

# ***PHASE-5***

## ***PROJECT-5 COVID-19 VACCINESS ANALYSIS***

### ***INTRODUCTION***

#### ***DATA COLLECTION***

##### **◆ *Research Databases***

*Acquire scientific papers, preprints, and peer-reviewed articles from reputable databases such as PubMed, IEEE Xplore, and Google Scholar. These sources offer valuable insights into vaccine development, efficacy, and safety*

##### **◆ *Clinical Trial Registries***

*Gather data from official clinical trial registries, including ClinicalTrials.gov and the WHO International Clinical Trials Registry Platform (ICTRP). This data will provide information on ongoing and completed vaccine trials, including phase, participant demographics, and outcomes.*

##### **◆ *News Media and Press Releases***

*Collect news articles and press releases from trusted news outlets, government health agencies, and pharmaceutical companies. This data source offers real-time updates on vaccine distribution, public reactions, and regulatory approvals.*

##### **◆ *Government and Health Organization Reports***

*Access official reports and publications from health authorities such as the World Health Organization (WHO), the Centers for Disease Control and Prevention (CDC), and the European Medicines Agency (EMA). These documents contain critical information on vaccine guidelines, safety assessments, and vaccination campaigns.*

## DATA PREPROCESSING

### ◆ **Handling Missing Values**

- Identify missing data in your dataset.
- Decide how to handle missing values: either by removing rows with missing data, filling them with a default value (e.g., mean, median, or mode), or using more advanced techniques like interpolation.

### ◆ **Encoding Categorical Features**

- Convert categorical variables into numerical representations that machine learning models can understand.
- For nominal variables (categories without a specific order), you can use one-hot encoding or label encoding.
- For ordinal variables (categories with a meaningful order), use ordinal encoding.

### ◆ **Feature Scaling**

- Normalize or standardize numerical features to bring them to a similar scale.
- Common techniques include Min-Max scaling (scaling to a specific range) or Z-score normalization (scaling to have mean=0 and standard deviation=1).

### ◆ **Handling Outliers**

- Identify and decide how to deal with outliers. You can choose to remove them or transform them using techniques like log transformation.

### ◆ **Data Splitting**

- Split your dataset into training, validation, and test sets to evaluate your model's performance properly.

## EXPLORATORY DATA ANALYSIS (EDA)

### ◆ **Data Overview**

*Begin the EDA by providing a summary of the dataset, including its size, number of features, and data types. Highlight any key variables of interest that will be central to the analysis.*

### ◆ **Descriptive Statistics**

*Calculate and present summary statistics for numerical features. This should include measures of central tendency (mean, median) and dispersion (standard deviation, range). For categorical variables, provide frequency counts and percentages.*

### ◆ **Data Distribution**

*Visualize the distribution of key variables using appropriate plots and charts. Histograms, box plots, and density plots can reveal the underlying distribution patterns and potential outliers.*

### ◆ **Correlation Analysis**

*Examine the relationships between variables by calculating correlation coefficients. Create correlation matrices and visualizations (e.g., heatmaps) to identify strong positive or negative correlations.*

### ◆ **Feature Importance**

*If machine learning will be applied, use techniques such as feature importance scores to assess the relevance of different features in achieving project objectives. Present the results to guide feature selection.*

### ◆ **Exploratory Visualization**

*Create visualizations that provide insights into the data. Examples include:*

*- Time Series Plots: If applicable, plot trends over time related to vaccine distribution, case numbers, or public sentiment.*

- *Word Clouds: Visualize frequently occurring words in text data to identify common themes.*

- *Scatterplots: Explore relationships between variables, such as vaccine coverage and disease incidence.*

## *Statistical Analysis*

### **◆ Hypothesis Testing**

*Formulate hypotheses based on project goals and data characteristics. Common hypotheses in COVID-19 vaccine analysis include:*

- *"COVID-19 vaccination significantly reduces the risk of infection."*
- *"Vaccine efficacy varies by vaccine type."*
- *"Vaccine distribution is associated with reduced disease incidence."*

*Select appropriate statistical tests based on the nature of the hypotheses, including t-tests, chi-square tests, ANOVA, and regression analysis.*

### **◆ Pattern Recognition**

*Use descriptive statistics, data visualization, and exploratory analysis techniques to identify patterns and trends within the dataset. Key tasks include:*

- *Central Tendency: Analyze measures of central tendency, such as mean, median, and mode, for numerical variables.*
- *Data Distribution: Examine distributions and identify skewness, kurtosis, and multimodality.*
- *Correlation Analysis: Investigate correlations between variables to identify relationships.*
- *Time Series Analysis: Perform time series analysis for trends over time, especially relevant for vaccine distribution and disease incidence data.*

### ◆ **Data Validation**

*Ensure data meets the assumptions of statistical methods used. Tasks include:*

- Normality Tests: Verify the normality of data distribution when applicable.*
- Homoscedasticity: Assess homoscedasticity, ensuring constant variance in regression analysis.*
- Multicollinearity: Identify and address multicollinearity issues in regression models.*

### ◆ **Interpretation and Conclusion**

*Interpret the statistical findings in the context of the project's objectives and hypotheses. Summarize key insights and implications. Address any limitations or assumptions made during the analysis.*

### ◆ **Reporting and Documentation**

*Document the entire statistical analysis process, including methodologies, results, and any code or scripts used. Provide clear visualizations and tables to convey findings effectively.*

## **VISUALIZATION**

### ◆ **Data Visualization**

*Data visualization involves creating graphical representations of data to facilitate understanding. Key data visualization tasks include:*

- 1. Descriptive Statistics: Present summary statistics using charts, histograms, and box plots to provide a snapshot of the data's distribution.*
- 2. Time Series Plots: Visualize trends over time, such as vaccine distribution, disease incidence, and public sentiment, using line charts or time series plots.*
- 3. Correlation Heatmaps: Create heatmaps to display correlation matrices, highlighting relationships between variables.*

4. *Geospatial Maps:* Develop geospatial maps to visualize regional variations in vaccine distribution, disease prevalence, or public sentiment.

5. *Word Clouds:* Use word clouds to visualize frequently occurring words in text data, providing insights into common themes.

6. *Scatterplots:* Illustrate relationships between variables through scatterplots, especially when investigating vaccine efficacy, public perception, or other factors.

7. *Bar and Pie Charts:* Create bar and pie charts for categorical data to display distributions and proportions.

## **INSIGHTS**

### **◆ Insight Generation**

- Summarize the most significant findings and trends identified during the analysis. Highlight insights related to vaccine efficacy, safety, distribution, public sentiment, and disease impact.

### **◆ Identifying Patterns**

- Discuss any recurring patterns or correlations uncovered in the data. Explain their implications in the context of COVID-19 vaccines.

### **◆ Public Perception**

- Analyze public sentiment trends and the factors influencing vaccine hesitancy or acceptance. Consider insights from social media data and surveys.

### **◆ Vaccine Variants**

- Assess the adaptability of existing vaccines to emerging variants of the virus and implications for future vaccination efforts.

### **◆ Hypothesis Validation**

- Review and validate hypotheses formulated during the analysis phase. Discuss the extent to which the data and analysis support or refute these hypotheses

## RECOMMENDATIONS

### ◆ **Public Health Recommendations**

- *Develop recommendations for public health authorities, healthcare providers, and policymakers. These recommendations may include vaccination strategies, communication plans, and targeted interventions.*

### ◆ **Vaccine Distribution Strategies**

- *Offer insights into optimizing vaccine distribution to ensure equitable access, especially in underserved populations or regions.*

### ◆ **Safety Monitoring**

- *Recommend strategies for ongoing safety monitoring and reporting of adverse events associated with COVID-19 vaccines.*

### ◆ **Future Research Directions**

- *Suggest areas for future research and analysis, such as the evaluation of booster shots, long-term vaccine impact, and vaccine adaptability to new variants.*

### ◆ **Actionable Insights**

- *Emphasize actionable insights that can lead to tangible improvements in vaccination campaigns, public perception, and overall COVID-19 response efforts.*

### ◆ **Communication**

- *Prepare clear and concise reports, presentations, or documents that convey the insights and recommendations effectively to various stakeholders.*

## INNOVATION

*Innovations in the analysis of COVID-19 vaccines have been crucial for monitoring their effectiveness, safety, and adaptation to emerging variants. Here are some key innovations in this area:*

**1. Real-World Data Analysis:** The use of real-world data from vaccinated populations has allowed for continuous monitoring of vaccine effectiveness and safety. Large-scale data analysis has been instrumental in identifying trends and anomalies.

**2. Variant Surveillance:** Ongoing genomic sequencing and analysis of SARS-CoV-2 variants have enabled researchers to assess how vaccines perform against different strains. This has led to adjustments in vaccine strategies and booster shots.

**3. Vaccine Efficacy Models:** Mathematical models have been developed to predict vaccine efficacy under various scenarios, helping public health officials make informed decisions about vaccine distribution and vaccination strategies.

**4. Adverse Event Detection:** Advanced data analytics have been applied to rapidly detect and investigate adverse events following vaccination. This helps ensure vaccine safety and allows for swift response when potential issues arise.

**5. Immunological Assays:** Innovations in immunological assays, such as neutralization assays and T-cell response studies, have provided insights into the immune response generated by vaccines.

**6. Vaccine Effectiveness Against Variants:** Studies have assessed how well vaccines protect against specific variants, informing decisions about booster shots and updated vaccine formulations.

**7. Vaccine Heterologous Boosting:** Research into the effectiveness of mixing and matching different vaccines (heterologous boosting) has been conducted to optimize vaccination strategies.

**8. Adaptive Clinical Trials:** Trials have been designed to adapt to changing circumstances, allowing for quick assessment of new vaccine candidates and modifications to existing ones.

**9. Global Data Sharing:** International collaboration and data sharing have been critical for analyzing vaccine performance globally and ensuring equitable access to vaccines.



**10. AI and Machine Learning:** Artificial intelligence and machine learning techniques have been used to analyze vast datasets, identify trends, and predict vaccine outcomes.

◆ **DATASET :-**

I took the dataset from([www.kaggle.com/data](https://www.kaggle.com/data)).  
The dataset is related to Covid-19 Vaccines Analysis.

**MY DATASET LINK:**

(<https://www.kaggle.com/datasets/apreda/covid-world-vaccination-progress>)

◆ **DETAILS OF MY DATASET:-**

In my dataset the column names contains:


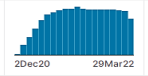



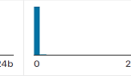

- 1)Country
- 2)Iso\_code
- 3)Date
- 4)Total\_vaccination
- 5)People\_vaccinated
- 6)People\_fully\_vaccinated
- 7)Daily\_vaccinations\_raw
- 8)Total\_vaccination
- 9)People\_vaccinated\_per\_hundred

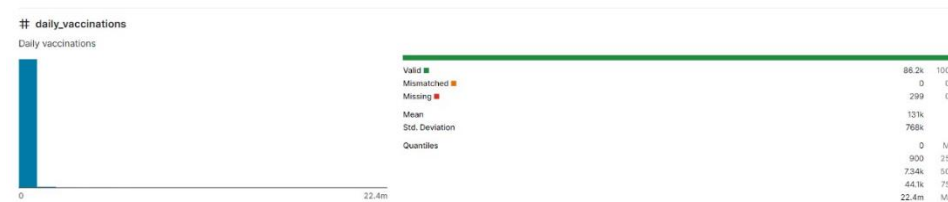
for eg: (flowchart):

country\_vaccinations.csv (17.73 MB)

Detail Compact Column 10 of 15 columns

COVID-19 vaccination

country	iso_code	date	# total_vaccinations	# people_vaccinated	# people_fully_vaccinated	# daily_vaccinations	# daily_vaccinations_smoothed
Country	ISO Code	Date	Total vaccinations	People vaccinated	People fully vaccinated	Daily vaccinations	Daily vaccinations smoothed
	223 unique values						
Afghanistan	AFG	2021-02-22	0.0	0.0			
Afghanistan	AFG	2021-02-23					1367.0
Afghanistan	AFG	2021-02-24					1367.0
Afghanistan	AFG	2021-02-25					1367.0
Afghanistan	AFG	2021-02-26					1367.0
Afghanistan	AFG	2021-02-27					1367.0
Afghanistan	AFG	2021-02-28	8200.0	8200.0			1367.0
Afghanistan	AFG	2021-03-01					1588.0
Afghanistan	AFG	2021-03-02					1794.0
Afghanistan	AFG	2021-03-03					2008.0



## METRICS USED FOR ACCURACY CHECK:-

**For Classification Tasks (e.g., binary or multiclass classification):**

1. **Accuracy:** This is a widely used metric that calculates the ratio of correctly predicted instances to the total number of instances. It's suitable for balanced datasets. However, it may not be the best choice for imbalanced datasets, where a class is significantly more prevalent than others.

2. **Precision:** Precision measures the ratio of correctly predicted positive instances to the total predicted positive instances. It's valuable when minimizing false positives is crucial, such as in spam detection or medical diagnosis.
3. **F1-Score:** The F1-score is the harmonic mean of precision and recall. It provides a balance between precision and recall and is useful when there's an uneven class distribution.

**For Regression Tasks (e.g., predicting numerical values):**

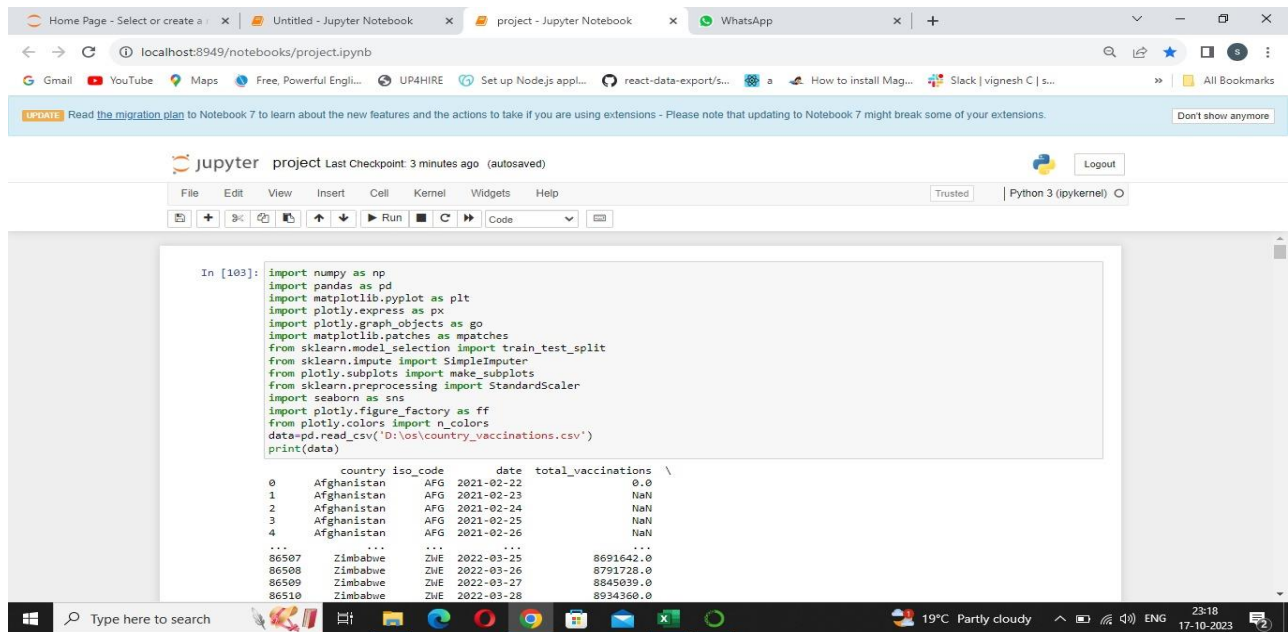
1. **Mean Absolute Error (MAE):** MAE calculates the average absolute difference between the predicted values and the actual values. It is less sensitive to outliers compared to RMSE.
2. **Mean Squared Error (MSE):** MSE calculates the average of the squared differences between predicted and actual values. It penalizes larger errors more heavily than MAE.
3. **Root Mean Squared Error (RMSE):** RMSE is the square root of MSE and provides an interpretable measure of prediction error in the same units as the target variable. It's sensitive to outliers.

**BEGIN BUILDING THE PROJECT BY LOAD THE DATASET**

To import the required libraries and read a CSV file,

```
data=pd.read_csv('D:\os\country_vaccinations.csv')
```

```
Print(data)
```



## PREPROCESS DATASET

### Import The Required Libraries:

*import numpy as np*

*import pandas as pd*

*import matplotlib.pyplot as plt*

*import plotly.express as px*

*import plotly.graph\_objects as go*

*import matplotlib.patches as mpatches*

*from sklearn.model\_selection import train\_test\_split*

*from sklearn.impute import SimpleImputer*

*from plotly.subplots import make\_subplots*

*from sklearn.preprocessing import StandardScaler*

```
import seaborn as sns

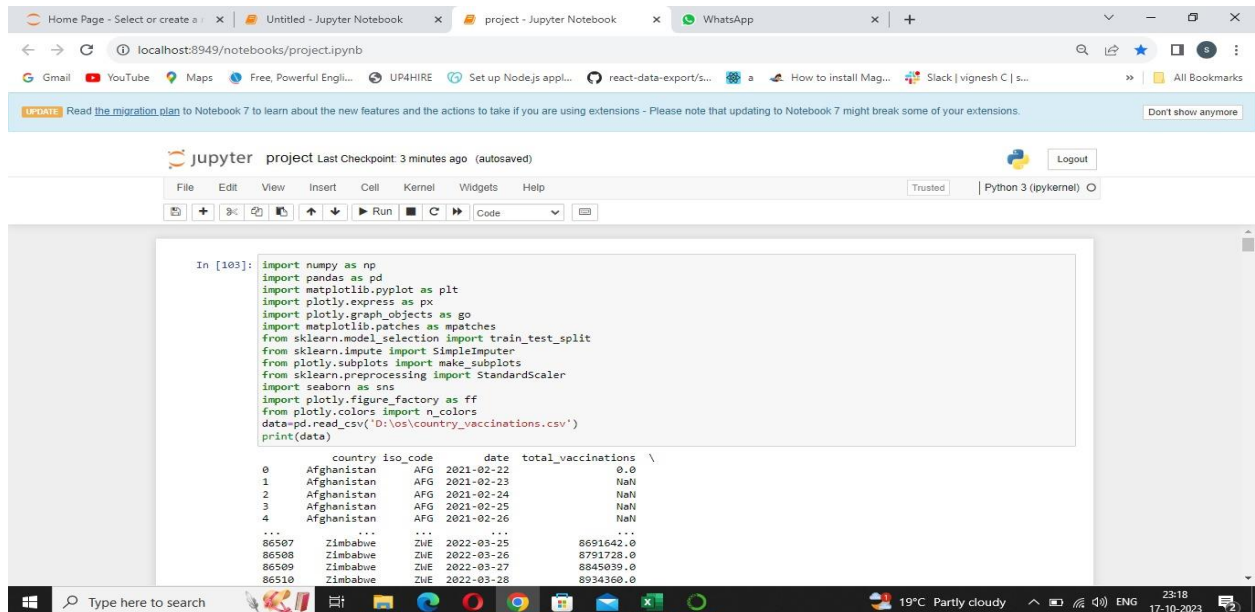
import plotly.figure_factory as ff

from plotly.colors import n_colors
```

## Importing the Dataset:

Read Dataset,

```
data=pd.read_csv('D:\os\country_vaccinations.csv')print(data)
```



The screenshot shows a Jupyter Notebook interface with a code cell containing the following Python code:

```
In [103]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import plotly.express as px
import plotly.graph_objects as go
import matplotlib.patches as mpatches
from sklearn.model_selection import train_test_split
from sklearn.impute import SimpleImputer
from plotly.subplots import make_subplots
from sklearn.preprocessing import StandardScaler
import seaborn as sns
import plotly.figure_factory as ff
from plotly.colors import n_colors
data=pd.read_csv('D:\os\country_vaccinations.csv')
print(data)
```

The output of the code is a DataFrame with the following columns: country, iso\_code, date, and total\_vaccinations. The data is displayed in a table format:

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

- Create Matrix,

```
matrix=data.values
```

```
print(matrix)
```

Home Page - Select or create a notebook | project - Jupyter Notebook | localhost:8949/notebooks/project.ipynb

UPDATE: Read the migration plan to Notebook 7 to learn about the new features and the actions to take if you are using extensions - Please note that updating to Notebook 7 might break some of your extensions. Don't show anymore

jupyter project Last Checkpoint: 22 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

```
In [9]: matrix=data.values
print(matrix)

[['Afghanistan' 'AFG' '2021-02-22' ...
  'Johnson&Johnson, Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing'
  'World Health Organization' 'https://covid19.who.int/']
 ['Afghanistan' 'AFG' '2021-02-23' ...
  'Johnson&Johnson, Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing'
  'World Health Organization' 'https://covid19.who.int/']
 ['Afghanistan' 'AFG' '2021-02-24' ...
  'Johnson&Johnson, Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing'
  'World Health Organization' 'https://covid19.who.int/']
 ...
 ['Zimbabwe' 'ZWE' '2022-03-27' ...
  'Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac, Sputnik V'
  'Ministry of Health'
  'https://www.arcgis.com/home/webmap/viewer.html?url=https://services9.arcgis.com/DnERH4rcjw7NU61v/ArcGIS/rest/services/Vaccin
e_Distribution_Program/FeatureServer&source=sd']
 ['Zimbabwe' 'ZWE' '2022-03-28' ...
  'Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac, Sputnik V'
  'Ministry of Health'
  'https://www.arcgis.com/home/webmap/viewer.html?url=https://services9.arcgis.com/DnERH4rcjw7NU61v/ArcGIS/rest/services/Vaccin
```

- *Other Information about dataset,*

Home Page - Select or create a notebook | project - Jupyter Notebook | localhost:8949/notebooks/project.ipynb

UPDATE: Read the migration plan to Notebook 7 to learn about the new features and the actions to take if you are using extensions - Please note that updating to Notebook 7 might break some of your extensions. Don't show anymore

jupyter project Last Checkpoint: 22 minutes ago (autosaved) Logout

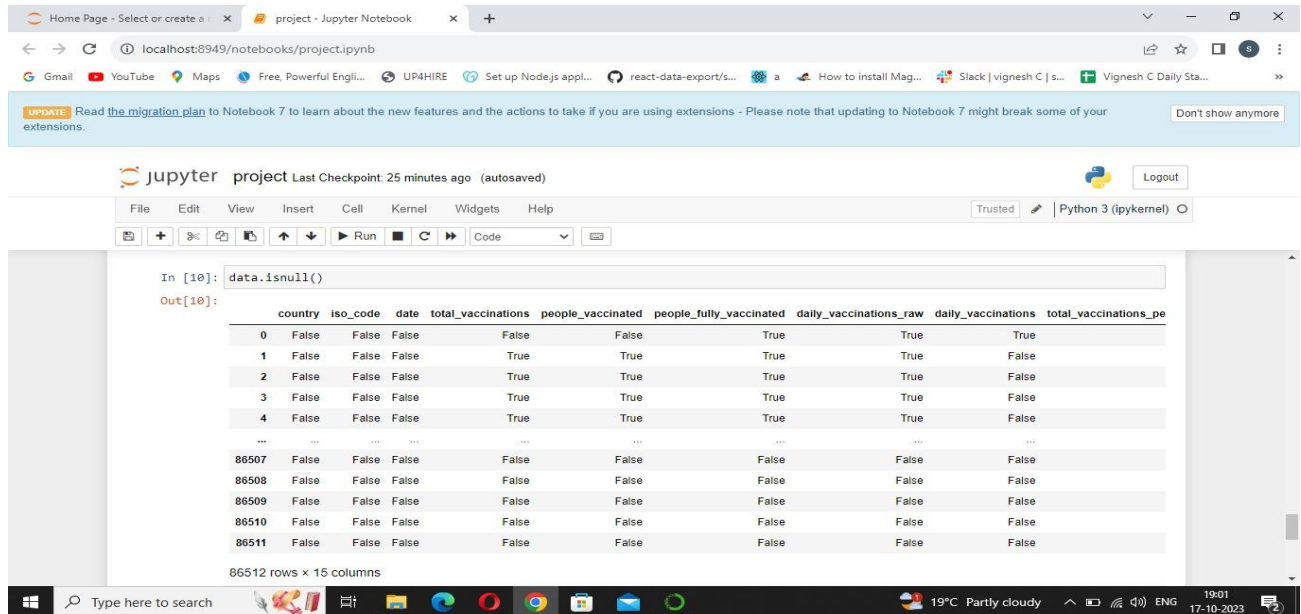
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

```
In [8]: print(data.info())

<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
None
```

## Handling The Missing Data:

*Before Handling the Missing data, we use `isnull()` to show the null values and using `isnull().sum` to get total number of null values.*



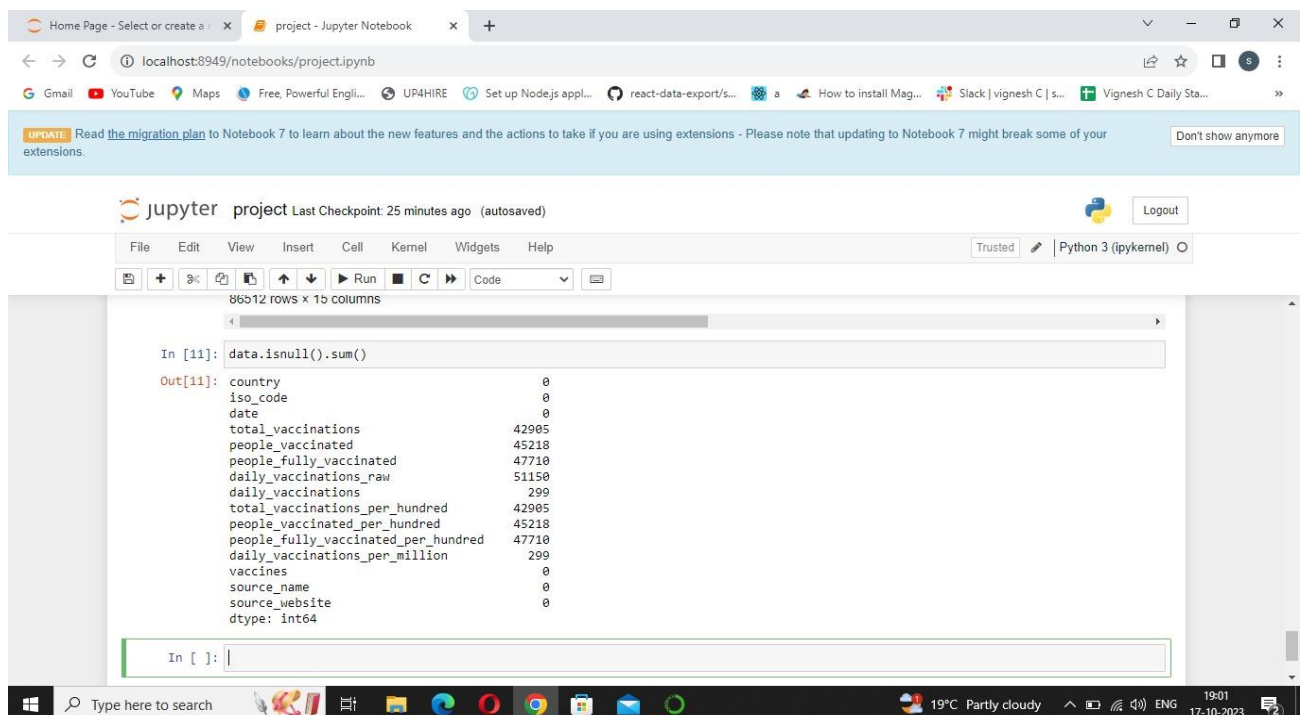
The screenshot shows a Jupyter Notebook interface with the following code and output:

```
In [10]: data.isnull()
```

Out[10]:

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations	total_vaccinations_pe
0	False	False	False	False	False	True	True	True	
1	False	False	False	True	True	True	True	False	
2	False	False	False	True	True	True	True	False	
3	False	False	False	True	True	True	True	False	
4	False	False	False	True	True	True	True	False	
...	...	...	...	...	...	...	...	...	...
86507	False	False	False	False	False	False	False	False	
86508	False	False	False	False	False	False	False	False	
86509	False	False	False	False	False	False	False	False	
86510	False	False	False	False	False	False	False	False	
86511	False	False	False	False	False	False	False	False	

86512 rows x 15 columns



The screenshot shows a Jupyter Notebook interface with the following code and output:

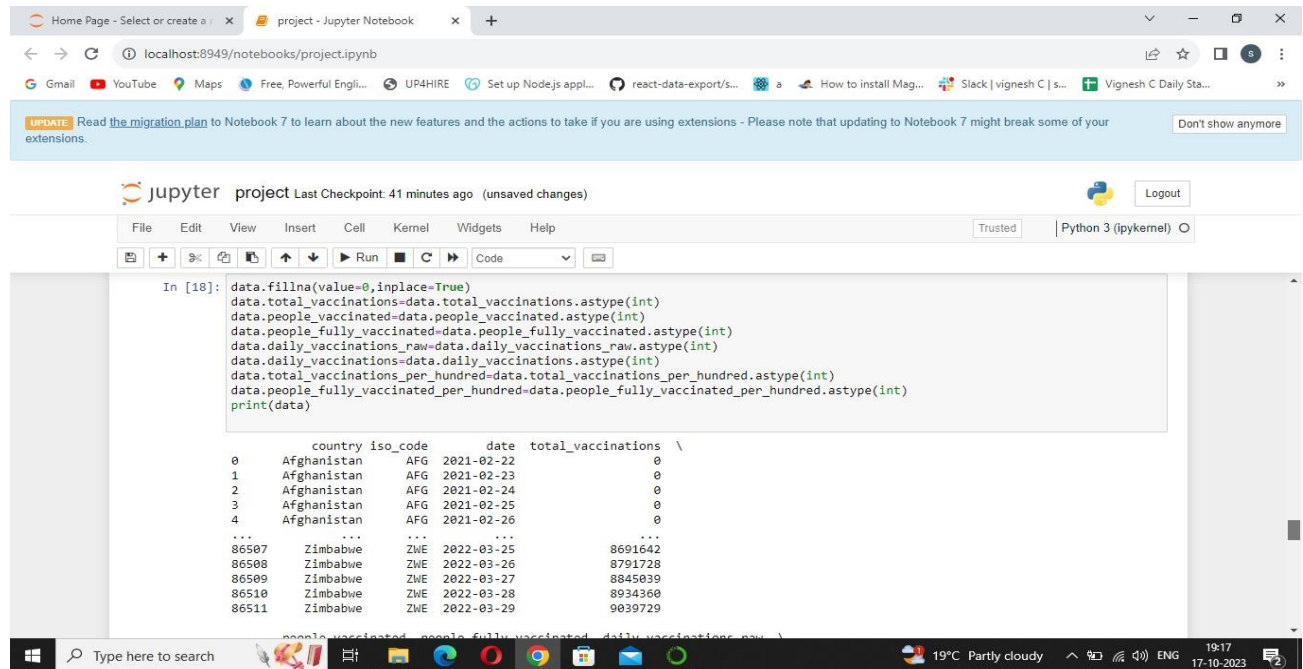
```
In [11]: data.isnull().sum()
```

Out[11]:

```
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
```



The below code in the image for all the data cleaning,

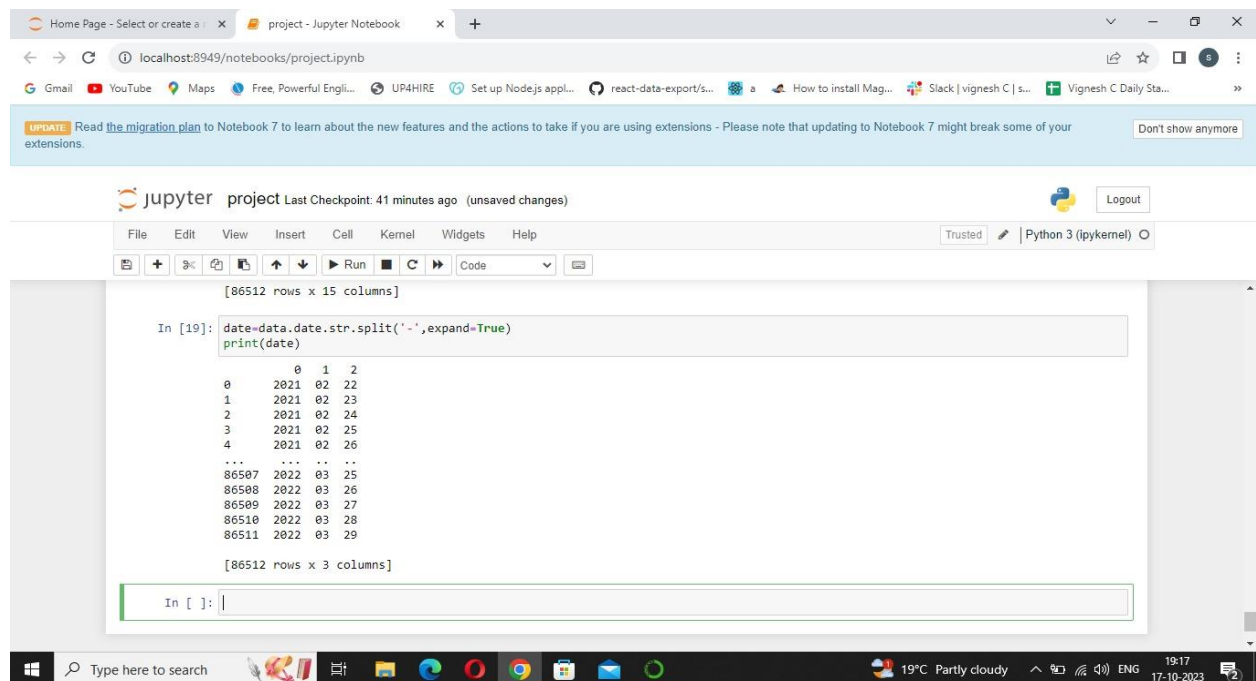


The screenshot shows a Jupyter Notebook window titled 'project - Jupyter Notebook'. The browser address bar indicates the notebook is running on localhost:8949. The notebook interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations and execution. The code cell contains the following Python code:

```
In [18]: data.fillna(value=0,inplace=True)
data.total_vaccinations=data.total_vaccinations.astype(int)
data.people_vaccinated=data.people_vaccinated.astype(int)
data.people_fully_vaccinated=data.people_fully_vaccinated.astype(int)
data.daily_vaccinations_raw=data.daily_vaccinations_raw.astype(int)
data.daily_vaccinations=data.daily_vaccinations.astype(int)
data.total_vaccinations_per_hundred=data.total_vaccinations_per_hundred.astype(int)
data.people_fully_vaccinated_per_hundred=data.people_fully_vaccinated_per_hundred.astype(int)
print(data)
```

The output of the code is a DataFrame with columns: country, iso\_code, date, total\_vaccinations, and people\_fully\_vaccinated\_per\_hundred. The output shows data for Afghanistan (rows 0-4) and Zimbabwe (rows 86507-86511).

	country	iso_code	date	total_vaccinations	people_fully_vaccinated_per_hundred
0	Afghanistan	AFG	2021-02-22	0	
1	Afghanistan	AFG	2021-02-23	0	
2	Afghanistan	AFG	2021-02-24	0	
3	Afghanistan	AFG	2021-02-25	0	
4	Afghanistan	AFG	2021-02-26	0	
...	...	...	...	...	...
86507	Zimbabwe	ZWE	2022-03-25	8691642	
86508	Zimbabwe	ZWE	2022-03-26	8791728	
86509	Zimbabwe	ZWE	2022-03-27	8845039	
86510	Zimbabwe	ZWE	2022-03-28	8934360	
86511	Zimbabwe	ZWE	2022-03-29	9039729	



The screenshot shows the same Jupyter Notebook window. The code cell contains the following Python code:

```
In [19]: date=data.date.str.split('-',expand=True)
print(date)
```

The output of the code is a DataFrame with columns: 0, 1, and 2, representing the year, month, and day respectively. The output shows data for Afghanistan (rows 0-4) and Zimbabwe (rows 86507-86511).

	0	1	2
0	2021	02	22
1	2021	02	23
2	2021	02	24
3	2021	02	25
4	2021	02	26
...	...	...	...
86507	2022	03	25
86508	2022	03	26
86509	2022	03	27
86510	2022	03	28
86511	2022	03	29

Below the output, the notebook shows the dimensions of the resulting DataFrame: [86512 rows x 3 columns].



Home Page - Select or create a project - Jupyter Notebook

localhost:8949/notebooks/project.ipynb

UPDATE: Read the migration plan to Notebook 7 to learn about the new features and the actions to take if you are using extensions - Please note that updating to Notebook 7 might break some of your extensions. Don't show anymore

jupyter project Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

In [22]:

```
data['year']=date[0]
data['month']=date[1]
data['day']=date[2]
data.year=pd.to_numeric(data.year)
data.month=pd.to_numeric(data.month)
data.day=pd.to_numeric(data.day)
data.date=pd.to_datetime(data.date)
data.head()
```

Out[22]:

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations	total_vaccinations_per
0	Afghanistan	AFG	2021-02-22	0	0	0	0	0	
1	Afghanistan	AFG	2021-02-23	0	0	0	0	1367	
2	Afghanistan	AFG	2021-02-24	0	0	0	0	1367	
3	Afghanistan	AFG	2021-02-25	0	0	0	0	1367	

19°C Partly cloudy 19:23 17-10-2023

Home Page - Select or create a project - Jupyter Notebook

localhost:8949/notebooks/project.ipynb

UPDATE: Read the migration plan to Notebook 7 to learn about the new features and the actions to take if you are using extensions - Please note that updating to Notebook 7 might break some of your extensions. Don't show anymore

jupyter project Last Checkpoint: an hour ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

In [23]:

```
data.date=pd.to_datetime(data.date)
data.head()
```

Out[23]:

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations	total_vaccinations_per
0	Afghanistan	AFG	2021-02-22	0	0	0	0	0	
1	Afghanistan	AFG	2021-02-23	0	0	0	0	1367	
2	Afghanistan	AFG	2021-02-24	0	0	0	0	1367	
3	Afghanistan	AFG	2021-02-25	0	0	0	0	1367	
4	Afghanistan	AFG	2021-02-26	0	0	0	0	1367	

19°C Partly cloudy 19:25 17-10-2023

*Some detailed features to specify the details using the below code,*

Home Page - Select or create a... project - Jupyter Notebook WhatsApp

localhost:8949/notebooks/project.ipynb

UPDATE Read the migration plan to Notebook 7 to learn about the new features and the actions to take if you are using extensions - Please note that updating to Notebook 7 might break some of your extensions. Don't show anymore

jupyter project Last Checkpoint: an hour ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

3 Afghanistan AFG 2021-02-25 0 0 0 0 1367

4 Afghanistan AFG 2021-02-26 0 0 0 0 1367

```
In [26]: print('data point starts from',data.date.min())
print('data point ends at',data.date.max())
print('total number of countries in the data set',len(data.country.unique()))
print('total number of unique vaccines in the data set',len(data.vaccines.unique()))
```

data point starts from 2020-12-02 00:00:00  
data point ends at 2022-03-29 00:00:00  
total number of countries in the data set 223  
total number of unique vaccines in the data set 84

In [ ]:

In [ ]:

In [ ]:

Type here to search 19°C Partly cloudy 19:32 17-10-2023

Home Page - Select or create a ...

project - Jupyter Notebook

WhatsApp

localhost:8949/notebooks/project.ipynb

Gmail YouTube Maps Free, Powerful Engin... UP4HIRE Set up Node.js appl... react-data-export/s... How to install Mag... Slack | vignesh C | s... Vignesh C Daily Sta...

UPDATE Read the migration plan to Notebook 7 to learn about the new features and the actions to take if you are using extensions - Please note that updating to Notebook 7 might break some of your extensions.

Don't show anymore

jupyter project Last Checkpoint: an hour ago (autosaved)

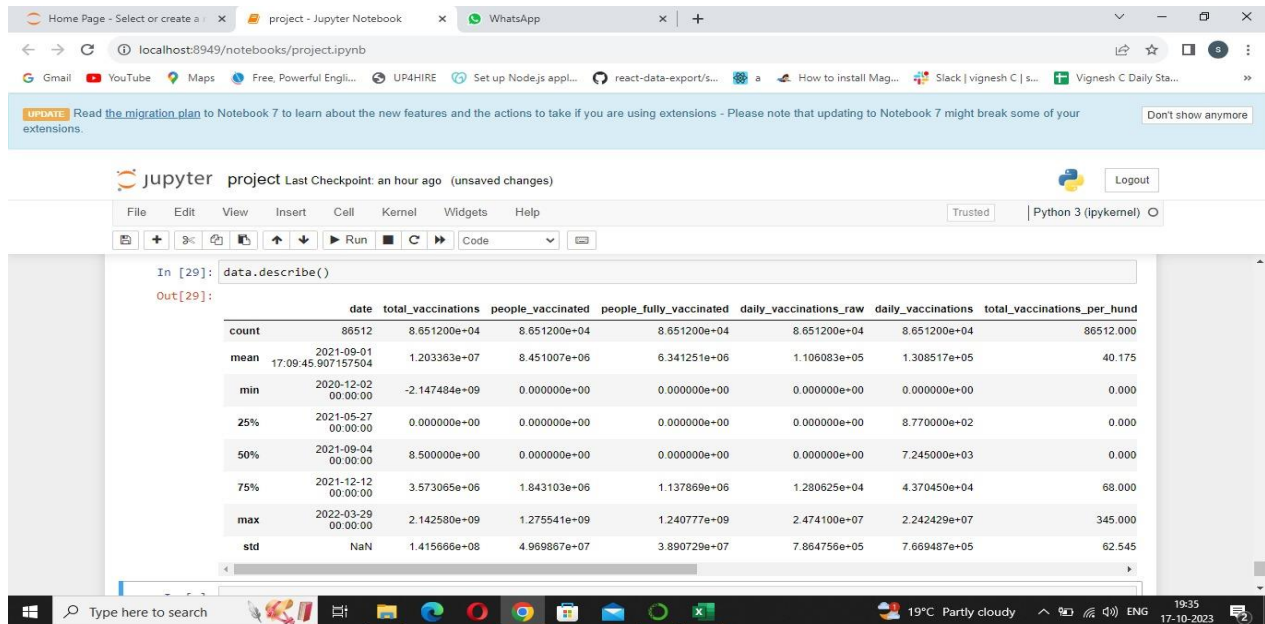
Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

In [27]:  
data.info()  
  
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 86512 entries, 0 to 86511  
Data columns (total 18 columns):  
# Column Non-Null Count Dtype  
---  
0 country 86512 non-null object  
1 iso\_code 86512 non-null object  
2 date 86512 non-null datetime64[ns]  
3 total\_vaccinations 86512 non-null int32  
4 people\_vaccinated 86512 non-null int32  
5 people\_fully\_vaccinated 86512 non-null int32  
6 daily\_vaccinations\_raw 86512 non-null int32  
7 daily\_vaccinations 86512 non-null int32  
8 total\_vaccinations\_per\_hundred 86512 non-null float64  
9 people\_vaccinated\_per\_hundred 86512 non-null float64  
10 people\_fully\_vaccinated\_per\_hundred 86512 non-null float64  
11 daily\_vaccinations\_per\_million 86512 non-null float64  
12 vaccines 86512 non-null object  
13 source\_name 86512 non-null object  
14 source\_website 86512 non-null object  
15 year 86512 non-null int64

Using Data visualization we are going to draw some visuals to get insights from dataset,

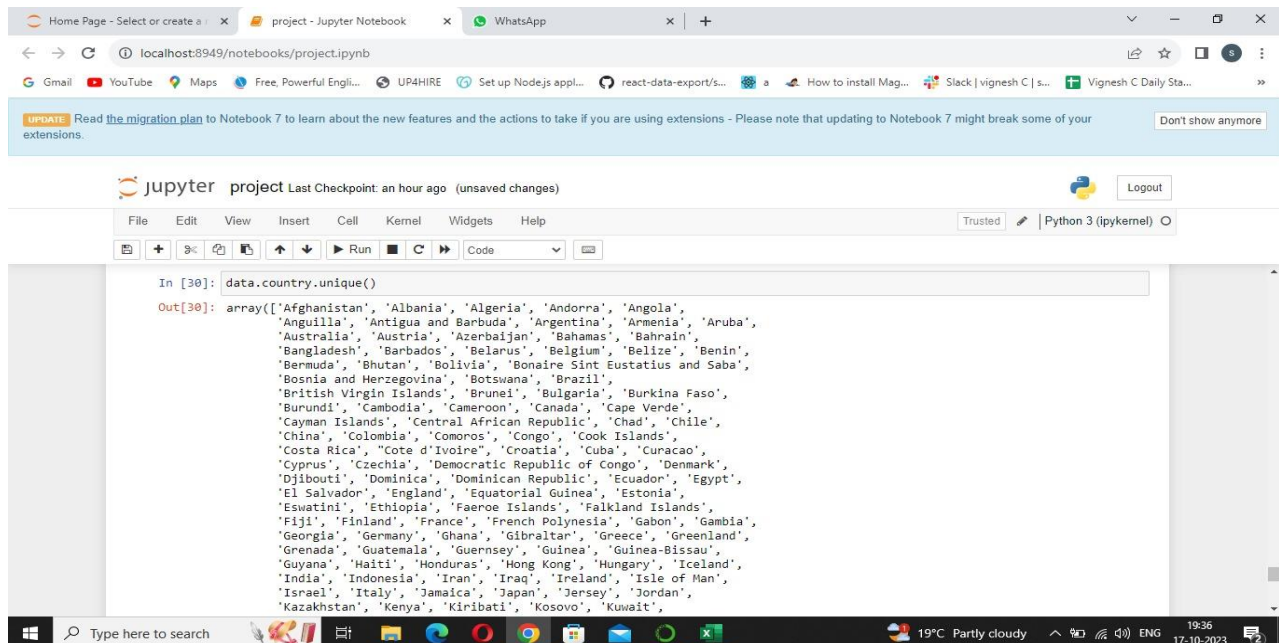
Describe () function is used to get the statistics of each feature in dataset to get count, min, max, standard deviation, median, etc.,



```
In [29]: data.describe()
```

	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations	total_vaccinations_per_hund
count	86512	8.651200e+04	8.651200e+04	8.651200e+04	8.651200e+04	8.651200e+04	86512.000
mean	2021-09-01 17:09:45.907157504	1.203363e+07	8.451007e+06	6.341251e+06	1.106083e+05	1.308517e+05	40.175
min	2020-12-02 00:00:00	-2.147484e+09	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000
25%	2021-05-27 00:00:00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	8.770000e+02	0.000
50%	2021-09-04 00:00:00	8.500000e+00	0.000000e+00	0.000000e+00	0.000000e+00	7.245000e+03	0.000
75%	2021-12-12 00:00:00	3.573065e+06	1.843103e+06	1.137869e+06	1.280625e+04	4.370450e+04	68.000
max	2022-03-29 00:00:00	2.142580e+09	1.275541e+09	1.240777e+09	2.474100e+07	2.242429e+07	345.000
std	NaN	1.415666e+08	4.969867e+07	3.890729e+07	7.864756e+05	7.669487e+05	62.545

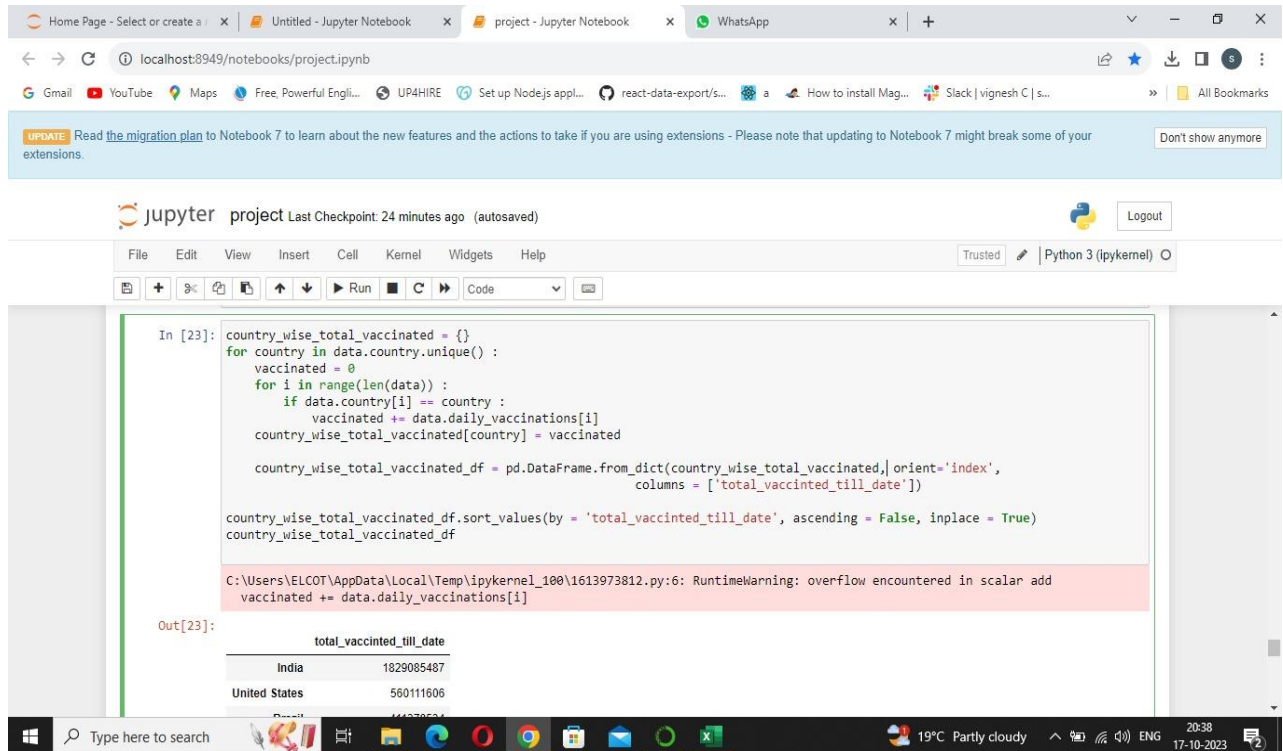
Unique () function helps to get unique values,



```
In [30]: data.country.unique()
```

```
Out[30]: array(['Afghanistan', 'Albania', 'Algeria', 'Andorra', 'Angola', 'Anguilla', 'Antigua and Barbuda', 'Argentina', 'Armenia', 'Aruba', 'Australia', 'Austria', 'Azerbaijan', 'Bahamas', 'Bahrain', 'Bangladesh', 'Barbados', 'Belarus', 'Belgium', 'Belize', 'Benin', 'Bermuda', 'Bhutan', 'Bolivia', 'Bonaire Sint Eustatius and Saba', 'Bosnia and Herzegovina', 'Botswana', 'Brazil', 'British Virgin Islands', 'Brunei', 'Bulgaria', 'Burkina Faso', 'Burundi', 'Cambodia', 'Cameroon', 'Canada', 'Cape Verde', 'Cayman Islands', 'Central African Republic', 'Chad', 'Chile', 'China', 'Colombia', 'Comoros', 'Congo', 'Cook Islands', 'Costa Rica', 'Cote d'Ivoire', 'Croatia', 'Cuba', 'Curacao', 'Cyprus', 'Czechia', 'Democratic Republic of Congo', 'Denmark', 'Djibouti', 'Dominica', 'Dominican Republic', 'Ecuador', 'Egypt', 'El Salvador', 'England', 'Equatorial Guinea', 'Estonia', 'Eswatini', 'Ethiopia', 'Faeroe Islands', 'Falkland Islands', 'Fiji', 'Finland', 'France', 'French Polynesia', 'Gabon', 'Gambia', 'Georgia', 'Germany', 'Ghana', 'Gibraltar', 'Greece', 'Greenland', 'Grenada', 'Guatemala', 'Guernsey', 'Guinea', 'Guinea-Bissau', 'Guyana', 'Haiti', 'Honduras', 'Hong Kong', 'Hungary', 'Iceland', 'India', 'Indonesia', 'Iran', 'Iraq', 'Ireland', 'Isle of Man', 'Israel', 'Italy', 'Jamaica', 'Japan', 'Jersey', 'Jordan', 'Kazakhstan', 'Kenya', 'Kiribati', 'Kosovo', 'Kuwait', ...])
```

*To see how many total vaccines have been used in each country using the code below,*



Home Page - Select or create a | x | Untitled - Jupyter Notebook | x | project - Jupyter Notebook | x | WhatsApp | x | +

localhost:8949/notebooks/project.ipynb

UPDATE Read the [migration plan](#) to Notebook 7 to learn about the new features and the actions to take if you are using extensions - Please note that updating to Notebook 7 might break some of your extensions. Don't show anymore

jupyter project Last Checkpoint: 24 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

```
In [23]: country_wise_total_vaccinated = {}
for country in data.country.unique():
    vaccinated = 0
    for i in range(len(data)):
        if data.country[i] == country:
            vaccinated += data.daily_vaccinations[i]
    country_wise_total_vaccinated[country] = vaccinated

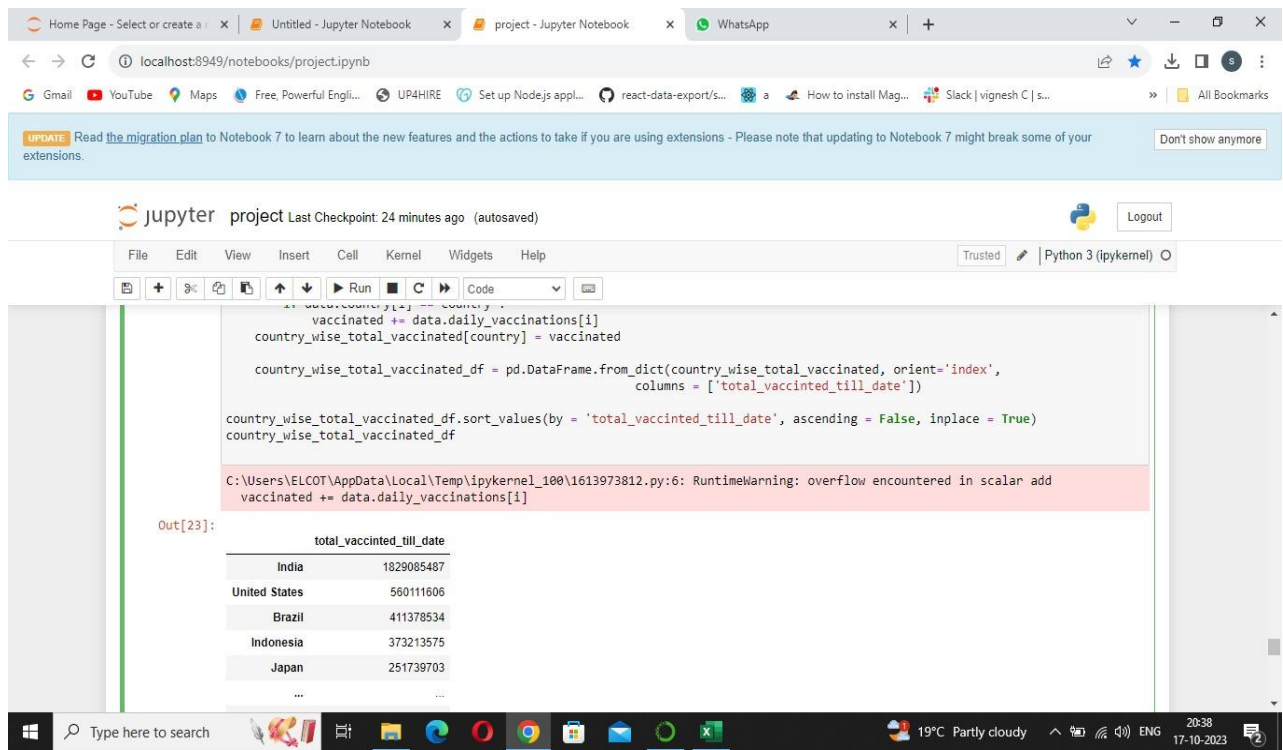
country_wise_total_vaccinated_df = pd.DataFrame.from_dict(country_wise_total_vaccinated, orient='index',
                                                         columns = ['total_vaccinted_till_date'])

country_wise_total_vaccinated_df.sort_values(by = 'total_vaccinted_till_date', ascending = False, inplace = True)
country_wise_total_vaccinated_df
```

C:\Users\ELCOT\AppData\Local\Temp\ipykernel\_100\1613973812.py:6: RuntimeWarning: overflow encountered in scalar add  
vaccinated += data.daily\_vaccinations[i]

Out[23]:

	total_vaccinted_till_date
India	1829085487
United States	560111606



Home Page - Select or create a | x | Untitled - Jupyter Notebook | x | project - Jupyter Notebook | x | WhatsApp | x | +

localhost:8949/notebooks/project.ipynb

UPDATE Read the [migration plan](#) to Notebook 7 to learn about the new features and the actions to take if you are using extensions - Please note that updating to Notebook 7 might break some of your extensions. Don't show anymore

jupyter project Last Checkpoint: 24 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

```
In [23]: country_wise_total_vaccinated = {}
for country in data.country.unique():
    vaccinated += data.daily_vaccinations[i]
    country_wise_total_vaccinated[country] = vaccinated

country_wise_total_vaccinated_df = pd.DataFrame.from_dict(country_wise_total_vaccinated, orient='index',
                                                         columns = ['total_vaccinted_till_date'])

country_wise_total_vaccinated_df.sort_values(by = 'total_vaccinted_till_date', ascending = False, inplace = True)
country_wise_total_vaccinated_df
```

C:\Users\ELCOT\AppData\Local\Temp\ipykernel\_100\1613973812.py:6: RuntimeWarning: overflow encountered in scalar add  
vaccinated += data.daily\_vaccinations[i]

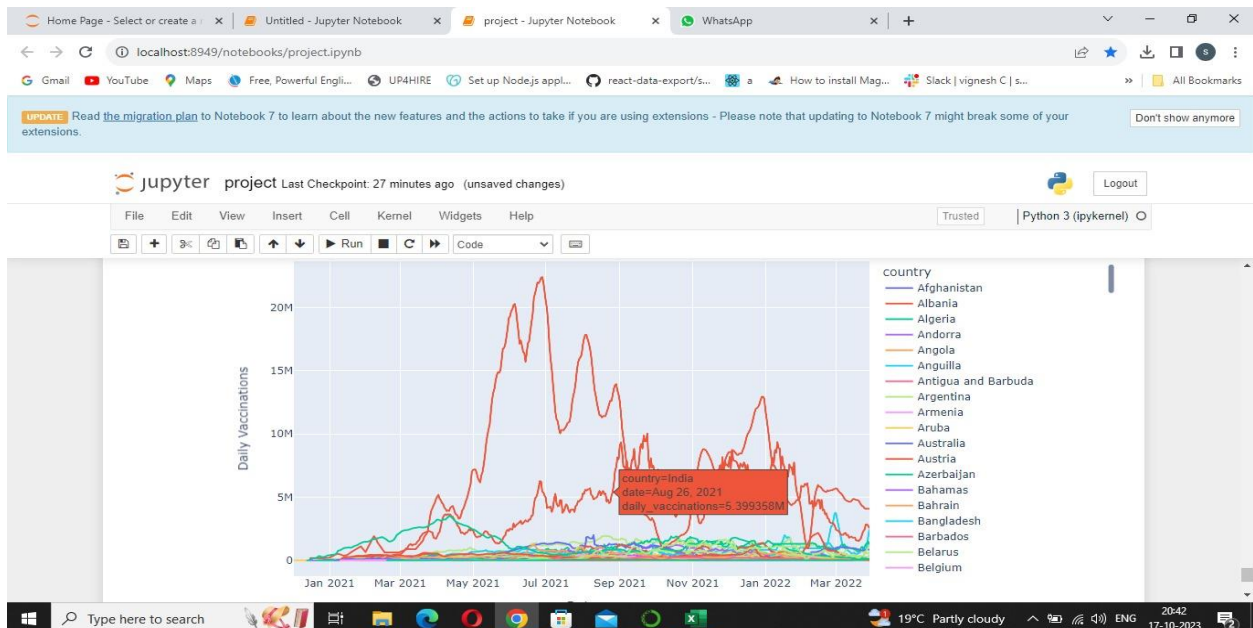
Out[23]:

	total_vaccinted_till_date
India	1829085487
United States	560111606
Brazil	411378534
Indonesia	373213575
Japan	251739703





*To draw a line plot where x-axis is Date and the y-axis is daily\_vaccination using the in the image,*



*Now, using the sklearn.preprocessing library contains class called imputer, helps in missing data by using the below:*

```
In [57]: imputer=SimpleImputer(strategy='mean')
x=np.array(data['people_vaccinated']).reshape(-1,1)
y=np.array(data['daily_vaccinations']).reshape(-1,1)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25)
imputer.fit_transform(x_train)
x_test=imputer.transform(x_test)
print(x_test)

[[18021592.92013698]
 [18021592.92013698]
 [ 290216.         ]
 ...
 [18021592.92013698]
 [ 6318501.         ]
 [18021592.92013698]]
```

## Encoding categorical data(one-hot encoding)

One-hot encoding is a technique used to convert categorical data into a numerical format that machine learning algorithms can work with. Here's how you can perform one-hot encoding in Python, assuming you have a dataset with categorical variables:

```
data.fillna(value=0, inplace=True)
data_encoded=pd.get_dummies(data,columns=['vaccines'])
print(data_encoded.head())
```

	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

	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

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

	vaccines_Oxford/AstraZeneca, Sputnik V	vaccines_Pfizer/BioNTech
0	False	False
1	False	False
2	False	False
3	False	False
4	False	False

	vaccines_Pfizer/BioNTech, Sinopharm/Beijing
0	False
1	False
2	False
3	False
4	False

	vaccines_Pfizer/BioNTech, Sinopharm/Beijing, Sputnik V
0	False

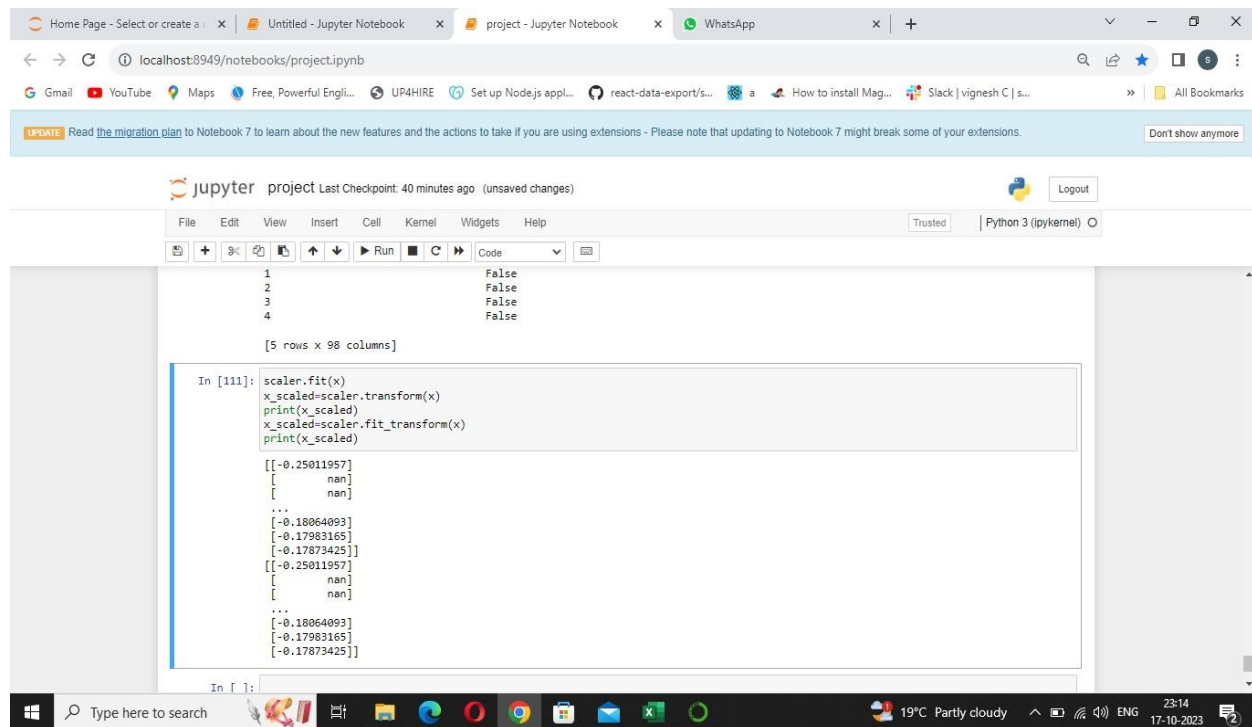
## Splitting the data set into test set and training set

By using, Import train\_test\_split

```
x=np.array(data['people_vaccinated']).reshape(-1,1)
y=np.array(data['daily_vaccinations']).reshape(-1,1)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25,random_state=42)

plt.scatter(x_test,y_test, color='b')
plt.show()
```

## Feature Scaling



The screenshot shows a Jupyter Notebook running in a web browser at localhost:8949/notebooks/project.ipynb. The notebook has a menu bar with File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. Below the menu bar is a toolbar with icons for saving, undo, redo, and running cells. The notebook content shows a code cell with the following code:

```
1
2
3
4
[5 rows x 98 columns]

In [111]: scaler.fit(x)
          x_scaled=scaler.transform(x)
          print(x_scaled)
          x_scaled=scaler.fit_transform(x)
          print(x_scaled)
```

The output of the code cell is:

```
[[ -0.25011957]
 [          nan]
 [          nan]
 ...
 [-0.18064093]
 [-0.17983165]
 [-0.17873425]]
[[ -0.25011957]
 [          nan]
 [          nan]
 ...
 [-0.18064093]
 [-0.17983165]
 [-0.17873425]]
```

## STATISTICAL ANALYSIS

*Statistical analysis for COVID-19 vaccine analysis data can provide more in-depth insights and support decision-making. Here are some statistical techniques and analyses you can perform:*

**1. Descriptive Statistics:** Calculate summary statistics (mean, median, standard deviation) for key vaccine-related variables, such as vaccination rates, doses administered, and adverse events.

**2. Hypothesis Testing:** Use statistical tests like t-tests or ANOVA to compare vaccination rates or outcomes between different groups (e.g., regions, age groups, vaccine types).

- **DESCRIPTIVE STATISTICS**

*Descriptive statistics provide a summary of your data, including measures like mean, median, and standard deviation.*



```
mean_vaccination = data['total_vaccinations'].mean()
```

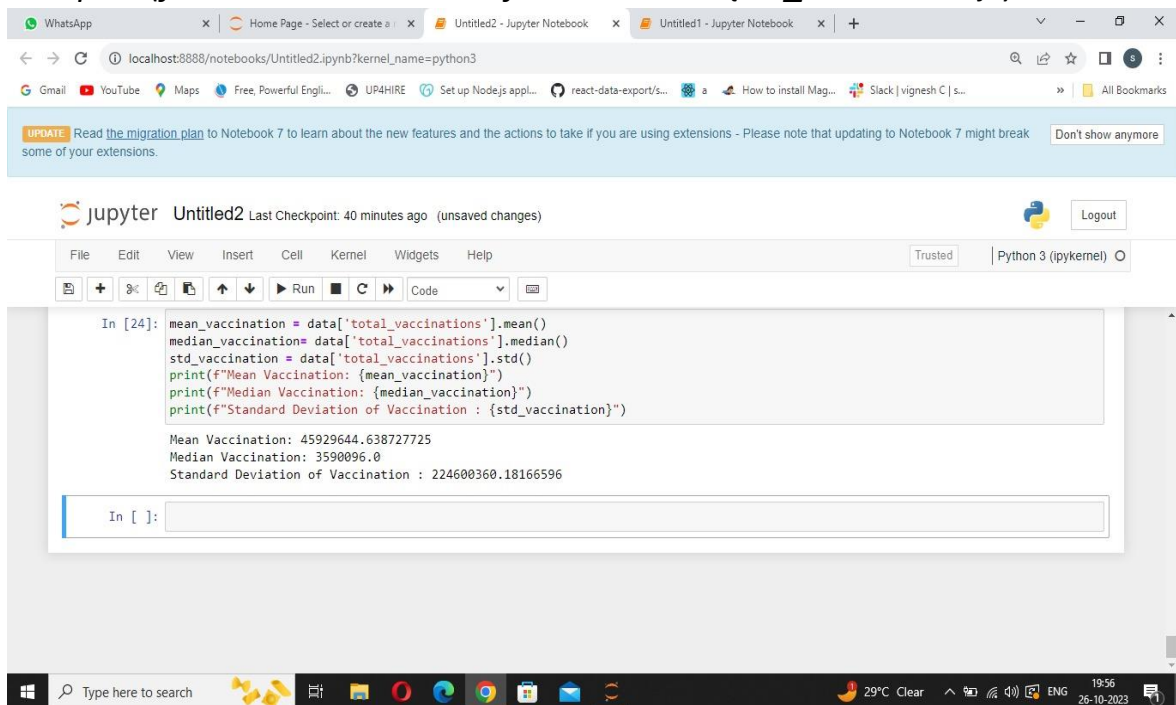
```
median_vaccination= data['total_vaccinations'].median()
```

```
std_vaccination = data['total_vaccinations'].std()
```

```
print(f"Mean Vaccination: {mean_vaccination}")
```

```
print(f"Median Vaccination: {median_vaccination}")
```

```
print(f"Standard Deviation of Vaccination : {std_vaccination}")
```



The screenshot shows a Jupyter Notebook interface with a browser window at the top. The notebook has a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations, zooming, and running code. The code cell contains the following Python code:

```
In [24]: mean_vaccination = data['total_vaccinations'].mean()
median_vaccination= data['total_vaccinations'].median()
std_vaccination = data['total_vaccinations'].std()
print(f"Mean Vaccination: {mean_vaccination}")
print(f"Median Vaccination: {median_vaccination}")
print(f"Standard Deviation of Vaccination : {std_vaccination}")
```

The output of the code is displayed below the cell:

```
Mean Vaccination: 45929644.63872725
Median Vaccination: 3590096.0
Standard Deviation of Vaccination : 224600360.18166596
```

The bottom of the screenshot shows a Windows taskbar with the search bar, task view button, and several application icons. The system tray on the right shows the temperature (29°C), weather (Clear), and the date and time (19:56, 26-10-2023).

- ```
mean_vaccination = data['people_vaccinated'].mean()
```

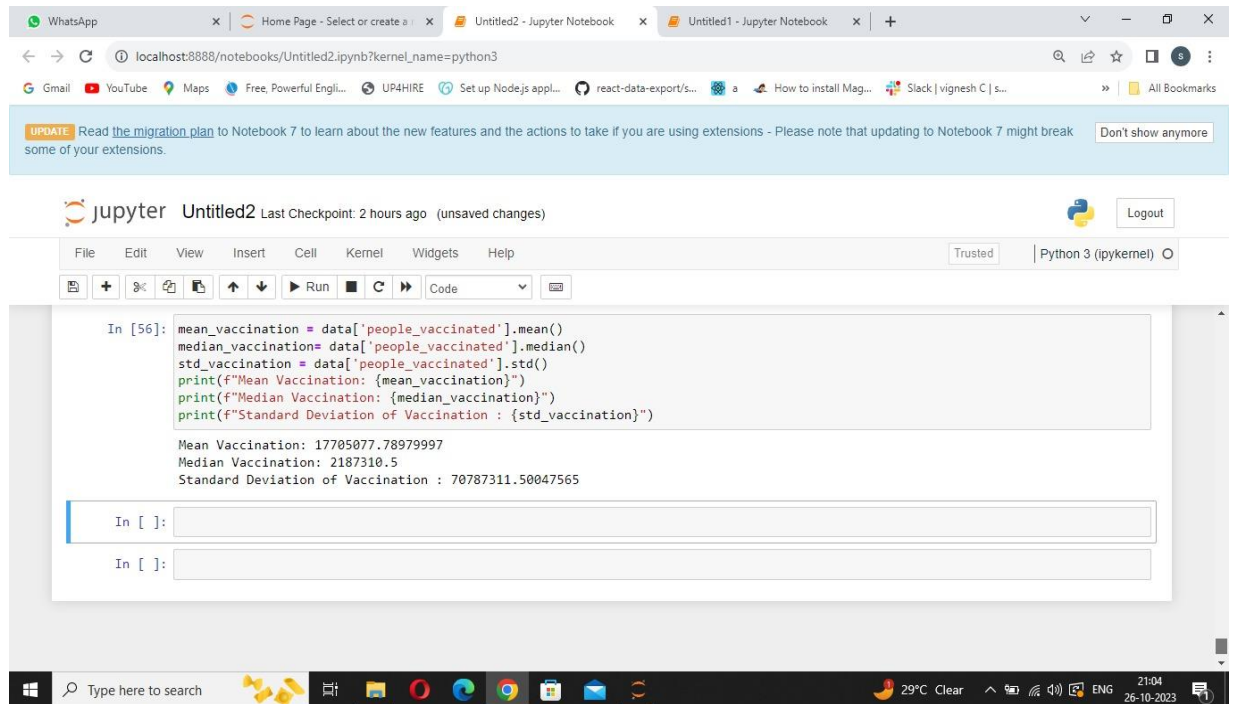
```
median_vaccination= data['people_vaccinated'].median()
```

```
std_vaccination = data['people_vaccinated'].std()
```

```
print(f"Mean Vaccination: {mean_vaccination}")
```

```
print(f"Median Vaccination: {median_vaccination}")
```

```
print(f"Standard Deviation of Vaccination : {std_vaccination}")
```



The screenshot shows a web browser window with a Jupyter Notebook interface. The browser's address bar shows 'localhost:8888/notebooks/Untitled2.ipynb?kernel\_name=python3'. The Jupyter Notebook interface includes a top bar with the Jupyter logo, the notebook name 'Untitled2', and a 'Logout' button. Below this is a menu bar with 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', 'Widgets', and 'Help'. A toolbar contains icons for file operations, running, and code execution. The main area displays a code cell with the following Python code:

```
In [56]: mean_vaccination = data['people_vaccinated'].mean()
median_vaccination = data['people_vaccinated'].median()
std_vaccination = data['people_vaccinated'].std()
print(f"Mean Vaccination: {mean_vaccination}")
print(f"Median Vaccination: {median_vaccination}")
print(f"Standard Deviation of Vaccination : {std_vaccination}")
```

The output of the code is displayed below the code cell:

```
Mean Vaccination: 17705077.78979997
Median Vaccination: 2187310.5
Standard Deviation of Vaccination : 70787311.50047565
```

Below the output, there are two empty input cells labeled 'In [ ]:'.

- `mean_vaccination = data['daily_vaccinations'].mean()`

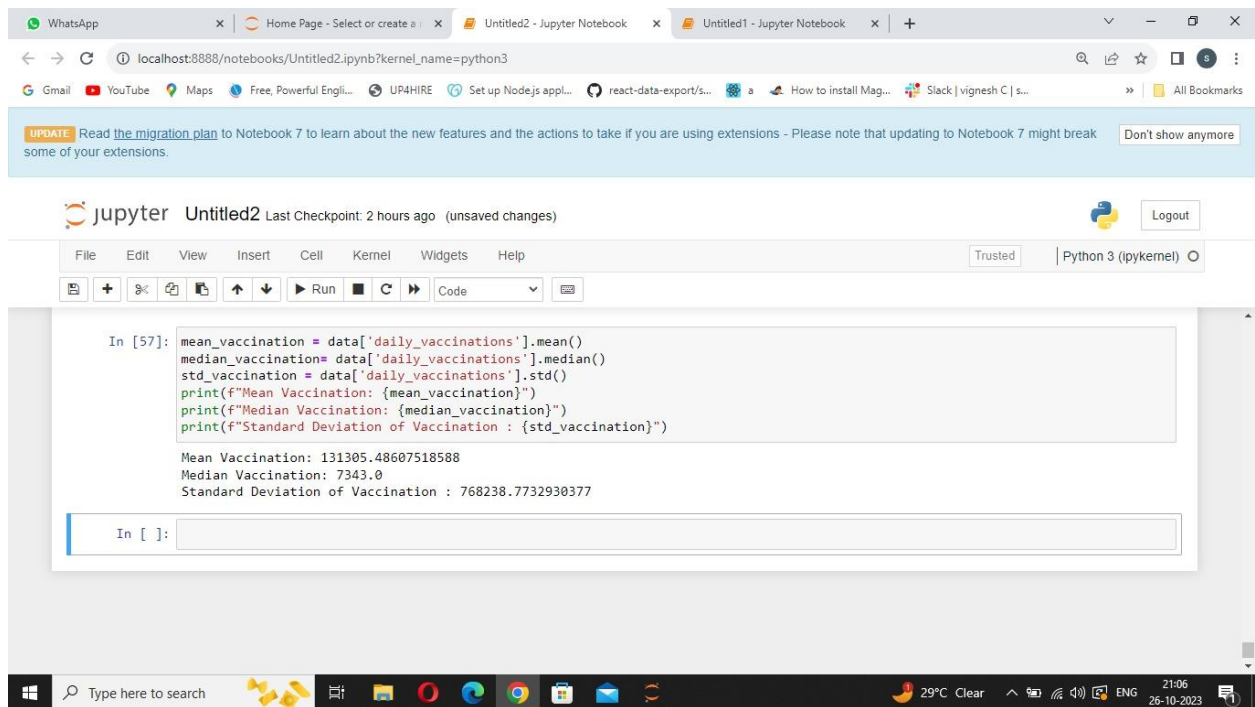
```
median_vaccination = data['daily_vaccinations'].median()
```

```
std_vaccination = data['daily_vaccinations'].std()
```

```
print(f"Mean Vaccination: {mean_vaccination}")
```

```
print(f"Median Vaccination: {median_vaccination}")
```

```
print(f"Standard Deviation of Vaccination : {std_vaccination}")
```



- **HYPOTHESIS TESTING**

*from scipy import stats*

*country\_A = data[data['country'] == 'country\_A']['total\_vaccinations']*

*country\_B = data[data['country'] == 'country\_B']['total\_vaccinations']*

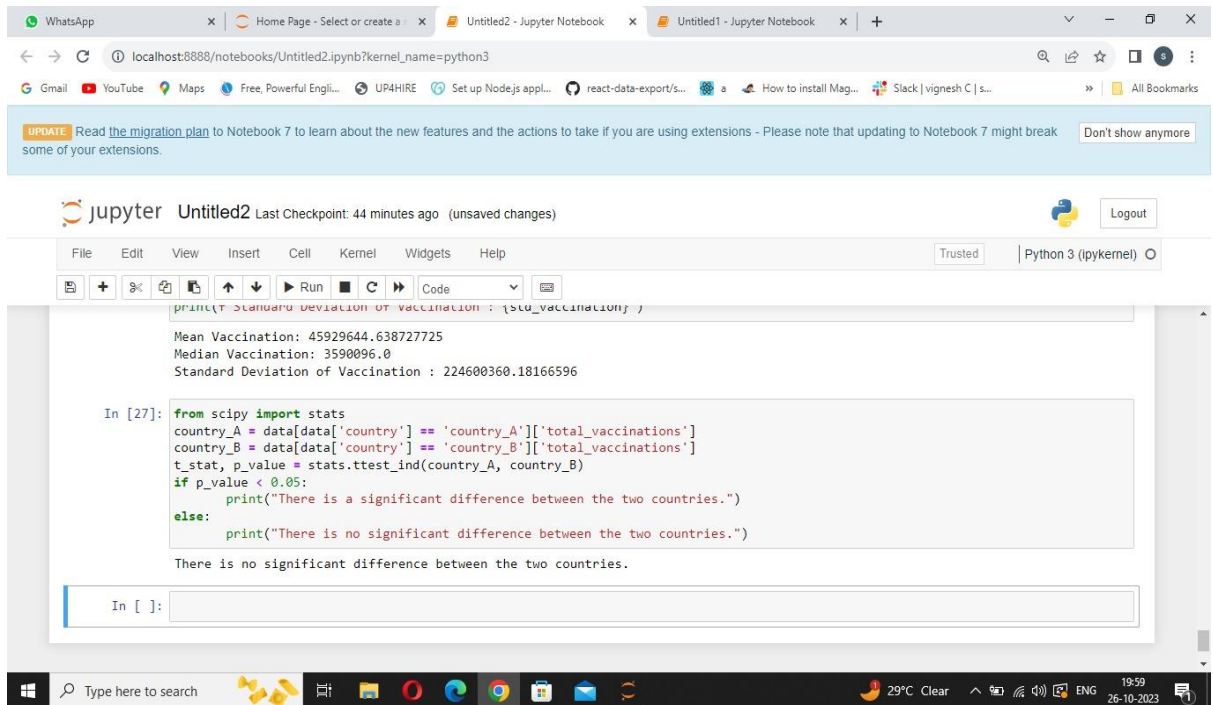
*t\_stat, p\_value = stats.ttest\_ind(country\_A, country\_B)*

*if p\_value < 0.05:*

*print("There is a significant difference between the two countries.")*

*else:*

*print("There is no significant difference between the two countries.")*



## VISUALISATION

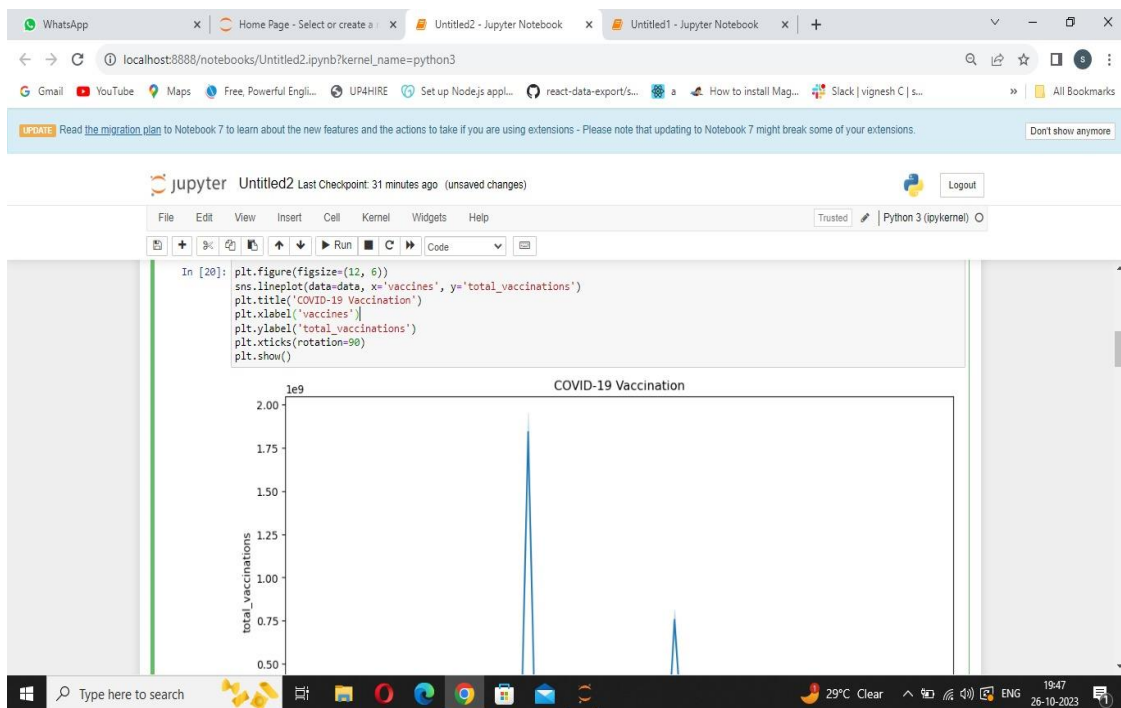
*Visualizations play a crucial role in understanding and presenting COVID-19 vaccine analysis. Here are some types of visualizations you can use:*

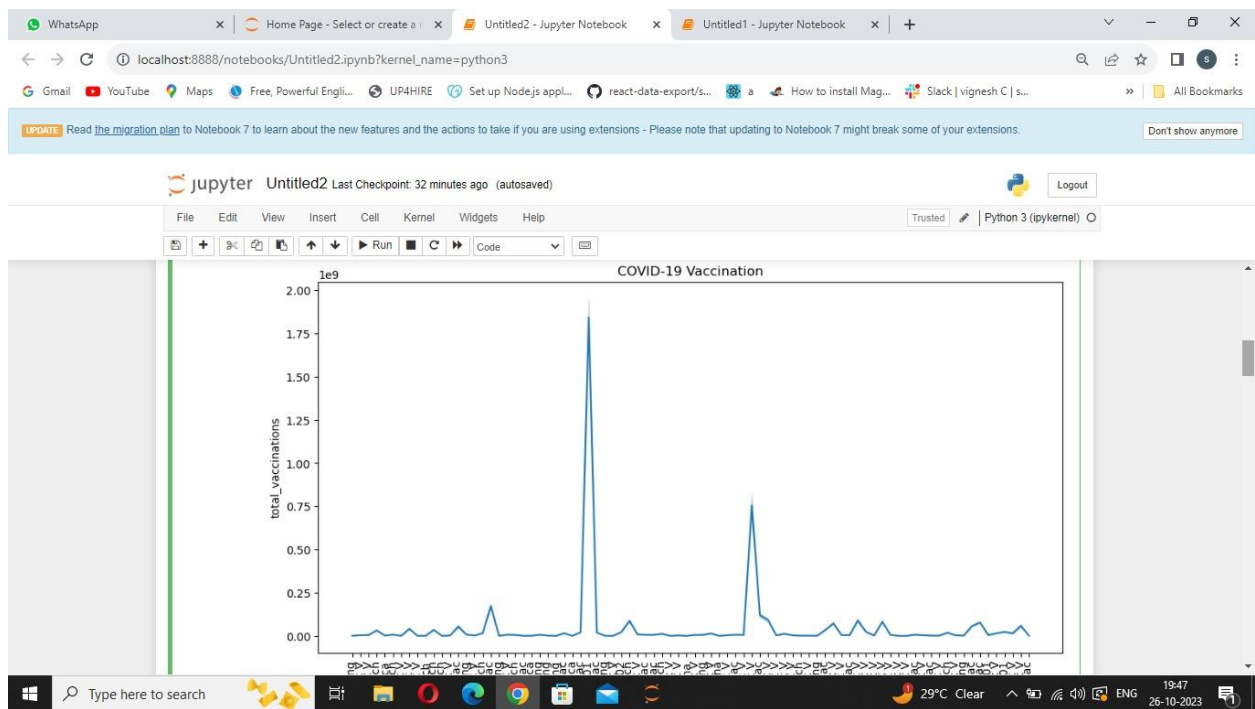
- 1. Bar Charts:** *Show the distribution of different vaccine types administered in a region or over time.*
- 2. Line Charts:** *Display trends in vaccination rates over time, including first and second doses administered.*
- 3. Area Charts:** *Visualize the cumulative number of vaccines administered over time to track progress.*
- 4. Stacked Bar Charts:** *Illustrate the breakdown of vaccine distribution by age group or gender.*
- 5. Heatmaps:** *Depict vaccination rates across regions or countries using color gradients.*

**6. Choropleth Maps:** Show vaccination coverage by shading areas on a map, with darker colors indicating higher coverage.

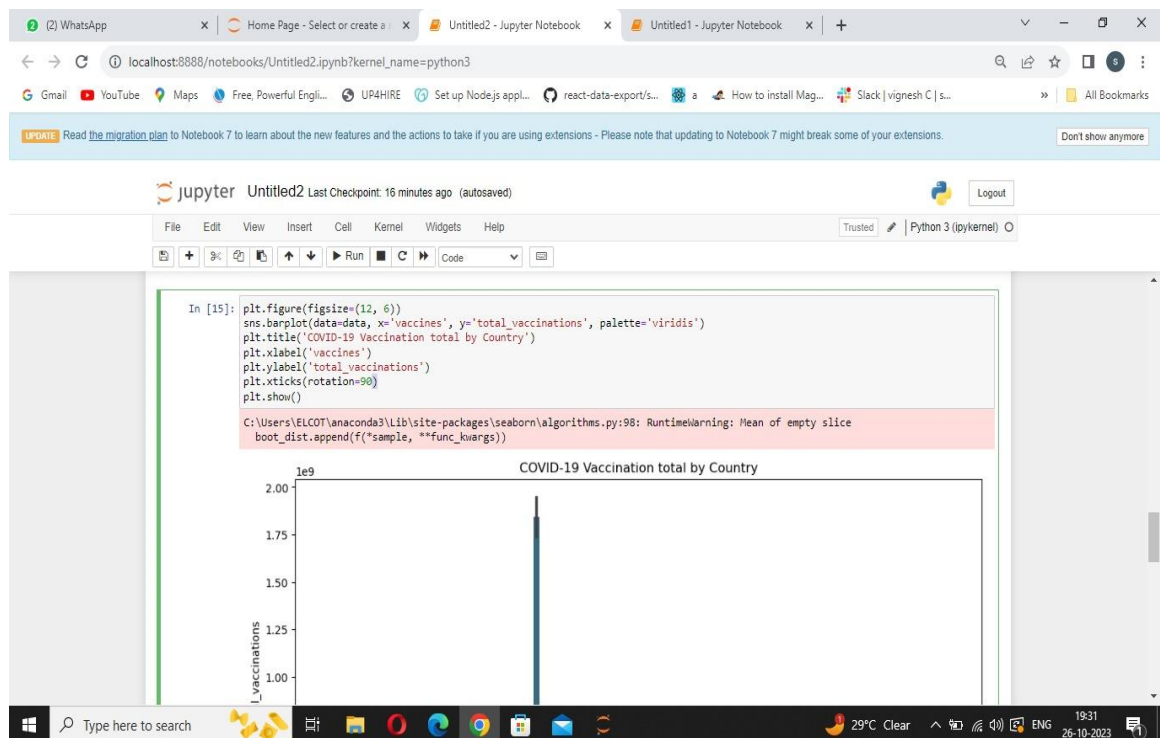
**7. Scatter Plots:** Explore correlations between vaccination rates and variables like COVID-19 cases, GDP, or healthcare infrastructure.

- create a line chart to utilize the progress of vaccination

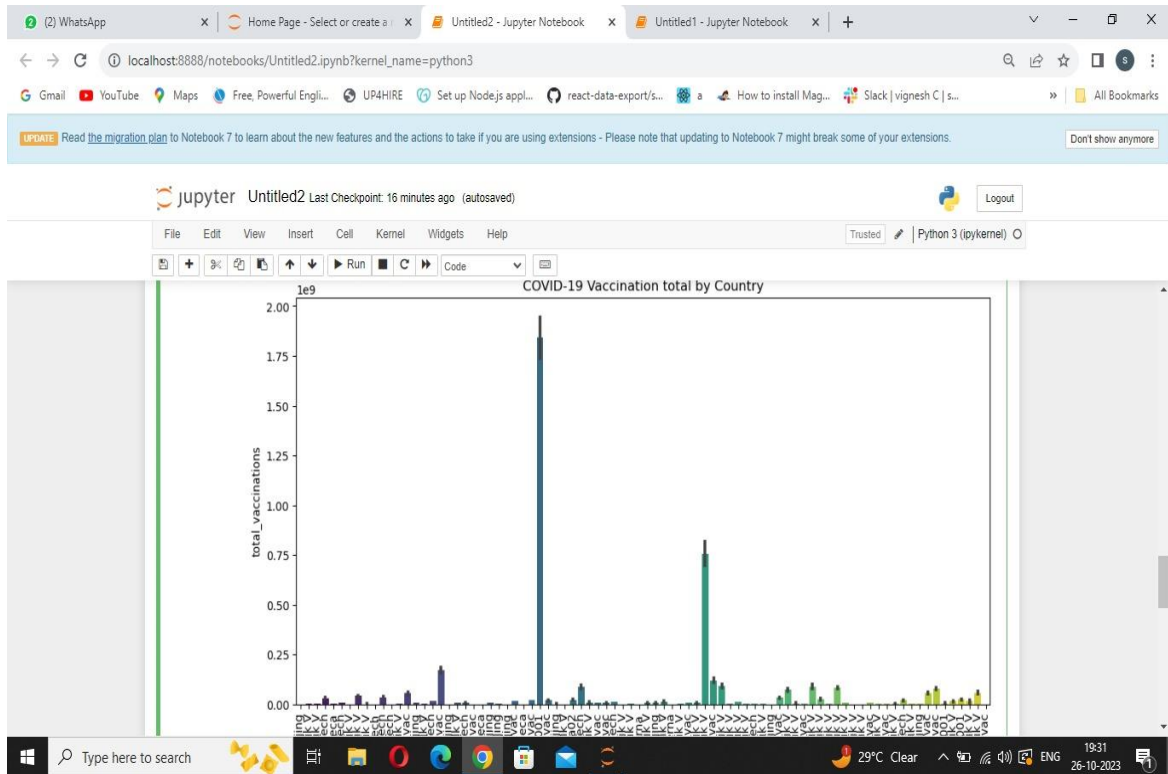




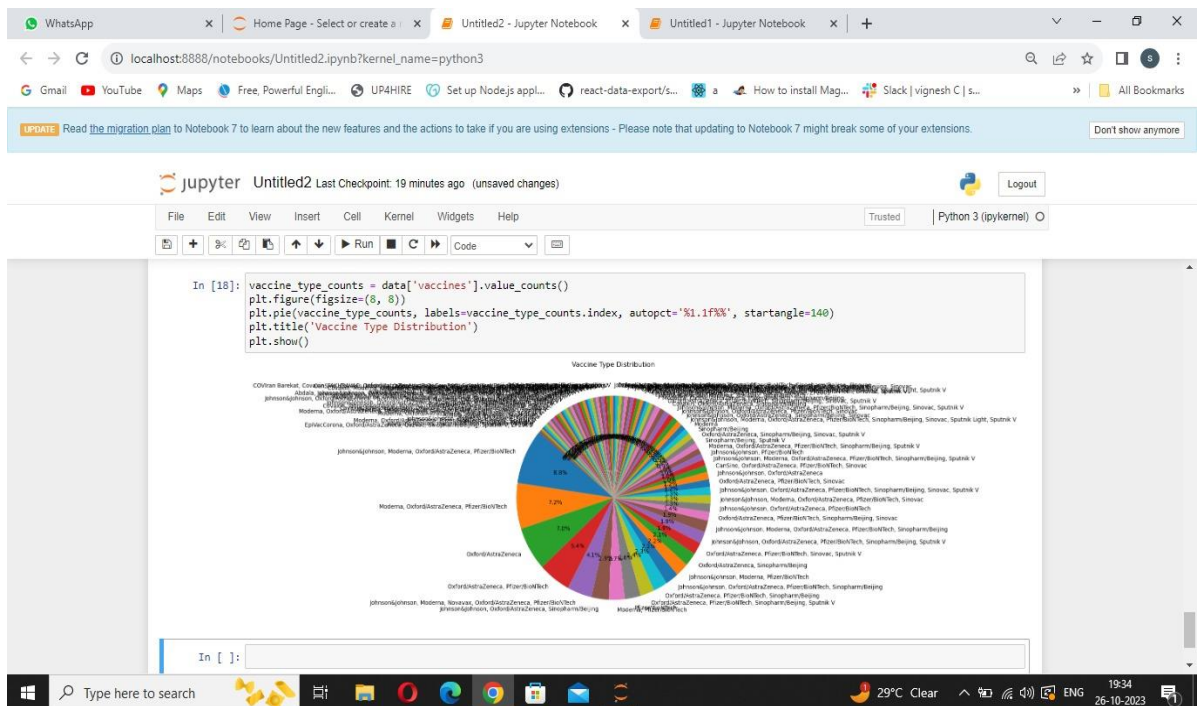
- *create a bar chart between total by country and total\_vaccination*







- **Create a pie chart to show the distribution of vaccine types**



## ***CONCLUSION***

***Thus, project covid-19 vaccines Analysis was completed successfully.***