t

COVID-19 LOCKDOWN AND VACCINATION:

ANALYSIS OF ITS EFFECT ON INDIAN STATES.

ST. XAVIERS COLLEGE, Mumbai(Autonomous)

SUBMITTED TO Prof. AARON JOHNS
2021-22

DEPARTMENT OF I.T. / STATISTICS

MSC BDA-1

St. Xavier's College (Autonomous), Mumbai

TITLE: COVID-19 LOCKDOWN AND VACCINATION: ANALYSIS OF ITS EFFECT ON INDIAN STATES.

Sr.No	Names	Roll.No	UID
1	Joel Zachariah Cherian	01	219001
2	Jenin Johna	30	219037
3	Reeve Kale	02	219002
4	Sambitha P Santhosh	25	219031
5	Irene Susan Jacob	03	219004

PROF. AARON JOHNS
(Asst.Professor)

PROF.ROY THOMAS
(HEAD OF IT DEPT)

TABLE OF CONTENTS

ACKNOWLEDGEMENT

We would like to convey our gratitude to the Department of Information Technology, St. Xavier's College,Mumbai,who gave us this wonderful opportunity to carry out this worthwhile project based on a relevant topic 'COVID-19 LOCKDOWN AND VACCINATION: ANALYSIS OF ITS EFFECT ON INDIAN STATES.' This project helped us in doing a huge amount of research which gave us an opportunity to learn something new.

We would like to express our heartfelt gratitude to Prof.Aaron Johns for the tremendous help and guidance at each and every step from the selection of the topic till the final analysis.

Last but not the least, we would also like to thank the Statistics and IT Department for the knowledge they have provided to us which played a very important role in the accomplishment of this project.

Thank you!!

INTRODUCTION

Coronavirus disease 2019 (COVID-19) is a contagious disease caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2). This ongoing pandemic has had an adverse impact on India and the world.The pandemic has left a glaring gap in the economy. The very first cases of COVID-19 in India were reported on January 30th, 2020 in three towns of Kerala, among three Indian medical students who had returned fromWuhan, the epicenter of the pandemic.

On the evening of 24 March 2020,the Government of India under the Prime Minister ordered a nationwide lockdown for 21 days, limiting movement of the entire population of the country as a preventive measure against COVID-19 pandemic. The lockdown was conducted in four phases: Phase-I, from 24th March-14th April,Phase 2,3,and 4 from 15th April-3rd May, 4th-17th May and 18th-31st May respectively. The Ministry of Home Affairs issued fresh guidelines from June, stating that the unlock phases would "have an economic focus". Lockdown restrictions were only imposed in containment zones, while activities were permitted in other zones in a phased manner.

India began its vaccination programme on 16 January 2021. As of November 9th, 2021, India has administered over 1.09 billion doses overall, including first and second doses of the currently-approved vaccines-Covishield, Covaxin and Sputnik V.

This project is focused mainly on the analysis of the effects of lockdown and vaccination on Indian states and union territories.

The datasets and materials needed for the study have been taken from the github repository (https://github.com/covid19india/api) which gets updated regularly with data related to Covid-19 and World Air Quality Index portal (WAQI) (www.aqicn.org) and city wise data sets from the Central Pollution Control Board of India under the Ministry of Environment, Forest and Climate Change.

OBJECTIVES

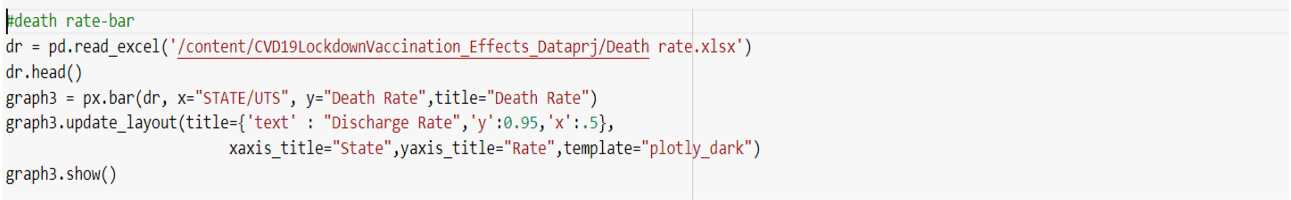
Through this project, we have aimed to determine the following objectives:

1. Data visualization of deaths, death rate, discharge rate and total cases of each Indian States/UTs.
2. Data visualization of Covid-19 vaccinations.
3.To exhibit the effects of lockdown on Air Pollution and its forecasting.
4.To determine whether a higher vaccination rate was needed to decrease the rate of positive cases.
5.To compare the trends of covid death rate before and after vaccination
6.To predict the rise and fall of cases using various methods.

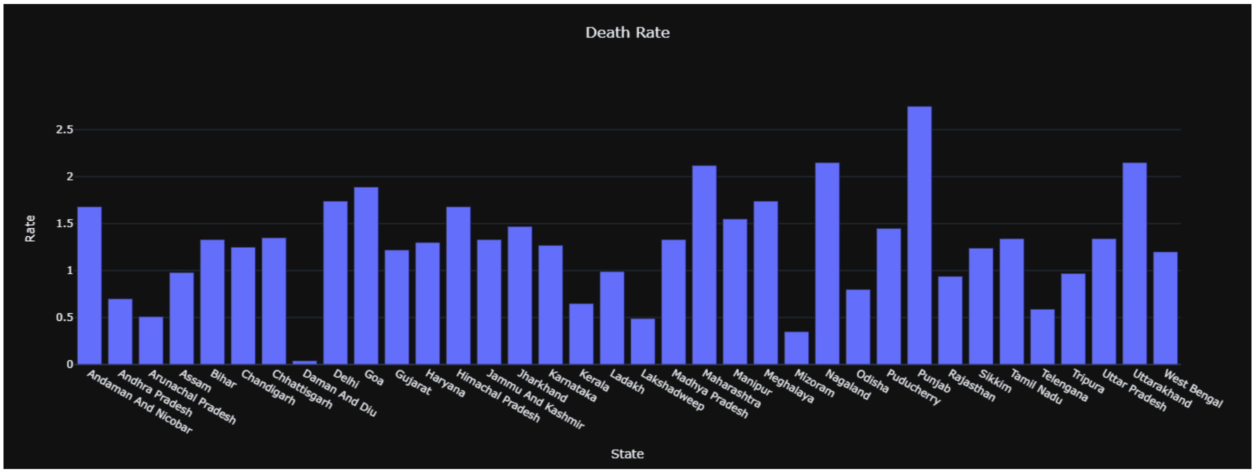
1.DATA VISUALIZATION OF TOTAL CASES,DEATH RATE AND RECOVERY RATE OF EACH INDIAN STATE/UTs

1. Deaths and Death Rate

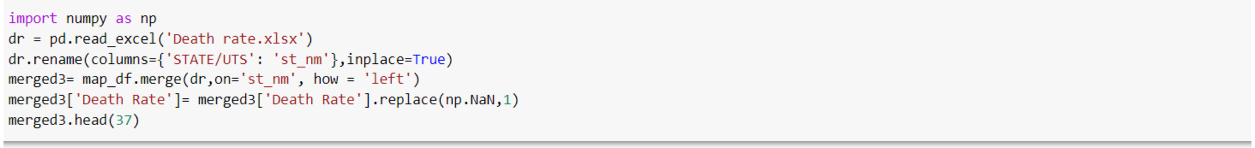
From the following chart, it can be concluded that Punjab has the highest death rate in India. It is followed by Nagaland, Uttarakhand and Maharashtra with a similar death rate.



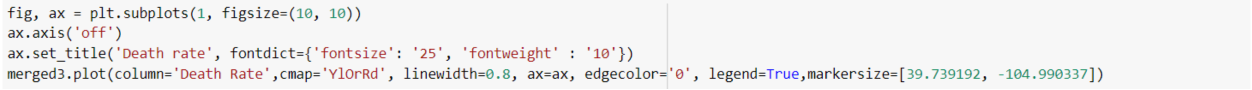
```
#death rate-bar
dr = pd.read_excel('/content/CVD19LockdownVaccination_Effects_Dataprj/Death rate.xlsx')
dr.head()
graph3 = px.bar(dr, x="STATE/UTS", y="Death Rate",title="Death Rate")
graph3.update_layout(title={'text' : "Discharge Rate",'y':0.95,'x':.5},
                     xaxis_title="State",yaxis_title="Rate",template="plotly_dark")
graph3.show()
```



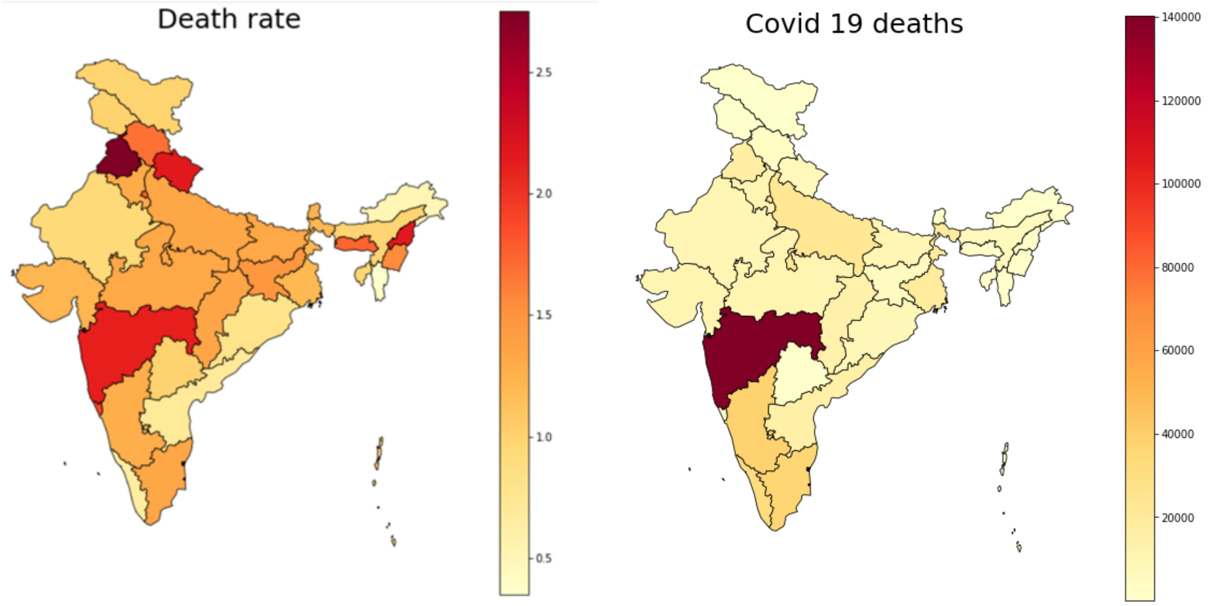
It can be visualised using the heat map as shown below:



```
import numpy as np
dr = pd.read_excel('Death rate.xlsx')
dr.rename(columns={'STATE/UTS': 'st_nm'},inplace=True)
merged3= map_df.merge(dr,on='st_nm', how = 'left')
merged3['Death Rate']= merged3['Death Rate'].replace(np.NaN,1)
merged3.head(37)
```

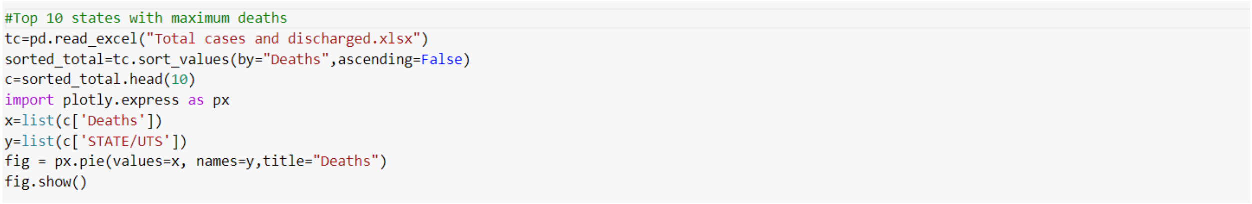


```
fig, ax = plt.subplots(1, figsize=(10, 10))
ax.axis('off')
ax.set_title('Death rate', fontdict={'fontsize': '25', 'fontweight' : '10'})
merged3.plot(column='Death Rate',cmap='YlOrRd', linewidth=0.8, ax=ax, edgecolor='0', legend=True,markersize=[39.739192, -104.990337])
```

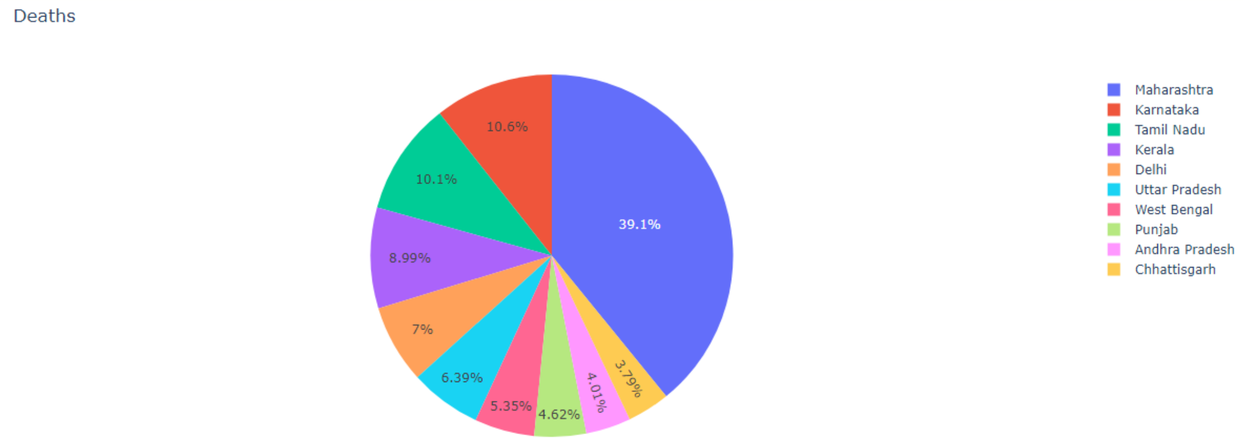


States with the highest deaths in India

The following Pie-chart shows that Maharashtra has the highest percentage (39.1%) of reported deaths, followed by Karnataka(10.6%).

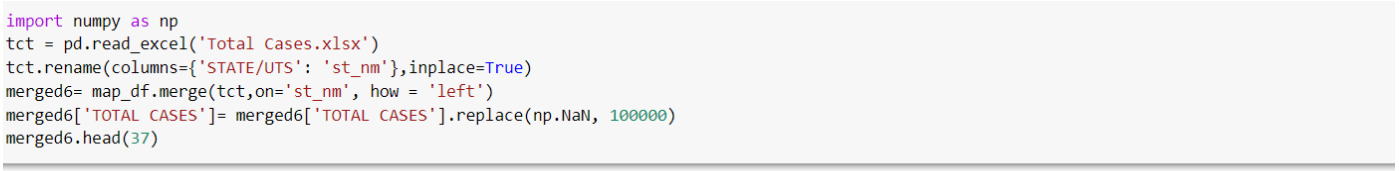


```
#Top 10 states with maximum deaths
tc=pd.read_excel("Total cases and discharged.xlsx")
sorted_total=tc.sort_values(by="Deaths",ascending=False)
c=sorted_total.head(10)
import plotly.express as px
x=list(c['Deaths'])
y=list(c['STATE/UTS'])
fig = px.pie(values=x, names=y,title="Deaths")
fig.show()
```

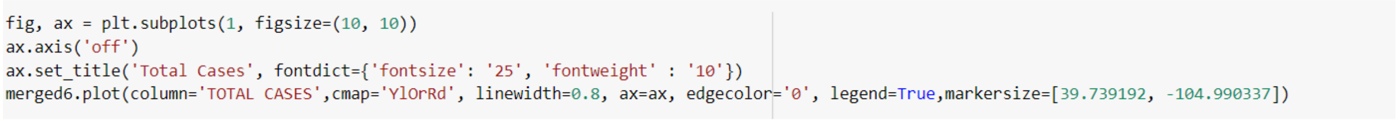


2. Total Cases

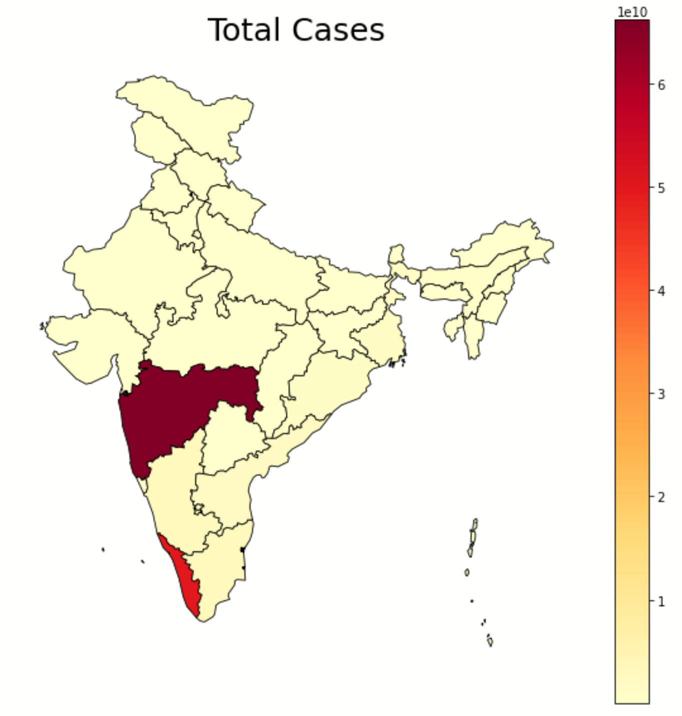
From the following charts it can be concluded that the percentage of total cases were highest in Maharashtra(51.7%) and Kerala(38.9%).

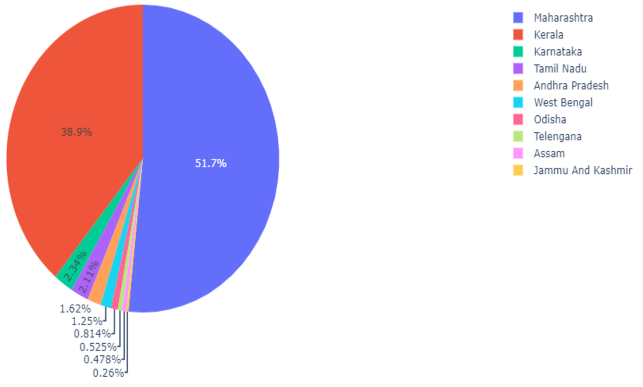


```
import numpy as np
tct = pd.read_excel('Total Cases.xlsx')
tct.rename(columns={'STATE/UTS': 'st_nm'},inplace=True)
merged6= map_df.merge(tct,on='st_nm', how = 'left')
merged6['TOTAL CASES']= merged6['TOTAL CASES'].replace(np.NaN, 100000)
merged6.head(37)
```



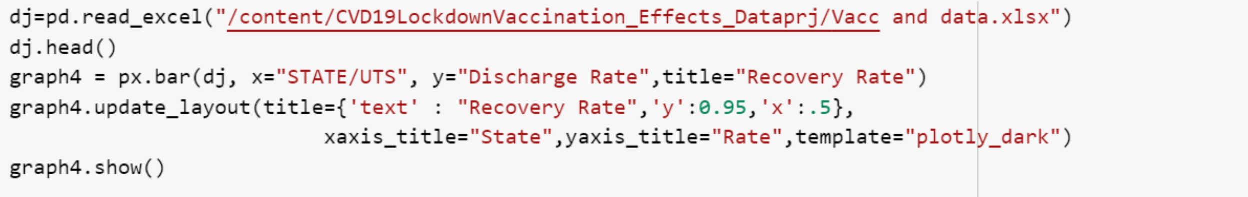
```
fig, ax = plt.subplots(1, figsize=(10, 10))
ax.axis('off')
ax.set_title('Total Cases', fontdict={'fontsize': '25', 'fontweight' : '10'})
merged6.plot(column='TOTAL CASES',cmap='YlOrRd', linewidth=0.8, ax=ax, edgecolor='0', legend=True,markersize=[39.739192, -104.990337])
```



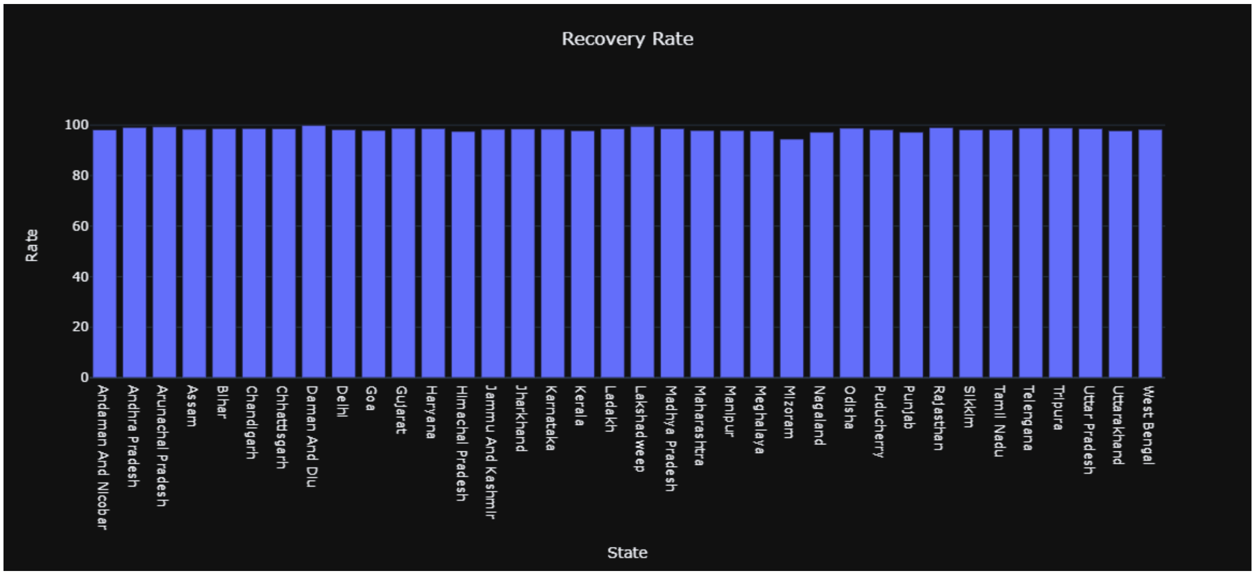


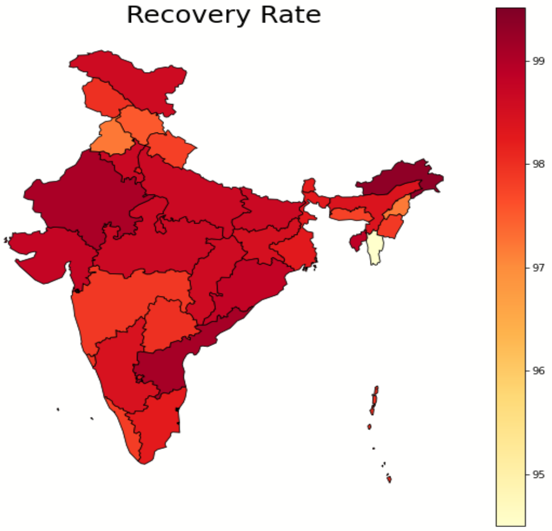
3. Recovery Rate

From the chart below, it can be seen that the recovery rate was almost similar in most of the states.



```
dj=pd.read_excel("/content/CVD19LockdownVaccination_Effects_Dataprj/Vacc and data.xlsx")
dj.head()
graph4 = px.bar(dj, x="STATE/UTS", y="Discharge Rate",title="Recovery Rate")
graph4.update_layout(title={'text' : "Recovery Rate",'y':0.95,'x':.5},
                          xaxis_title="State",yaxis_title="Rate",template="plotly_dark")
graph4.show()
```

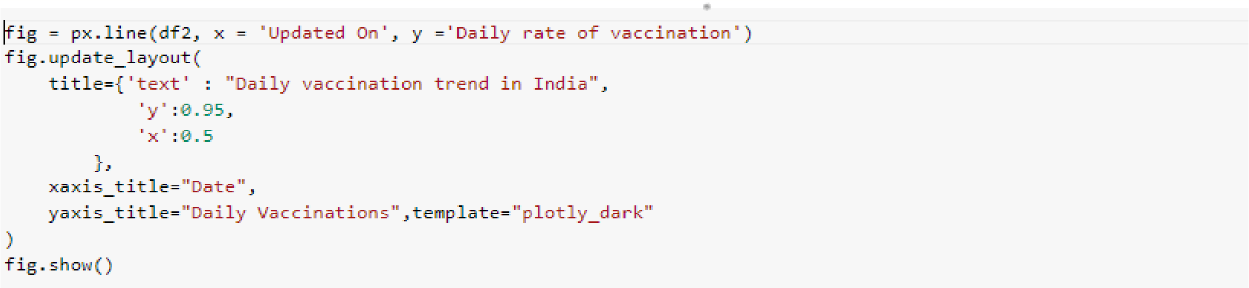




2. VACCINATION DISTRIBUTION IN INDIA

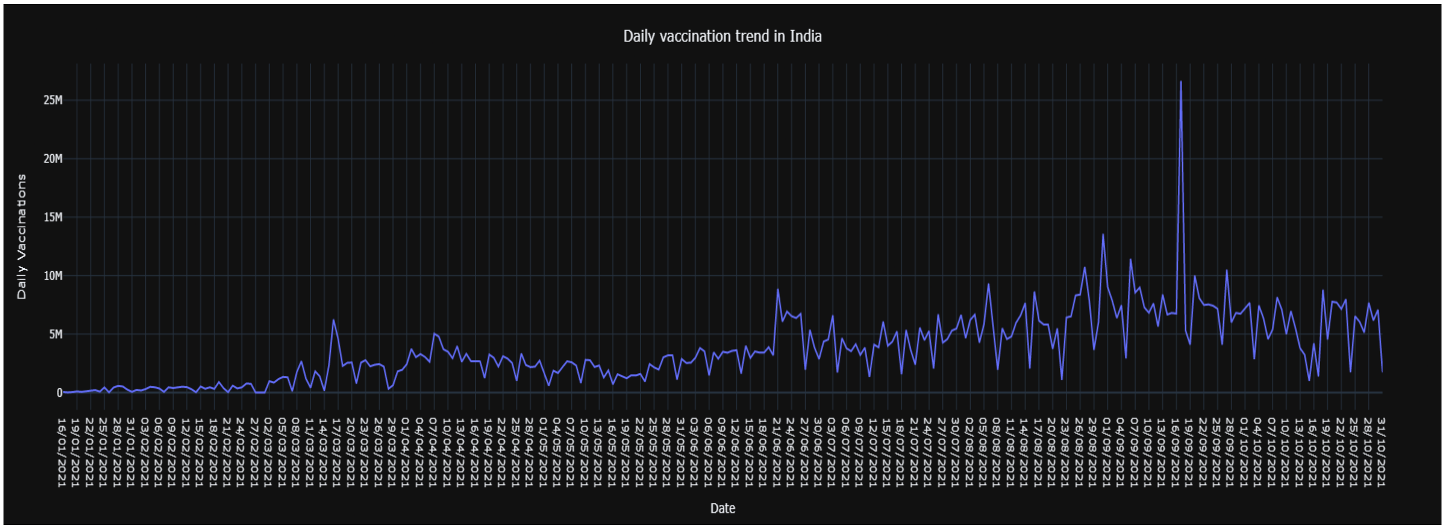
Here we are taking the data needed from the github repository(https://github.com/covid19india/api) which gets weekly updated with data related to Covid-19 and vaccinations. We then removed the daily rate of vaccinations from the 'total vaccines administered' per day data , the same we did it for the 3 different vaccines data and removed it's daily rate of those vaccines taken by people in India.

The Daily Rate at which the vaccine are administered in India

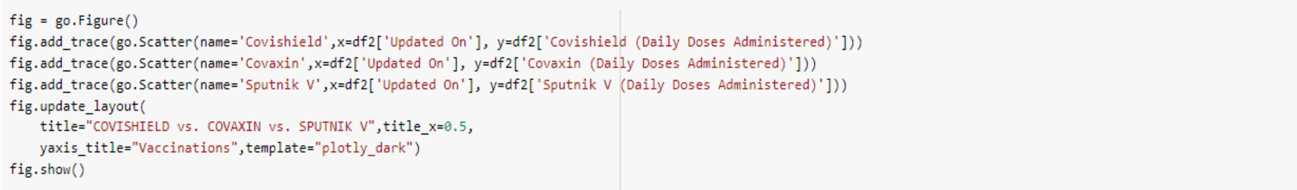


```
fig = px.line(df2, x = 'Updated On', y ='Daily rate of vaccination')
fig.update_layout(
    title={'text' : "Daily vaccination trend in India",
            'y':0.95,
            'x':0.5
        },
    xaxis_title="Date",
    yaxis_title="Daily Vaccinations",template="plotly_dark"
)
fig.show()
```

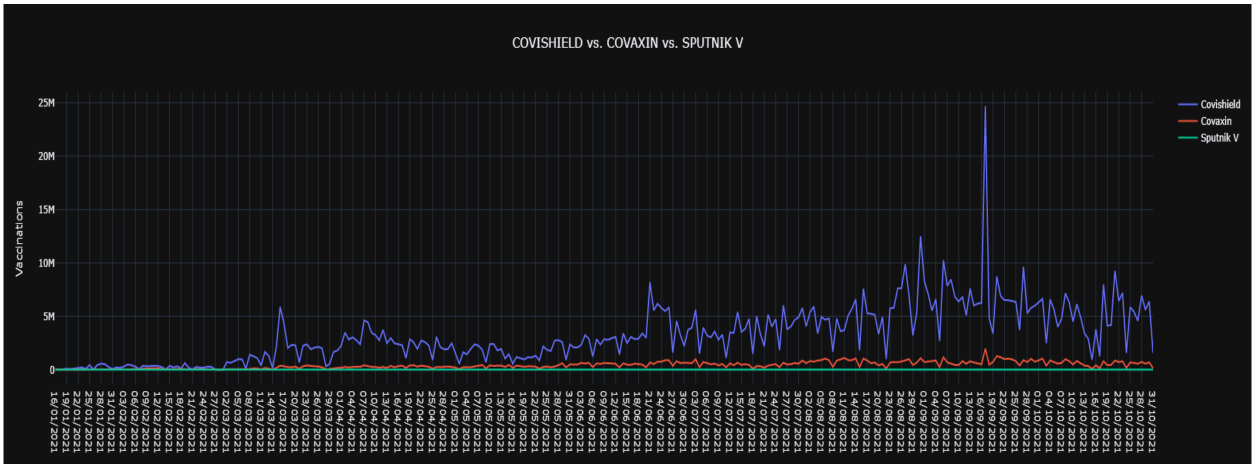
Output



The Daily rate for distribution of different types of vaccine In India

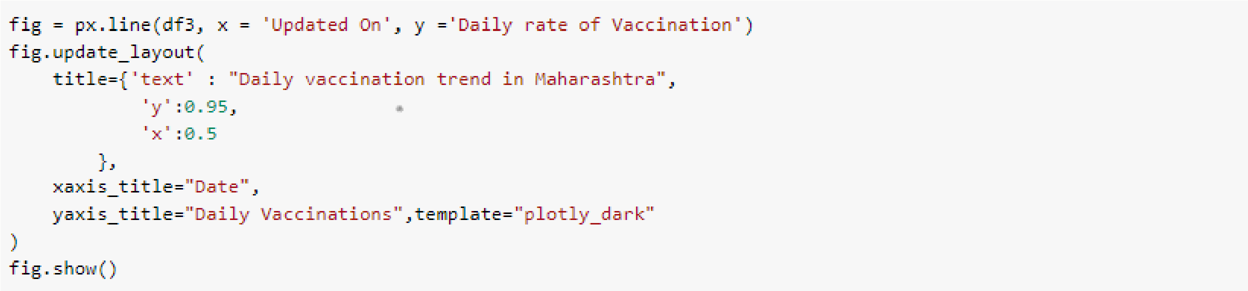


```
fig = go.Figure()
fig.add_trace(go.Scatter(name='Covishield',x=df2['Updated On'], y=df2['Covishield (Daily Doses Administered)']))
fig.add_trace(go.Scatter(name='Covaxin',x=df2['Updated On'], y=df2['Covaxin (Daily Doses Administered)']))
fig.add_trace(go.Scatter(name='Sputnik V',x=df2['Updated On'], y=df2['Sputnik V (Daily Doses Administered)']))
fig.update_layout(
    title="COVISHIELD vs. COVAXIN vs. SPUTNIK V",title_x=0.5,
    yaxis_title="Vaccinations",template="plotly_dark")
fig.show()
```

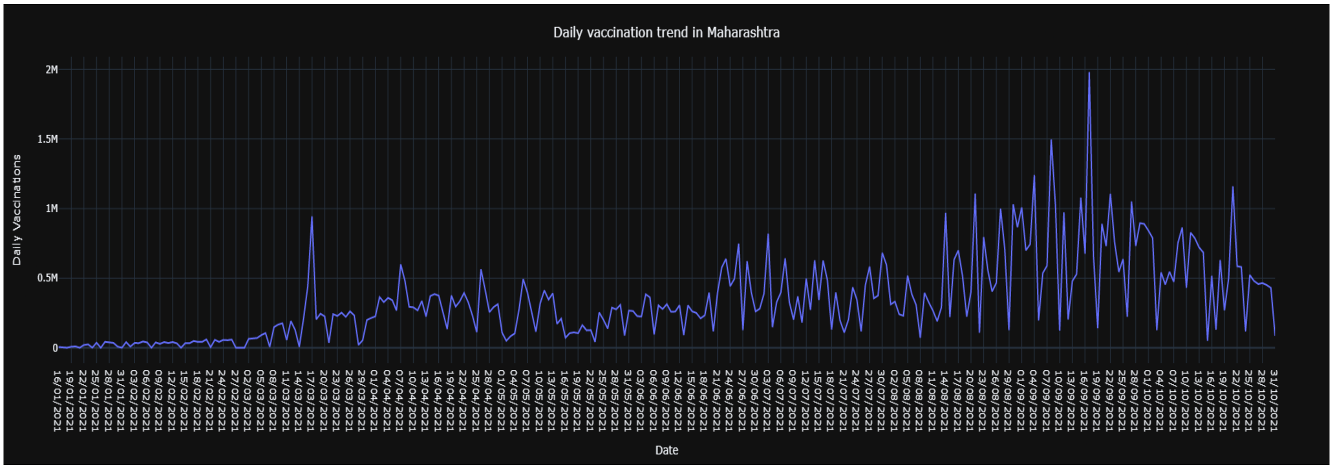


Here we can clearly observe that the daily vaccination rate in India has picked up in the last six months. Even Though vaccine from different pharmaceuticals were introduced in India the major contribution was made by Covishield followed by Covaxin

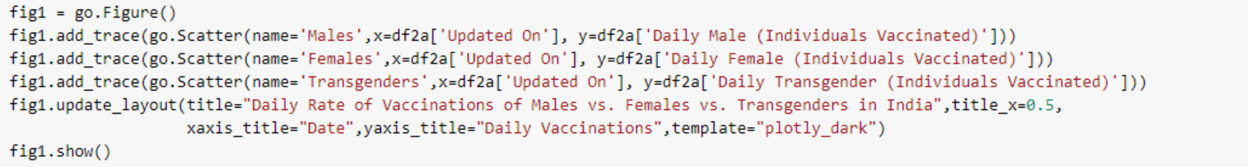
The Daily Rate at which the vaccine is administered in Maharashtra.



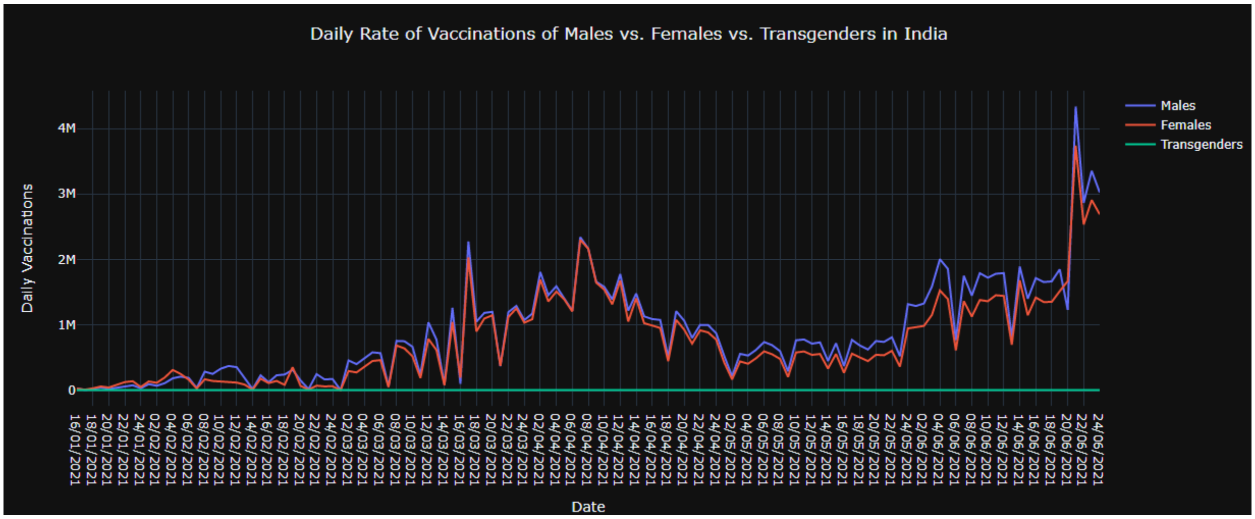
```
fig = px.line(df3, x = 'Updated On', y ='Daily rate of Vaccination')
fig.update_layout(
    title={'text' : "Daily vaccination trend in Maharashtra",
            'y':0.95,
            'x':0.5
        },
    xaxis_title="Date",
    yaxis_title="Daily Vaccinations",template="plotly_dark"
)
fig.show()
```



Gender wise distribution of the daily rate at which vaccine is administered in India



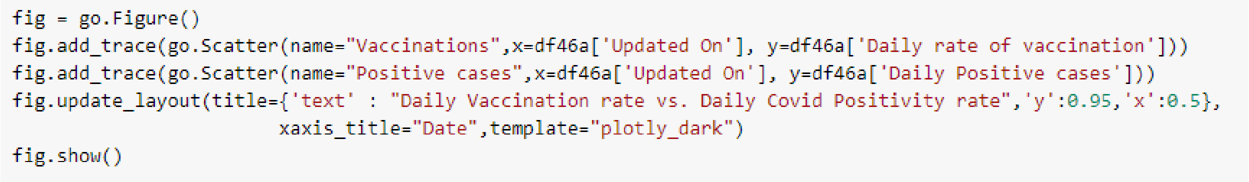
```
fig1 = go.Figure()
fig1.add_trace(go.Scatter(name='Males',x=df2a['Updated On'], y=df2a['Daily Male (Individuals Vaccinated)']))
fig1.add_trace(go.Scatter(name='Females',x=df2a['Updated On'], y=df2a['Daily Female (Individuals Vaccinated)']))
fig1.add_trace(go.Scatter(name='Transgenders',x=df2a['Updated On'], y=df2a['Daily Transgender (Individuals Vaccinated)']))
fig1.update_layout(title="Daily Rate of Vaccinations of Males vs. Females vs. Transgenders in India",title_x=0.5,
                   xaxis_title="Date",yaxis_title="Daily Vaccinations",template="plotly_dark")
fig1.show()
```



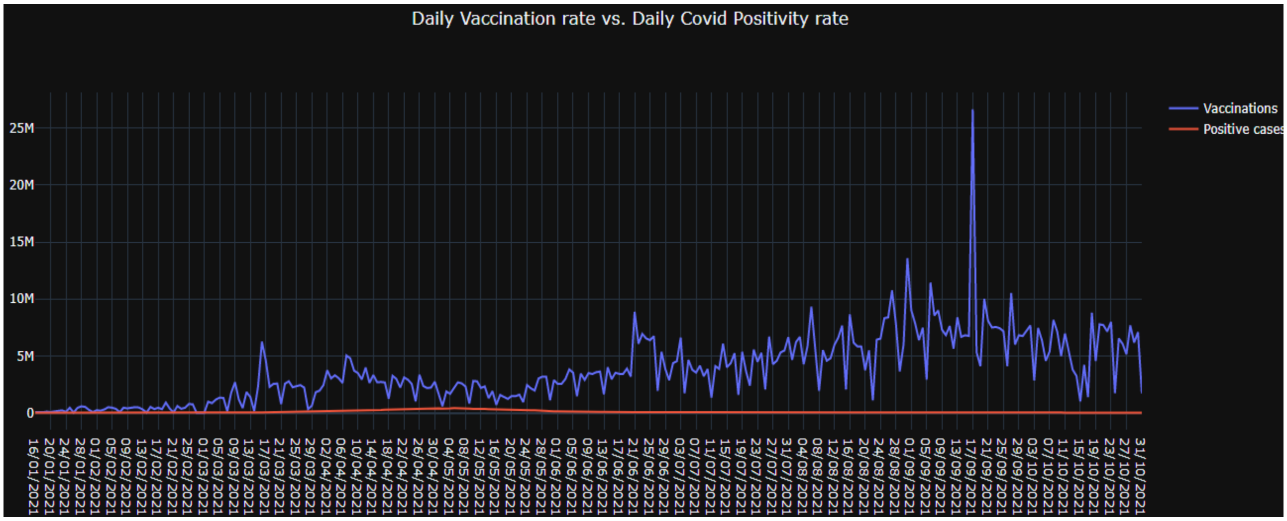
3. STUDY ON DAILY POSITIVE CASES AFTER VACCINATION

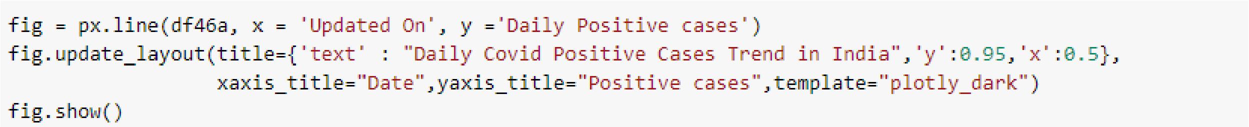
This study is to find how effective the current vaccination rate is for decreasing the daily positive rates. For the study, we have taken data of daily doses of vaccine and daily covid cases.We did visualization on python and R to see the trend of vaccination and the daily cases. We made a linear regression model of vaccination and daily rates to see how the rates behave when there is a change in the number of vaccines received each day.

TO SHOW THE TRENDS OF DAILY COVID POSITIVE CASES WITH THE GRADUAL INCREASE IN VACCINATIONS

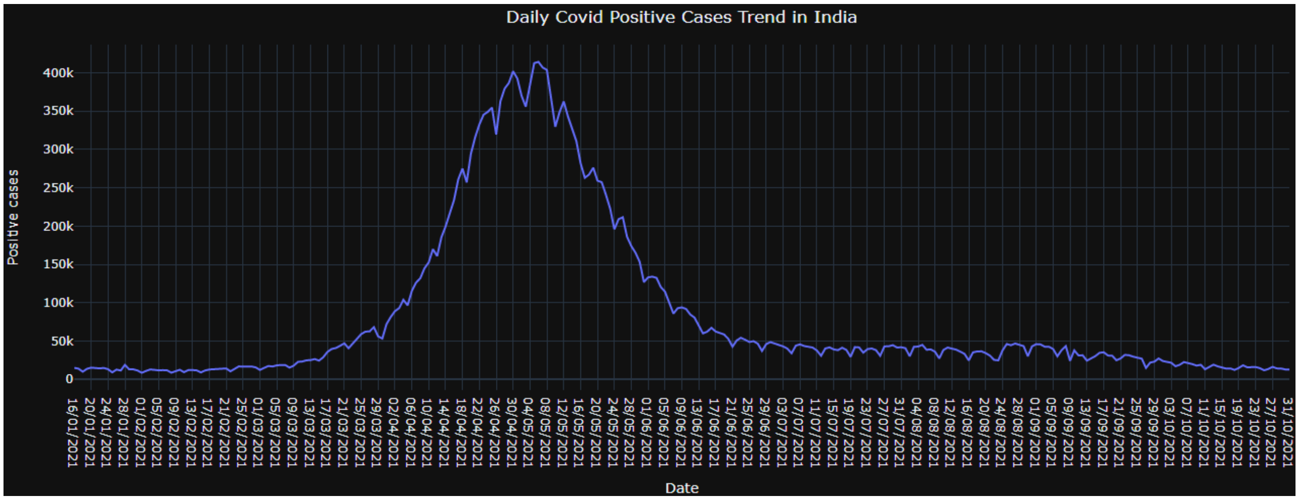


```
fig = go.Figure()
fig.add_trace(go.Scatter(name="Vaccinations",x=df46a['Updated On'], y=df46a['Daily rate of vaccination']))
fig.add_trace(go.Scatter(name="Positive cases",x=df46a['Updated On'], y=df46a['Daily Positive cases']))
fig.update_layout(title={'text' : "Daily Vaccination rate vs. Daily Covid Positivity rate",'y':0.95,'x':0.5},
                        xaxis_title="Date",template="plotly_dark")
fig.show()
```





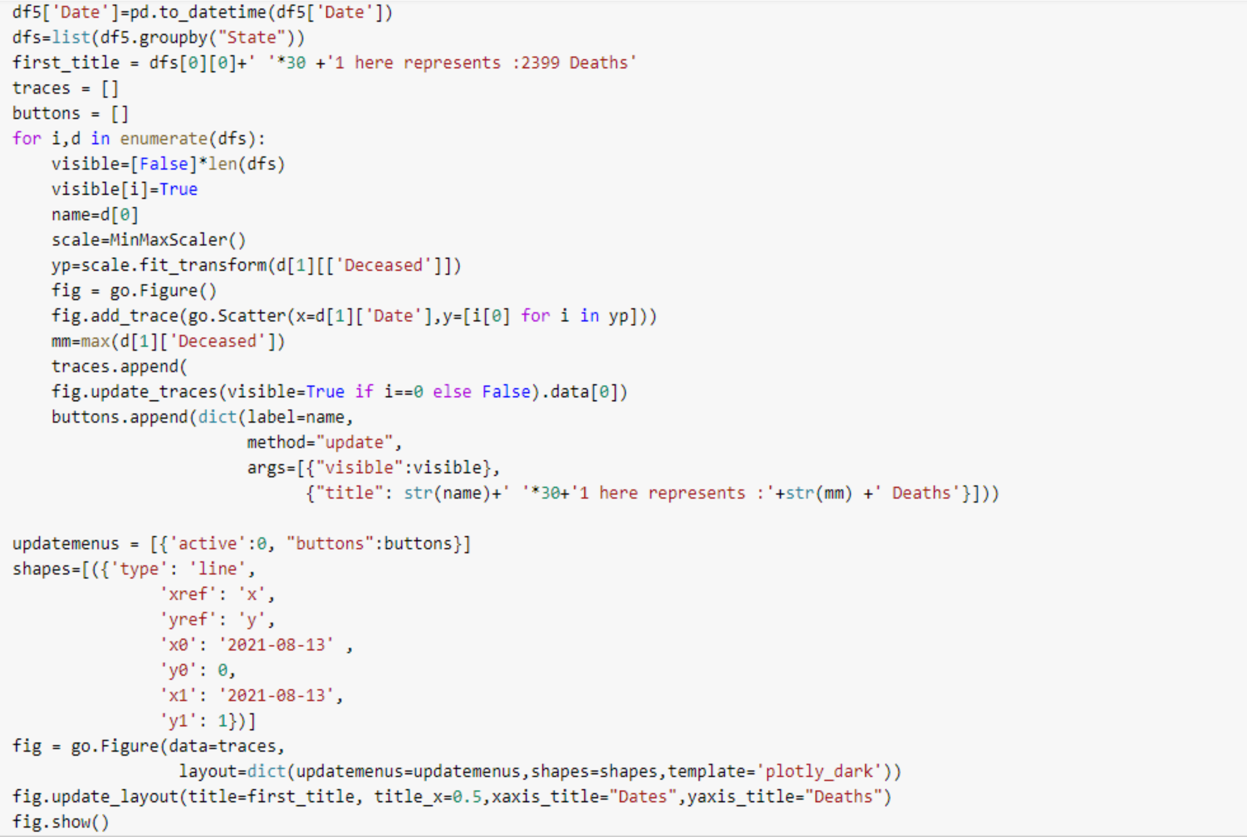
```
fig = px.line(df46a, x = 'Updated On', y ='Daily Positive cases')
fig.update_layout(title={'text' : "Daily Covid Positive Cases Trend in India",'y':0.95,'x':0.5},
                  xaxis_title="Date",yaxis_title="Positive cases",template="plotly_dark")
fig.show()
```


Daily Covid Positive Cases Trend in India
Positive cases
400k
350k
300k
250k
200k
150k
100k
50k
0
16/01/2021
20/01/2021
24/01/2021
28/01/2021
01/02/2021
05/02/2021
09/02/2021
13/02/2021
17/02/2021
21/02/2021
25/02/2021
01/03/2021
05/03/2021
09/03/2021
13/03/2021
17/03/2021
21/03/2021
25/03/2021
29/03/2021
02/04/2021
06/04/2021
10/04/2021
14/04/2021
18/04/2021
22/04/2021
26/04/2021
30/04/2021
04/05/2021
08/05/2021
12/05/2021
16/05/2021
20/05/2021
24/05/2021
28/05/2021
01/06/2021
05/06/2021
09/06/2021
13/06/2021
17/06/2021
21/06/2021
25/06/2021
29/06/2021
03/07/2021
07/07/2021
11/07/2021
15/07/2021
19/07/2021
23/07/2021
27/07/2021
31/07/2021
04/08/2021
08/08/2021
12/08/2021
16/08/2021
20/08/2021
24/08/2021
28/08/2021
01/09/2021
05/09/2021
09/09/2021
13/09/2021
17/09/2021
21/09/2021
25/09/2021
29/09/2021
03/10/2021
07/10/2021
11/10/2021
15/10/2021
19/10/2021
23/10/2021
27/10/2021
31/10/2021
Date

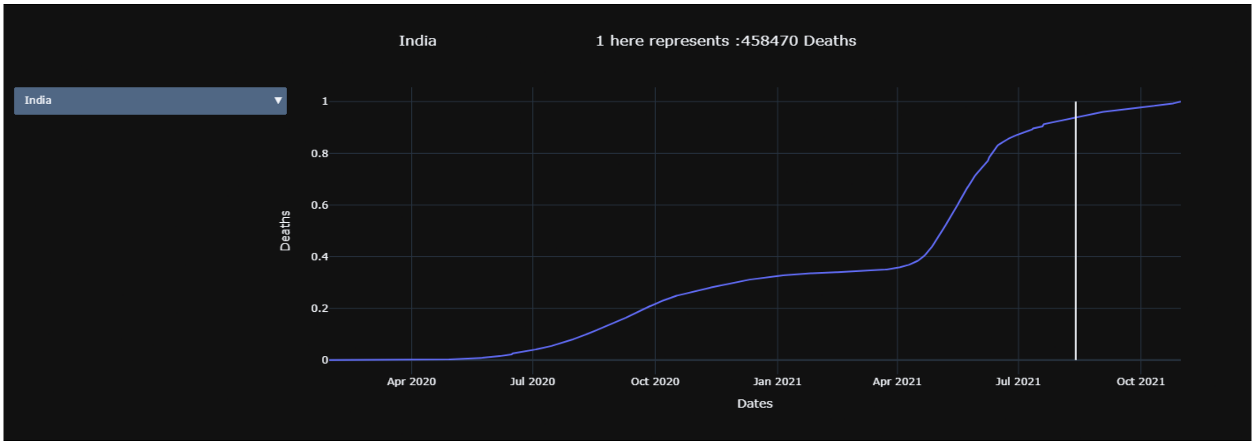
4.TO COMPARE THE TRENDS OF COVID DEATH RATES DURING AND AFTER VACCINATIONS

Plotly python library was used to plot the graph. Minmaxscaler scaler was used to scale all the data features in the range[0,1].

Exponential growth is observed after June 2021. It represents the second wave of COVID-19 that hit India, in which the death rates were high. As of this day, the total number of vaccinations have crossed the 1 billion mark. The white line in the graph represents the cases on the day the total number of vaccinations crossed 50%, i.e, 13th August,2021.



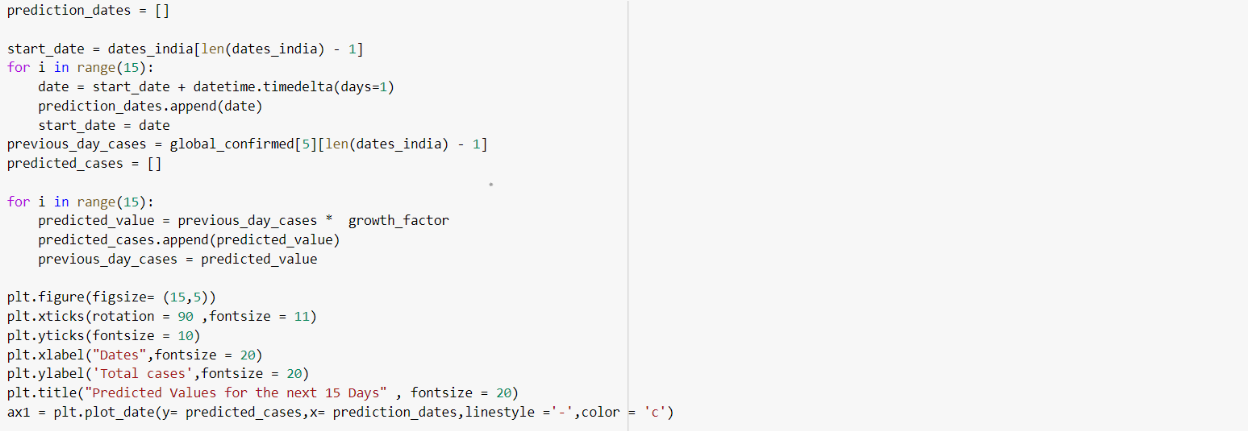
```
df5['Date']=pd.to_datetime(df5['Date'])
dfs=list(df5.groupby("State"))
first_title = dfs[0][0]+' '*30 +'1 here represents :2399 Deaths'
traces = []
buttons = []
for i,d in enumerate(dfs):
    visible=[False]*len(dfs)
    visible[i]=True
    name=d[0]
    scale=MinMaxScaler()
    yp=scale.fit_transform(d[1][['Deceased']])
    fig = go.Figure()
    fig.add_trace(go.Scatter(x=d[1]['Date'],y=[i[0] for i in yp]))
    mm=max(d[1]['Deceased'])
    traces.append(
    fig.update_traces(visible=True if i==0 else False).data[0])
    buttons.append(dict(label=name,
                        method="update",
                        args=[{"visible":visible},
                              {"title": str(name)+' '*30+'1 here represents :'+str(mm) +' Deaths'}]))

updatemenus = [{'active':0, "buttons":buttons}]
shapes=[({'type': 'line',
               'xref': 'x',
               'yref': 'y',
               'x0': '2021-08-13' ,
               'y0': 0,
               'x1': '2021-08-13',
               'y1': 1})]
fig = go.Figure(data=traces,
                 layout=dict(updatemenus=updatemenus,shapes=shapes,template='plotly_dark'))
fig.update_layout(title=first_title, title_x=0.5,xaxis_title="Dates",yaxis_title="Deaths")
fig.show()
```



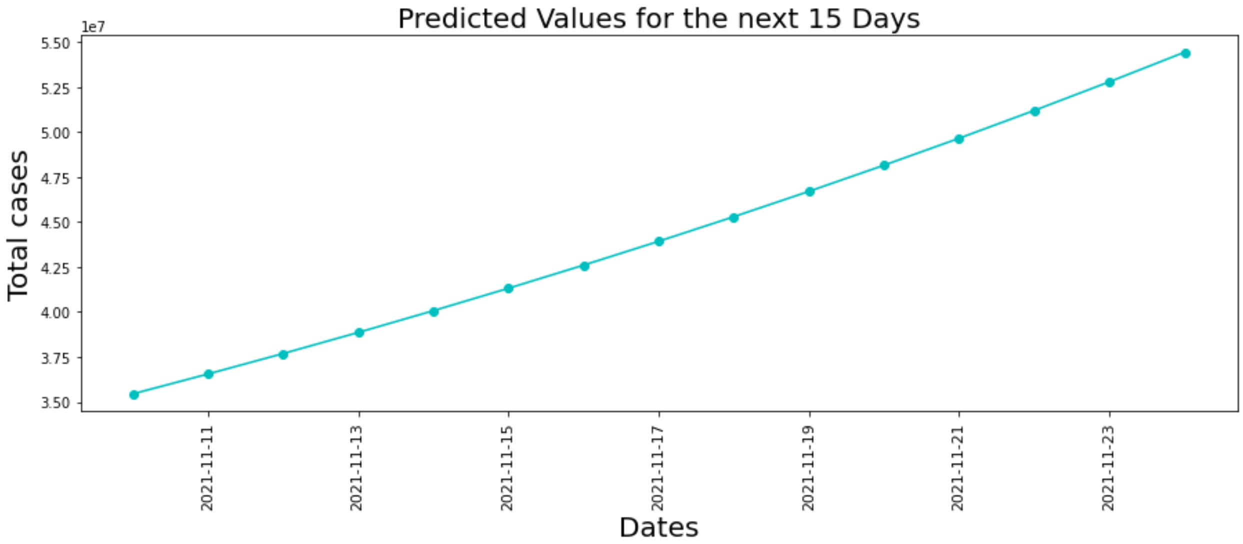
6.TO PREDICT THE RISE AND FALL OF CASES USING VARIOUS METHODS

PREDICTING THE CASES FOR THE NEXT 15 DAYS USING GROWTH FACTOR

We have taken the confirmed cases,df1, from 2nd April 2020. In order to compute the average growth rate, we divided each days confirmed cases with the previous days number of cases. Then we took the average. To predict the next 15 days cases, it was multiplied ith the previous days number of cases. A linear graph is obtained. The best case scenario for no new COVID cases would be when the growth factor is 1. This is not an accurate method to evaluate the real case scenario of the COVID-19 cases trend prediction.



```
prediction_dates = []

start_date = dates_india[len(dates_india) - 1]
for i in range(15):
    date = start_date + datetime.timedelta(days=1)
    prediction_dates.append(date)
    start_date = date
previous_day_cases = global_confirmed[5][len(dates_india) - 1]
predicted_cases = []

for i in range(15):
    predicted_value = previous_day_cases *  growth_factor
    predicted_cases.append(predicted_value)
    previous_day_cases = predicted_value

plt.figure(figsize= (15,5))
plt.xticks(rotation = 90 ,fontsize = 11)
plt.yticks(fontsize = 10)
plt.xlabel("Dates",fontsize = 20)
plt.ylabel('Total cases',fontsize = 20)
plt.title("Predicted Values for the next 15 Days" , fontsize = 20)
ax1 = plt.plot_date(y= predicted_cases,x= prediction_dates,linestyle ='-',color = 'c')
```



USING PROPHET METHOD

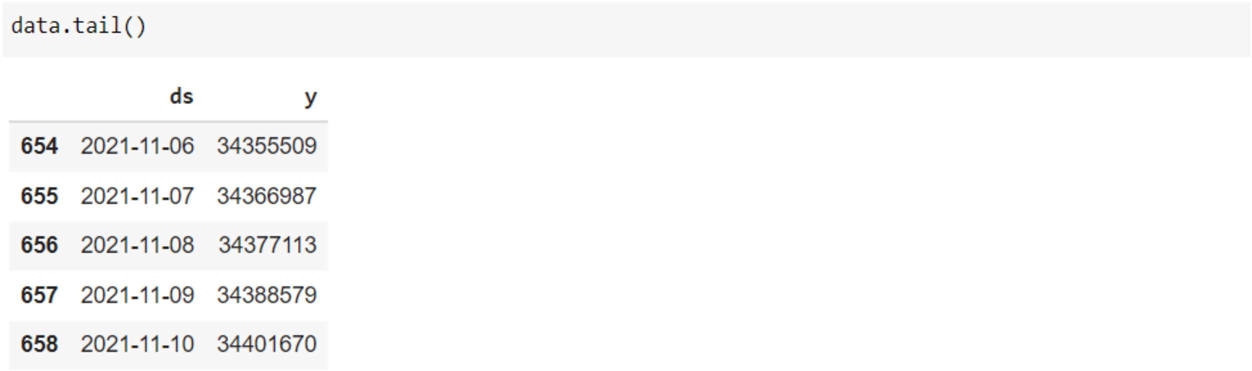
Prophet is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. They are combined in the following equation:

$y(t)= g(t) + s(t) + h(t) + εt$

where,

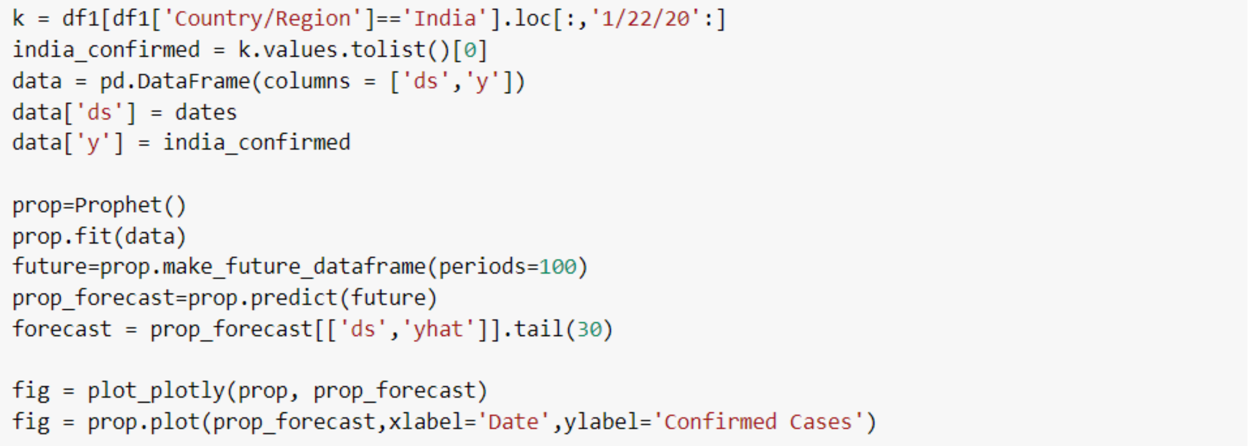
g(t): piecewise linear or logistic growth curve for modeling non-periodic changes in time series
s(t): periodic changes (e.g. weekly/yearly seasonality)
h(t): effects of holidays (user provided) with irregular schedules
εt: error term accounts for any unusual changes not accommodated by the model

All the necessary libraries for prediction were first imported, which includes, Pandas, Matplotlib and fbprophet, an open source library from facebook.The data for covid cases wass taken from the github website as it was updated daily. The dates which are in columns and converted to a list.
For the prophet model to work,the data must be in a specific format. The train_test_split was imported from the sklearn library and it was divided into two parts- train data and test data.
The make future dataframe method was then used to create dates needed for the prediction model.The period is specified to be approximately two months.Nextly, the predict method was used to forecast.
The concat method from pandas was used to plot the predicted values and the actual values from the raw data. The yellow and blue lines represent the predicted and the actual values respectively. The forecasted data takes a linear slope, the actual data is plateauing.This is due to scaling. It sums that this graph is overfitted even though it's not. The blue line in the plot represents the line of best fit and the shaded region is given by yhat_upper and yhat_lower which shows the margin of error. The black points are the actual value from the featured data. The pearsonr test is performed to find a correlation between the actual and the forecasted data. It comes out to be 0.96, a strong correlation.

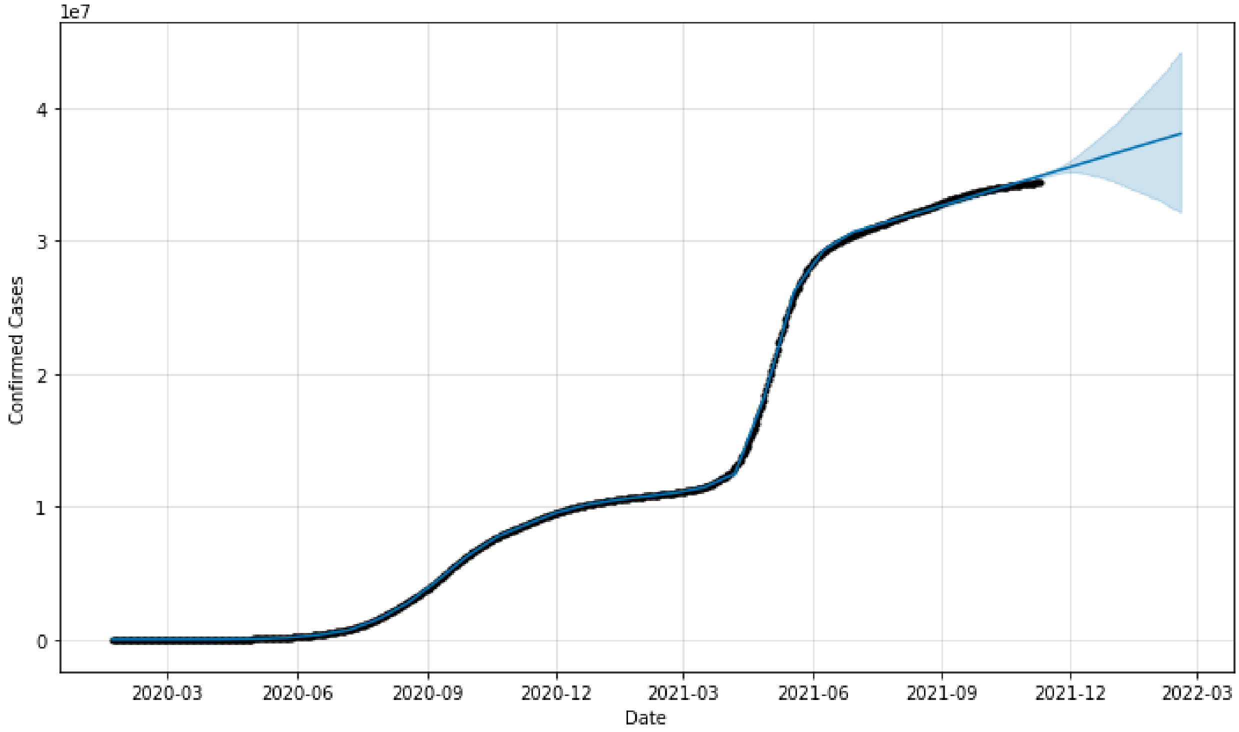


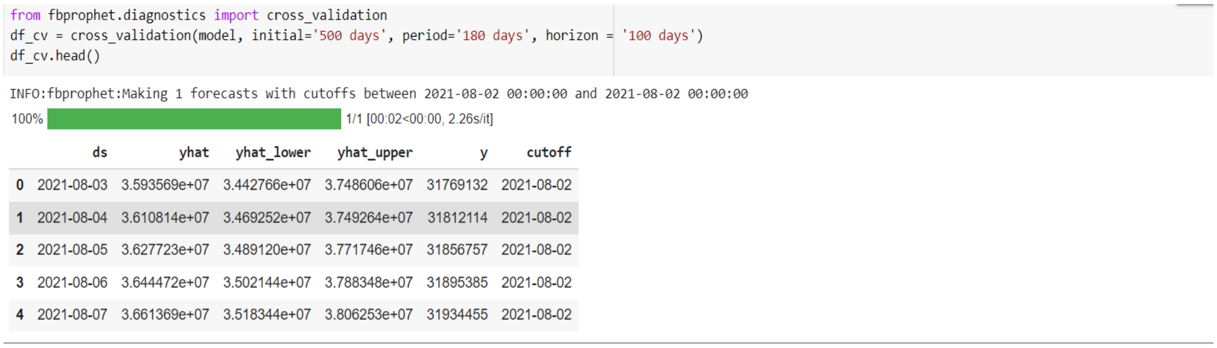
```
data.tail()
```

	ds	y
654	2021-11-06	34355509
655	2021-11-07	34366987
656	2021-11-08	34377113
657	2021-11-09	34388579
658	2021-11-10	34401670



```
k = df1[df1['Country/Region']=='India'].loc[:,'1/22/20':]
india_confirmed = k.values.tolist()[0]
data = pd.DataFrame(columns = ['ds','y'])
data['ds'] = dates
data['y'] = india_confirmed

prop=Prophet()
prop.fit(data)
future=prop.make_future_dataframe(periods=100)
prop_forecast=prop.predict(future)
forecast = prop_forecast[['ds','yhat']].tail(30)

fig = plot_plotly(prop, prop_forecast)
fig = prop.plot(prop_forecast,xlabel='Date',ylabel='Confirmed Cases')
```

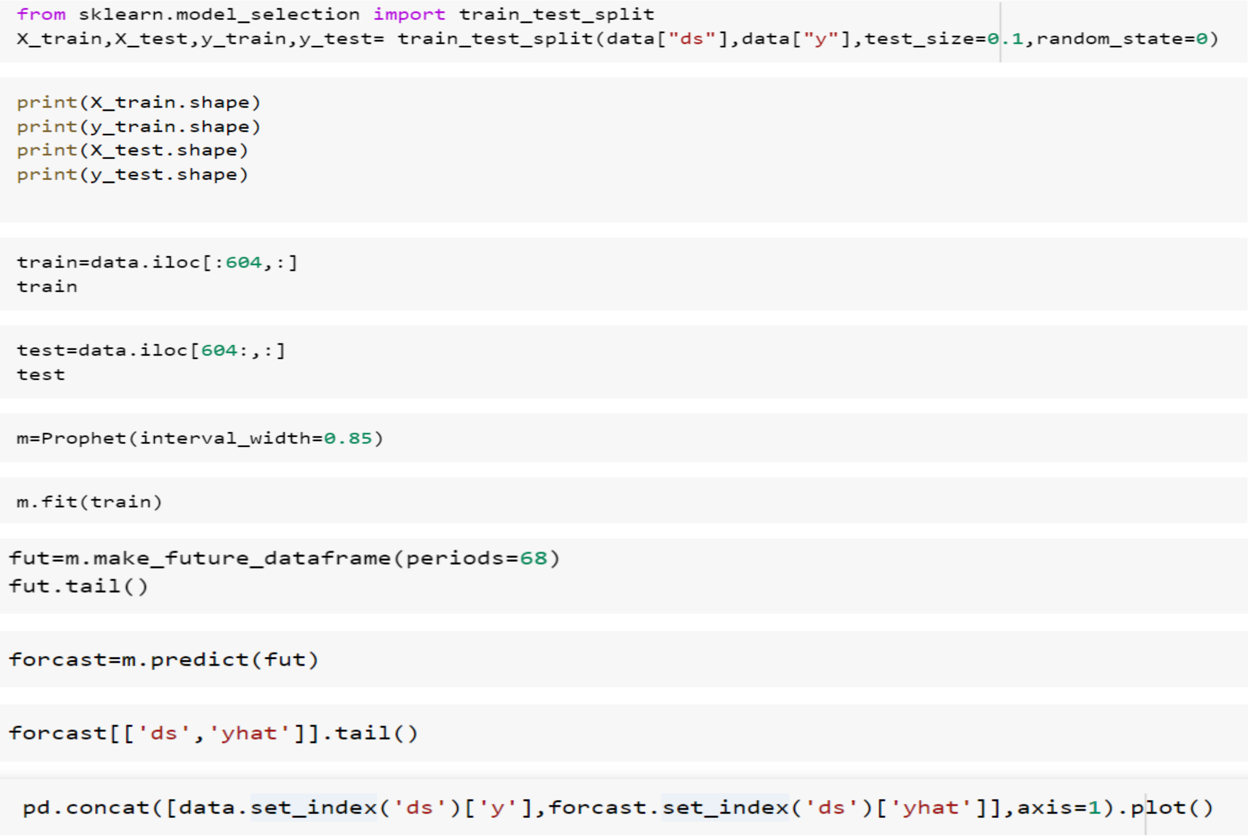




```
from fbprophet.diagnostics import cross_validation
df_cv = cross_validation(model, initial='500 days', period='180 days', horizon = '100 days')
df_cv.head()
```

```
INFO:fbprophet:Making 1 forecasts with cutoffs between 2021-08-02 00:00:00 and 2021-08-02 00:00:00
100% 1/1 [00:02<00:00, 2.26s/it]
```

	ds	yhat	yhat_lower	yhat_upper	y	cutoff
0	2021-08-03	3.593569e+07	3.442766e+07	3.748606e+07	31769132	2021-08-02
1	2021-08-04	3.610814e+07	3.469252e+07	3.749264e+07	31812114	2021-08-02
2	2021-08-05	3.627723e+07	3.489120e+07	3.771746e+07	31856757	2021-08-02
3	2021-08-06	3.644472e+07	3.502144e+07	3.788348e+07	31895385	2021-08-02
4	2021-08-07	3.661369e+07	3.518344e+07	3.806253e+07	31934455	2021-08-02



```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test= train_test_split(data["ds"],data["y"],test_size=0.1,random_state=0)
```

```
print(X_train.shape)
print(y_train.shape)
print(X_test.shape)
print(y_test.shape)
```

```
train=data.iloc[:604,:]
train
```

```
test=data.iloc[604:,:]
test
```

```
m=Prophet(interval_width=0.85)
```

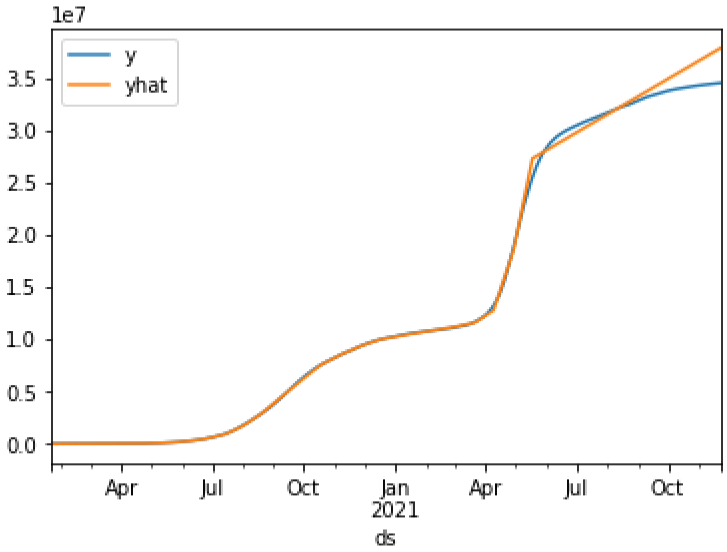
```
m.fit(train)
```

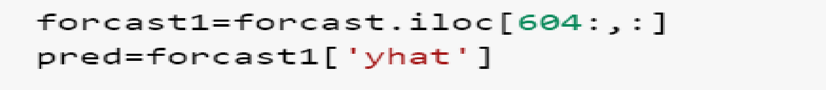
```
fut=m.make_future_dataframe(periods=68)
fut.tail()
```

```
forcast=m.predict(fut)
```

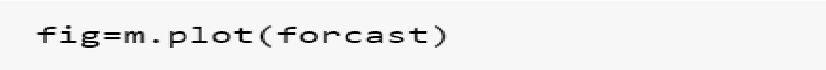
```
forcast[['ds','yhat']].tail()
```

```
pd.concat([data.set_index('ds')['y'],forcast.set_index('ds')['yhat']],axis=1).plot()
```

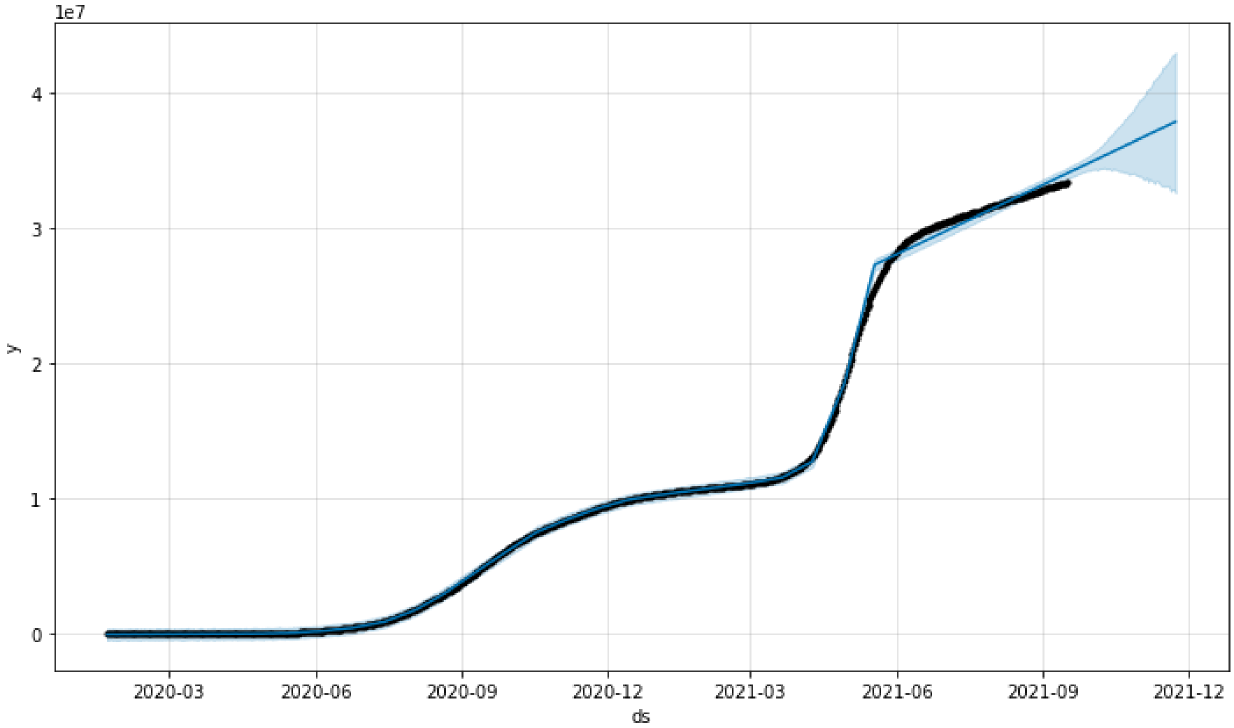


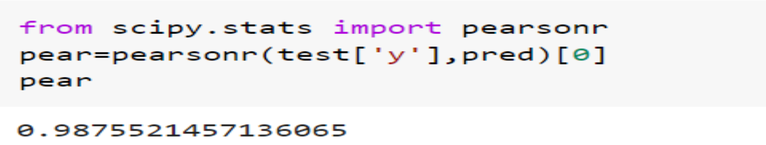


```
forcast1=forcast.iloc[604:,:]
pred=forcast1['yhat']
```



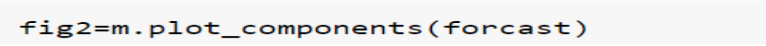
```
fig=m.plot(forcast)
```



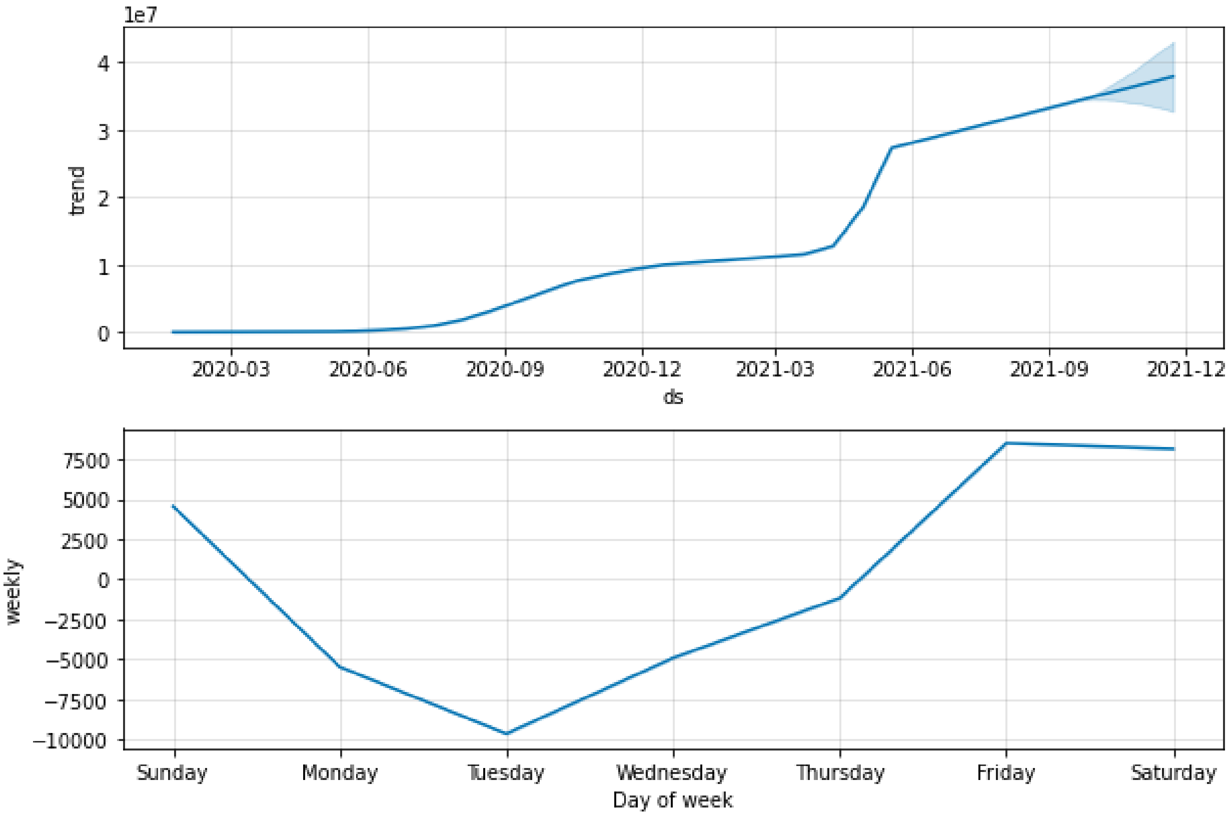


```
from scipy.stats import pearsonr
pear=pearsonr(test['y'],pred)[0]
pear
```

```
0.9875521457136065
```



```
fig2=m.plot_components(forcast)
```



REFERENCES

1.https://github.com/covid19india/api

2.https://medium.com/analytics-vidhya/generate-a-static-choropleth-india-map-using-corona-virus-pandemic-data-19e9cbf5a07d

3.Class Notes