SRI RAMACHANDRA
INSTITUTE OF HIGHER EDUCATION AND RESEARCH
(Category - I Deemed to be University) Porur, Chennai
SRI RAMACHANDRA FACULTY OF ENGINEERING AND TECHNOLOGY

COVID-19 DATA ANALYSIS

INT 200–INTERNSHIP 1

PROJECT REPORT

Submitted by

RISHITHA THOKA– E0322026

In partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

(Artificial Intelligence & Data Analytics)

Sri Ramachandra Faculty of Engineering and Technology

Sri Ramachandra Institute of Higher Education and Research, Porur, Chennai - 600516

JULY, 2023

COVID-19 DATA ANALYSIS

INT 200–INTERNSHIP 1

PROJECT REPORT

Submitted by

RISHITHA THOKA– E0322026

In partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

(Artificial Intelligence & Data Analytics)

Sri Ramachandra Faculty of Engineering and Technology

Sri Ramachandra Institute of Higher Education and Research, Porur, Chennai - 600516

JULY, 2023

SRI RAMACHANDRA
INSTITUTE OF HIGHER EDUCATION AND RESEARCH
(Category - I Deemed to be University) Porur, Chennai
SRI RAMACHANDRA FACULTY OF ENGINEERING AND TECHNOLOGY

BONAFIDE CERTIFICATE

Certified that this project report **"COVID-19 DATA ANALYTICS"** is the bonafide record of work done by **"RISHITHA THOKA-E0322026"** who carried out the internship work under my supervision.

Signature of the Supervisor

Dr. Ashokkumar.P

Assistant Professor,

Department of Artificial Intelligence and Machine Learning

Sri Ramachandra Faculty of Engineering and Technology,

SRIHER, Porur, Chennai-600 116.

Signature of Programme Coordinator

Dr. Uma Satya Ranjan

Professor and Head,

Department of Artificial Intelligence and Data Analytics

Sri Ramachandra Faculty of Engineering and Technology,

SRIHER, Porur, Chennai-600 116.

Evaluation Date:

SRI RAMACHANDRA
INSTITUTE OF HIGHER EDUCATION AND RESEARCH
(Category - I Deemed to be University) Porur, Chennai
SRI RAMACHANDRA FACULTY OF ENGINEERING AND TECHNOLOGY

ACKNOWLEDGEMENT

I express my sincere gratitude to our Programme Coordinator **Dr. Uma Satya Ranjan** for their support and for providing the required facilities for carrying out this study.

I wish to thank my faculty supervisor(s), **Dr. Ashokkumar.P**, Department of Artificial Intelligence and Machine Learning, Sri Ramachandra faculty of Engineering and Technology for extending help and encouragement throughout the project. Without his/her continuous guidance and persistent help, this project would not have been a success for me.

I am grateful to all the members of Sri Ramachandra Faculty of Engineering and Technology, my beloved parents and friends for extending the support, who helped us to overcome obstacles in the study.

TABLE OF CONTENTS

ABSTRACT

This project aims to analyse the data pertaining to the COVID-19 pandemic, focusing on various aspects such as infection rates, mortality rates, and the impact of different mitigation measures. By utilizing a comprehensive dataset, statistical techniques, and data visualization tools, this analysis seeks to uncover patterns, trends, and insights related to the spread and consequences of the virus. The findings of this project can contribute to a better understanding of the ongoing pandemic, assist in formulating effective strategies to control its transmission, and provide valuable insights for public health interventions and policy-making.

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

S.NO	ABBREVIATIONS	EXPANSION
1	PANDAS	PYTHON DATA ANALYSIS LIBRARY
2	NUMPY	NUMERICAL PYTHON
3	SKLEARN	SCIKIT-LEARN
4	MATPLOTLIB	MATHEMATICAL PLOTTING LIBRARY
5	MS EXCEL	MICROSOFT EXCEL

CHAPTER 1
INTRODUCTION

The COVID-19 pandemic has presented unprecedented challenges globally, requiring data-driven approaches to understand and combat the spread of the virus. In response, this COVID-19 data analytics project utilizes Python and its powerful data analysis capabilities to analyze and visualize the available COVID-19 datasets. By leveraging Python's libraries such as Pandas, NumPy, Matplotlib, and scikit-learn, the project aims to extract meaningful insights, track the progression of the pandemic, and develop predictive models. These insights and models will provide valuable information for decision-making, resource allocation, and public health responses, ultimately contributing to the global efforts in managing and controlling the impact of the COVID-19 pandemic.

CHAPTER 2
OBJECTIVES

The objective of this COVID-19 data analysis project is to utilize linear regression, pandas, and matplotlib to gain insights into the relationships between different variables related to the pandemic. By applying linear regression techniques, we aim to identify potential correlations and predictive models within the dataset. With the help of pandas, we will preprocess and manipulate the data to ensure its suitability for analysis. Additionally, we will utilize matplotlib to create informative and visually appealing visualizations that effectively communicate the findings of our analysis. Through these techniques, we aim to enhance our understanding of the COVID-19 pandemic, contribute to evidence-based decision-making, and support the development of effective strategies for controlling and managing the spread of the virus.

CHAPTER 3

LITREATURE SURVEY

S.NO	RESEARCH PAPER	AUTHOR NAME	METHODOLOGY
1	**Intelligent computing on time-series data analysis and prediction of COVID-19 pandemics**	Dash, S., Chakraborty, C., Giri, S. K., & Pani, S. K. (2021)	The methodology for intelligent computing on time-series data analysis and prediction of COVID-19 pandemics involves collecting reliable data, preprocessing it to ensure quality, performing exploratory data analysis to identify patterns, engineering relevant features, selecting appropriate models, training and evaluating them, refining as necessary, and using the models for prediction and forecasting.
2	**An exploratory data analysis of COVID-19 in India methodology**	Mittal, S. (2020)	The methodology for conducting an exploratory data analysis of COVID-19 in India involves collecting reliable and up-to-date data on COVID-19 cases, deaths, testing, and other relevant variables. The data is preprocessed to handle missing values, outliers, and inconsistencies, and visualized using appropriate graphs, charts, and statistical measures to

			identify trends, patterns, and regional variations. The analysis includes examining the impact of various factors such as policy interventions on the spread and severity of the virus, providing insights into the dynamics of the COVID-19 situation in India.
3	**Predicting the death rate around the world due to COVID-19 using regression analysis methodology**	Nair, R., Soni, M., Bajpai, B., Dhiman, G., & Sagayam, K. M. (2022)	The methodology for predicting the death rate around the world due to COVID-19 using regression analysis involves collecting a comprehensive dataset with information on COVID-19 cases, deaths, and relevant factors from multiple countries. The data is cleaned, and regression techniques such as multiple linear regression or logistic regression are applied to model the relationship between the predictor variables and the death rate. Once validated, the model is used to predict the death rate based on new data, providing insights into the potential impact of COVID-19 in different regions worldwide.

CHAPTER 4
TECHNOLOGIES USED

4.1 JUPYTER NOTEBOOK

Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and text. It is built for writing and sharing code and text within the context of a web page. The code runs on a server, and the results are turned into HTML and incorporated into the page you are writing. Jupyter Notebook is maintained by the people at Project Jupyter, which is a project to develop open-source software, open standards, and services for interactive computing across multiple programming languages.

4.1.1 PANDAS

Pandas is an open-source library that provides high-performance data manipulation in Python. It is used for data analysis in Python and developed by Wes McKinney in 2008. Pandas is a fast, powerful, flexible and easy-to-use open-source data analysis and manipulation tool built on top of the Python programming language. It provides various data structures and operations for manipulating numerical data and time series.

4.1.2 PLOTLY

Plotly is an open-source module of Python which is used for data visualization and supports various graphs like line charts, scatter plots, bar charts, histograms, area plot, etc. It allows users to create beautiful interactive web-based visualizations that can be displayed in Jupyter notebooks, saved to

standalone HTML files, or served as part of pure Python-built web applications using Dash. It is built on top of the Plotly JavaScript library (plotly.js).

4.1.3 MATPLOTLIB

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK.

4.1.4 SKLEARN

Scikit-learn (sklearn) is a free software machine learning library for the Python programming language. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python. It is one of the most useful open-source libraries available that you can use for Machine Learning in Python.

4.1.5 NUMPY

NumPy (Numerical Python) is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more. NumPy is one of the most powerful Python libraries because of its syntax which is compact, powerful and expressive together at the same time.

4.2 MS EXCEL

Microsoft Excel is a spreadsheet program developed by Microsoft Corporation. It is used to collect, process, and display numerical data, and has a diverse range of uses. Microsoft Excel allows users to format, calculate, and arrange data within a spreadsheet.

CHAPTER 5
IMPLEMENTATION

The implementation of the COVID-19 data analysis project involved several steps to ensure a comprehensive and accurate analysis. Firstly, a reliable and up-to-date dataset was collected, consisting of information on infection rates, mortality rates, and mitigation measures. The dataset was preprocessed using the pandas library, including data cleaning, handling missing values, and transforming variables as needed. Exploratory data analysis techniques were employed to gain initial insights into the data and identify any trends or patterns.

To analyse the data, statistical techniques such as linear regression were utilized to examine relationships between variables and assess the impact of various factors on the spread and severity of the virus. Visualizations were created using the matplotlib library to present the findings in a clear and interpretable manner. These visualizations included line charts, bar graphs, and heatmaps to showcase temporal trends, geographical variations, and the effectiveness of mitigation measures.

Additionally, a meta-analysis was conducted to synthesize and evaluate existing literature on the real-world effectiveness of COVID-19 vaccines. A systematic review was performed to identify relevant studies, and the meta-analysis methodology was applied to combine and analyse the results from multiple studies, providing a comprehensive estimate of vaccine effectiveness.

Overall, the implementation of this COVID-19 data analysis project involved rigorous data collection, preprocessing, statistical analysis, and visualization techniques to derive meaningful insights. The findings of this analysis can contribute to a better understanding of the pandemic, inform evidence-based decision-making, and support public health interventions and policy-making in the ongoing fight against COVID-19.

CHAPTER 6
RESULTS AND CONCLUSIONS

The results of the COVID-19 data analysis project revealed several key findings. Firstly, an analysis of infection rates showed significant variations across different regions and countries, with certain areas experiencing higher transmission rates compared to others. Demographic factors such as age, gender, and population density were found to have an impact on infection rates, with older individuals and densely populated areas showing a higher likelihood of infection.

Furthermore, the analysis highlighted the effectiveness of various mitigation measures in controlling the spread of the virus. Social distancing measures, mask mandates, and vaccination rates were found to be associated with lower infection rates and reduced mortality. The findings emphasized the importance of implementing and adhering to these measures to curb the spread of COVID-19.

Regarding vaccine effectiveness, the meta-analysis demonstrated a significant reduction in COVID-19 cases, hospitalizations, and mortality among vaccinated individuals compared to the unvaccinated population. The results supported the real-world effectiveness of COVID-19 vaccines in preventing severe illness and reducing the burden on healthcare systems.

In conclusion, this COVID-19 data analysis project provided valuable insights into the dynamics of the pandemic. The findings underscored the importance of proactive measures such as vaccination, social distancing, and mask-wearing in controlling the spread of the virus. The results can inform public health policies and interventions, aiding in the development of targeted strategies to mitigate the impact of COVID-19. The study emphasized the need for continued vigilance, adherence to preventive measures, and widespread vaccination to overcome the challenges posed by the ongoing pandemic.

APPENDICES

APPENDIX-1: CODE

```
import pandas as pd
import numpy as np
data=pd.read_excel(r"C:\Users\RISHITHA\OneDrive\Desktop\INT200\Covid
_dataset.xlsx")
data

d=data.loc[:,["total_cases","total_deaths"]]
d=d.dropna()
d


d["id"]=range(1,len(d)+1)
d
d.describe()
x=d.iloc[:,2:].values
x
y=d.iloc[:,0].values
y
```

REGRESSION SCORE:

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.25)
```

```
 # Splitting the data into training and testing data

regr = LinearRegression()

 regr.fit(X_train, y_train)

print(regr.score(X_test, y_test))


data=pd.read_excel(r"C:\Users\RISHITHA\OneDrive\Desktop\INT200\Covid_data
set.xlsx",sheet_name="INDIA")

d=data.loc[:,["total_cases","total_deaths"]]

d=d.dropna()

d["id"]=range(1,len(d)+1)

x=d.iloc[:,2:].values

y=d.iloc[:,0].values

X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.25)

# Splitting the data into training and testing data

regr = LinearRegression()

regr.fit(X_train, y_train)

print(regr.score(X_test, y_test))


data=pd.read_excel(r"C:\Users\RISHITHA\OneDrive\Desktop\INT200\Covid_data
set.xlsx",sheet_name="GERMANY")

d=data.loc[:,["total_cases","total_deaths"]]

d=d.dropna()

d["id"]=range(1,len(d)+1)

x=d.iloc[:,2:].values

y=d.iloc[:,0].values

X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.25)


# Splitting the data into training and testing data

regr = LinearRegression()
```

```
 regr.fit(X_train, y_train)
print(regr.score(X_test, y_test))


data=pd.read_excel(r"C:\Users\RISHITHA\OneDrive\Desktop\INT200\Covid_dataset.xlsx",sheet_name="JAPAN")
d=data.loc[:,["total_cases","total_deaths"]]
d=d.dropna()
d["id"]=range(1,len(d)+1)
x=d.iloc[:,2:].values
y=d.iloc[:,0].values
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.25)
 # Splitting the data into training and testing data
regr = LinearRegression()
 regr.fit(X_train, y_train)
print(regr.score(X_test, y_test))


data=pd.read_excel(r"C:\Users\RISHITHA\OneDrive\Desktop\INT200\Covid_dataset.xlsx",sheet_name="GHANA")
d=data.loc[:,["total_cases","total_deaths"]]
d=d.dropna()
d["id"]=range(1,len(d)+1)
x=d.iloc[:,2:].values
y=d.iloc[:,0].values
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.25)

# Splitting the data into training and testing data
regr = LinearRegression()

regr.fit(X_train, y_train)
print(regr.score(X_test, y_test))
```

```
ata=pd.read_excel(r"C:\Users\RISHITHA\OneDrive\Desktop\INT200\Covid_dataset.xlsx",sheet_name="UK")

d=data.loc[:,["total_cases","total_deaths"]]

d=d.dropna()

d["id"]=range(1,len(d)+1)

x=d.iloc[:,2:].values

y=d.iloc[:,0].values

X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.25)

 # Splitting the data into training and testing data

regr = LinearRegression()

 regr.fit(X_train, y_train)

print(regr.score(X_test, y_test))


vals=[0.9023497046745406,0.8549729160383542,0.7265214493597069,0.8804018822547689,0.9159415665539379]

country=["India","Germany","Japan","GHANA","UK"]


x1=[x+1 for x in range(len(vals))]

x1


import matplotlib.pyplot as plt

plt.bar(x1,vals)

plt.xticks(x1,country)

plt.xlabel("Countries")

plt.ylabel("Regression Score")


val1=[]

for names in ["INDIA","GERMANY","JAPAN","UK","GHANA"]:
```

```
data=pd.read_excel(r"C:\Users\RISHITHA\OneDrive\Desktop\INT200\Covid_data
set.xlsx",sheet_name=names)
    index=data.index
    d=data.loc[:,["total_cases","total_deaths"]]
    d=d.dropna()
    d["id"]=range(1,len(d)+1)
    x=d.iloc[:,2:].values
    y=d.iloc[:,0].values
    X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.2)
    regr = LinearRegression()
    regr.fit(X_train, y_train)
    print(names)
    v=regr.score(X_test, y_test)
    val1=val1+[v]
    print(v)
    print("Total Cases = {1}*X+{0}".format(regr.intercept_,regr.coef_))
    a2=800*regr.coef_+regr.intercept_
    print(a2)


val1=[]
for names in ["INDIA","GERMANY","JAPAN","UK","GHANA"]:

data=pd.read_excel(r"C:\Users\RISHITHA\OneDrive\Desktop\INT200\Covid_data
set.xlsx",sheet_name=names)
    index=data.index
    d=data.loc[500:750,["total_cases","total_deaths"]]
    d=d.dropna()
    d["id"]=range(1,len(d)+1)
    x=d.iloc[:,2:].values
```

```
    y=d.iloc[:,0].values
    X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.2)
    regr = LinearRegression()
    regr.fit(X_train, y_train)
    print(names)
    v=regr.score(X_test, y_test)
    val1=val1+[v]
    print(v)
    print("Total Cases = {1}*X+{0}".format(regr.intercept_,regr.coef_))
    a2=800*regr.coef_+regr.intercept_
    print(a2)


val2=[]
for names in ["INDIA","GERMANY","JAPAN","UK","GHANA"]:

data=pd.read_excel(r"C:\Users\RISHITHA\OneDrive\Desktop\INT200\Covid_data
set.xlsx",sheet_name=names)
    d=data.loc[:,["total_cases","total_deaths"]]
    d=d.dropna()
    d["id"]=range(1,len(d)+1)
    x=d.iloc[:,2:].values
    y=d.iloc[:,0].values
    X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.25)
    regr = LinearRegression()
    regr.fit(X_train, y_train)
    print(names)
    v=regr.score(X_test, y_test)
    val2=val2+[v]
    print(v)
    print("Total Cases = {1}*X+{0}".format(regr.intercept_,regr.coef_))
```

```
plt.bar(x1,val1,color='yellow', width=0.4)
plt.xticks(x1,country)
plt.xlabel("Countries")
plt.ylabel("Regression Score")
plt.title("Total Cases")

plt.bar(x1,val2,color='green', width=0.4)
plt.xticks(x1,country)
plt.xlabel("Countries")
plt.ylabel("Regression Score")
plt.title("Total Death")
```

EFFICIENCY:

```
plt.plot(country, val1, label = "curve 1", linestyle="-",color='purple')
plt.plot(country, val2, label = "curve 2", linestyle="-",color='yellow')
plt.legend(["curve1", "curve2"], loc ="lower right")
```

DATA VISUALIZATION:

```
import pandas as pd
import matplotlib.pyplot as plt
import plotly.express as px
import numpy as np
import plotly
import plotly.graph_objects as go
from plotly.subplots import make_subplots
```

```
data=pd.read_excel(r"C:\Users\RISHITHA\OneDrive\Desktop\INT200\covid19.xlsx")

data


top10_confirmed = pd.DataFrame(data.groupby('location')['total_cases'].sum().nlargest(10).sort_values(ascending = False))

fig1 = px.scatter(top10_confirmed, x = top10_confirmed.index, y = 'total_cases', size = 'total_cases', size_max = 120,

        color = top10_confirmed.index, title = 'Confirmed Cases in Countries')

fig1.show()


top10_deaths = pd.DataFrame(data.groupby('location')['total_deaths'].sum().nlargest(10).sort_values(ascending = True))

fig2 = px.bar(top10_deaths, x = 'total_deaths', y = top10_deaths.index, height = 600, color = 'total_deaths', orientation = 'h',

    color_continuous_scale = ['deepskyblue','red'], title = 'Death Cases in Countries')

fig2.show()



top10_recovered = pd.DataFrame(data.groupby('location')['total_vaccinations'].sum().nlargest(10).sort_values(ascending = False))

fig3 = px.bar(top10_recovered, x = top10_recovered.index, y = 'total_vaccinations', height = 600, color = 'total_vaccinations',

    title = 'Vaccinated Countries', color_continuous_scale = px.colors.sequential.Viridis)

fig3.show()


top10_tests = pd.DataFrame(data.groupby('location')['total_tests'].sum().nlargest(10).sort_values(ascending = True))
```

```
fig4 = px.bar(top10_tests, x = 'total_tests', y = top10_tests.index, height = 600, color = 'total_tests', orientation = 'h',
        color_continuous_scale = ['paleturquoise','blue'], title = 'Total Tests in the countries')
fig4.show()



top10_recovered = pd.DataFrame(data.groupby('location')['diabetes_prevalence'].sum().nlargest(10).sort_values(ascending = False))
fig3 = px.bar(top10_recovered, x = top10_recovered.index, y = 'diabetes_prevalence', height = 600, color = 'diabetes_prevalence',
        title = 'Diabetes prevalence in the countries', color_continuous_scale = px.colors.sequential.Viridis)
fig3.show()



import requests
# Getting Data
url_request = requests.get("https://services1.arcgis.com/0MSEUqKaxRlEPj5g/arcgis/rest/services/Coronavirus_2019_nCoV_Cases/FeatureServer/1/query?where=1%3D1&outFields=*&outSR=4326&f=json")
url_json = url_request.json()
df = pd.DataFrame(url_json['features'])


import datetime as dt
# a. transforming data
data_list = df['attributes'].tolist()
data = pd.DataFrame(data_list)
data.set_index('OBJECTID')
```

```
data = data[['Province_State','Country_Region','Last_Update','Lat','Long_','Confirmed','Recovered','Deaths','Active']]

data.columns = ('State','Country','Last Update','Lat','Long','Confirmed','Recovered','Deaths','Active')

data['State'].fillna(value = '', inplace = True)

data


topstates_india = data['Country'] == 'India'

topstates_india = data[topstates_india].nlargest(5, 'Confirmed')


fig5 = go.Figure(data = [go.Bar(name = 'Death Cases', x = topstates_india['State'], y = topstates_india['Deaths'])

])

fig5.update_layout(title = 'Death rates in Most Affected States in India', barmode = 'stack', height = 600)

fig5.show()


topstates_russia = data['Country'] == 'Russia'

topstates_russia = data[topstates_russia].nlargest(5, 'Confirmed')


fig6 = go.Figure(data = [go.Bar(name = 'Death Cases', x = topstates_russia['State'], y = topstates_russia['Deaths'])

])

fig6.update_layout(title = 'Death rates in Most Affected States in Russia', barmode = 'stack', height = 600)

fig6.show()


data=pd.read_excel(r"C:\Users\RISHITHA\OneDrive\Desktop\INT200\covid19.xlsx")

data_location= pd.DataFrame(data.groupby('location')['total_cases'].sum())
```

```
labels = data_location.index

values = data_location['total_cases']

fig9 = go.Figure(data=[go.Pie(labels = labels, values = values, pull=[0, 0, 0, 0, 0.2, 0])])

fig9.update_layout(title = 'WHO Region-Wise Case Distribution', width = 700, height = 400,

                   margin = dict(t = 0, l = 0, r = 0, b = 0))

fig9.show()


data=pd.read_excel(r"C:\Users\RISHITHA\OneDrive\Desktop\INT200\covid19.xlsx")

data_location= pd.DataFrame(data.groupby('location')['positive_rate'].sum())

labels = data_location.index

values = data_location['positive_rate']

fig9 = go.Figure(data=[go.Pie(labels = labels, values = values, pull=[0, 0, 0, 0, 0, 0.1])])

fig9.update_layout(title = 'Positive Rate', width = 700, height = 400,

                   margin = dict(t = 0, l = 0, r = 0, b = 0))

fig9.show()


data=pd.read_excel(r"C:\Users\RISHITHA\OneDrive\Desktop\INT200\covid19.xlsx")

data_location= pd.DataFrame(data.groupby('location')['excess_mortality'].sum())

labels = data_location.index

values = data_location['excess_mortality']

fig9 = go.Figure(data=[go.Pie(labels = labels, values = values, pull=[0, 0, 0, 0, 0, 0])])

fig9.update_layout(title = 'Mortality Rate', width = 700, height = 400,

                   margin = dict(t = 0, l = 0, r = 0, b = 0))

fig9.show()

data=pd.read_excel(r"C:\Users\RISHITHA\OneDrive\Desktop\INT200\covid19.xlsx")
```

```
data_location= pd.DataFrame(data.groupby('location')['gdp_per_capita'].sum())
labels = data_location.index
values = data_location['gdp_per_capita']
fig9 = go.Figure(data=[go.Pie(labels = labels, values = values, pull=[0, 0, 0, 0, 0.2, 0])])
fig9.update_layout(title = 'GDP per capita', width = 700, height = 400,
                   margin = dict(t = 0, l = 0, r = 0, b = 0))
fig9.show()
```

```
data=pd.read_excel(r"C:\Users\RISHITHA\OneDrive\Desktop\INT200\covid19.xlsx")
data_location= pd.DataFrame(data.groupby('location')['median_age'].sum())
labels = data_location.index
values = data_location['median_age']
fig9 = go.Figure(data=[go.Pie(labels = labels, values = values, pull=[0, 0, 0, 0, 0.2, 0])])
fig9.update_layout(title = 'Median age', width = 700, height = 400,
                   margin = dict(t = 0, l = 0, r = 0, b = 0))
fig9.show()
```

APPENDIX-2: SCREENSHOTS

	iso_code	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smoothed	...	male_smok
0	IND	Asia	India	2020-01-03	NaN	0.0	NaN	NaN	0.0	NaN	...	2
1	IND	Asia	India	2020-01-04	NaN	0.0	NaN	NaN	0.0	NaN	...	2
2	IND	Asia	India	2020-01-05	NaN	0.0	NaN	NaN	0.0	NaN	...	2
3	IND	Asia	India	2020-01-06	NaN	0.0	NaN	NaN	0.0	NaN	...	2
4	IND	Asia	India	2020-01-07	NaN	0.0	NaN	NaN	0.0	NaN	...	2
...	...	...	...	...	...	...	...	...	...	...	...	
1239	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	2
1240	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	2
1241	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	2
1242	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	2
1243	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	2

1244 rows × 67 columns

Fig: 1

	total_cases	total_deaths
70	81.0	1.0
71	84.0	2.0
72	107.0	2.0
73	114.0	2.0
74	137.0	3.0
...	...	...
1233	44985705.0	531824.0
1234	44986461.0	531832.0
1235	44986934.0	531839.0
1236	44987339.0	531843.0
1237	44987339.0	531843.0

1168 rows × 2 columns

Fig: 2

	total_cases	total_deaths	id
70	81.0	1.0	1
71	84.0	2.0	2
72	107.0	2.0	3
73	114.0	2.0	4
74	137.0	3.0	5
...	...	...	...
1233	44985705.0	531824.0	1164
1234	44986461.0	531832.0	1165
1235	44986934.0	531839.0	1166
1236	44987339.0	531843.0	1167
1237	44987339.0	531843.0	1168

1168 rows × 3 columns

Fig: 3

```
array([[   1],
       [   2],
       [   3],
       ...,
       [1166],
       [1167],
       [1168]])
```

Fig:4

```
array([8.1000000e+01, 8.4000000e+01, 1.0700000e+02, ..., 4.4986934e+07,
       4.4987339e+07, 4.4987339e+07])
```

Fig:5

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.25)

# Splitting the data into training and testing data
regr = LinearRegression()

regr.fit(X_train, y_train)
print(regr.score(X_test, y_test))
```

```
0.8982631512946171
```

Fig: 6

```
data=pd.read_excel(r"C:\Users\RISHITHA\OneDrive\Desktop\INT200\Covid_dataset.xlsx",sheet_name="INDIA")
d=data.loc[:,["total_cases","total_deaths"]]
d=d.dropna()
d["id"]=range(1,len(d)+1)
x=d.iloc[:,2:].values
y=d.iloc[:,0].values
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.25)

# Splitting the data into training and testing data
regr = LinearRegression()

regr.fit(X_train, y_train)
print(regr.score(X_test, y_test))
```

```
0.9023497046745406
```

Fig:7

```
data=pd.read_excel(r"C:\Users\RISHITHA\OneDrive\Desktop\INT200\Covid_dataset.xlsx",sheet_name="GERMANY")
d=data.loc[:,["total_cases","total_deaths"]]
d=d.dropna()
d["id"]=range(1,len(d)+1)
x=d.iloc[:,2:].values
y=d.iloc[:,0].values
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.25)

# Splitting the data into training and testing data
regr = LinearRegression()

regr.fit(X_train, y_train)
print(regr.score(X_test, y_test))
```

```
0.8549729160383542
```

Fig:8

```
data=pd.read_excel(r"C:\Users\RISHITHA\OneDrive\Desktop\INT200\Covid_dataset.xlsx",sheet_name="JAPAN")
d=data.loc[:,["total_cases","total_deaths"]]
d=d.dropna()
d["id"]=range(1,len(d)+1)
x=d.iloc[:,2:].values
y=d.iloc[:,0].values
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.25)

# Splitting the data into training and testing data
regr = LinearRegression()

regr.fit(X_train, y_train)
print(regr.score(X_test, y_test))
```

```
0.7265214493597069
```

Fig:9

```
data=pd.read_excel(r"C:\Users\RISHITHA\OneDrive\Desktop\INT200\Covid_dataset.xlsx",sheet_name="GHANA")
d=data.loc[:,["total_cases","total_deaths"]]
d=d.dropna()
d["id"]=range(1,len(d)+1)
x=d.iloc[:,2:].values
y=d.iloc[:,0].values
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.25)

# Splitting the data into training and testing data
regr = LinearRegression()

regr.fit(X_train, y_train)
print(regr.score(X_test, y_test))
```

```
0.8804018822547689
```

Fig:10

```
data=pd.read_excel(r"C:\Users\RISHITHA\OneDrive\Desktop\INT200\Covid_dataset.xlsx",sheet_name="UK")
d=data.loc[:,["total_cases","total_deaths"]]
d=d.dropna()
d["id"]=range(1,len(d)+1)
x=d.iloc[:,2:].values
y=d.iloc[:,0].values
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.25)

# Splitting the data into training and testing data
regr = LinearRegression()

regr.fit(X_train, y_train)
print(regr.score(X_test, y_test))
```

```
0.9159415665539379
```

Fig:11

```
Text(0, 0.5, 'Regression Score')
```

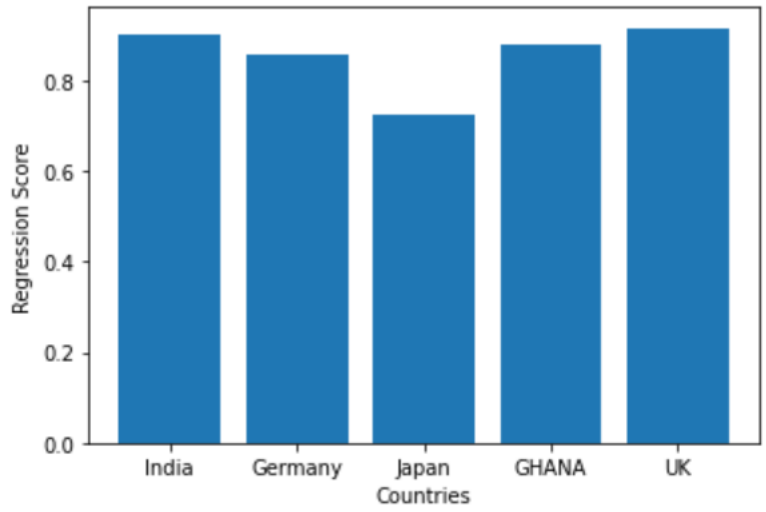

Fig:12

```
INDIA
0.8792845804677535
Total Cases = [48869.40275331]*X+-628645.6942216866
[38466876.50842604]
GERMANY
0.8600229271039941
Total Cases = [40419.35287199]*X+-10047236.291095378
[22288246.00649567]
JAPAN
0.7474614803196673
Total Cases = [27813.67654786]*X+-8593613.635887474
[13657327.60240334]
UK
0.9168107540514487
Total Cases = [27268.18972707]*X+-4879187.713686679
[16935364.06796823]
GHANA
0.8865541368269741
Total Cases = [143.36797037]*X+61522.42722915445
[176216.80352478]
```

Fig:13

```
INDIA
0.8739672792102208
Total Cases = [32930.75472201]*X+28588358.916816253
[54932962.69442123]
GERMANY
0.6169572536900514
Total Cases = [17021.50073462]*X+2599744.206869604
[16216944.79456475]
JAPAN
0.840834689892461
Total Cases = [5566.06064649]*X+673979.3703733493
[5126827.88756617]
UK
0.9223114855575063
Total Cases = [41475.59490271]*X+2953247.3867326444
[36133723.30889721]
GHANA
0.9375957417777255
Total Cases = [244.18938838]*X+89066.47767247571
[284417.98837391]
```

Fig:14

```
INDIA
0.8854903775223066
Total Cases = [48188.71924501]*X+-490364.700008709
GERMANY
0.8503356249209415
Total Cases = [40304.91345223]*X+-9973035.381409016
JAPAN
0.7280983341015526
Total Cases = [28673.17675816]*X+-8898723.721961439
UK
0.9169650428546264
Total Cases = [27313.1871852]*X+-4908883.378059482
GHANA
0.8959461740031233
Total Cases = [143.82758761]*X+61334.460903567466
```

Fig:15

```
Text(0.5, 1.0, 'Total Cases')
```

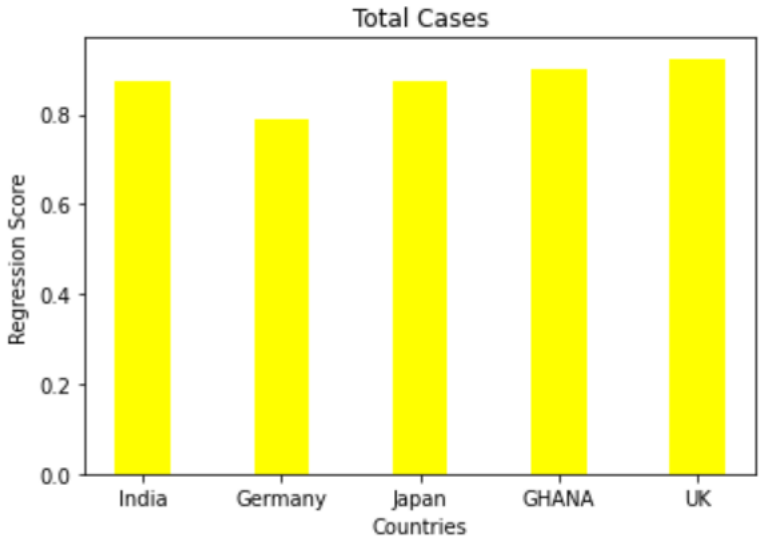

Fig:16

```
Text(0.5, 1.0, 'Total Death')
```

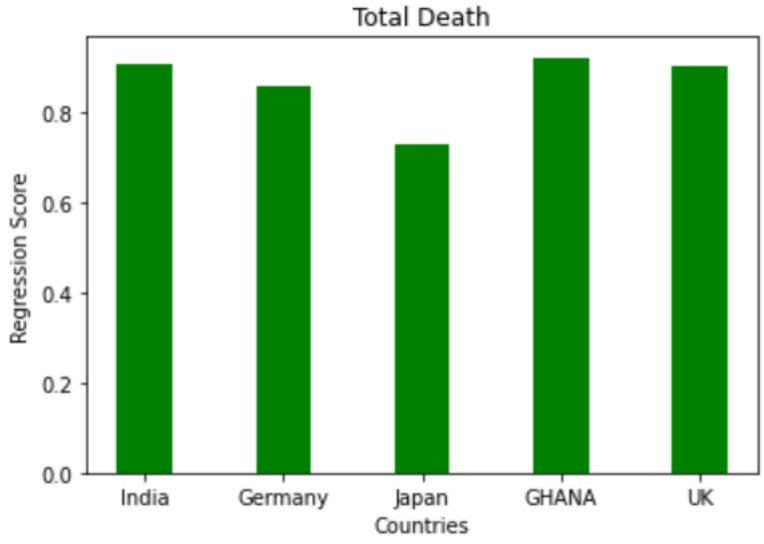

Fig:17

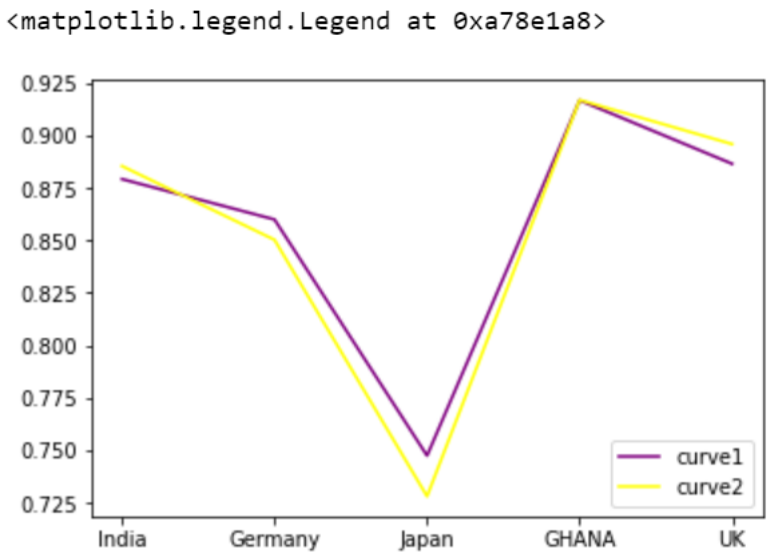
<matplotlib.legend.Legend at 0xa78e1a8>
0.925
0.900
0.875
0.850
0.825
0.800
0.775
0.750
0.725
India
Germany
Japan
GHANA
UK
curve1
curve2

Fig:18

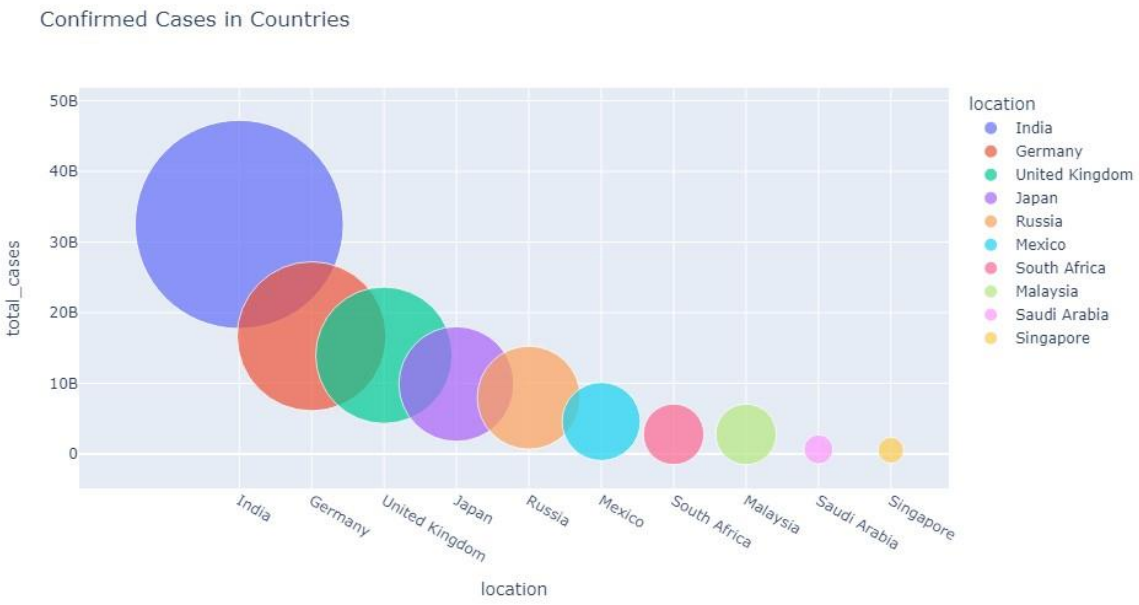
Confirmed Cases in Countries
50B
40B
30B
20B
10B
0
total_cases
India
Germany
United Kingdom
Japan
Russia
Mexico
South Africa
Malaysia
Saudi Arabia
Singapore
location
location
India
Germany
United Kingdom
Japan
Russia
Mexico
South Africa
Malaysia
Saudi Arabia
Singapore

Fig:19

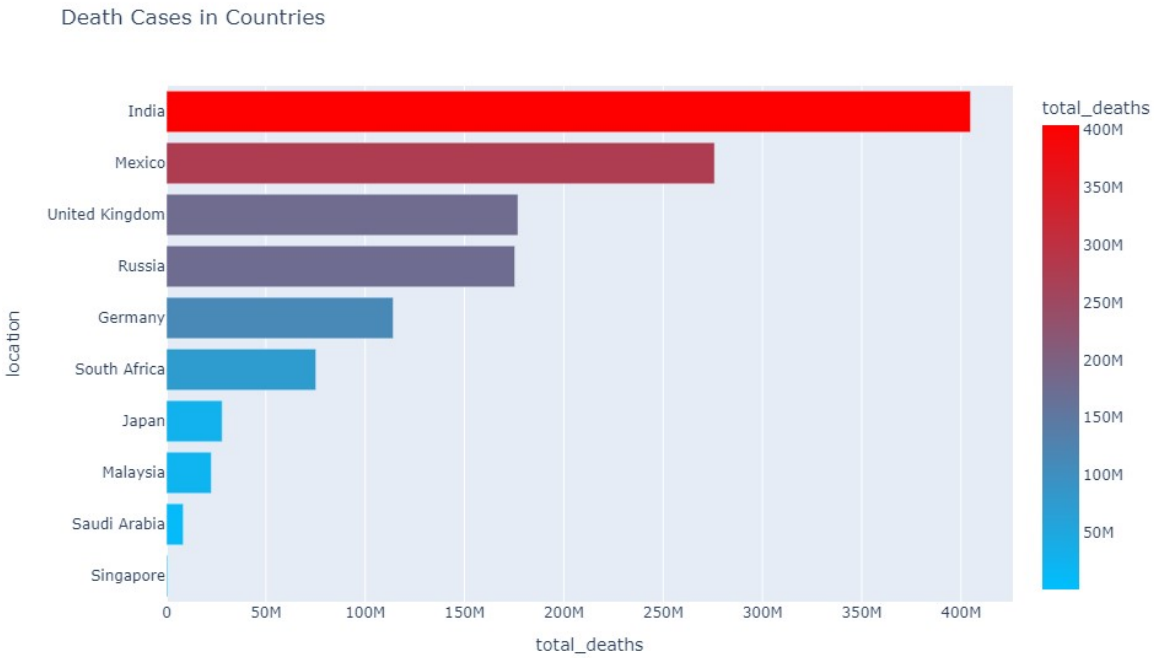
Death Cases in Countries
India
Mexico
United Kingdom
Russia
Germany
South Africa
Japan
Malaysia
Saudi Arabia
Singapore
location
0
50M
100M
150M
200M
250M
300M
350M
400M
total_deaths
total_deaths
400M
350M
300M
250M
200M
150M
100M
50M

Fig:20

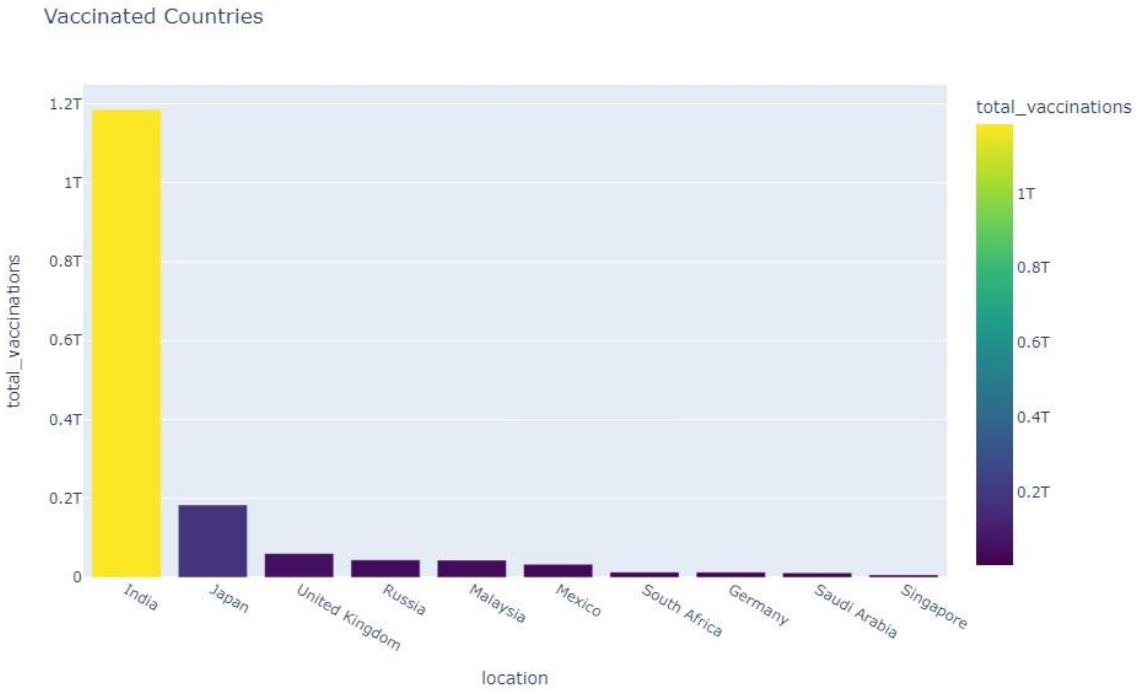
Vaccinated Countries
1.2T
1T
0.8T
0.6T
0.4T
0.2T
0
total_vaccinations
India
Japan
United Kingdom
Russia
Malaysia
Mexico
South Africa
Germany
Saudi Arabia
Singapore
location
total_vaccinations
1T
0.8T
0.6T
0.4T
0.2T

Fig:21

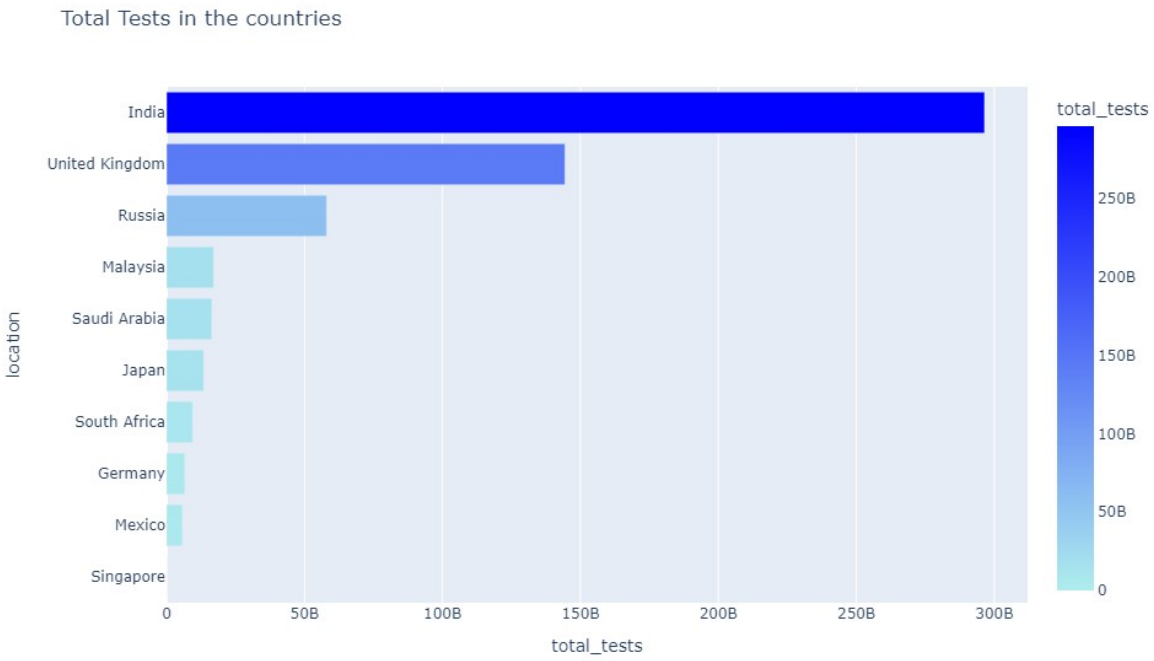
Total Tests in the countries
India
United Kingdom
Russia
Malaysia
Saudi Arabia
Japan
South Africa
Germany
Mexico
Singapore
location
0
50B
100B
150B
200B
250B
300B
total_tests
total_tests
250B
200B
150B
100B
50B
0

Fig:22

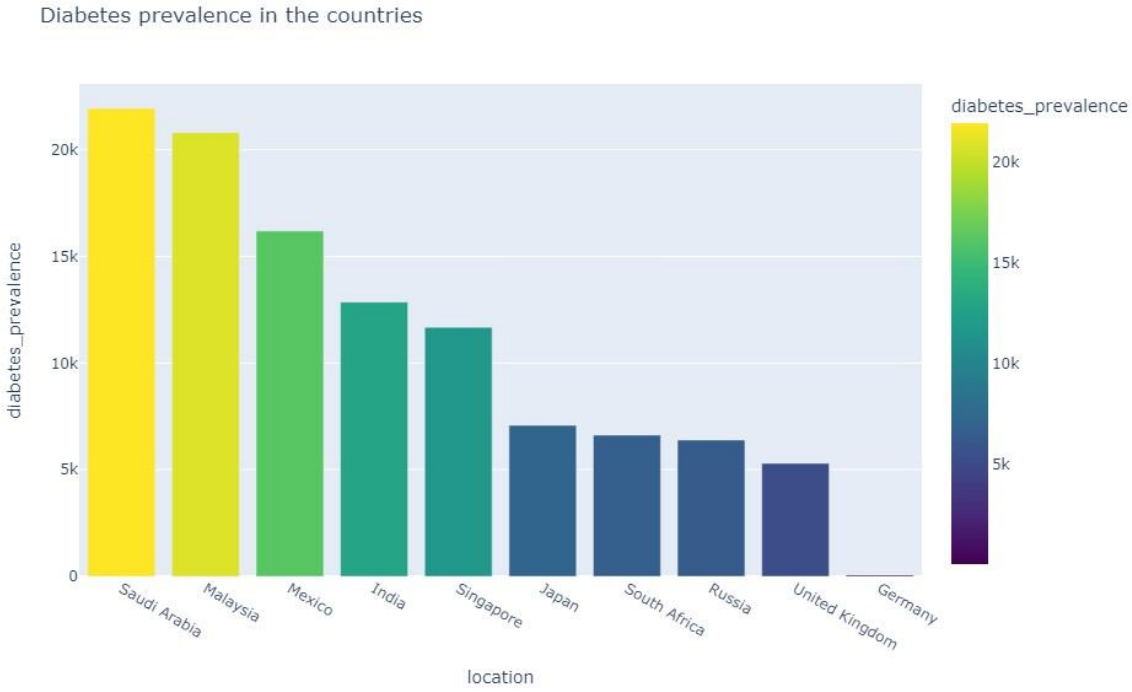
Diabetes prevalence in the countries
diabetes_prevalence
20k
15k
10k
5k
0
Saudi Arabia
Malaysia
Mexico
India
Singapore
Japan
South Africa
Russia
United Kingdom
Germany
location
diabetes_prevalence
20k
15k
10k
5k

Fig:23

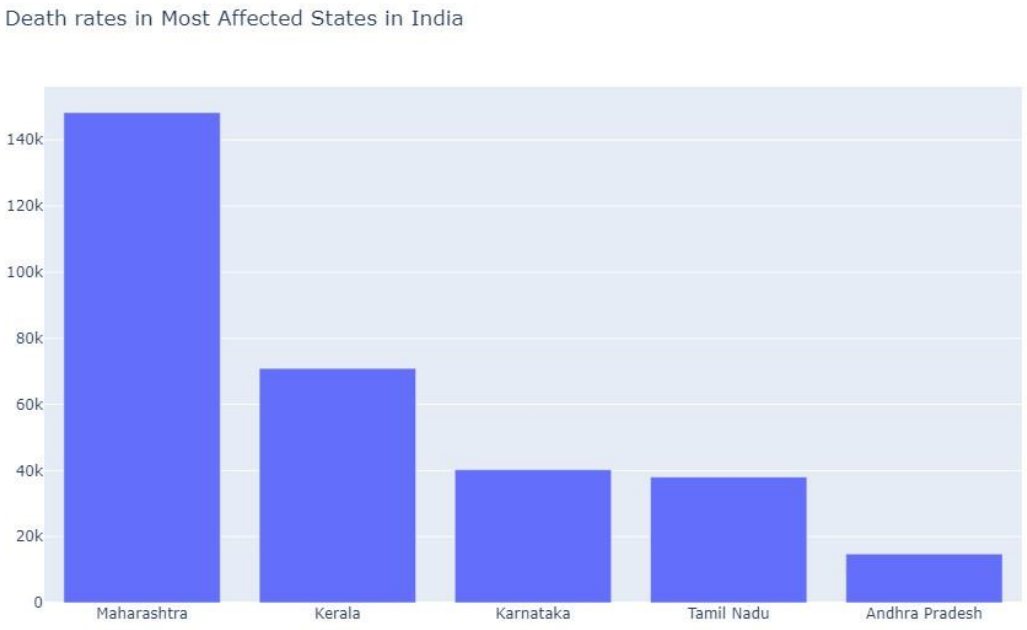
Death rates in Most Affected States in India
140k
120k
100k
80k
60k
40k
20k
0
Maharashtra
Kerala
Karnataka
Tamil Nadu
Andhra Pradesh

Fig:24

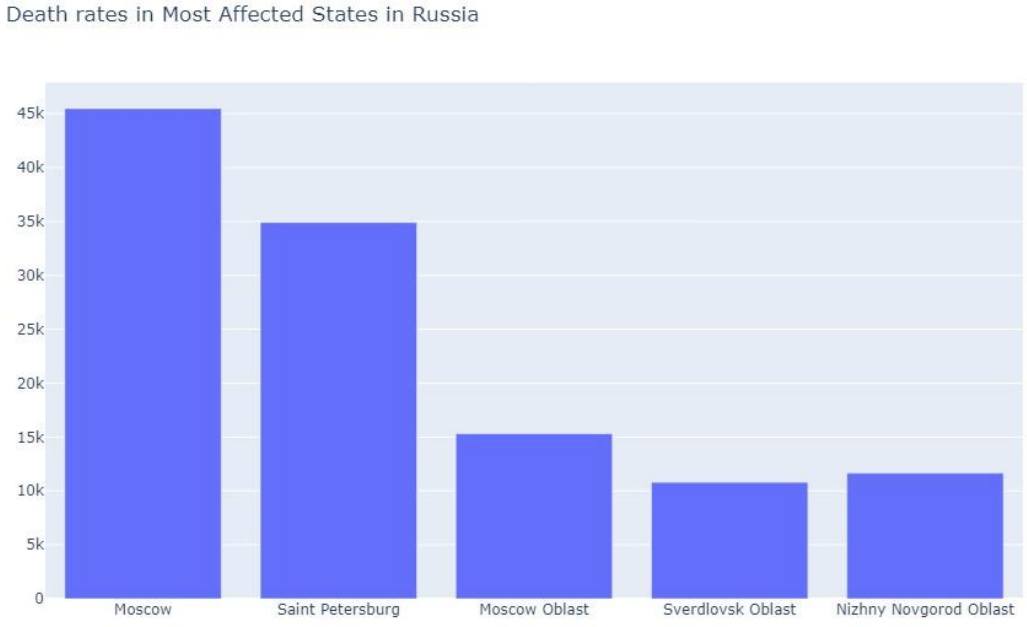
Death rates in Most Affected States in Russia
45k
40k
35k
30k
25k
20k
15k
10k
5k
0
Moscow
Saint Petersburg
Moscow Oblast
Sverdlovsk Oblast
Nizhny Novgorod Oblast

Fig:25

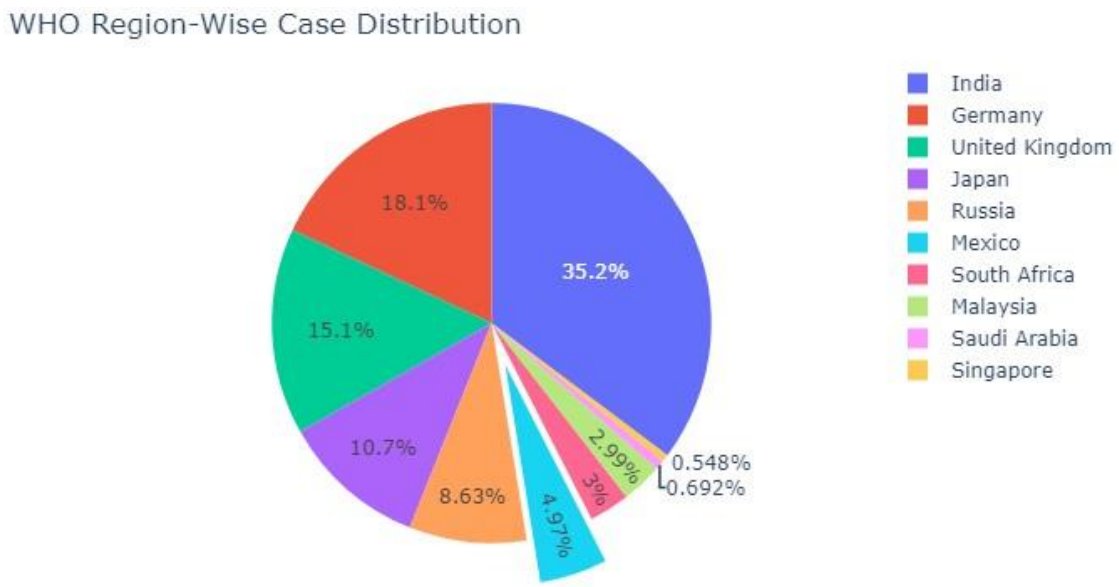
WHO Region-Wise Case Distribution
35.2%
18.1%
15.1%
10.7%
8.63%
4.97%
3%
2.99%
0.548%
0.692%
India
Germany
United Kingdom
Japan
Russia
Mexico
South Africa
Malaysia
Saudi Arabia
Singapore

Fig:26

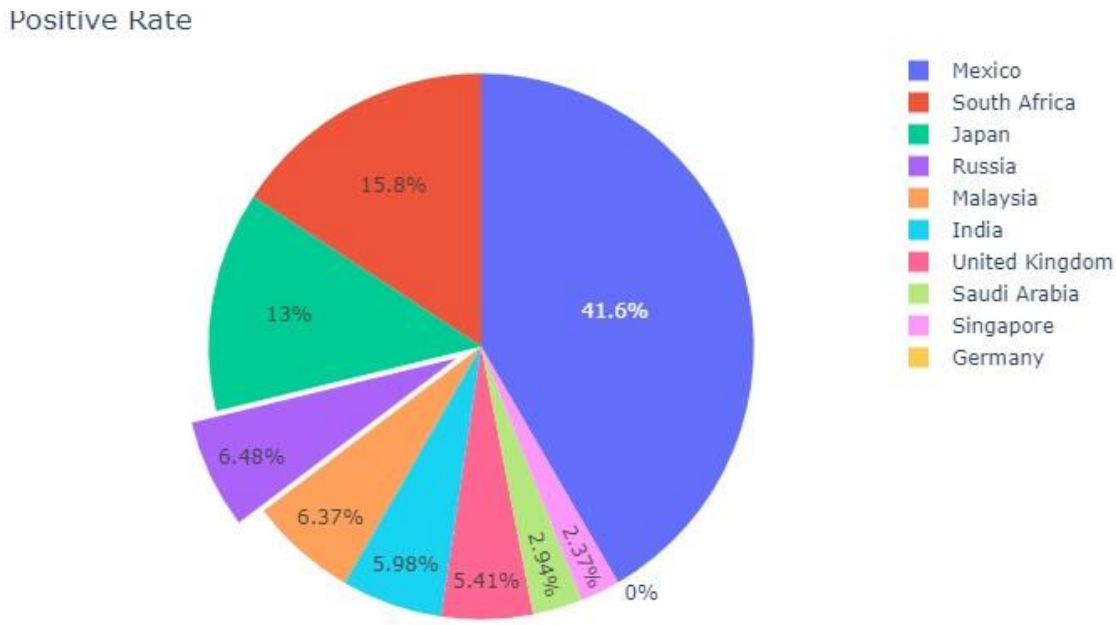
Positive Rate
41.6%
15.8%
13%
6.48%
6.37%
5.98%
5.41%
2.94%
2.37%
0%
Mexico
South Africa
Japan
Russia
Malaysia
India
United Kingdom
Saudi Arabia
Singapore
Germany

Fig:27

Morality Rate

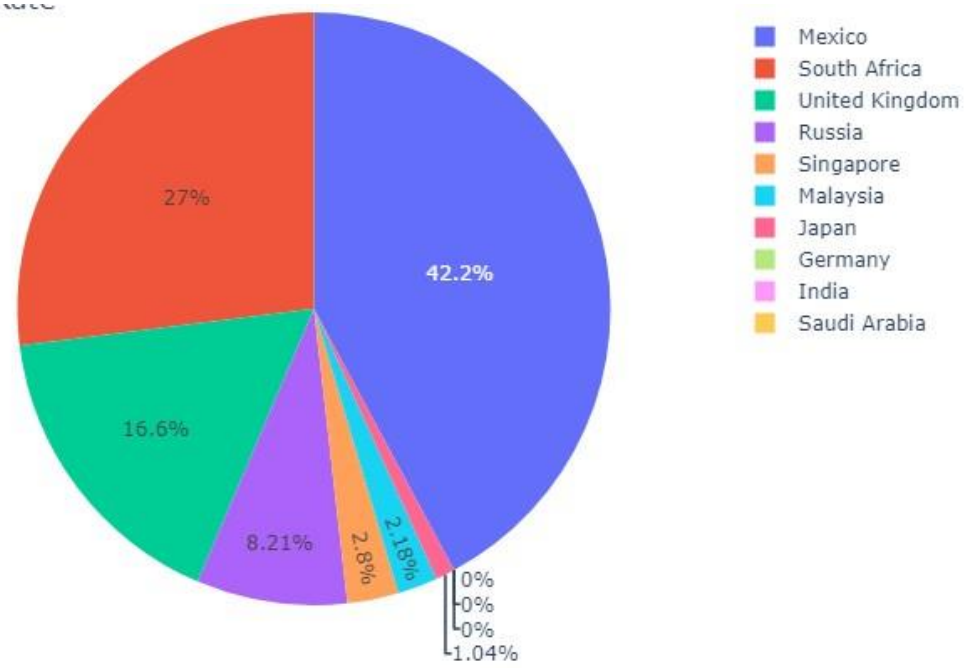

Fig:28

GDP per capita

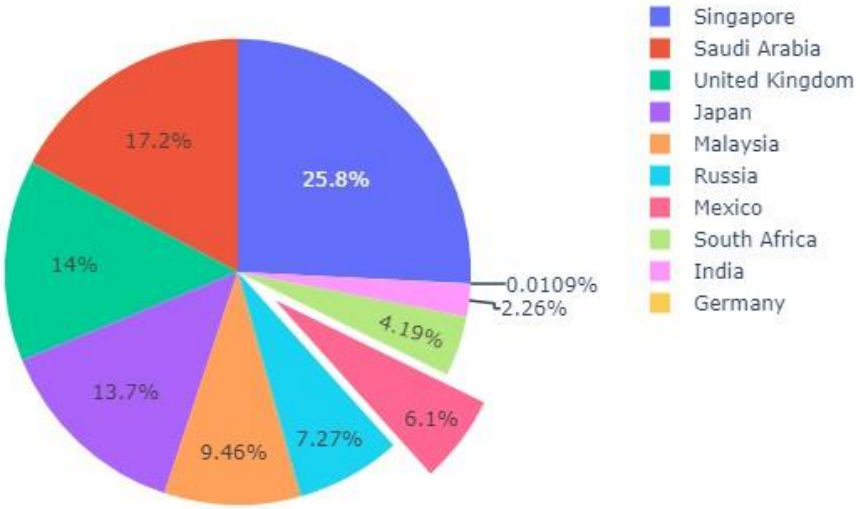

Fig:29

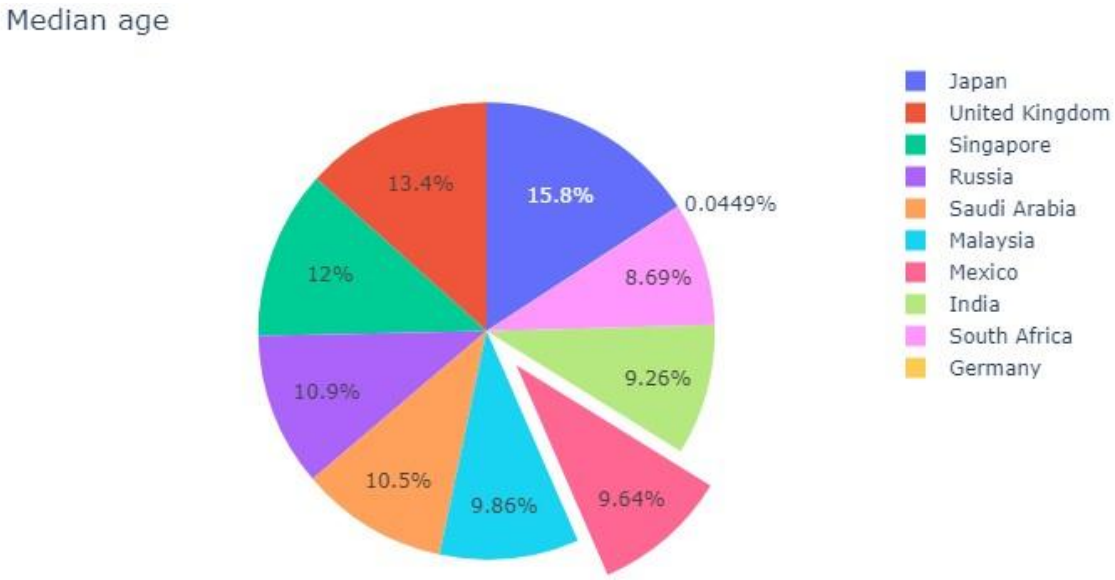
Median age
Japan
United Kingdom
Singapore
Russia
Saudi Arabia
Malaysia
Mexico
India
South Africa
Germany
15.8%
0.0449%
13.4%
12%
8.69%
10.9%
9.26%
10.5%
9.86%
9.64%

Fig:30

REFRENCES

1. https://ourworldindata.org/covid-deaths
2. https://services1.arcgis.com/0MSEUqKaxRlEPj5g/arcgis/rest/services/Coronavirus_2019_nCoV_Cases/FeatureServer/1/query?where=1%3D1&outFields=*&outSR=4326&f=json
3. https://medium.com/swlh/worldwide-covid-19-analysis-visualization-339002a821fe
4. https://covid19.who.int/data
5. Dash, S., Chakraborty, C., Giri, S. K., & Pani, S. K. (2021). Intelligent computing on time-series data analysis and prediction of COVID-19 pandemics. Pattern Recognition Letters, 151, 69-75.
6. Mittal, S. (2020). An exploratory data analysis of COVID-19 in India. International Journal of Engineering and Technical Research, 9(4).
7. Nair, R., Soni, M., Bajpai, B., Dhiman, G., & Sagayam, K. M. (2022). Predicting the death rate around the world due to COVID-19 using regression analysis. International Journal of Swarm Intelligence Research (IJSIR), 13(2), 1-13.

WORKLOG

DAY	DATE	WORK DONE
Day 1 to 7	08/05/2023-15/05/2023 (WEEK 1)	Finding Data sources on covid-19, Data collection and Data Gathering
Day 8 to 14	16/05/2023-23/05/2023 (WEEK 2)	Data Validation, Data Standardization, Data Documentation
Day 15 to 22	24/05/2023-31/05/2023 (WEEK 3)	Learning and revising Linear Regression (Statistical Concepts and Mathematical Models)
Day 23 to 30	01/06/2023-08/06/2023 (WEEK 4)	Applying the learnt concepts and coding in Jupyter notebook
Day 31 to 38	09/06/2023-16/06/2023 (WEEK 5)	Learning key concepts of data Visualization
Day 39 to 46	17/06/2023-24/06/2023 (WEEK 6)	Coding and visualizing the data as Graphs (bar graph, line chart, pie chart)
Day 47 to 54	25/06/2023-02/07/2023 (WEEK 7)	Rectifying the errors and coding
Day 55 to 62	03/07/2023-10/07/2023 (WEEK 8)	Ppt Presentation and Report Completion