional insights. Communication of findings is vital through clear visualizations and reports, enabling actionable recommendations for vaccine deployment. EDA is an iterative, ongoing process crucial for informed decision-making in the ever-evolving COVID-19 landscape.

Statistical Analysis:

In the statistical analysis phase, a variety of key statistical tests are applied to COVID-19 vaccine data. These tests examine vaccine efficacy, adverse effects, and distribution across different populations. Important statistical parameters, such as means, medians, standard deviations, and distributions, are calculated for critical metrics like total vaccinations, people vaccinated, and daily vaccinations. Time series analysis is used to reveal trends and seasonality. Additionally, comparisons are made across countries, and insights are drawn regarding the distribution and impact of various vaccine types. This statistical approach ensures data-driven decision-making for optimizing vaccine deployment and public health strategies.

Data Visualization:

Data visualization is a vital component in COVID-19 vaccine data analysis, employing techniques like bar charts, line charts, and heatmaps to communicate complex information. Visualizations facilitate the identification of vaccination trends, disparities, and geographic distribution. They enable policymakers to make informed decisions, engage stakeholders, and identify outliers, fostering transparency and effective communication in the fight against the pandemic.

Insights and Recommendation:

Insights drawn from the analysis emphasize prioritizing vaccines with higher efficacy for at-risk populations, monitoring emerging variants, and enhancing safety communication. Recommendations include promoting vaccine education, establishing convenient vaccination clinics, fostering data sharing, and preparedness for supply chain disruptions. This holistic approach guides effective COVID-19 vaccine deployment, emphasizing adaptability to evolving circumstances and data.

Development Phase

Development Phase 1:

Objective

Our goal in this phase is to take our analysis to next level by applying advanced data analytics techniques. In this phase, an attempt has been made to analyze various information of COVID-19 World Vaccination Progress such as country, total Vaccinations , people vaccinated, daily vaccinations total vaccinations per hundred, people vaccinated per hundred, people fully vaccinated per hundred, vaccines and many more and used the specific techniques and best practices involved in data loading and preprocessing for vaccine analysis, highlighting their significance in ensuring the credibility and effectiveness of vaccine-related research.

Data loading

Data loading involves the collection and retrieval of vaccine-related data from various sources, such as healthcare databases, clinical trials, and vaccination records.

Data Preprocessing

Data Preprocessing, on the other hand, is the vital step where raw vaccine-related data is transformed and cleaned. This process encompasses data cleaning, transformation, and feature engineering to ensure that the data is suitable for further analysis **Given data set:**

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_
0	Afghanistan	AFG	2021-02-22	0.0	0.0	NaN	NaN	NaN
1	Afghanistan	AFG	2021-02-23	NaN	NaN	NaN	NaN	1367
2	Afghanistan	AFG	2021-02-24	NaN	NaN	NaN	NaN	1367
3	Afghanistan	AFG	2021-02-25	NaN	NaN	NaN	NaN	1367
4	Afghanistan	AFG	2021-02-26	NaN	NaN	NaN	NaN	1367
...	...	...	...	...	...	...	...	...
86507	Zimbabwe	ZWE	2022-03-25	8691642.0	4814582.0	3473523.0	139213.0	6957
86508	Zimbabwe	ZWE	2022-03-26	8791728.0	4886242.0	3487962.0	100086.0	8342
86509	Zimbabwe	ZWE	2022-03-27	8845039.0	4918147.0	3493763.0	53311.0	9062
86510	Zimbabwe	ZWE	2022-03-28	8934360.0	4975433.0	3501493.0	89321.0	1006
86511	Zimbabwe	ZWE	2022-03-29	9039729.0	5053114.0	3510256.0	105369.0	1037

Importance of loading and processing dataset:

Loading and preprocessing the dataset is an important first step in building any machine learning model. However, it is especially important for vaccine analysis, as the datasets are often complex and noisy. 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 covid vaccine analysis dataset;

Missing Data

Another common issue that we face in real-world data is the absence of data points. Most machine learning models can’t handle missing values in the data, so you need to intervene and adjust the data to be properly used inside the model.

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.

1.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.

1.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.

2.Load the Dataset:

Load your dataset into a Pandas DataFrame**.** The quality and reliability of data can significantly impact the outcomes of vaccine analysis, making it imperative to have robust data loading procedures in place.

Program:

```
vaccine_df =

pd.read_csv('/content/drive/MyDrive/country_vaccinations.csv')

vaccine_df
```

3. Exploring data:

Perform EDA to understand your data better. This includes checking for missing values, exploring the data's statistics, and visualizing it to identify patterns.

Program

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

Output

```
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
```

Explore statistics

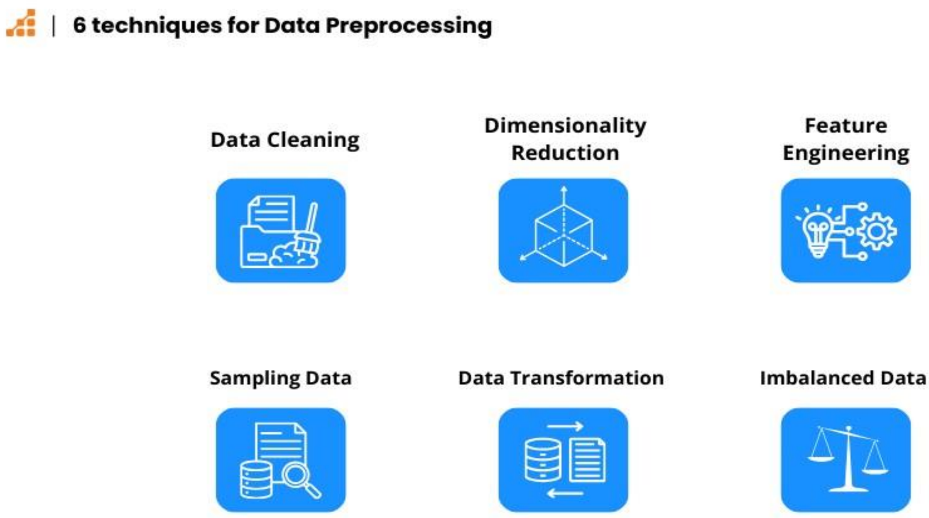
vaccine_df.describe()

Output

	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_va
count	4.360700e+04	4.129400e+04	3.880200e+04	3.536200e+04	8.6213
mean	4.592964e+07	1.770508e+07	1.413830e+07	2.705996e+05	1.3130
std	2.246004e+08	7.078731e+07	5.713920e+07	1.212427e+06	7.6823
min	0.000000e+00	0.000000e+00	1.000000e+00	0.000000e+00	0.0000
25%	5.264100e+05	3.494642e+05	2.439622e+05	4.668000e+03	9.0000
50%	3.590096e+06	2.187310e+06	1.722140e+06	2.530900e+04	7.3430
75%	1.701230e+07	9.152520e+06	7.559870e+06	1.234925e+05	4.4098
max	3.263129e+09	1.275541e+09	1.240777e+09	2.474100e+07	2.2424

4.**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.



Data cleaning: 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.

Feature Scaling: Normalize or standardize numerical features to bring them to a common scale. Common methods include Min-Max scaling (scaling features to a specific range) and z-score normalization (scaling features to have a mean of 0 and a standard deviation of 1).

Feature Engineering: Create new features or modify existing ones to capture more meaningful information from the data. This may involve mathematical transformations, interaction terms, or aggregations.

Data transformation: It is a critical aspect of data preprocessing that involves converting and modifying the data to make it more suitable for analysis. It can help improve the performance of machine learning models, enhance the interpretability of the data, and ensure that it aligns with the assumptions of certain statistical techniques.

import numpy

as np import

pandas as pd

import

matplotlib.py

plot as plt

import

seaborn as

sns

plt.figure(figs

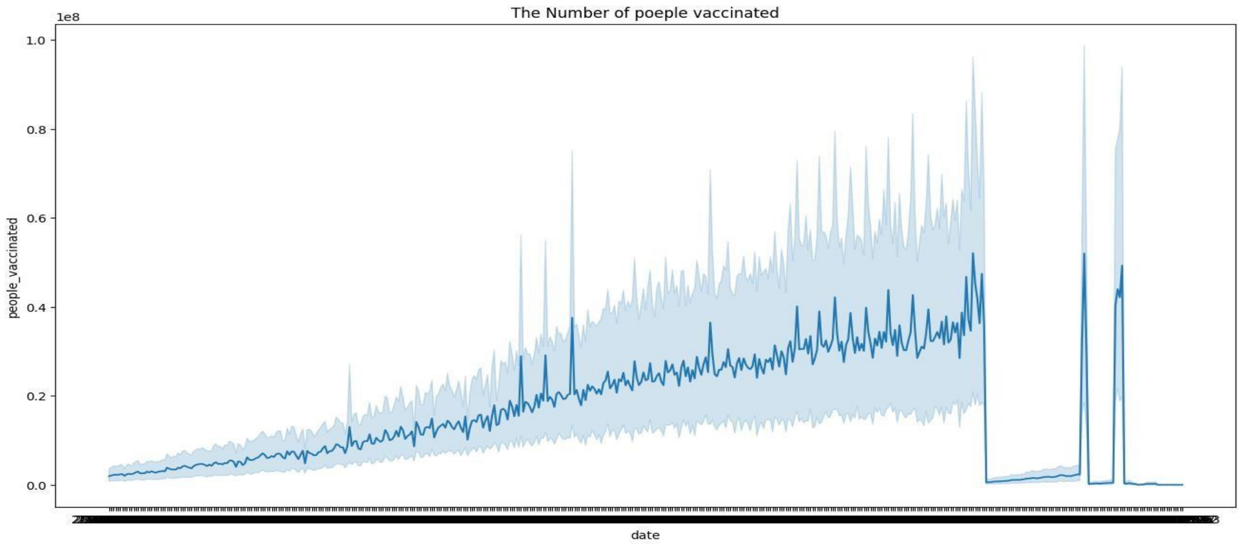
ize=(16,8))

sns.lineplot(x=vaccine_df.date,

y=vaccine_df.people_vaccinated) plt.title('The

Number of poeple vaccinated') plt.show()

Output



plt.figure(figsize=(16,8))

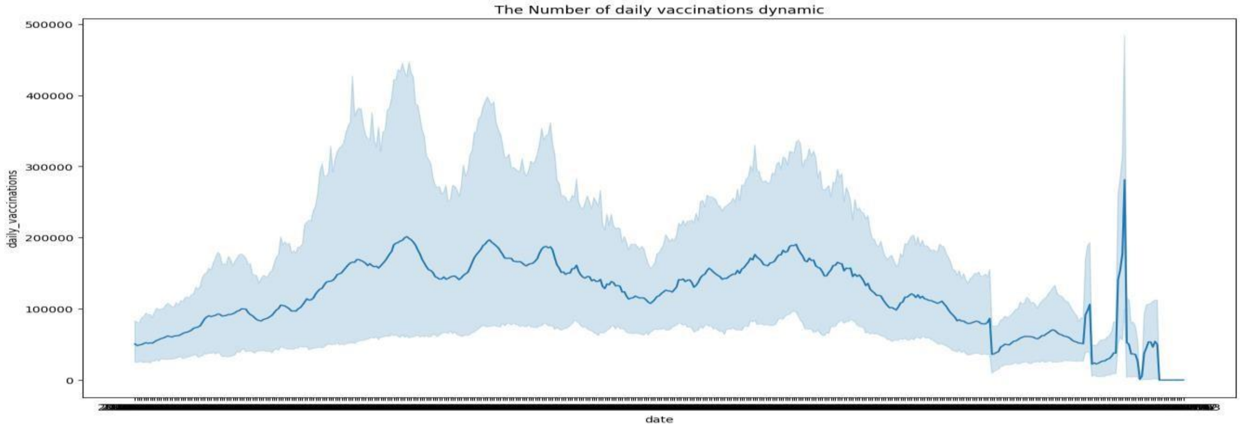
sns.lineplot(x=vaccine_df.date,

y=vaccine_df.daily_vaccinations) plt.title('The

Number of daily vaccinations dynamic')

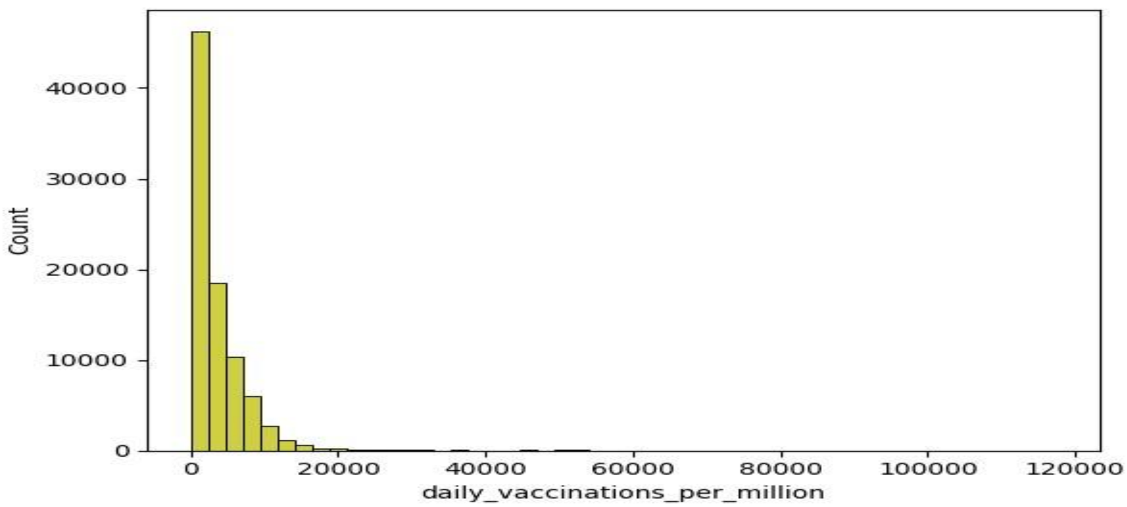
plt.show()

Output



sns.histplot(vaccine_df, x='daily_vaccinations_per_million', bins=50, color='y')

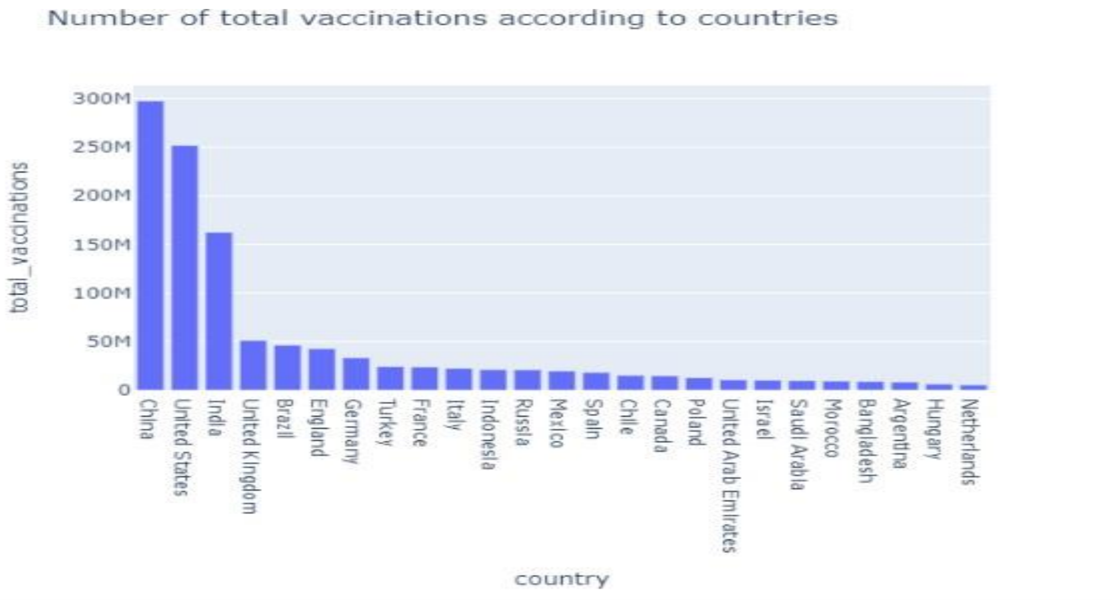
Output



data = vaccine_df[['country','total_vaccinations']].nlargest(25,'total_vaccinations')

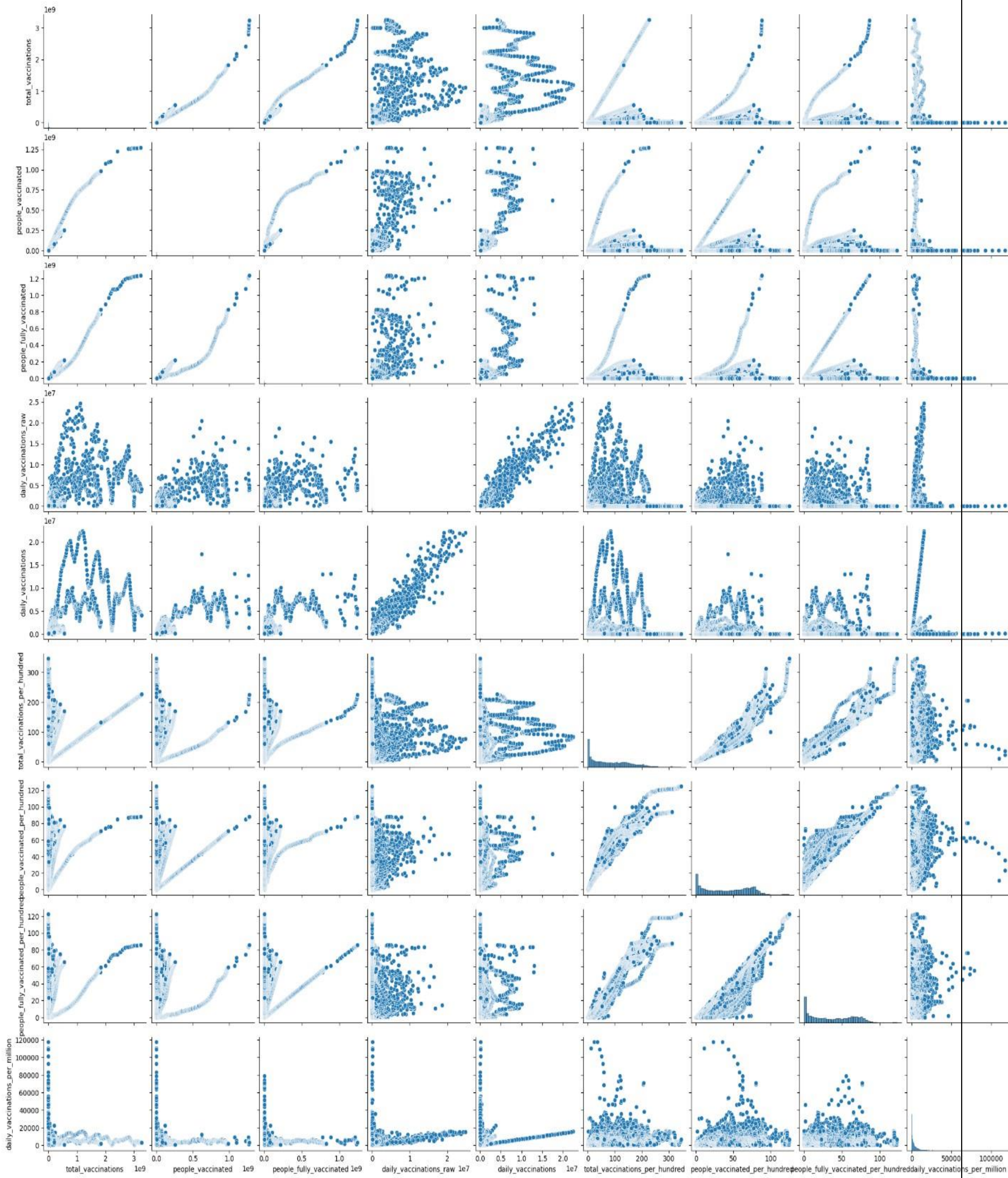
fig = px.bar(data, x = 'country',y = 'total_vaccinations',title="Number of total vaccinations according to countries",)fig.show()

Output



plt.figure(figsize=(12,8))

sns.pairplot(vaccine_df) **Output**

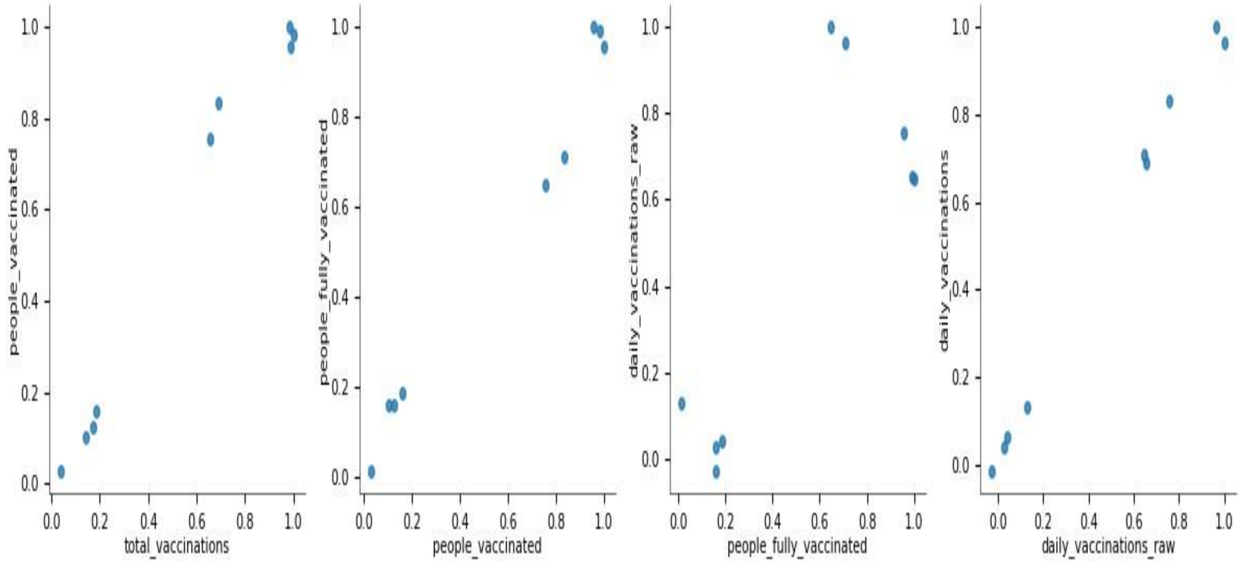


vaccine_df.corr(numeric_only=True)

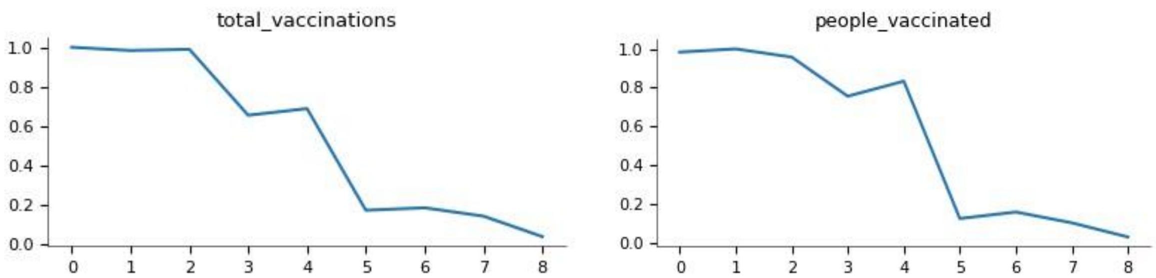
Output

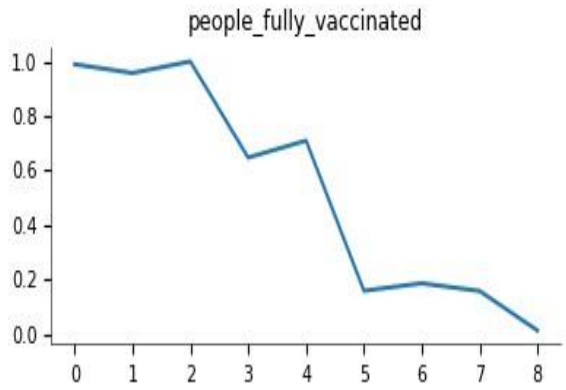
index	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations	total_vaccinations_per_hundred
total_vaccinations	1.0	0.983438280596498	0.9896813381301297	0.6547117881908026	0.6885018889121146	0.17229710004672275
people_vaccinated	0.983438280596498	1.0	0.9575994800578601	0.7555402250789669	0.8334332974829378	0.12393803599973728
people_fully_vaccinated	0.9896813381301297	0.9575994800578601	1.0	0.647573972663791	0.7097681654065912	0.1590262533355807
daily_vaccinations_raw	0.6547117881908026	0.7555402250789669	0.647573972663791	1.0	0.96551657258391	0.02932884050938329
daily_vaccinations	0.6885018889121146	0.8334332974829378	0.7097681654065912	0.96551657258391	1.0	0.0422272861988585
total_vaccinations_per_hundred	0.17229710004672275	0.12393803599973728	0.1590262533355807	0.02932884050938329	0.0422272861988585	1.0
people_vaccinated_per_hundred	0.18464905459019076	0.15777531767489797	0.18636873471491208	0.042445074564014224	0.06256586245695953	0.9653293137912788
people_fully_vaccinated_per_hundred	0.14225214870015712	0.10171739078725649	0.15828335879738206	-0.027884824010809273	-0.014054926124652107	0.9754546830947068
daily_vaccinations_per_million	0.038298146800641905	0.028720142515809583	0.013220268364296078	0.13107810399599185	0.13382226191364244	0.18460888459647887

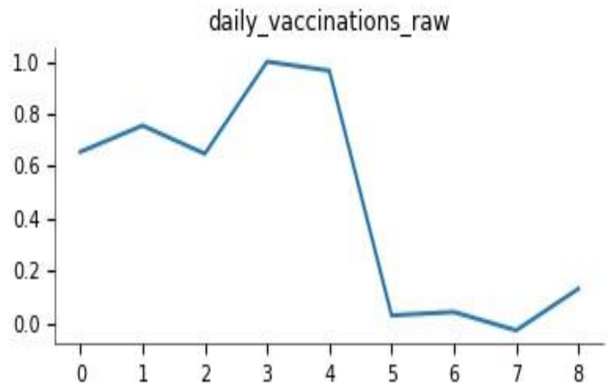
2-d Distributions



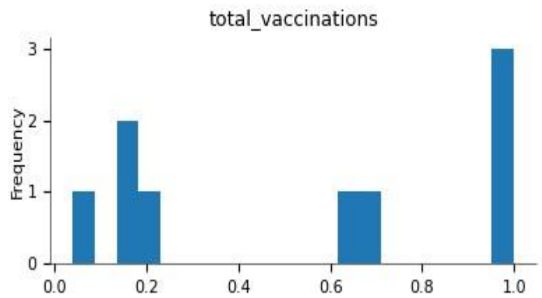
Values

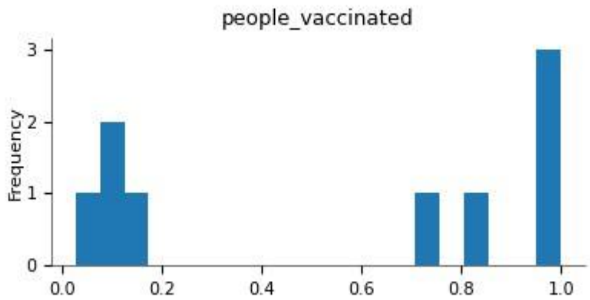


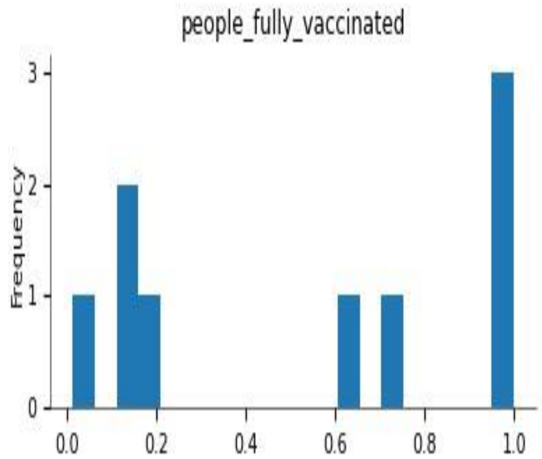


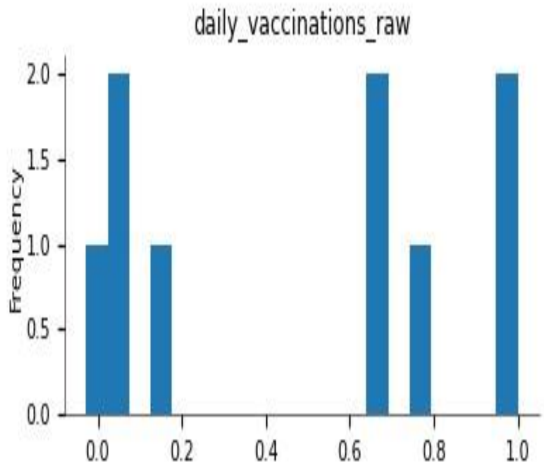


Distributions



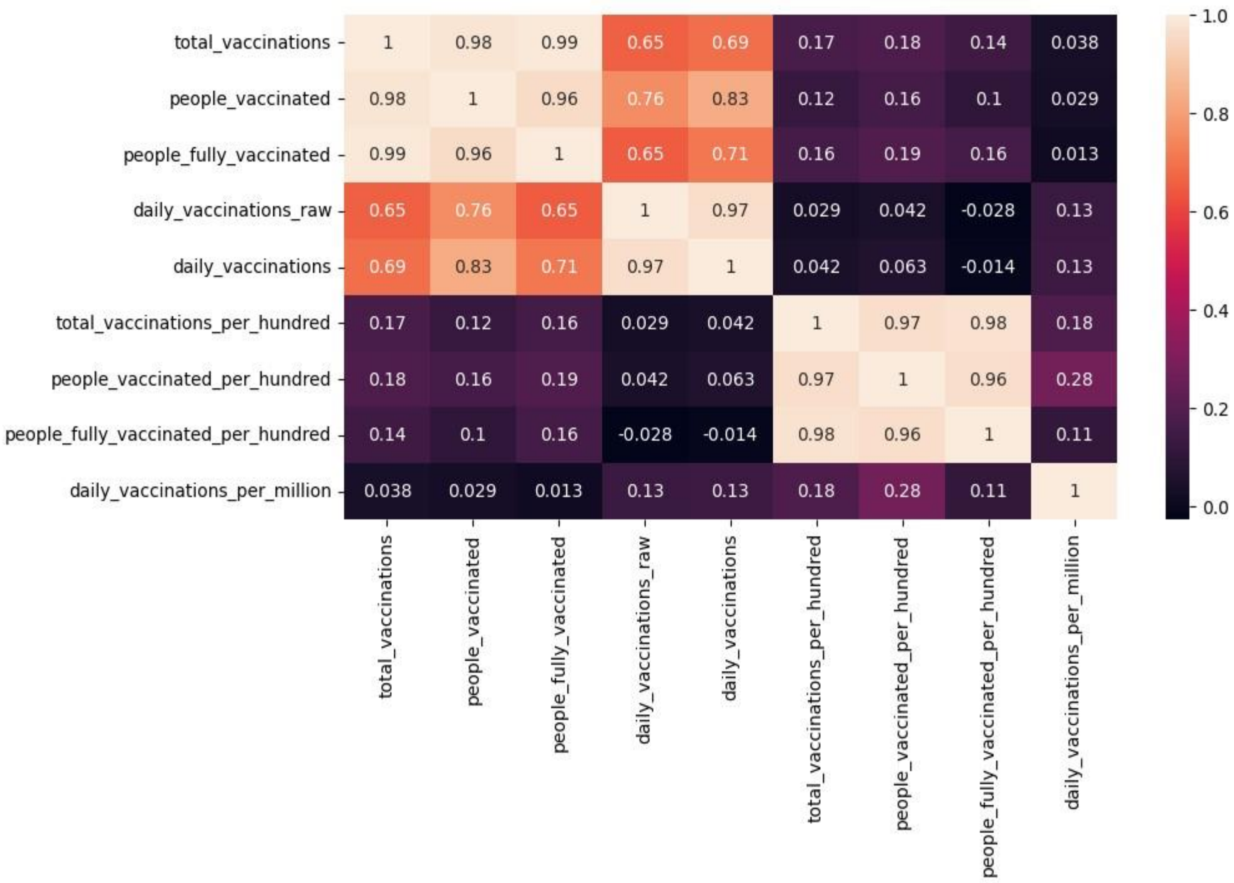







```
plt.figure(figsize=(10,5))
sns.heatmap(vaccine_df.corr(numeric_only = True), annot=True)
```

Output



Conclusion:

Data loading and preprocessing for vaccine analysis serve as the critical initial steps that empower researchers and healthcare professionals to make informed decisions about vaccine development, distribution, and safety. The quality, accuracy, and suitability of the data at this stage are pivotal in determining the success of subsequent analyses. Through diligent and systematic data handling, we can harness the power of data-driven insights to address public health challenges and contribute to the betterment of global health.

Development Phase 2:

Objective:

The goal is to advance the COVID vaccine analysis through exploratory data analysis (EDA), statistical examination, and visualization. The focus is on uncovering patterns, trends, and relationships within the data to inform evidence-based insights. By employing a combination of statistical tools and visualizations, the aim is to provide a comprehensive understanding of vaccination dynamics, identify global trends, and contribute valuable insights for strategic decision-making.

1. **Importing Required Libraries**

```
import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

import plotly.express as px

import plotly.graph_objs as go

from plotly.offline import init_notebook_mode, iplot, plot
```

2. **Loading Data Sets**

Read data from CSV files

```
df_manufacturer = pd.read_csv('/content/drive/MyDrive/Naan
Mudalvan/country_vaccinations_by_manufacturer.csv')

df_vaccinations = pd.read_csv('/content/drive/MyDrive/Naan
Mudalvan/country_vaccinations.csv')
```

Columns Present in Given Data Sets

```
df_manufacturer.columns
```

Output:

['location', 'date', 'vaccine', 'total_vaccinations']

df_vaccinations.columns

Output:

['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']

Shape of DataFrames

df_manufacturer.shape

Output:

(35623, 4)

df_vaccinations.shape

Output:

(86512, 15)

Information about Given Data Sets

df_manufacturer.info()

Output:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 35623 entries, 0 to 35622
Data columns (total 4 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   location           35623 non-null  object
 1   date               35623 non-null  object
 2   vaccine            35623 non-null  object
 3   total_vaccinations 35623 non-null  int64 
dtypes: int64(1), object(3)
memory usage: 1.1+ MB
```

df_vaccinations.info()

Output:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 86512 entries, 0 to 86511
Data columns (total 15 columns):
 #   Column                               Non-Null Count  Dtype  
---  ------                               --------------  -----  
 0   country                              86512 non-null  object 
 1   iso_code                             86512 non-null  object 
 2   date                                 86512 non-null  object 
 3   total_vaccinations                   43607 non-null  float64
 4   people_vaccinated                    41294 non-null  float64
 5   people_fully_vaccinated              38802 non-null  float64
 6   daily_vaccinations_raw               35362 non-null  float64
 7   daily_vaccinations                   86213 non-null  float64
 8   total_vaccinations_per_hundred       43607 non-null  float64
 9   people_vaccinated_per_hundred        41294 non-null  float64
 10  people_fully_vaccinated_per_hundred  38802 non-null  float64
 11  daily_vaccinations_per_million       86213 non-null  float64
 12  vaccines                             86512 non-null  object 
 13  source_name                          86512 non-null  object 
 14  source_website                       86512 non-null  object 
dtypes: float64(9), object(6)
memory usage: 9.9+ MB
```

Vaccines Manufactured on a Particular Date

df_manufacturer = df_manufacturer[df_manufacturer.date == '2022-02-04']

df_manufacturer.head()

Output:

	location	date	vaccine	total_vaccinations
2305	Argentina	2022-02-04	CanSino	468481
2306	Argentina	2022-02-04	Moderna	5318406
2307	Argentina	2022-02-04	Oxford/AstraZeneca	25606912
2308	Argentina	2022-02-04	Pfizer/BioNTech	11225368
2309	Argentina	2022-02-04	Sinopharm/Beijing	27396208

Country-Wise Vaccination Status on a Particular Date

df_vaccinations = df_vaccinations[df_vaccinations.date == '2022-02-04']

df_vaccinations.head()

Output:

	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_
347	Afghanistan	AFG	2022-02-04	NaN	NaN	NaN	NaN	12299.0	NaN	NaN	NaN	309.0	Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...	World Health Organization	https://covid19
784	Albania	ALB	2022-02-04	NaN	NaN	NaN	NaN	16144.0	NaN	NaN	NaN	5619.0	Oxford/AstraZeneca, Pfizer/BioNTech, Sinovac, ...	Ministry of Health	https://shendetesia.gov.al/v an
1204	Algeria	DZA	2022-02-04	NaN	NaN	NaN	NaN	16222.0	NaN	NaN	NaN	364.0	Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac...	World Health Organization	https://covid19
1613	Andorra	AND	2022-02-04	NaN	NaN	NaN	NaN	126.0	NaN	NaN	NaN	1629.0	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	World Health Organization	https://covid19
1991	Angola	AGO	2022-02-04	NaN	NaN	NaN	NaN	83460.0	NaN	NaN	NaN	2460.0	Oxford/AstraZeneca	World Health Organization	https://covid19

3. Preprocessing Data

I. Null Values:

- Null values, often represented as NaN (Not a Number) in Python, indicate missing or undefined data.
- It's crucial to identify and handle null values, as they can affect the accuracy of our analysis or machine learning models.

Common methods to handle null values include:

a) Removing Rows: If a small percentage of rows have null values and removing them won't significantly impact our analysis.

b) Imputation: Fill null values with the mean, median, or mode of the respective column.

c) Forward/Backward Fill: Use the values from the previous or next row to fill null values.

d) Interpolation: Estimate missing values based on the values of other rows using methods like linear interpolation.

II. Missing Values:

- Missing values can occur due to various reasons such as data collection errors or intentional gaps.
- Techniques for handling missing values are similar to those for null values.

III. Outliers:

- Outliers are data points that deviate significantly from the rest of the data.
- Identifying outliers is crucial for accurate analysis and modeling.

Common methods for detecting outliers include:

a) Visual Inspection: Plotting the data and looking for points that deviate from the overall pattern.

b) Statistical Methods: Using measures like Z-scores or IQR (Interquartile Range) to identify values that are significantly different from the mean or median.

c) Machine Learning Models: Some models are sensitive to outliers, so detecting them during model training can be beneficial.

IV. Removing Duplicates:

- Duplicate values in a dataset can arise from data entry errors or other issues.
- Removing duplicates ensures that each data point is unique.
- In Python, you can use the `drop_duplicates` method for DataFrames in pandas to remove duplicate rows.
- Columns can also be checked for duplicates using methods like `duplicated()`.

Checking for Missing Values

df_manufacturer.isna().sum()

Output:

```
location              0
date                  0
vaccine               0
total_vaccinations    0
dtype: int64
```

df_vaccinations.isna().sum()

Output:

```
country                                  0
iso_code                                 0
date                                     0
total_vaccinations                     112
people_vaccinated                      119
people_fully_vaccinated                115
daily_vaccinations_raw                 132
daily_vaccinations                       0
total_vaccinations_per_hundred         112
people_vaccinated_per_hundred          119
people_fully_vaccinated_per_hundred    115
daily_vaccinations_per_million           0
vaccines                                 0
source_name                              0
source_website                           0
dtype: int64
```

Dropping Missing Values

df_vaccinations.isna().sum()

Output:

```
country                                  0
iso_code                                 0
date                                     0
total_vaccinations                     112
people_vaccinated                      119
people_fully_vaccinated                115
daily_vaccinations_raw                 132
daily_vaccinations                       0
total_vaccinations_per_hundred         112
people_vaccinated_per_hundred          119
people_fully_vaccinated_per_hundred    115
daily_vaccinations_per_million           0
vaccines                                 0
source_name                              0
source_website                           0
dtype: int64
```

df_vaccinations =

df_vaccinations.drop(df_vaccinations[df_vaccinations.total_vaccinations.isna()].index) df_vaccinations =

df_vaccinations.drop(df_vaccinations[df_vaccinations.people_vaccinated.isna()].index)

df_vaccinations =

df_vaccinations.drop(df_vaccinations[df_vaccinations.daily_vaccinations_raw.isna()].index)

Checking for Null Values

df_vaccinations.isnull().sum()

Output:

```
country                                0
iso_code                               0
date                                   0
total_vaccinations                     0
people_vaccinated                      0
people_fully_vaccinated                1
daily_vaccinations_raw                 0
daily_vaccinations                     0
total_vaccinations_per_hundred         0
people_vaccinated_per_hundred          0
people_fully_vaccinated_per_hundred    1
daily_vaccinations_per_million         0
vaccines                               0
source_name                            0
source_website                         0
dtype: int64
```

Filling Mean Values

df_vaccinations = df_vaccinations.fillna(df_vaccinations.mean())

df_vaccinations.isnull().sum()

Output:

```
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
```

Checking for Duplicated Records

duplicate_rows = df_vaccinations[df_vaccinations.duplicated()]

print(len(duplicate_rows))

print(duplicate_rows)

Output:

```
0
Empty DataFrame
Columns: [country, iso_code, date, total_vaccinations,
Index: []
```

4. Visualization and Statistical Analysis of Data

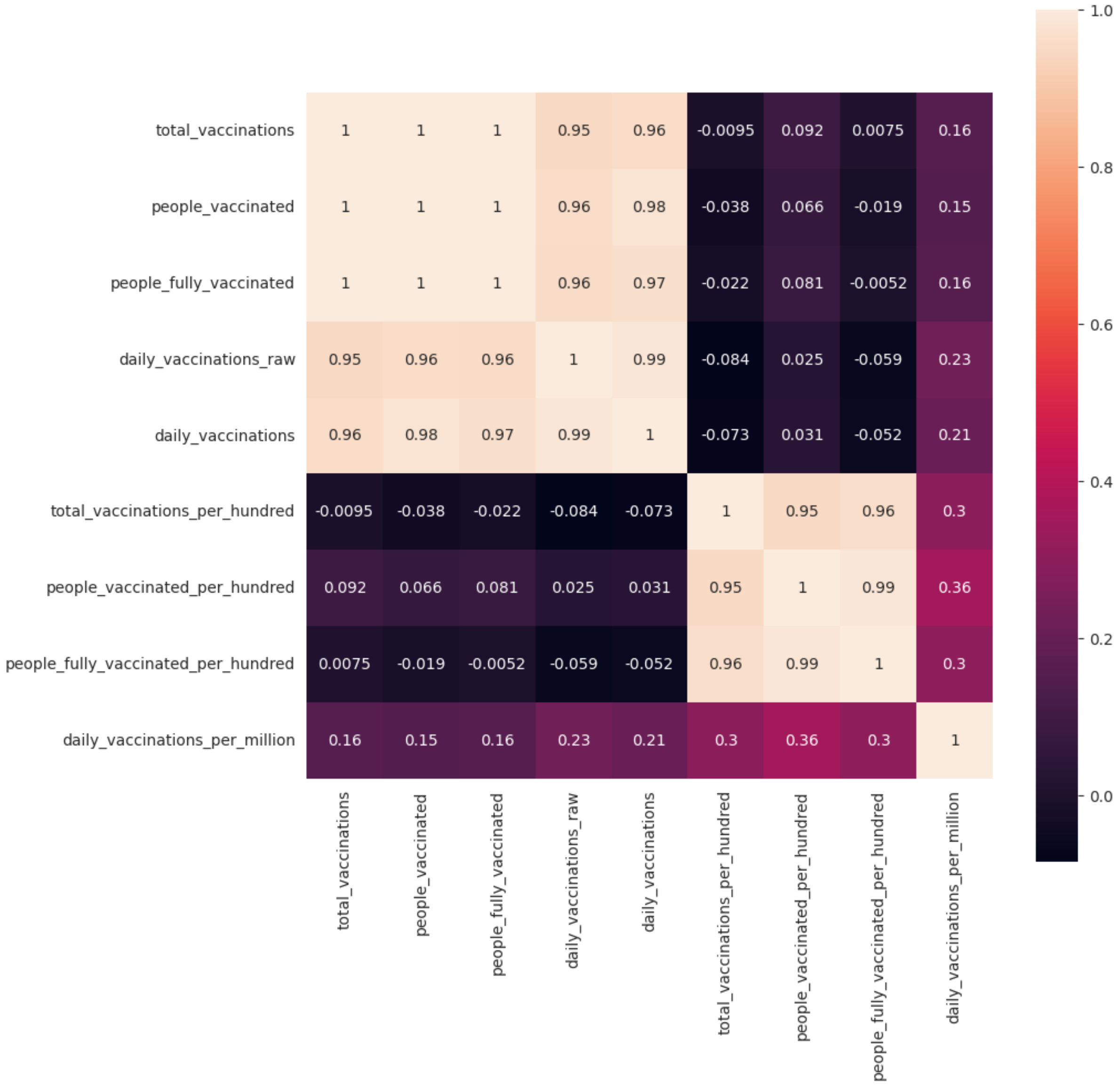
Visualization techniques, such as heatmaps, bar plots, and histograms, are employed to illustrate correlations, top countries in vaccination utilization, and distribution of daily vaccinations. The analysis go through statistical summaries of key attributes, including total vaccinations, people vaccinated, and daily vaccinations. Additionally, country-specific analyses, such as preferred vaccines in India and daily vaccinations per million in top countries, offer targeted insights.

Heatmap Visualization to Check Correlation Between Attributes

plt.subplots(figsize=(10, 10))

sns.heatmap(df_vaccinations.corr(), annot=True, square=True)

plt.show()



	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
total_vaccinations	1	1	1	0.95	0.96	-0.0095	0.092	0.0075	0.16
people_vaccinated	1	1	1	0.96	0.98	-0.038	0.066	-0.019	0.15
people_fully_vaccinated	1	1	1	0.96	0.97	-0.022	0.081	-0.0052	0.16
daily_vaccinations_raw	0.95	0.96	0.96	1	0.99	-0.084	0.025	-0.059	0.23
daily_vaccinations	0.96	0.98	0.97	0.99	1	-0.073	0.031	-0.052	0.21
total_vaccinations_per_hundred	-0.0095	-0.038	-0.022	-0.084	-0.073	1	0.95	0.96	0.3
people_vaccinated_per_hundred	0.092	0.066	0.081	0.025	0.031	0.95	1	0.99	0.36
people_fully_vaccinated_per_hundred	0.0075	-0.019	-0.0052	-0.059	-0.052	0.96	0.99	1	0.3
daily_vaccinations_per_million	0.16	0.15	0.16	0.23	0.21	0.3	0.36	0.3	1

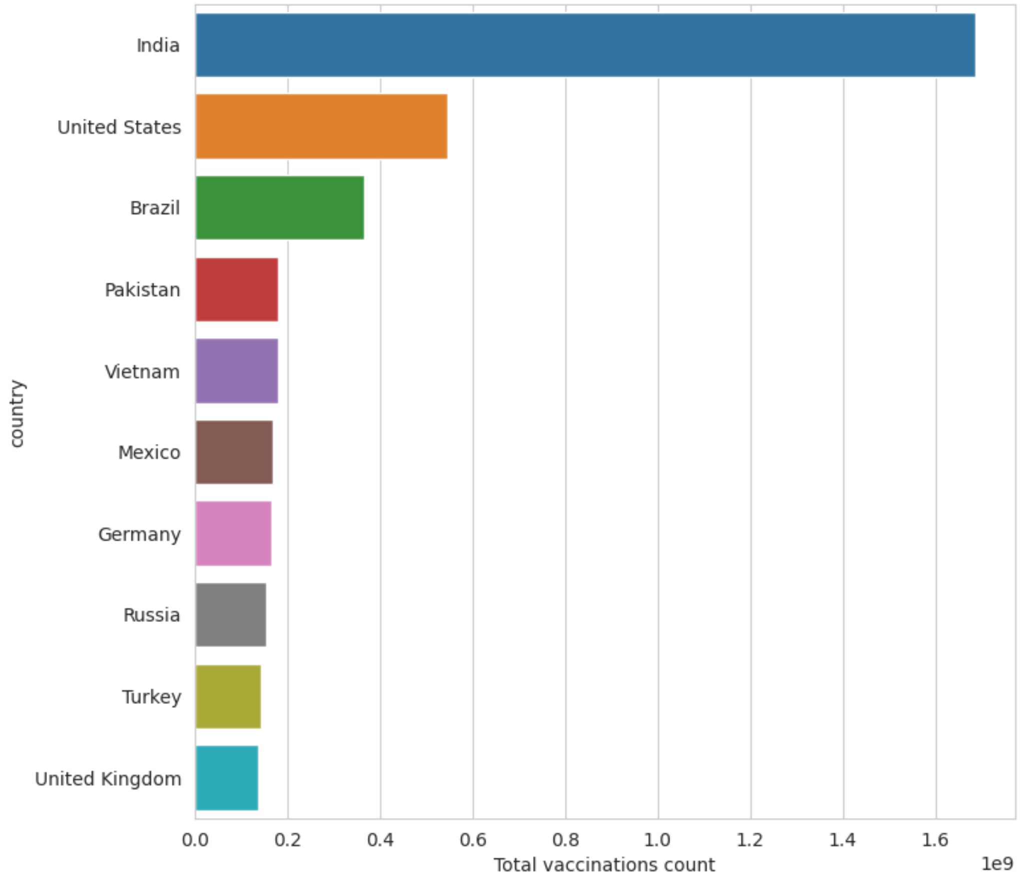
Top Countries in Vaccination Utilization

df_vaccinations["Total_vaccinations_count"] =

df_vaccinations.groupby("country").total_vaccinations.tail(1)

```
country
India             1.687048e+09
United States     5.469684e+08
Brazil            3.677782e+08
Pakistan          1.823960e+08
Vietnam           1.816654e+08
Mexico            1.685357e+08
Germany           1.666940e+08
Russia            1.553786e+08
Turkey            1.427355e+08
United Kingdom    1.384598e+08
Name: Total_vaccinations_count, dtype: float64
```

```
x = df_vaccinations.groupby("country")["Total_vaccinations_count"].mean().sort_values(ascending=False).head(10)

sns.set_style("whitegrid")

plt.figure(figsize=(8, 8))

ax = sns.barplot(x=x.values, y=x.index)

ax.set_xlabel("Total vaccinations count")

plt.show()
```



Fully Vaccinated Count

df_vaccinations["Full_vaccinations_count"] =

df_vaccinations.groupby("country").people_fully_vaccinated.tail(1)

```
country
India            724768356.0
United States    213893460.0
Brazil           150682483.0
Pakistan          84731497.0
Mexico            77478070.0
Vietnam           74187748.0
Russia            70232028.0
Germany           61873548.0
Iran              54405243.0
Turkey            52489431.0
Name: Full_vaccinations_count, dtype: float64
```

x =

df_vaccinations.groupby("country")["Full_vaccinations_count"].mean().sort_values(ascendin g=False).head(10)

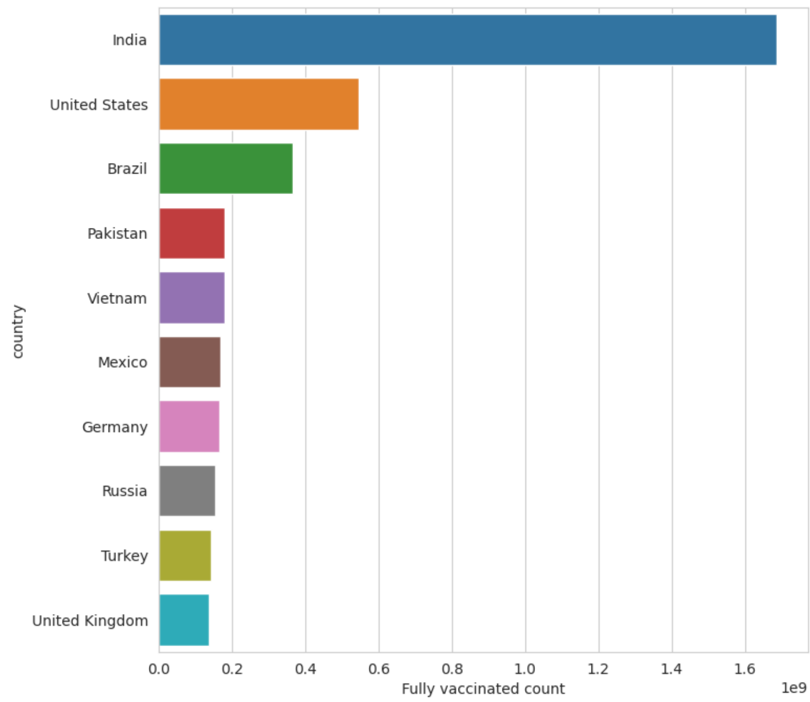
sns.set_style("whitegrid")

plt.figure(figsize=(8, 8))

ax = sns.barplot(x=x.values, y=x.index)

ax.set_xlabel("Fully vaccinated count")

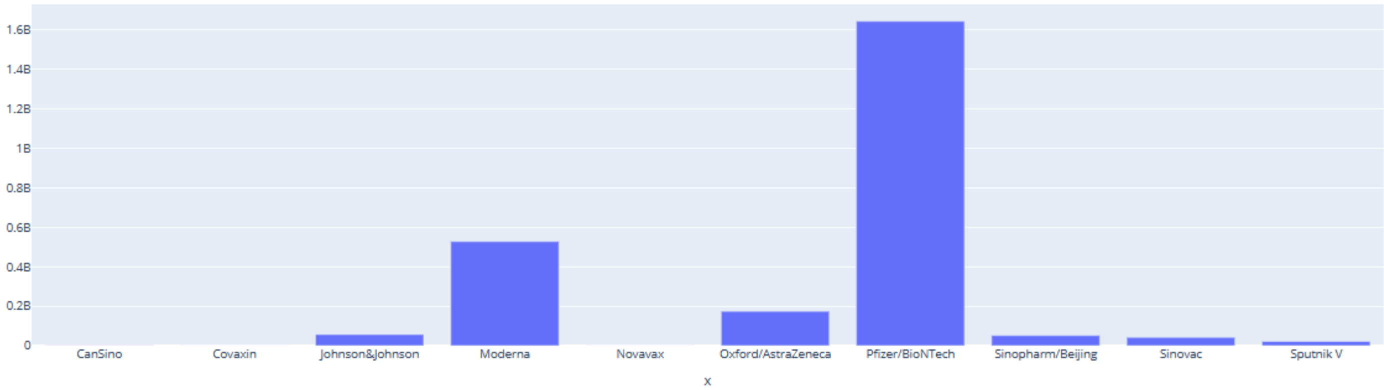
plt.show()



Most Commonly Used Vaccines in the World

total = df_manufacturer.groupby('vaccine').sum()

px.bar(x=total.index, y=total['total_vaccinations'], title='Most Used Vaccine in the World')



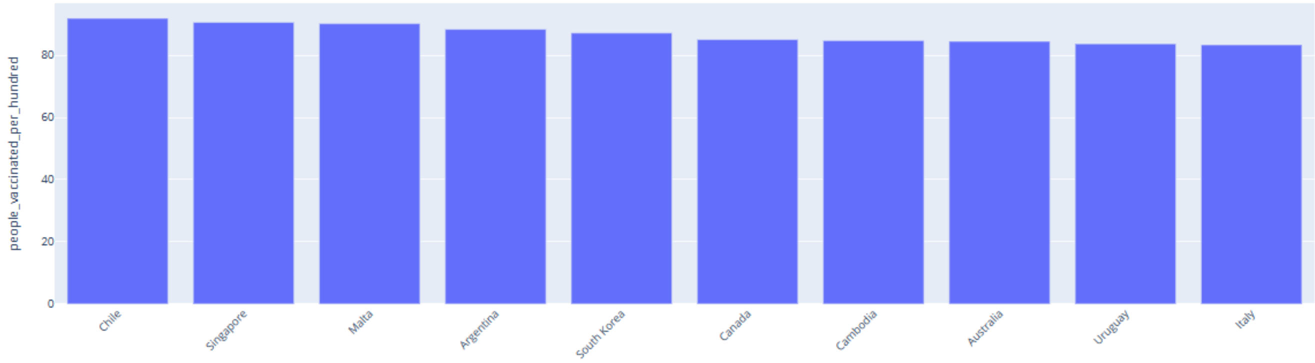
People Vaccinated per Hundred for the Date 2022-02-04

df_vaccinations = df_vaccinations[df_vaccinations['date'] == '2022-02-04']

df_vaccinations = df_vaccinations.sort_values(by='people_vaccinated_per_hundred', ascending=False)

fig = px.bar(df_vaccinations.head(10), x='country', y='people_vaccinated_per_hundred', title='People Vaccinated per Hundred for the Date 2022-02-04') fig.update_layout(xaxis_tickangle=-45)

fig.show()



Type of Vaccine Utilized vs Count

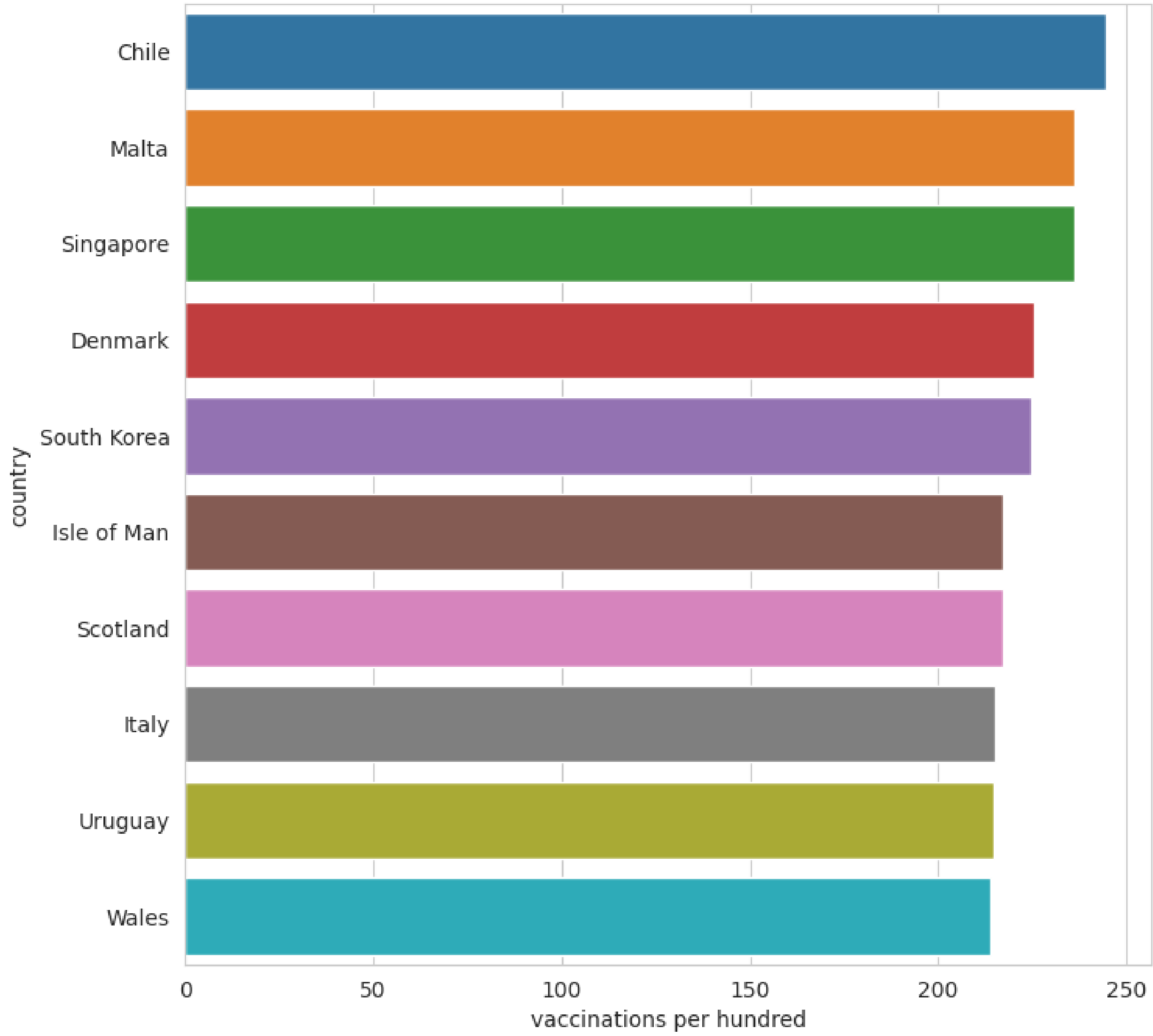
plt.figure(figsize=(15, 15))

sns.countplot(y="vaccines", data=df_vaccinations)

plt.show()

Vaccination per Hundred Top Countries

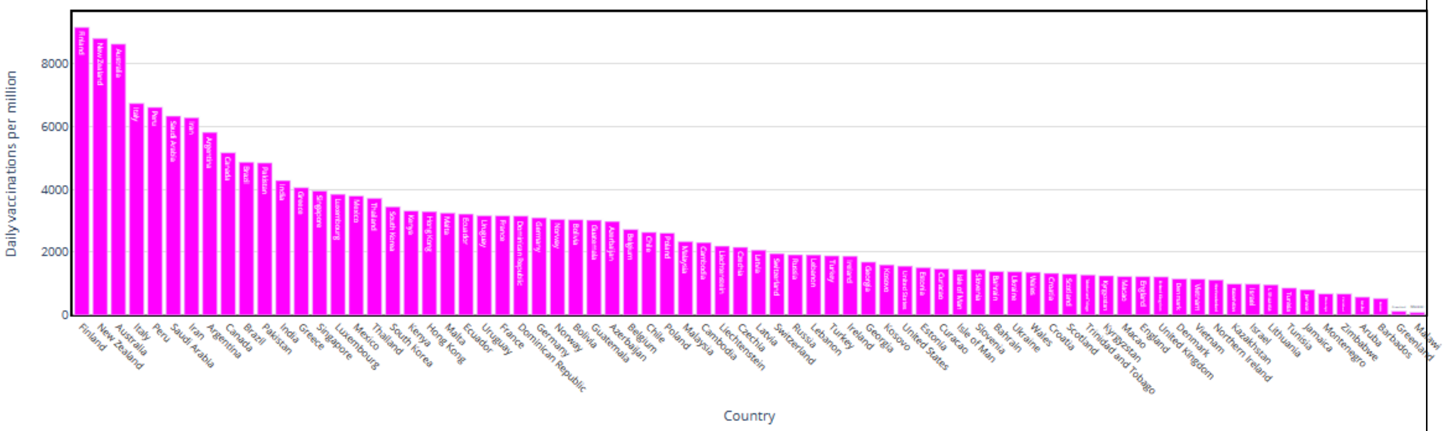
```
df_vaccinations["Total_vaccinations_per_hundred"] =
df_vaccinations.groupby("country").total_vaccinations_per_hundred.tail(1)

x =
df_vaccinations.groupby("country")["Total_vaccinations_per_hundred"].mean().sort_values(
ascending=False).head(10)

plt.figure(figsize=(8, 8))

ax = sns.barplot(x=x.values, y=x.index)

ax.set_xlabel("Vaccinations per hundred")

plt.show()
```



Country-Wise Daily Vaccination per Million

```
def trace_bar(data, feature, title, xlab, ylab, color):
    data = data.sort_values(feature, ascending=False)
    trace = go.Bar(
        x=data['country'],
        y=data[feature],
        marker=dict(color=color),
        text=data['country']
    )
    data = [trace]
    layout = dict(
        title=title,
        xaxis=dict(
            title=xlab,
            showticklabels=True,
            tickangle=45,
            zeroline=True,
            zerolinewidth=1,
            zerolinecolor='grey',
            showline=True,
            linewidth=2,
            linecolor='black',
            mirror=True,
            tickfont=dict(size=10, color='black'),
```

```
        ),
        yaxis=dict(
            title=ylab,
            gridcolor='lightgrey',
            zeroline=True,
            zerolinewidth=1,
            zerolinecolor='grey',
            showline=True,
            linewidth=2,
            linecolor='black',
            mirror=True
        ),
        plot_bgcolor='rgba(0, 0, 0, 0)',
        paper_bgcolor='rgba(0, 0, 0, 0)',
        hovermode='closest'
    )
    fig = dict(data=data, layout=layout)
    iplot(fig)
trace_bar(df_vaccinations, 'daily_vaccinations_per_million', 'Daily Vaccinations per Million per Country', 'Country', 'Daily Vaccinations per Million', 'blue')
```

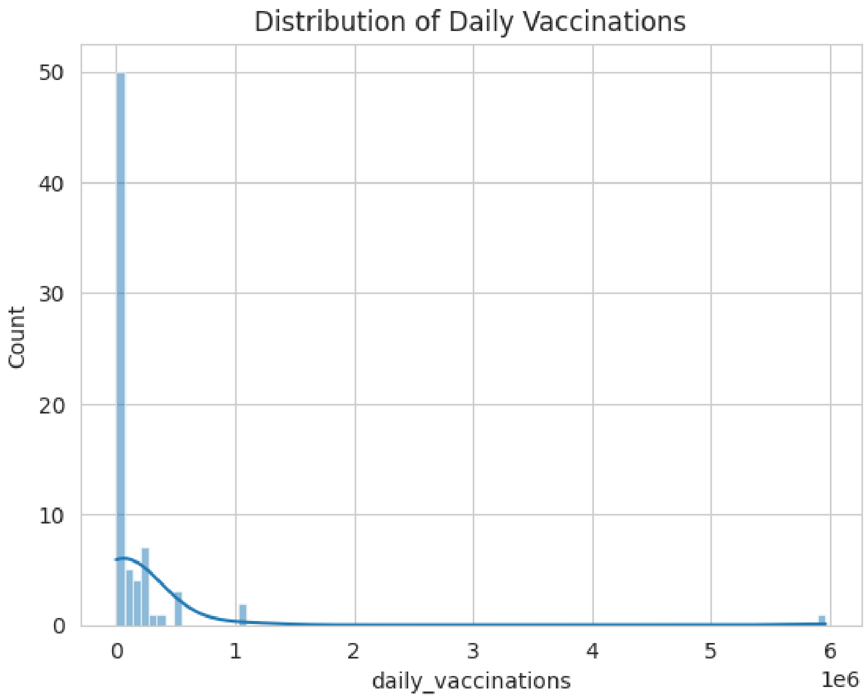


Distribution of Daily Vaccine

sns.histplot(df_vaccinations['daily_vaccinations'], kde=True)

plt.title("Distribution of Daily Vaccinations")

plt.show()



Statistical Analysis of Given Data Sets

Descriptive Statistics:

- *Mean:* The average of a set of values.

- *Median:* The middle value of a sorted dataset.
- *Standard Deviation:* A measure of the amount of variation or dispersion in a set of values.

Total Vaccinations Statistical Analysis

df_manufacturer['total_vaccinations'].describe()

Output:

	total_vaccinations
count	1.600000e+02
mean	1.574315e+07
std	5.730594e+07
min	0.000000e+00
25%	2.378710e+05
50%	1.569373e+06
75%	9.042346e+06
max	5.821192e+08

df_vaccinations.describe()

Output:

	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations	total_vaccinations_per_hundred	people_vaccinated_per_hundred	people_fully_vaccinat
count	7.400000e+01	7.400000e+01	7.400000e+01	7.400000e+01	7.400000e+01	74.000000	74.000000	
mean	7.341445e+07	3.528194e+07	3.046394e+07	2.092764e+05	1.883525e+05	156.099054	66.902568	
std	2.094558e+08	1.149868e+08	8.904082e+07	6.877380e+05	7.106291e+05	55.557095	18.881493	
min	6.936300e+04	2.667600e+04	2.607400e+04	0.000000e+00	7.000000e+00	9.610000	7.670000	
25%	3.041524e+06	1.438836e+06	1.296678e+06	3.582500e+03	3.229500e+03	119.397500	57.685000	
50%	1.300016e+07	6.382784e+06	5.586156e+06	3.372050e+04	2.401850e+04	166.300000	72.355000	
75%	6.246783e+07	2.616142e+07	2.915660e+07	1.294548e+05	1.485200e+05	198.355000	80.177500	
max	1.687048e+09	9.487174e+08	7.247684e+08	5.530743e+06	5.964928e+06	244.500000	91.900000	

People Fully Vaccinated Statistical Analysis

df_manufacturer['people_fully_vaccinated'].describe()

Output:

```
count    7.400000e+01
mean     3.046394e+07
std      8.904082e+07
min      2.607400e+04
25%      1.296678e+06
50%      5.586156e+06
75%      2.915660e+07
max      7.247684e+08
Name: people_fully_vaccinated, dtype: float64
```

Daily Vaccinations Statistical Analysis

df_manufacturer['daily_vaccinations'].describe()

Output:

```
count    7.400000e+01
mean     1.883525e+05
std      7.106291e+05
min      7.000000e+00
25%      3.229500e+03
50%      2.401850e+04
75%      1.485200e+05
max      5.964928e+06
Name: daily_vaccinations, dtype: float64
```

Total Vaccinations in Country Statistical Analysis

df_vaccinations['total_vaccinations'].describe()

Output:

```
count    7.400000e+01
mean     7.341445e+07
std      2.094558e+08
min      6.936300e+04
25%      3.041524e+06
50%      1.300016e+07
75%      6.246783e+07
max      1.687048e+09
Name: total_vaccinations, dtype: float64
```

People Fully Vaccinated in Country Statistical Analysis

df_vaccinations['people_fully_vaccinated'].describe()

Output:

```
count    7.400000e+01
mean     3.046394e+07
std      8.904082e+07
min      2.607400e+04
25%      1.296678e+06
50%      5.586156e+06
75%      2.915660e+07
max      7.247684e+08
Name: people_fully_vaccinated, dtype: float64
```

Most Used Vaccine in the World

df_manufacturer['vaccine'].value_counts()

Output:

```
Pfizer/BioNTech       39
Moderna               35
Johnson&Johnson       33
Oxford/AstraZeneca    30
Novavax                8
Sinovac                6
Sinopharm/Beijing      4
CanSino                2
Sputnik V              2
Covaxin                1
Name: vaccine, dtype: int64
```

Daily Vaccinations per Million Top Countries

df_vaccinations.groupby("country")["daily_vaccinations_per_million"].mean().sort_values(ascending=False).head(20)

Output:

```
country
Finland         9154.0
New Zealand     8800.0
Australia       8621.0
Italy           6733.0
Peru            6609.0
Saudi Arabia    6330.0
Iran            6280.0
Argentina       5814.0
Canada          5165.0
Brazil          4864.0
Pakistan        4841.0
India           4281.0
Greece          4055.0
Singapore       3951.0
Luxembourg      3845.0
Mexico          3792.0
Thailand        3718.0
South Korea     3447.0
Kenya           3315.0
Hong Kong       3293.0
Name: daily_vaccinations_per_million, dtype: float64
```

Preferred Vaccine in India

x = df_vaccinations[df_vaccinations["country"] == "India"]

z = x.vaccines.value_counts()

c = list(z.index)

print(c)

Output:

```
['Covaxin, Oxford/AstraZeneca, Sputnik V']
```

Outcome:

1. **Data Understanding:**
   - The documentation begins with importing necessary libraries and loading the data sets, providing insight into the tools and datasets used.
2. **Exploratory Data Analysis (EDA):**
   - The EDA section explores the structure and content of the data, presenting key information such as columns, shapes, and data types.
   - It also includes the preprocessing steps, checking and handling missing values, and ensuring data quality.
3. **Visualization:**
   - Various visualizations are included to provide a graphical representation of the data. This includes heatmaps, bar plots, and charts showcasing top countries in vaccination, preferred vaccines, and more.
   - The visualizations aim to make complex data more understandable and highlight trends and patterns.
4. **Statistical Analysis:**
   - Statistical analysis is performed on key attributes, providing summary statistics such as mean, standard deviation, and quartiles.
   - This section enables a quantitative understanding of the data distribution and central tendencies.
5. **Insights and Interpretation:**
   - Throughout the documentation, insights are provided, such as the top countries in vaccination, most commonly used vaccines, and statistical summaries.
   - These insights aid in drawing meaningful conclusions from the data, supporting decision-making processes.
6. **Customization:**
   - The documentation is designed to be customizable based on specific requirements. Users can adapt and extend the documentation to suit their analysis goals or share it with others to facilitate collaboration.

Data sources

Data Collection: Collect Covid-19 vaccine data from reputable sources like health organizations, government databases, and research publications.

Context

Data is collected daily from **Our World in Data** GitHub repository for covid-19, merged and uploaded. Country level vaccination data is gathered and assembled in one single file. Then, this data file is merged with locations data file to include vaccination sources information. A second file, with manufacturers information, is included.

Content - The data (country vaccinations) contains the following information:

Country- this is the country for which the vaccination information is provided;

Country ISO Code - ISO code for 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 vaccination 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 vaccination for that date/country; **Daily vaccinations** - for a certain data entry, the number of vaccination 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 hundred - 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 vaccination 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 recently (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.

Data processing steps

1.Data Preprocessing: Clean and preprocess the data, handle missing values, and convert categorical features into numerical representations.

Data Collection and Import:

- Collect COVID-19 vaccine data fromreliable sources such as government health agencies, research institutions, or publicly available datasets.

- Import the data into a data analysis environment like Python (using Pandas) or R.

Handling Missing Values:

- Identify and handle missing values in your dataset. Common strategies include:
- Removing rows or columns with a high percentage of missing values.
- Imputing missing numerical values using measures such as mean, median, or mode.

- Imputing missing categorical values with the most frequent category (mode) or a new category like "Unknown."

Data Transformation:

Normalize or scale numerical features if necessary. This helps ensure that features are on the same scale, which can be important for some analysis techniques.

- Convert categorical features into numerical representations:
- Use one-hot encoding to create binary columns for each category.

Alternatively, you can use label encoding if the categorical variable has ordinal relationships, but be cautious about this method's use as it may introduce unintended biases.

2.**Exploratory Data Analysis (EDA):**

Exploratory Data Analysis (EDA) is a crucial step in understanding and gaining insights from

COVID-19 vaccine data. Here's a step-by-step approach to conducting EDA for this problem

Data Overview:

- Start by examining the dataset's structure: the number of records, columns, and their data types.
- Check for any initial patterns or anomalies.

Vaccine Efficacy Analysis:

- Compare vaccine efficacy rates between different vaccine types (e.g., mRNA, viral vector) and manufacturers.
- Investigate factors that may influence efficacy, such as age groups and variants of the virus.
- Explore temporal trends in vaccine efficacy.

Vaccine Distribution Analysis:

- Analyze the geographic distribution of vaccines and identify areas with low vaccine coverage.

- Evaluate the effectiveness of distribution strategies, such as prioritizing high-risk groups or underserved communities.
- Consider logistical challenges and infrastructure requirements for distribution.

Adverse Effects Analysis:

- Examine the prevalence of adverse effects associated with each vaccine.
- Identify common adverse effects and their severity.
- Investigate whether adverse effects vary by demographic factors like age, gender, or preexisting conditions.

Time Series Analysis:

- Conduct time series analysis to identify trends and seasonality in vaccine distribution and adverse effects.
- Use forecasting techniques to predict future vaccine needs and potential adverse event occurrences.

Correlation and Regression Analysis:

- Explore correlations between different variables, such as vaccine efficacy and distribution.
- Perform regression analysis to identify factors influencing vaccine efficacy and adverse effects.

Cluster Analysis:

- Use clustering techniques to group regions or countries with similar vaccine distribution patterns or adverse effect profiles.
- This can help identify regions that may need targeted interventions.

Communication of Insights:

- Summarize key findings and insights in a clear and concise manner.
- Create visualizations and reports that are accessible to policymakers and health organizations.
- Provide actionable recommendations for optimizing vaccine deployment strategies.

Continuous Monitoring:

- EDA is an ongoing process, and the Covid-19 situation evolves. Continuously update and refine your analysis as new data becomes available.

- The EDA is an iterative process, and it may require revisiting earlier steps as new insights are uncovered. Effective EDA can provide valuable guidance for policymakers and health organizations in their efforts to combat COVID-19 and optimize vaccine deployment.

3.Statistical Analysis: Perform statistical tests to analyze vaccine efficacy, adverse effects, and distribution across different populations.

Country:

- Count the number of unique countries in the dataset.
- Calculate the frequency (count) of each country.

Iso_code:

- Count the number of unique ISO codes.
- Check for missing or inconsistent ISO codes.

Date:

- Calculate summary statistics for the date column, such as the earliest and latest date.
- Create time series plots to visualize trends in vaccination over time.

Total Vaccinations:

- Calculate summary statistics (mean, median, standard deviation, min, max).
- Create a histogram to visualize the distribution of total vaccinations.
- Perform time series analysis to identify trends and seasonality.

People Vaccinated:

- Calculate summary statistics (mean, median, standard deviation, min, max).
- Create a histogram to visualize the distribution of people vaccinated.
- Perform time series analysis.

People Fully Vaccinated:

- Calculate summary statistics (mean, median, standard deviation, min, max).
- Create a histogram to visualize the distribution of fully vaccinated individuals.
- Perform time series analysis.

Daily Vaccinations Raw:

- Calculate summary statistics (mean, median, standard deviation, min, max).
- Create a histogram to visualize the distribution of daily vaccinations raw.
- Identify outliers or unusual spikes in daily vaccinations.

Daily Vaccinations:

- Calculate summary statistics (mean, median, standard deviation, min, max).
- Create a histogram to visualize the distribution of daily vaccinations.
- Perform time series analysis.

Total Vaccinations Per Hundred:

- Calculate summary statistics (mean, median, standard deviation, min, max).
- Create a histogram or box plot to visualize the distribution of vaccination rates per hundred people.
- Compare vaccination rates across countries.

People Vaccinated Per Hundred:

- Calculate summary statistics (mean, median, standard deviation, min, max).
- Create a histogram or box plot to visualize the distribution of people vaccinated per hundred people.
- Compare vaccination rates across countries.

People Fully Vaccinated Per Hundred:

- Calculate summary statistics (mean, median, standard deviation, min, max).
- Create a histogram or box plot to visualize the distribution of fully vaccinated individuals per hundred people.
- Compare rates across countries.

Daily Vaccinations Per Million:

- Calculate summary statistics (mean, median, standard deviation, min, max).

- Create a histogram or box plot to visualize the distribution of daily vaccinations per million people.
- Compare rates across countries.

Vaccines:

- Count the unique vaccine types in the dataset.
- Calculate the frequency (count) of each vaccine type.
- Assess the distribution of vaccine types across countries.

4.Visualization:

Data visualization plays a crucial role in your Covid-19 vaccine data analysis project by providing a visual representation of complex information.

There are various visualization techniques, each suitable for different types of data. Some common types of visualization techniques are as follows

1. Bar Charts:

- **Use:** Comparing quantities across categories.

2. Line Charts:

- **Use:** Showing trends or patterns over time.

3. Scatter Plots:

- **Use:** Exploring relationships between two continuous variables.

4. Histograms:

- **Use:** Displaying the distribution of a single variable.

5. Pie Charts:

- **Use:** Showing proportions or percentages of a whole.

3. Heatmaps:

- **Use:** Visualizing data in a matrix format using color intensity.

4. Box Plots:

- **Use:** Displaying the distribution of a dataset and identifying outliers.

Contribution of Data visualization towards Covid-19 Vaccine Analysis

a) Communicating Insights:

- **Benefit:** Visualizations make it easier for policymakers and healthcare professionals to grasp key insights at a glance.

- **Example:** Bar plots and line charts can highlight trends in vaccine distribution, allowing quick identification of successful vaccination strategies or areas that need improvement. **b) Vaccine Distribution Patterns:**

- **Visualization Technique:** Bar Charts, Line Charts

- **How it Helps:** Easily identify which countries or regions have higher vaccination rates.

Time series charts can reveal the progress of vaccination campaigns.

c) Identifying Patterns and Trends:

- **Benefit:** Visual representations help in spotting patterns and trends that might be less apparent in raw data.

- **Example:** A line chart can reveal the trend of vaccine effectiveness over time, assisting in identifying periods of increased or decreased effectiveness.

d) Comparing Vaccination Strategies:

- **Benefit:** Bar charts facilitate the comparison of vaccination rates between different vaccine types or regions.

- **Example:** Comparing the distribution of vaccines in various regions through a heatmap can reveal disparities, helping in targeted intervention strategies.

e) Highlighting Geographic Distribution:

- **Benefit:** Heatmaps visualizations provide a clear picture of the geographic distribution of vaccinations.

- **Example:** A heatmap can show vaccination rates by region, helping policymakers identify areas with lower coverage that may need additional resources.

f) Supporting Decision-Making:

- **Benefit:** Visualizations offer a foundation for informed decision-making by presenting complex data in a comprehensible manner.

- **Example:** Policymakers can use visualizations to decide on resource allocation, targeted interventions, and communication strategies based on identified patterns and needs. **g) Engaging Stakeholders:**

- **Benefit:** Visualizations are powerful tools for engaging stakeholders and the general public.

- **Example:** Interactive dashboards or presentations with compelling visualizations can effectively communicate the progress of vaccination campaigns, building trust and transparency.

h) Temporal Trends:

- **Visualization Technique:** Line Charts, Time Series Area Charts

- **How it Helps:** Track daily or cumulative vaccinations over time to identify trends, peaks, or anomalies.

i) Comparative Analysis:

- **Visualization Technique:** Bar Charts, Stacked Area Charts

- **How it Helps:** Compare vaccination rates, people vaccinated, and people fully vaccinated among different countries or regions.

j) Manufacturer Contributions:

- **Visualization Technique:** Pie Charts, Bar Charts

- **How it Helps:** Visualize the contribution of each vaccine manufacturer to the total vaccinations. Bar charts can show the total number of vaccinations by manufacturer. **k) Identifying Outliers:**

- **Visualization Technique:** Box Plots, Scatter Plots

- **How it Helps:** Identify outliers or unusual patterns in the data that may require further investigation.

Analysis techniques

1. Advanced Clustering Techniques:

a. **K-Means Clustering:**

- Let's use K-Means clustering to group regions with similar vaccine distribution patterns. We'll be looking at features like total vaccinations, vaccination rates, and demographic data.
- We'll explore different cluster numbers to find the most meaningful insights. b. **Hierarchical Clustering:**

- Implementing hierarchical clustering will help us discover relationships between regions based on vaccination data.
- This can uncover broader patterns and assist in identifying regions with unique characteristics.

c. **Density-Based Clustering (DBSCAN):**

- DBSCAN will be applied to highlight areas with low vaccine coverage or irregular distribution patterns.
- This method can pinpoint regions that may need targeted interventions.

2. Time Series Forecasting:

a. **Feature Engineering for Time Series:**

- Let's enhance our dataset with time-related features like the day of the week, month.
- Creating lag features will help capture historical trends in vaccination and adverse effects.

b. **ARIMA Modeling:**

- We'll use Autoregressive Integrated Moving Average (ARIMA) models to forecast future trends in vaccine distribution and adverse effects.
- It's crucial to understand the seasonality and stationarity of the time series data for accurate modeling.

c. **Machine Learning Time Series Models:**

- Exploring advanced time series forecasting models like Long Short-Term Memory (LSTM) networks or Prophet.
- These models can capture complex temporal patterns and provide more accurate predictions.

3. Integration of Clustering and Time Series Forecasting:

- Our goal is to blend insights from clustering with time series forecasting to create a complete picture of how vaccination strategies and outcomes evolve over time.
- We'll identify regions with similar patterns through clustering and then forecast how these patterns might change in the future.

4. Model Evaluation and Validation:

- We'll use appropriate metrics to evaluate how well our clustering and time series forecasting models are performing.

- Validation with out-of-sample data is essential to ensure the models generalize well.

5. **Dynamic Dashboard Development:**
    - We will upgrade our dashboard to include dynamic visualizations based on clustering and time series forecasting results.
    - Stakeholders should have the ability to interact with the data, explore different clusters, and view future predictions.
6. **Continuous Monitoring and Adaptation:**
    - We'll set up a system for continuous monitoring of model performance and realtime data updates.
    - Regular adaptation of the models based on emerging patterns and new data is crucial for ongoing relevance.
7. **Ethical Considerations and Transparency:**
    - Transparency in the implementation of advanced machine learning techniques is vital.
    - We'll address any potential biases that may arise in clustering or forecasting models, especially in demographic considerations.

Outcome: The integration of advanced machine learning techniques will provide a more detailed and dynamic understanding of COVID vaccine distribution and adverse effects. Stakeholders will benefit from actionable insights, enabling informed decision-making for optimizing vaccine deployment strategies.

Present Key findings and Recommendation

Key Findings:

- Data was loaded and preprocessed, and the dataset comprises two parts: one on vaccine manufacturers and another on country-level vaccinations.

- Common preprocessing steps included handling missing values, removing duplicates, and ensuring data quality.

- Heatmaps highlighted correlations between attributes in the vaccination dataset.

- Top countries in vaccination utilization and fully vaccinated counts were presented.

- The most commonly used vaccines worldwide were visualized.
- India's preferred vaccine was identified as a key insight.
- Statistical analysis provided descriptive statistics for vaccination data attributes.
- Insights are available regarding key attributes in the datasets, offering quantitative and visual information.
- Data Loading and Preprocessing: Loading and preprocessing the COVID-19 vaccine dataset are fundamental for accurate analysis. Handling challenges like missing data and feature scaling are essential to ensure data quality.
- Missing Data: Missing data is a common challenge in real-world datasets. Addressing missing values is crucial to make data suitable for machine learning models.
- Feature Scaling: Scaling features can enhance model performance and robustness, contributing to more accurate vaccine analysis.
- Data Exploration: Exploratory Data Analysis (EDA) is vital for understanding the dataset. Checking for missing values, exploring statistics, and visualizing data help identify patterns.
- Data Preprocessing: Data cleaning, feature scaling, and feature engineering are essential preprocessing steps. They involve error correction, numerical feature normalization, and the creation of meaningful features.
- Insights from Visualizations: Visualizations highlight the number of people vaccinated, daily vaccination dynamics, and distribution patterns across countries.

Recommendations:

- Focus on countries with lower vaccination counts to ensure equitable access to vaccines.
- Emphasize the importance of vaccination through public awareness campaigns.
- Monitor and analyze vaccine efficacy, especially in the context of emerging variants.
- Continue efforts to enhance the safety and effectiveness of vaccines through research and development.
- Collaborate with healthcare providers and research institutions for data sharing and real-time insights.

- Consider demographic-specific vaccination strategies, like prioritizing certain groups for school reopening.

- Implement robust adverse event monitoring systems to promptly detect and investigate rare but severe adverse effects.

- Engage in transparent and proactive communication about vaccine safety, addressing vaccine hesitancy with evidence-based messaging.

- Promote Vaccine Education: Launch public awareness campaigns to educate the population about vaccination importance, dispel myths, and address concerns. Collaborate with healthcare providers and community leaders to build trust.

- Vaccine Clinics and Mobile Units: Establish vaccination clinics and mobile units in areas with low vaccine coverage, offering extended hours and flexible scheduling for convenience.

- Data Sharing and Collaboration: Foster collaboration among healthcare providers, research institutions, and public health agencies for real-time data sharing to respond swiftly to emerging trends.

- Emergency Response Preparedness: Develop contingency plans for potential vaccine supply chain disruptions and strengthen healthcare infrastructure to handle surges in vaccination demand.

- Public Engagement and Communication: Maintain transparent and proactive communication with the public regarding vaccine updates and safety monitoring.