COVID VACCINES ANALYSIS

TEAM MEMBER

962221106086-RASKIN

Phase – 3 Document Submission

Project : COVID VACCINES ANALYSIS

OBJECTIVE :

The objectives of COVID-19 vaccine analysis can vary depending on the specific purpose of the analysis. However, some common objectives include:

- To assess the safety and effectiveness of COVID-19 vaccines. This includes evaluating the vaccines' ability to prevent infection, serious illness, hospitalization, and death.
- To identify the optimal vaccination schedule. This includes determining the number of doses needed and the ideal time interval between doses.
- To understand how COVID-19 vaccines work and how they interact with the immune system. This information can be used to develop new and more effective vaccines.

- To identify and monitor the emergence of vaccine-resistant variants of the SARS-CoV-2 virus. This information can be used to update vaccines and develop new booster shots.

PHASE- 3

Loading and preprocessing the dataset.

The Covid-19 vaccine analysis by collecting and preprocessing the data.

INTRODUCTION:

To conduct a COVID-19 vaccine analysis, you will first need to collect and preprocess the dataset. This dataset may include information about vaccine distribution, administration, coverage, and possibly other relevant data. Here's a general outline of the steps you should follow:

1. Data Collection:

- Identify the sources from which you will collect COVID-19 vaccine data. Common sources include government health agencies, research institutions, and open data repositories.

- Download or access the data. Ensure that it's in a format that you can work with, such as CSV, Excel, or a database.

2. Data Exploration:

- Begin by loading the data into a data analysis tool, such as Python with libraries like Pandas, or R.

- Get a sense of what the dataset contains by examining its structure and the first few rows. Use the following code snippet in Python as an example:

```python
import pandas as pd

# Load the dataset into a Pandas DataFrame
data = pd.read_csv('covid_vaccine_data.csv')

# Display the first few rows
print(data.head())
```

3. Data Cleaning:

- Clean the data to ensure that it's accurate and suitable for analysis.

- Handle missing values, duplicates, and outliers.

- Convert data types if needed (e.g., dates to datetime objects).

4. Data Preprocessing:

- Preprocess the data according to the specific analysis you want to perform. This might include aggregating data, creating new features, or filtering relevant columns.

5. Data Visualization(Optional but highly recommended):

- Create visualizations to gain insights into the data. Visualization libraries like Matplotlib or Seaborn can be helpful for this.

6. Statistical Analysis:

- Perform the statistical analysis that you've planned. This could include calculating vaccine coverage rates, identifying trends over time, or analyzing factors influencing vaccine distribution and adoption.

7. Machine Learning (if applicable):

- If your analysis involves predictive modeling or machine learning, preprocess the data accordingly, create training and testing datasets, and build your model.

8. Interpretation and Reporting:

- Interpret the results of your analysis and draw conclusions. Create clear and informative reports or visualizations to present your findings.

It's important to note that the specific dataset and analysis will determine the exact preprocessing and analysis steps you need to follow. Make sure to choose the appropriate tools and techniques for your project, and ensure that you are working with up-to-date and reliable data sources, as the COVID-19 situation is continually evolving.

PROGRAM:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

selected_country_1= 'Japan'
selected_country_2= 'Japan'

df1 =
p
d.read_csv('../input/covid19-global-dataset/worldometer_coronavirus_daily_data.csv',h
eader=0)
df1 = df1[df1['country']==selected_country_1]
df2 =
p
d.read_csv('../input/covid-world-vaccination-progress/country_vaccinations.csv',heade
r=0)
df2 = df2[df2['country']==selected_country_2]
df1.index = pd.to_datetime(df1['date'])
```

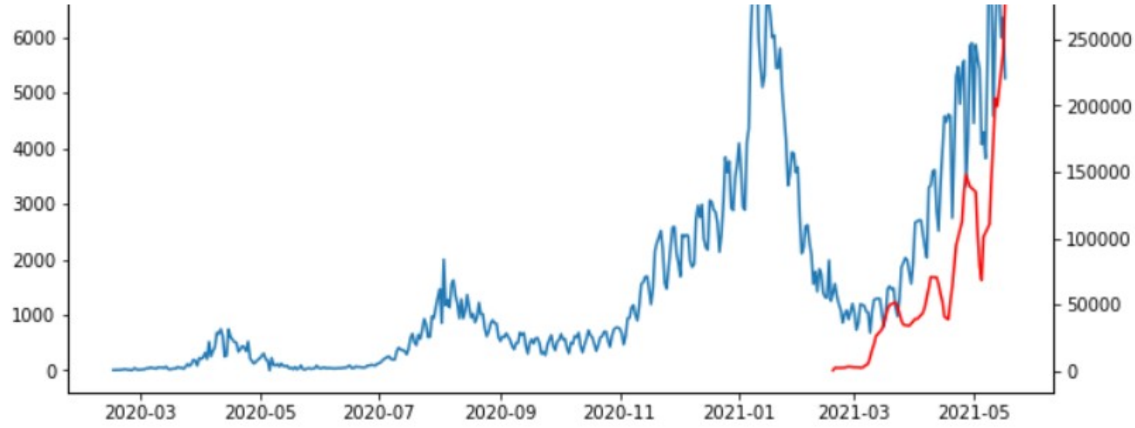
```
df2.index = pd.to_datetime(df2['date'])
#print(df2)


#analyses
#from here data will be analysed

fig,ax = plt.subplots(figsize=(10,5))

ax.plot(df1.index,df1['daily_new_cases'])
ax1 = ax.twinx()
ax1.plot(df2.index,df2['daily_vaccinations'],'r')
plt.show(
```

GRAPH:

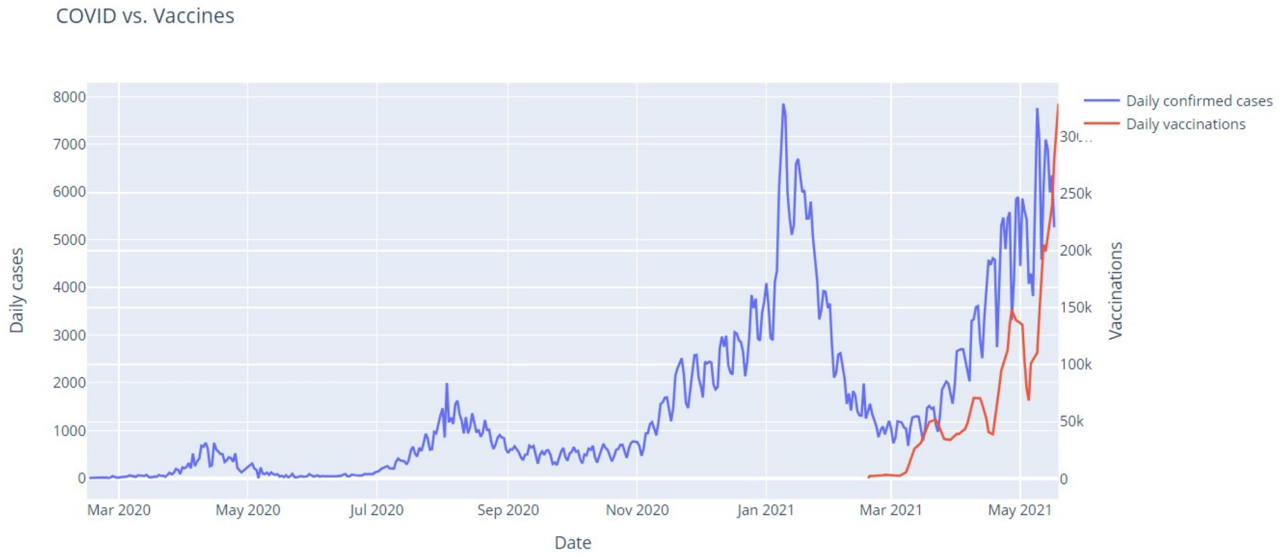


```
# improve visualization

from plotly.offline import iplot
import plotly.graph_objs as go

daily_new_cases =
go.Scatter(x=df1.index,y=df1['daily_new_cases'],yaxis='y1',name='Daily confirmed
cases')
daily_vaccination =
go.Scatter(x=df2.index,y=df2['daily_vaccinations'],yaxis='y2',name='Daily
vaccinations')

layout_obj = go.Layout(title='COVID vs. Vaccines',
xaxis=dict(title='Date'),yaxis=dict(title='Daily
cases'),yaxis2=dict(title='Vaccinations',side='right',overlaying='y'))
fig = go.Figure(data=[daily_new_cases,daily_vaccination],layout=layout_obj)
iplot(fig)
fig.write_html('outcome.html'
```


COVID vs. Vaccines
Daily confirmed cases
Daily vaccinations
Daily cases
8000
7000
6000
5000
4000
3000
2000
1000
0
Vaccinations
30
250k
200k
150k
100k
50k
0
Mar 2020
May 2020
Jul 2020
Sep 2020
Nov 2020
Jan 2021
Mar 2021
May 2021
Date