SUMMER INTERNSHIP REPORT

Department of Computer Science and Engineering

July - 2021

Covid-19 End Prediction using Machine Learning

SUBMITTED BY

K. Nikhil Sai	Nandini. Ch	Sai Praveen
AP19110010218	AP19110010333	AP19110010215
Dept. of CSE	Dept. of CSE	Dept. of CSE

Under the Guidance of

Dr. Ravi Kant Kumar
(Assistant Professor)
Dept. of Computer Science and Engineering
(SRM University-AP, Andhra Pradesh)

DATA SHEET

<table>
<tr><td>Roll Numbers</td><td>:</td><td colspan="2">1. AP19110010218
2. AP19110010333
3. AP19110010215</td></tr>
<tr><td>Names of the student</td><td>:</td><td colspan="2">1. K. Nikhil sai
2. Ch. Nandini
3. Sai Praveen</td></tr>
<tr><td>Branch & Section</td><td>:</td><td colspan="2">CSE- E & CSE- F</td></tr>
<tr><td>Batch</td><td>:</td><td colspan="2">2019-2023</td></tr>
<tr><td>Type of internship</td><td>:</td><td colspan="2">Project with Faculty</td></tr>
<tr><td>Company Name/Institute Name</td><td>:</td><td colspan="2">SRM University , AP</td></tr>
<tr><td>Company/Institute Website</td><td>:</td><td colspan="2">www.srmap.edu</td></tr>
<tr><td>Start Date (MM/DD/YYYY)</td><td>:</td><td colspan="2">04-06-2021</td></tr>
<tr><td>End Date (MM/DD/YYYY)</td><td>:</td><td colspan="2">14-07-2021</td></tr>
<tr><td>Duration (No. of days)</td><td>:</td><td colspan="2">41 days</td></tr>
<tr><td>Status of the internship</td><td>:</td><td colspan="2">Completed</td></tr>
<tr><td>Name of internship mentor
(SRM Faculty)</td><td>:</td><td colspan="2">Dr. Ravi Kant Kumar</td></tr>
<tr><td rowspan="3">Names & Signatures of the student</td><td rowspan="3">:</td><td>K. Nikhil sai</td><td>K. Nikhil sai</td></tr>
<tr><td>Ch. Nandini</td><td>Ch. Nandini</td></tr>
<tr><td>Sai Praveen</td><td>Sai Praveen</td></tr>
</table>

TABLE OF CONTENTS

ACKNOWLEDGEMENT:

First and foremost, we would like to thank our mentor for this project, Mr. Ravikant Kumar for the valuable guidance and advice. He inspired us greatly to work on this project. His willingness to motivate us contributed tremendously to our project. We also would like to thank him for his continuous support for the successful compilation of the project. Besides, we would like to thank SRM University, AP for providing us with a good environment and facilities to complete this project. We thank them for offering this project. It allowed us to participate and learn about ML techniques. Finally, an honorable mention goes to our families and friends for their understanding and support of us in completing this project. Without the help of the particular that mentioned above, we would face many difficulties while doing this.

ABSTRACT:

The transmission of COVID-19 disease has shown a great impact on society. The whole world has been fighting this epidemic since late February 2020. The main objective of this project is to predict the spread and end of the COVID-19 disease. Because of the COVID-19 outbreak, the world's economy has been affected so far and an accurate prognosis of its epidemic is significant. Prediction for the end of this disease is not an easy task as it requires plenty of data and also various parameters involved in the prediction. This project proposes certain machine learning techniques and ARIMA models with numerical approximations from the dataset provided we forecast the number of reported cases and disease transmission.

INTRODUCTION:

According to WHO, the first case of coronavirus was reported on December 31, 2019, in Wuhan city of China. Another 44 cases of a similar type were reported in China within 4 days. This is identified to be a communicable disease that vigorously spread throughout the world leading to the death of many people. People with underlying health problems are more likely to develop serious illnesses as this disease is more likely to affect the respiratory system. People throughout the world are researching in many different ways to end this disease and are also predicting models using machine learning techniques.
The model is built to forecast the number of confirmed cases, recovered cases, and death cases based on the data available. For the prediction model, the time series forecasting method is applied.

Skills acquired through the internship:

1. Data Analysis with Pandas
2. Visualization (Matplotlib, plotly, seaborn)
3. Time Series Forecasting model

PREDICTION:

STEPS FOLLOWED TO PREDICT THE END OF COVID-19:

1)DATA COLLECTION:

- Collecting data and datasets from different websites.

2)DATA CLEANING:

- Cleaning the dataset by using pandas.

3)DATA ANALYSIS & VISUALIZATION:

- Analyzing the covid-19 data of Maharashtra.
- Visualizing the total cases, deaths, and new cases using Matplotlib and Plotly.

4)MODEL BUILDING (Another File):

- ARIMA model is used for forecasting the new cases.

5)PREDICTING FOR THE NEXT 6 MONTHS:

- Using the ARIMA model we predicted the new cases for the next 6-8 months.

WHY PREDICTION?

COVID-19 has affected every country around the world. The number of infected and dead people due to this disease has been raised.

Prediction models contribute knowledge of the disease and its prevalence. These strategies examine past occurrences and scenarios to produce the best predictions for the future. These forecasts may aid people in preparing for potential outcomes. They play a major role in getting accurate predictions. These models can either be predicted by mathematical models or by machine learning techniques. Data should be collected from different sources in large quantities to prepare an accurate model. Various parameters such as environmental factors, quarantine period, disease spreading rate, immunity levels of the person, person’s past health issues, etc., are taken before forecasting the pandemic.

METHODOLOGY:

AR MODEL(AutoRegressive Model):

It is a statistical model which predicts future value based on its past values is called lags.

- The model that depends only on one lag in the past is given below:

$$Y_t = \omega + \phi Y_{t-1} + e_t$$

- This model is called the long-memory model.
  If the recursion in the time goes back until the beginning of the series, those are called long memory models.

MA MODEL(Moving Average Model):

It is a model which predicts the future based on past errors called errors.

- It depends only on the lag of error in the past.
- The model that depends only on one lag in the past:

$$Y_t = \omega + \theta e_{t-1} + e_t$$

- These are called short memory models.
- There is no effect on the present predicted value if there are any big errors long enough ago.

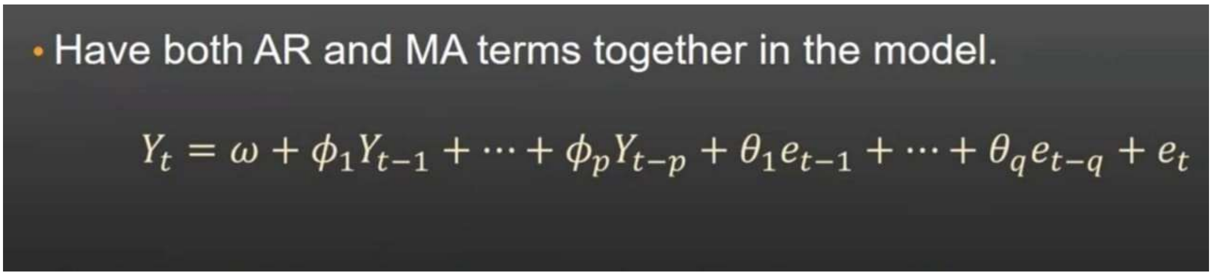
ARIMA MODEL:
Auto-Regressive Integrated Moving Average Model

ARIMA model is a forecasting algorithm, the information contained in the past values of the time series can be utilized to forecast future values on its own. It is a type of model that describes a time series based on its previous values. As a result, that equation can be used to predict future values. This model is used to gain a better understanding of the data and to forecast future points in the series. It is applied to time series forecasting and provides complementary methods to the problem.

An ARIMA model is characterized by 3 terms:

- p, d, and q.

1. p - order of AR.
2. q - order of MA.
3. d - differencing we did for our time series stationary.
4. AR term - It is based on past values.
5. MA term - It is based on past errors.



The first step to build an ARIMA model is to make the time series stationary because 'Auto Regressive' in ARIMA means it is a linear regression model that uses its lags as predictors. Linear regression models work best when the predictors are not correlated and are independent of each other

What is p?
p is the order of the AR (Auto-Regressive) term. It refers to the number of Y lags to be utilized as predictors.

What is q?
q is the order of the MA (Moving Average) term. It refers to the number of lagged forecast errors that should go into the ARIMA Model.

What is d?
The most common approach to make the series stationary is to difference it i.e, subtract the previous value from the current value depending on the complexity of the series, more than one differencing may be needed. The value of d is the minimum number of differencing needed to make the series stationary. If the time series is stationary, then d = 0.

SARIMA Model:

If a time series exhibits seasonal patterns, seasonal terms must be included and it becomes SARIMA, short for ‘Seasonal ARIMA’.

Seasonal Order:
(P,Q,D,Seasonality)

Importing libraries and required:

```
#importing libraries
import numpy as np
import pandas as pd

# for warnings
import warnings
warnings.filterwarnings('ignore')

# visualization
import matplotlib.pyplot as plt
import matplotlib.dates as mdates

#for plotly
import plotly.express as px
import plotly.graph_objects as go

#statmodels
import statsmodels.api as sm
from statsmodels.tsa.stattools import adfuller
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.graphics.tsaplots import plot_pacf
from statsmodels.graphics.tsaplots import plot_acf

# for evaluation
from sklearn import metrics
```

- Here we imported the required libraries into our notebook.

DATASET:

We had collected the dataset from
https://www.kaggle.com/sudalairajkumar/covid19-in-india

Columns: Date, Time ,State, cured, deaths, confirmed.

Import the data

```
#reading covid-19 india dataset
df=pd.read_csv('covid_19_india.csv')
```

```
df.head()
```

	Sno	Date	Time	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	Confirmed
0	1	2020-01-30	6:00 PM	Kerala	1	0	0	0	1
1	2	2020-01-31	6:00 PM	Kerala	1	0	0	0	1
2	3	2020-02-01	6:00 PM	Kerala	2	0	0	0	2
3	4	2020-02-02	6:00 PM	Kerala	3	0	0	0	3
4	5	2020-02-03	6:00 PM	Kerala	3	0	0	0	3

- Let's change the column name ‘state/union territory’ to ‘state’ and ‘confirmed’ to ‘ total’.
- Then let's create a new column named ‘active’ cases.

```
#changing column names
df.rename(columns={'State/UnionTerritory':'state','Confirmed':'total'},inplace= True)
df["active"]=df['total']-df['Deaths']-df['Cured']
```

```
#last day of dataset
lastday=df.loc[:,['Date']]
list=[]
for i in lastday.Date:
    list.append(i)
list[-1]
```

```
'2021-07-07'
```

```
#getting all the states on the lastdate of dataset
today_data=df[df.Date==list[-1]]
today_data.head()
```

	Sno	Date	Time	state	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	total	active
16814	16815	2021-07-07	8:00 AM	Andaman and Nicobar Islands	-	-	7343	128	7487	16
16815	16816	2021-07-07	8:00 AM	Andhra Pradesh	-	-	1861937	12898	1908065	33230
16816	16817	2021-07-07	8:00 AM	Arunachal Pradesh	-	-	34525	181	37879	3173
16817	16818	2021-07-07	8:00 AM	Assam	-	-	493306	4717	522267	24244
16818	16819	2021-07-07	8:00 AM	Bihar	-	-	711913	9612	722746	1221

- Let's sort the today_data from highest to lowest,
- So we will get the top five highest cases states

```
#sorting the values of total column from highest to lowest
top_cases=today_data.sort_values(by='total',ascending=False)
```

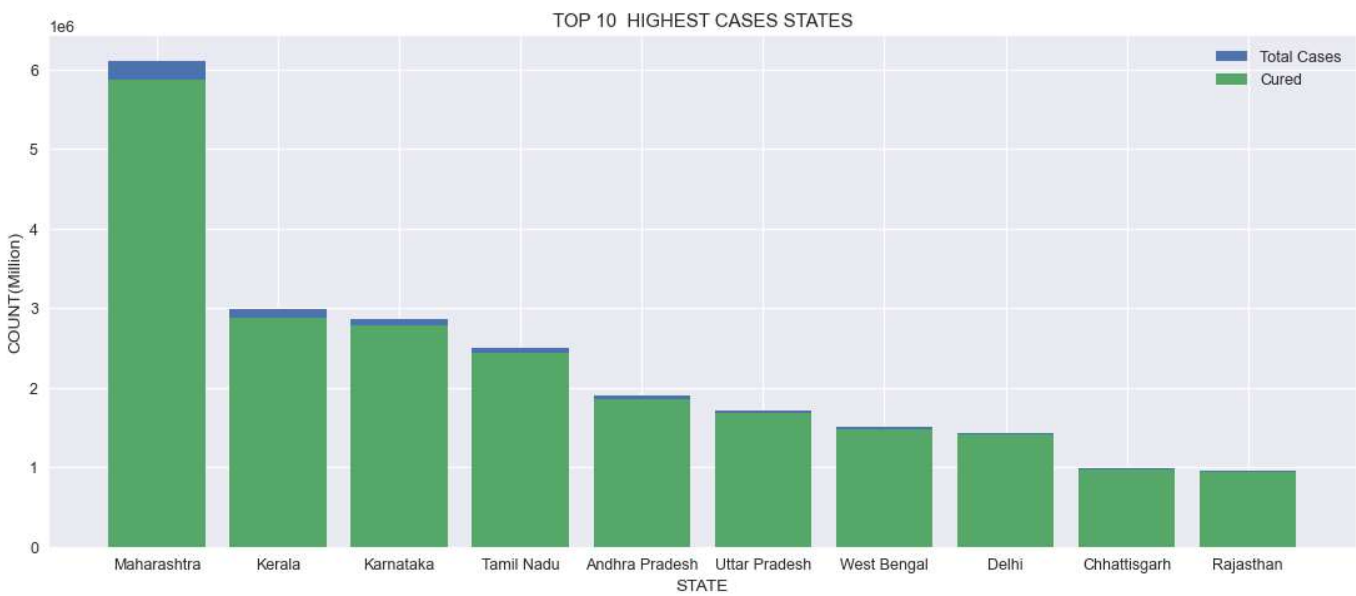
```
top_cases.head()
```

	Sno	Date	Time	state	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	total	active
16834	16835	2021-07-07	8:00 AM	Maharashtra	-	-	5872268	123531	6113335	117536
16830	16831	2021-07-07	8:00 AM	Kerala	-	-	2877557	13960	2996094	104577
16829	16830	2021-07-07	8:00 AM	Karnataka	-	-	2784030	35526	2859595	40039
16844	16845	2021-07-07	8:00 AM	Tamil Nadu	-	-	2435872	33132	2503481	34477
16815	16816	2021-07-07	8:00 AM	Andhra Pradesh	-	-	1861937	12898	1908065	33230

- From the above, we conclude that Maharastra has the highest number of cases
- We got the top 5 highest case states.
- Let’s see the top 10 highest case states by visualization.

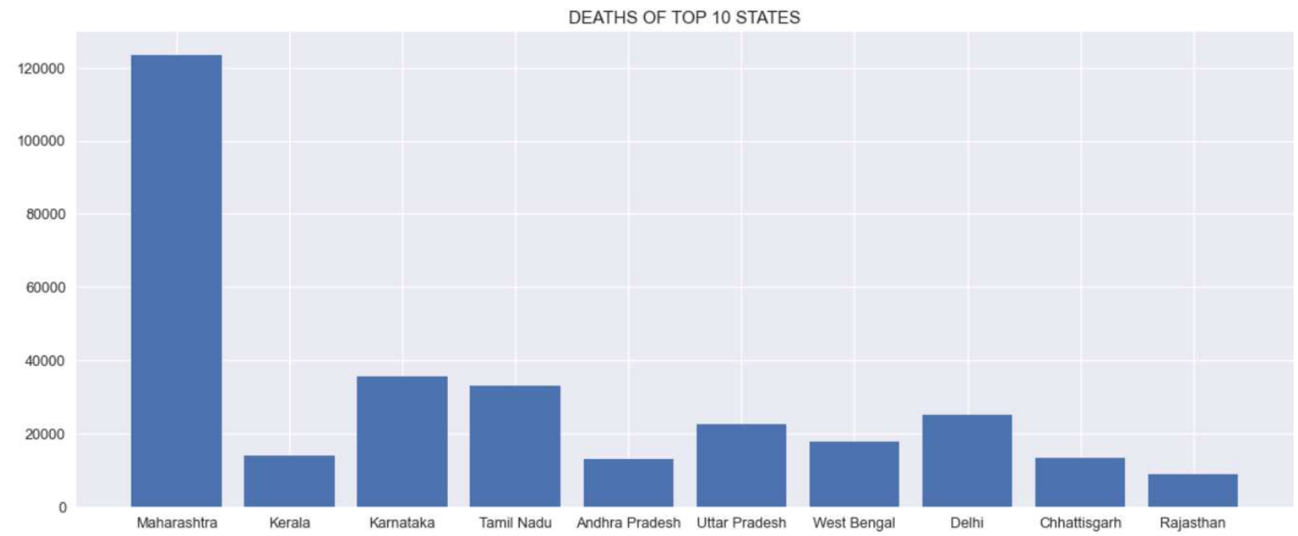
```
#USING matplotlib lets plot the  top 10 total cases of states
```

```
x=top_cases.state[:10]
y=top_cases.total[:10]
z=top_cases.Cured[:10]
plt.title('TOP 10  HIGHEST CASES STATES')
plt.bar(x,y,label='Total Cases')
plt.bar(x,z,label='Cured')
plt.xlabel('STATE')
plt.ylabel('COUNT(Million)')
plt.rcParams['figure.figsize']=(15,6)
plt.legend()
plt.show()
```



- After the top 10 highest case states, now, let us plot the deaths of these top 10 states.

```
#plotting the deaths of top 10 states
a=top_cases.state[:10]
b=x=top_cases.Deaths[:10]
plt.bar(a,b)
plt.title('DEATHS OF TOP 10 STATES')
plt.show()
```



ANALYSIS:

ANALYSIS OF MAHARASHTRA:

- Let us analyze the total active cases, number of people cured, deaths, and total cases of Maharashtra.

MAHARASHTRA

```
ms=df[df.state=='Maharashtra']
```

```
ms
```

	Sno	Date	Time	state	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	total	active
76	77	2020-03-09	6:00 PM	Maharashtra	2	0	0	0	2	2
91	92	2020-03-10	6:00 PM	Maharashtra	5	0	0	0	5	5
97	98	2020-03-11	6:00 PM	Maharashtra	2	0	0	0	2	2
120	121	2020-03-12	6:00 PM	Maharashtra	11	0	0	0	11	11
133	134	2020-03-13	6:00 PM	Maharashtra	14	0	0	0	14	14
...	...	...	...	...	...	...	...	...	...	...
16690	16691	2021-07-03	8:00 AM	Maharashtra	-	-	5836920	122353	6079352	120079
16726	16727	2021-07-04	8:00 AM	Maharashtra	-	-	5845315	122724	6088841	120802
16762	16763	2021-07-05	8:00 AM	Maharashtra	-	-	5848693	123030	6098177	126454
16798	16799	2021-07-06	8:00 AM	Maharashtra	-	-	5861720	123136	6104917	120061
16834	16835	2021-07-07	8:00 AM	Maharashtra	-	-	5872268	123531	6113335	117536

486 rows × 10 columns

```
ms=ms.loc[:,['Date','state','Cured','Deaths','total','active']]
ms.head()
```

	Date	state	Cured	Deaths	total	active
76	2020-03-09	Maharashtra	0	0	2	2
91	2020-03-10	Maharashtra	0	0	5	5
97	2020-03-11	Maharashtra	0	0	2	2
120	2020-03-12	Maharashtra	0	0	11	11
133	2020-03-13	Maharashtra	0	0	14	14

```
ms.describe()
```

	Cured	Deaths	total	active
count	4.860000e+02	486.000000	4.860000e+02	486.000000
mean	1.674463e+06	39741.835391	1.870149e+06	155944.508230
std	1.710989e+06	31861.231600	1.831266e+06	168833.300211
min	0.000000e+00	0.000000	2.000000e+00	2.000000
25%	1.197165e+05	9299.500000	2.187718e+05	46101.000000
50%	1.556812e+06	44884.500000	1.706879e+06	96492.500000
75%	2.066541e+06	52468.500000	2.216942e+06	193650.750000
max	5.872268e+06	123531.000000	6.113335e+06	701614.000000

- Used ‘describe’ for the stat of total cases, cured, deaths and active cases as per date wise in Maharashtra.
- Removed unwanted columns from the dataset.

```
ms.isnull().sum()
```

```
Date      0
state     0
Cured     0
Deaths    0
total     0
active    0
dtype: int64
```

```
#changing date column into datetime type
ms.Date=pd.to_datetime(ms.Date)
```

```
ms.head()
```

	Date	state	Cured	Deaths	total	active
76	2020-03-09	Maharashtra	0	0	2	2
91	2020-03-10	Maharashtra	0	0	5	5
97	2020-03-11	Maharashtra	0	0	2	2
120	2020-03-12	Maharashtra	0	0	11	11
133	2020-03-13	Maharashtra	0	0	14	14

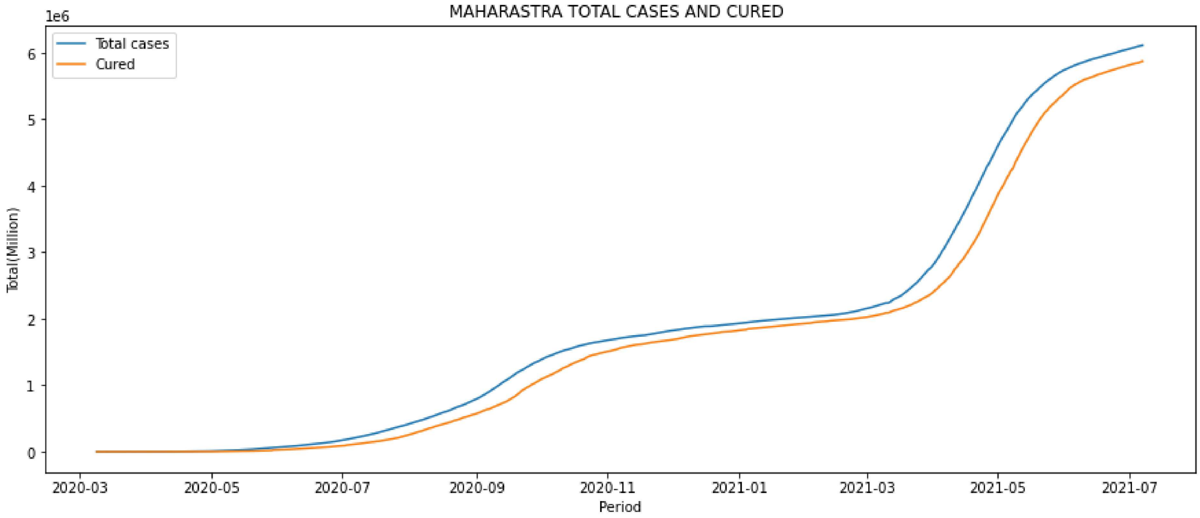
- No null values in the dataset and changed the type of date column into datetime

- Plotting the graph of total cases and the number of people cured in Maharashtra by assuming the time period on the x-axis and the total number of people(in millions) on the y-axis.

VISUALIZATION:

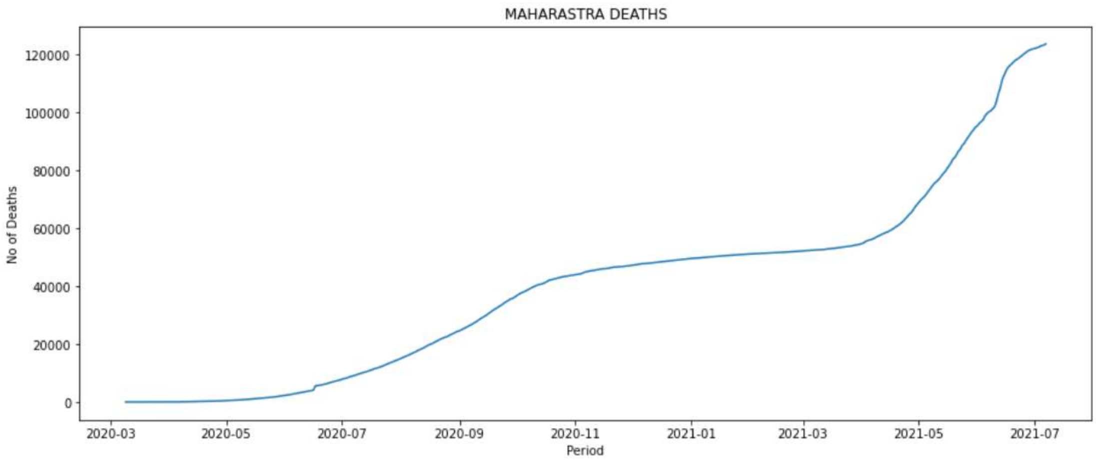
```
#Lets visualize Maharastra state
# period vs Total cases and cured
x=ms.Date
y=ms.total
plt.title('MAHARASTRA TOTAL CASES AND CURED ')
plt.plot(x,y,label='Total cases')
plt.plot(x,ms.Cured,label='Cured')
plt.xlabel('Period')
plt.ylabel('Total(Million)')
plt.legend()
plt.show()
```

- The total cases and cured cases started increasing in the line graph.

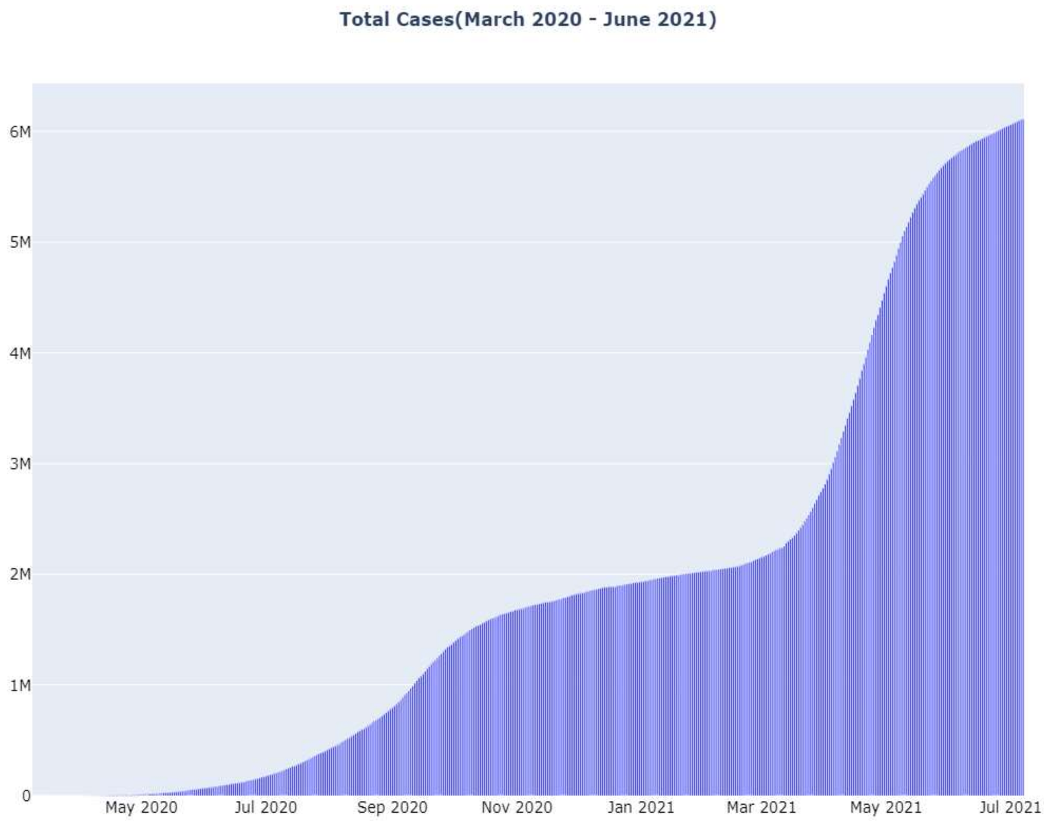


- Plotting the line graph for the total number of deaths in Maharashtra

```
#period vs no of Deaths
plt.plot(x,ms.Deaths)
plt.title('MAHARASTRA DEATHS')
plt.xlabel('Period')
plt.ylabel('No of Deaths')
plt.show()
```

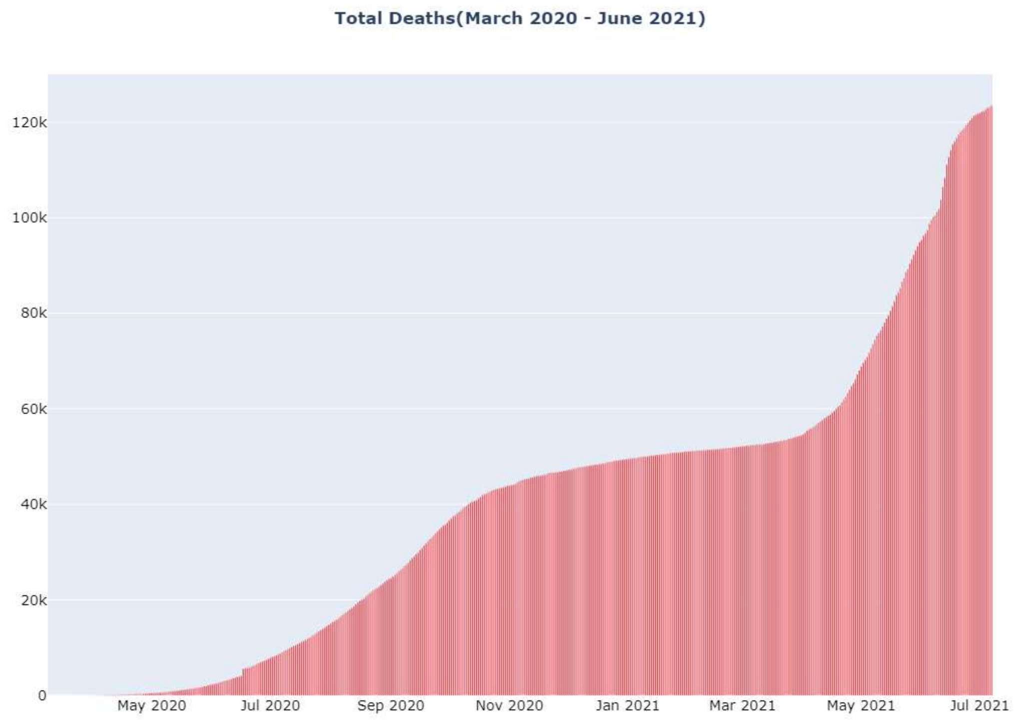


- From the above, we can observe deaths raised in July 2020 and in April 2021.
- Let's plot the same(above) using Plotly for better visualization.
- The below graph indicates the number of total cases between March 2020 to July 2021. This gives a view of how the total cases raised.

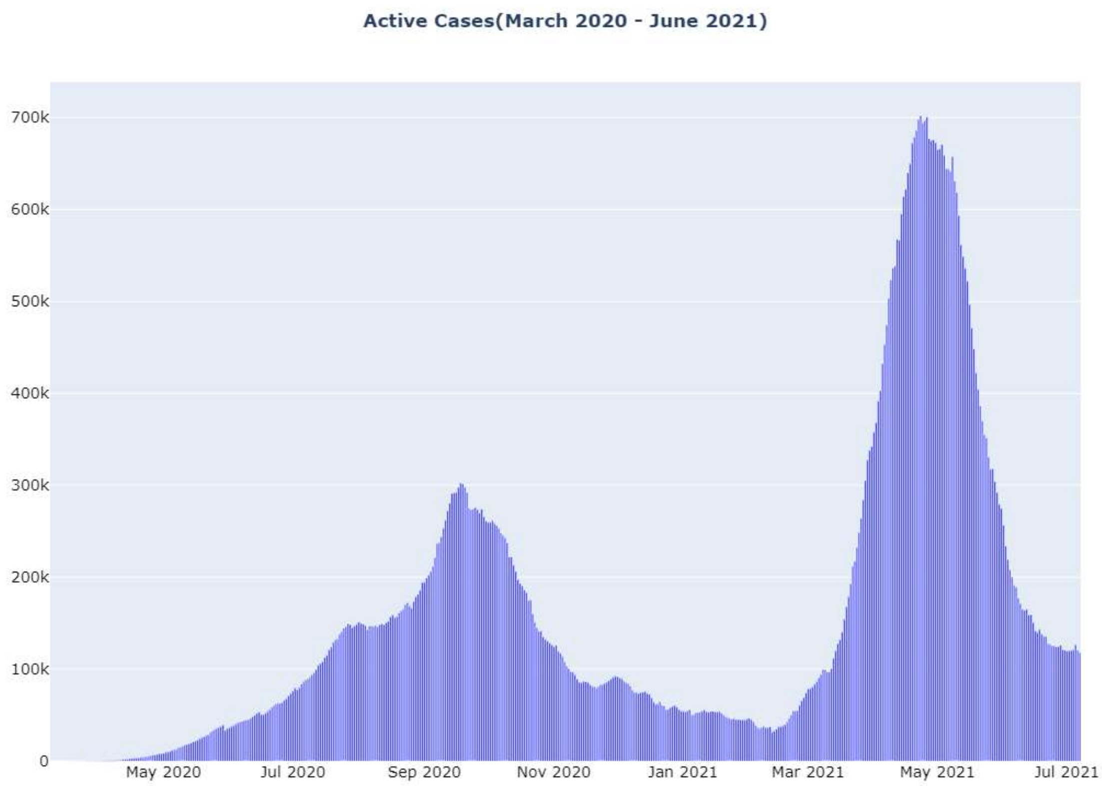


- The below graph indicates the number of people who are cured of the disease between March 2020 to July 2021. This interprets that the number of people cured of the disease is high.

- The below graph indicates the number of people deceased between March 2020 to July 2021.



- The below graph indicates the total active cases between March 2020 to July 2021. By observation, we can say that the number of active cases went uphill at the time of September and October 2020 and gradually decreased with time being.
- The total active cases were very low in March 2021 and a sudden increase of cases happened in May 2021. This was very high compared to September and October 2020.



CREATING A NEW COLUMN USING EXCEL:

```
#saved maharastra dataset to csv
ms.to_csv('maharashtra.csv')
#taking this csv into excel file,  I created a new column ('NEW CASES') in excel
```

```
#reading our new csv file
ums=pd.read_csv('maharashtra (1) - maharashtra (1).csv')
```

```
ums.head()
```

	Date	state	Cured	Deaths	total	active	new_cases
0	2020-03-09	Maharashtra	0	0	2	2	2
1	2020-03-10	Maharashtra	0	0	5	5	3
2	2020-03-11	Maharashtra	0	0	7	7	2
3	2020-03-12	Maharashtra	0	0	11	11	4
4	2020-03-13	Maharashtra	0	0	14	14	3

```
ums.isnull().sum()
```

```
Date         0
state        0
Cured        0
Deaths       0
total        0
active       0
new_cases    0
dtype: int64
```

- Above we saved the 'ms' dataset to csv then imports the csv file in excel and created a new column 'new_cases' using row(n)-row(n-1).
- Later imported the updated csv 'ums' into our notebook.

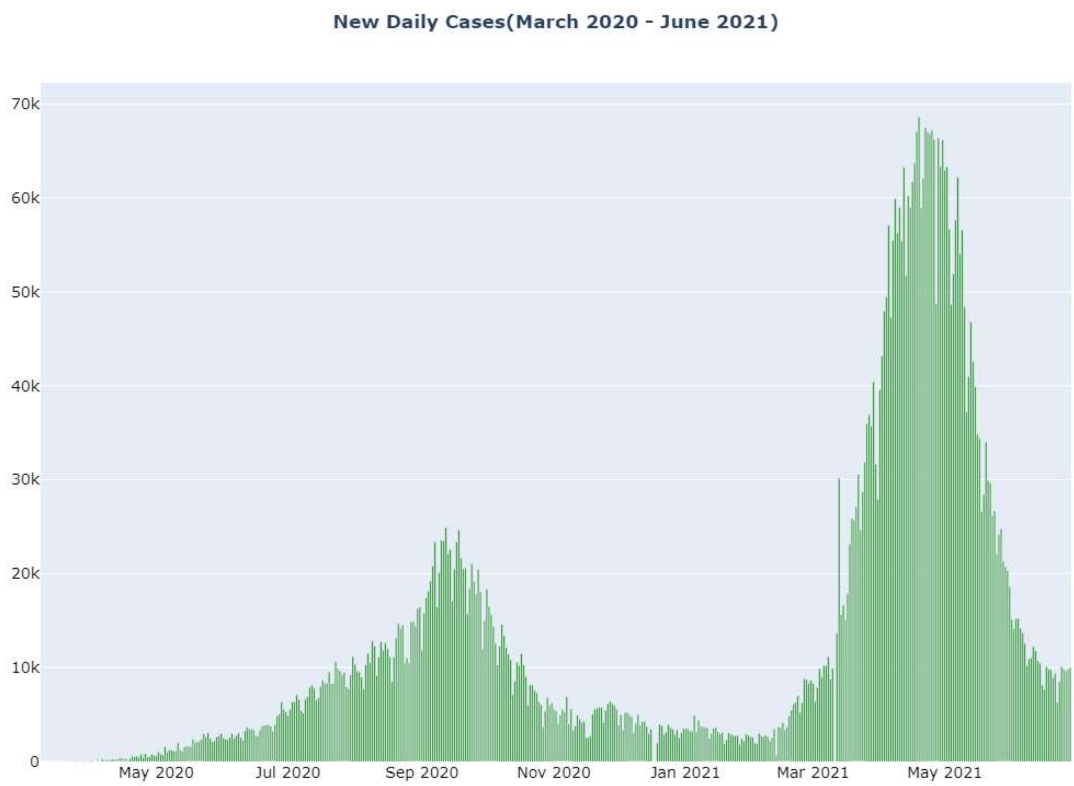
```
ifzero=ums[ums.new_cases==0 ]
ifzero
```

	Date	state	Cured	Deaths	total	active	new_cases
5	2020-03-14	Maharashtra	0	0	14	14	0
7	2020-03-16	Maharashtra	0	0	32	32	0
25	2020-04-03	Maharashtra	42	16	335	277	0
27	2020-04-05	Maharashtra	42	24	490	424	0
283	2020-12-17	Maharashtra	1769897	48434	1886807	68476	0
284	2020-12-18	Maharashtra	1774255	48499	1886807	64053	0
367	2021-03-11	Maharashtra	2089294	52556	2238398	96548	0

```
m_ms=ums[~(ums.new_cases==0)]
m_ms.head()
```

	Date	state	Cured	Deaths	total	active	new_cases
0	2020-03-09	Maharashtra	0	0	2	2	2
1	2020-03-10	Maharashtra	0	0	5	5	3
2	2020-03-11	Maharashtra	0	0	7	7	2
3	2020-03-12	Maharashtra	0	0	11	11	4
4	2020-03-13	Maharashtra	0	0	14	14	3

- Above we removed the rows where new_cases == 0.
- When new_cases==0, on that day there is no testing in Maharashtra. So we don't require that data for our prediction.
- Let us plot our new column 'new_cases' using plotly.
- The below graph indicates the 'daily new cases' between March 2020 to July 2021. By observation, we can say that the number of daily new cases went uphill at the time of September and October 2020, and a high increase in daily cases between March 2021 and June 2021.



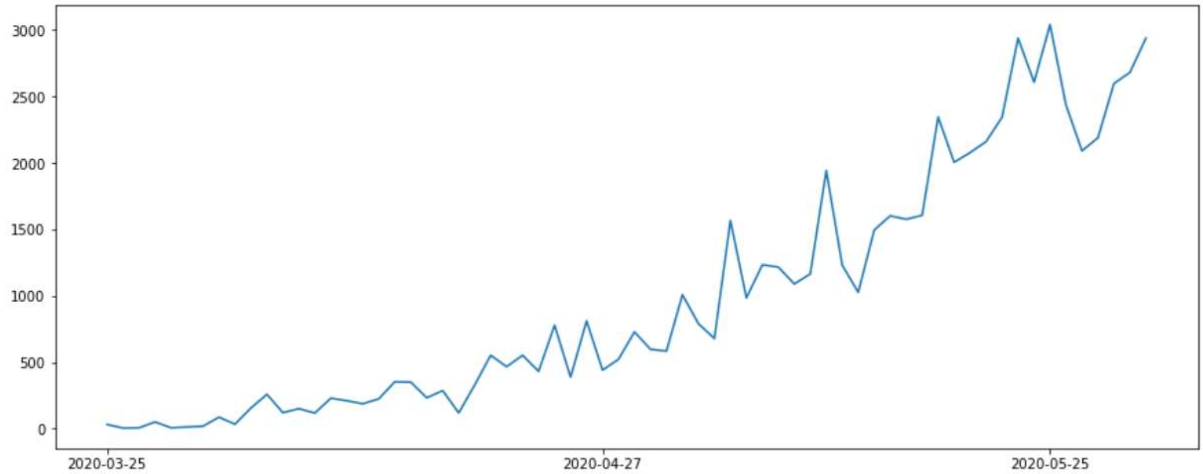
LOCKDOWNS AND UNLOCKS ANALYSIS:

```
lockdown1=m_ms[(m_ms['Date']>='2020-03-25')&(m_ms['Date']<='2020-05-31')]
```

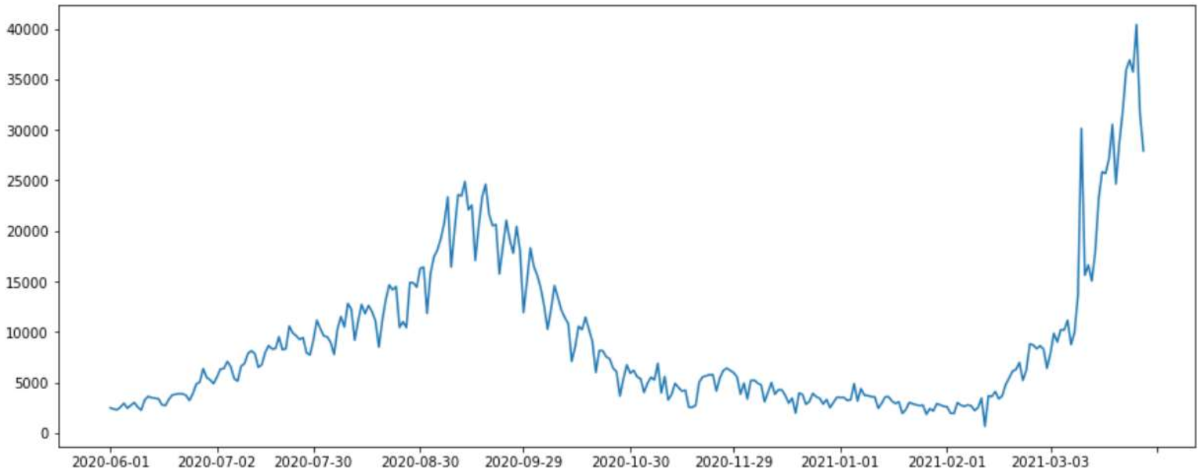
```
unlock1=m_ms[(m_ms['Date']>='2020-06-01')&(m_ms['Date']<='2021-03-31')]
```

```
lockdown2=m_ms[(m_ms['Date']>='2021-04-02')&(m_ms['Date']<='2021-06-01')]
```

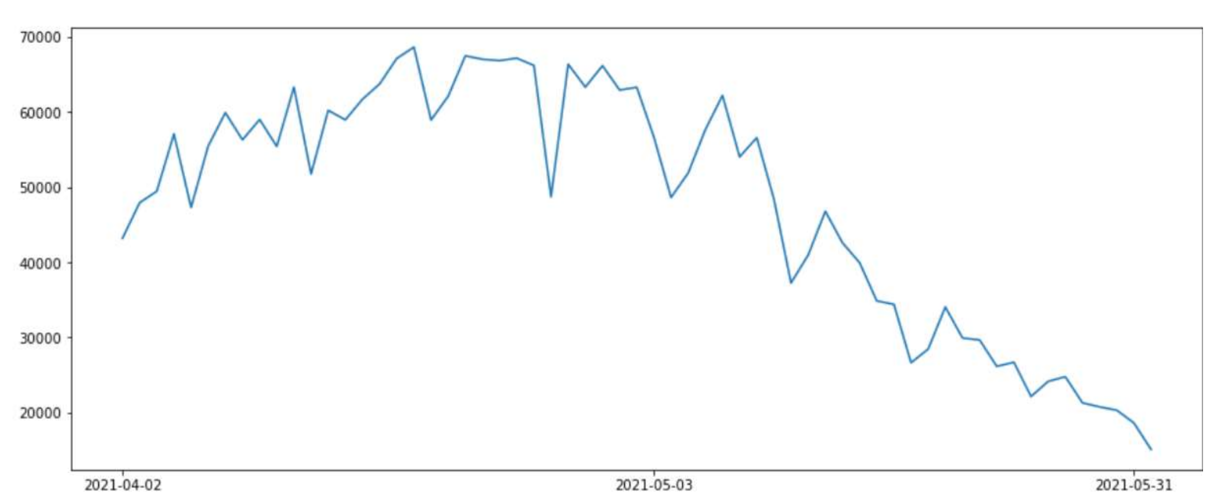
Lockdown 1:



Unlock:



Lockdown 2:



- In the state, lockdown 1 is declared from 25-03-2020 to 31-05-2020. The number of cases at this time also gradually increased day by day.
- The unlock is declared from 01-06-2020 to 31-03-2021. The count of the cases is -/+ more in unlock compared to the count of cases at the time of lockdown-1.
- At some point, the cases counted to the peak, and later on, they slowly started decreasing.
- During lockdown-1 and unlock the cases lasted a maximum of up to 25000 per day.
- At the time of mid-April 2021, the sudden spread of disease started and the number of cases per day increased.
- Then lockdown-2 was declared from 19-04-2021
- As time progressed the daily case count started increasing. Concerning the factors affecting the disease the cases count isn't stable and is increasing and decreasing.

<u>Note:</u>

- We updated our dataset in the visualization part and not in the model building.
- In plots, graphs are plotted till July 2021 but titles are not updated to July 2021.

IMPLEMENTATION:

We used the ARIMA Model for the prediction of the “number of daily cases “. We took 4 lags and 5 errors by choosing the best values for p and q using a for loop. Checked Stationarity by Dickey-Fuller Test (ADF test). Our predicted values depend on its past 4 lags and 5 errors with 7 days of seasonal component.

```
m_ms.head()
```

	Date	state	Cured	Deaths	total	active	new_cases
0	2020-03-09	Maharashtra	0	0	2	2	2
1	2020-03-10	Maharashtra	0	0	5	5	3
2	2020-03-11	Maharashtra	0	0	7	7	2
3	2020-03-12	Maharashtra	0	0	11	11	4
4	2020-03-13	Maharashtra	0	0	14	14	3

```
m_ms.set_index('Date',inplace=True)
```

```
m_ms.head()
```

	state	Cured	Deaths	total	active	new_cases
Date						
2020-03-09	Maharashtra	0	0	2	2	2
2020-03-10	Maharashtra	0	0	5	5	3
2020-03-11	Maharashtra	0	0	7	7	2
2020-03-12	Maharashtra	0	0	11	11	4
2020-03-13	Maharashtra	0	0	14	14	3

- We should convert the Date column as an index in the ARIMA model as we are using Time series data.

- So, we changed the index as the Date column using set_index.

Checking Stationarity:

- To check stationarity we used the Dickey-Fuller test.
- Dicker Fuller Test is a common statistical test used to test whether a given time series is stationary or not.

```
adfuller_test =adfuller(m_ms['new_cases'])
print('adfuller test P-value:', adfuller_test[1])
```

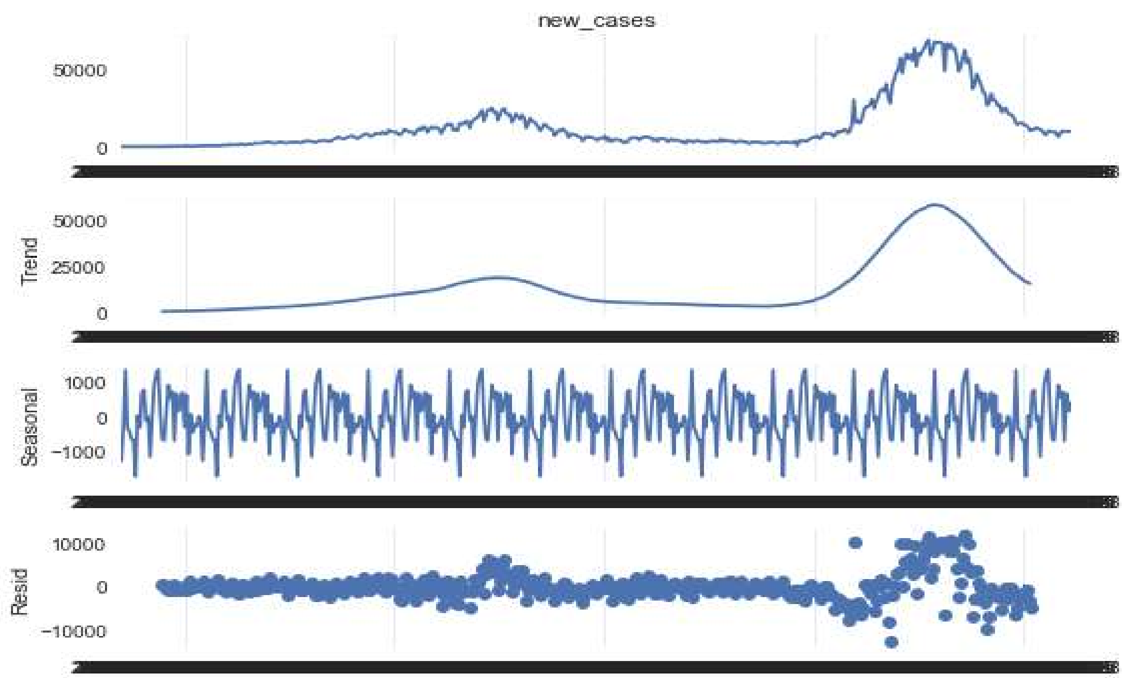
```
adfuller test P-value: 0.0023868640775724123
```

- Here, the p-value is less than 0.05 so our data is stationary.
- P-value > 0.05 , our data is not stationary.
- P-value < 0.05 , our data is stationary.

SEASONAL DECOMPOSITION:

- Seasonal decomposition involves thinking of a series as a combination of level, trend, seasonality.

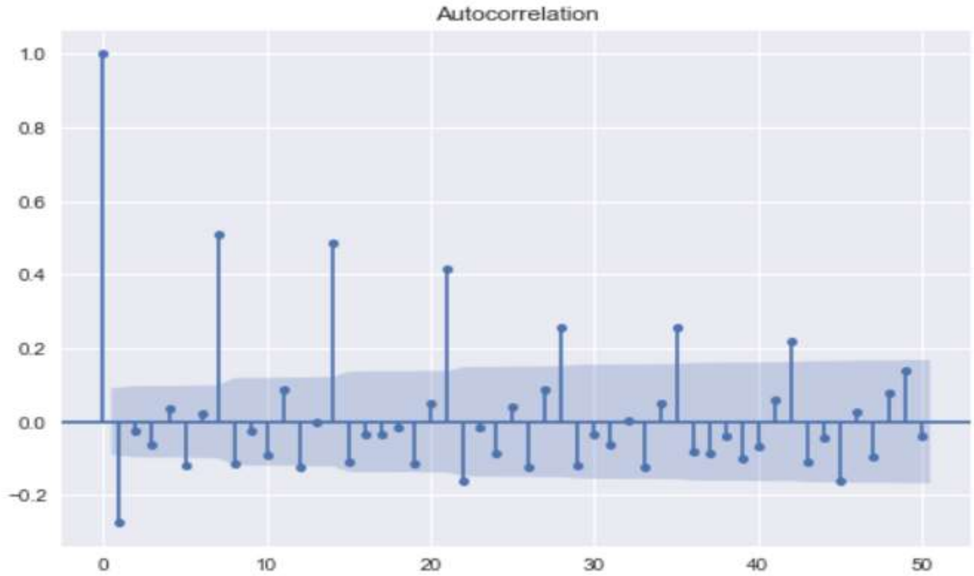
```
plt.style.use('seaborn')
seasonal_decompose(m_ms['new_cases'],period=40).plot();
```



ACF:

- The coefficient correlation between two values in a time series is called the AutoCorrelation Function.
- We select the MA using AutoCorrelationFunction.
- Here our best MA values are 1 and 5.

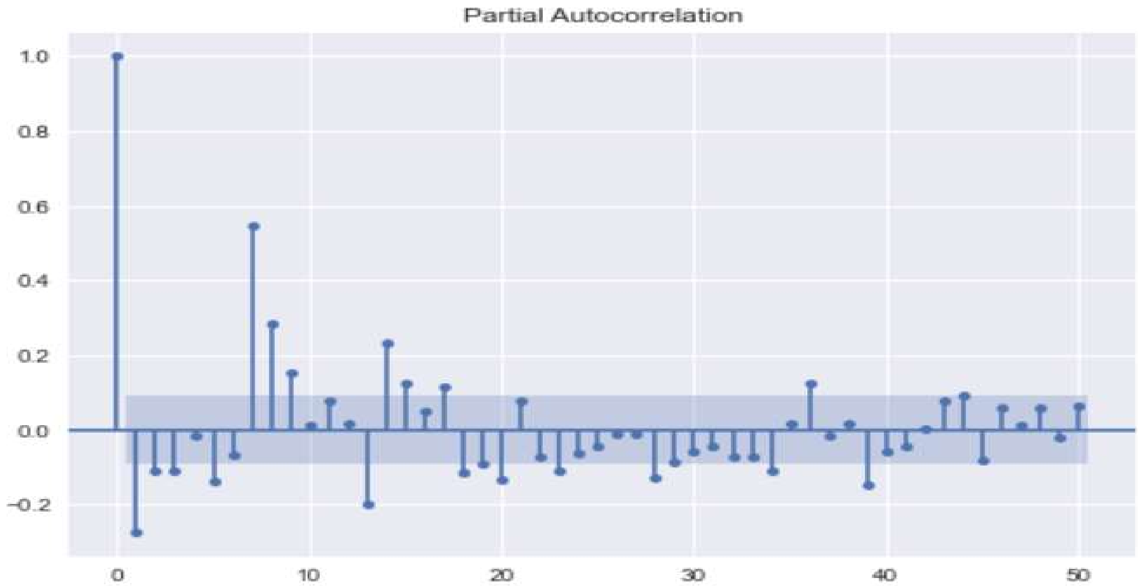
```
diff=m_ms['new_cases'].diff().dropna()
plot_acf(diff,lags=50);
```



PACF:

- Partial autocorrelation is the summary of the relation between a value with value at previous time steps.
- We select AR with Partial autocorrelation.

```
plot_pacf(diff,lags=50);
```



- To find the d, we used ndiffs from Arima. Firstly, we should fit the training set with the SARIMAX model. Then we need to predict the test data. So, we used summary_frame for getting the data frame of predicted values.
- In the predicted dataset, we can find the mean, mean_lower, and mean_upper. Mean_lower and Mean_upper are the range of predicted values.
- Then, we calculated the root mean squared error(RMSE).
- For better forecasting, we should fit the complete data into our model.
- So days = 240 (8 months).
- By using Plotly we plotted the total daily new_cases and our next 240 days predicted new_cases.

```
days=240
prediction=sarimax.get_forecast(steps=days)
pred_date=prediction.summary_frame(alpha=0.05).set_index(pd.date_range(start='2021-06-28',periods=days,freq='D'))
```

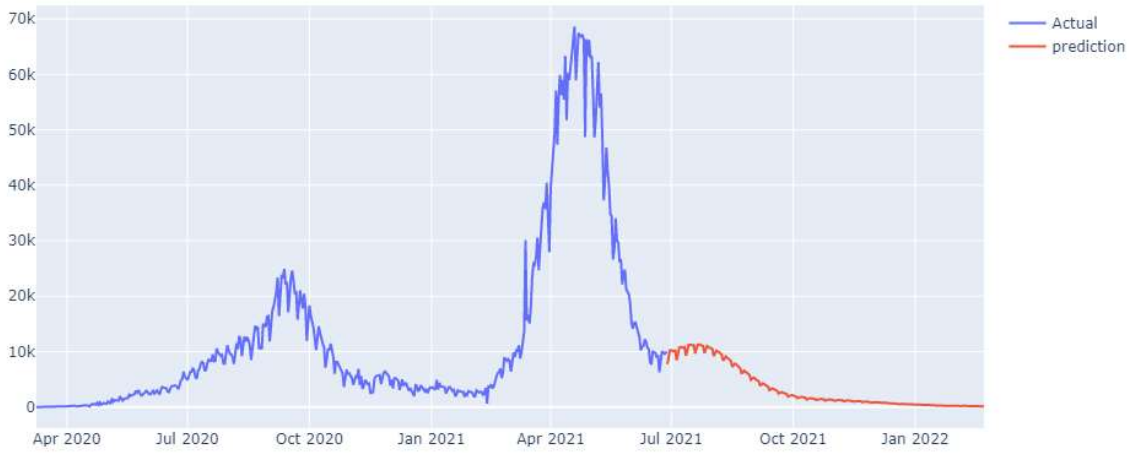
```
fig=go.Figure()

fig.add_trace(go.Scatter(
    name="Actual",
     x=m_ms.index, y=m_ms["new_cases"]))

fig.add_trace(go.Scatter(
    name="prediction",mode="lines",
     x=pred_date.index, y=pred_date['mean']))
```

RESULT:

8 months - Predicted cases:



CONCLUSION:

- Our goal is to predict the cases < 500, So we can conclude that virus transmission is decreased.
- By our model, at the end of December 2021, we will get less than < 500 Cases.

PRECAUTIONS:

- Washing hands frequently with soap and water, especially after blowing your nose, coughing, or sneezing, or after arriving from a public place.
- Avoid touching your face with unwashed hands. Use a hand sanitizer that contains a minimum of 60% alcohol.
- Avoid close contact with people.
- Cover your face with a shield and mask when you are surrounded by people.

REFERENCES:

[1] Tiwari, Sunita, Sushil Kumar, and Kalpna Guleria. "Outbreak trends of coronavirus disease–2019 in India: a prediction." Disaster medicine and public health preparedness 14, no. 5 (2020): e33-e38.

[2] Jia, Lin, Kewen Li, Yu Jiang, and Xin Guo. "Prediction and analysis of coronavirus disease 2019." arXiv preprint arXiv:2003.05447 (2020).

[3] Kuniya, Toshikazu. "Prediction of the epidemic peak of coronavirus disease in Japan, 2020." Journal of clinical medicine 9, no. 3 (2020): 789.

[4] Salehi, Ahmad Waleed, Preety Baglat, and Gaurav Gupta. "Review on machine and deep learning models for the detection and prediction of Coronavirus." Materials Today: Proceedings 33 (2020): 3896-3901.

[5] Zhong, Linhao, Lin Mu, Jing Li, Jiaying Wang, Zhe Yin, and Darong Liu. "Early prediction of the 2019 novel coronavirus outbreak in the mainland China based on simple mathematical model." Ieee Access 8 (2020): 51761-51769.

[6] Bentout, Soufiane, Abdennasser Chekroun, and Toshikazu Kuniya. "Parameter estimation and prediction for coronavirus disease outbreak 2019 (COVID-19) in Algeria." AIMS Public Health 7, no. 2 (2020): 306.

[7] Abdi, Milad. "Coronavirus disease 2019 (COVID-19) outbreak in Iran: Actions and problems." Infection Control & Hospital Epidemiology 41, no. 6 (2020): 754-755.