A Project Report

on

COVID-19 CASE DETECTION USING ML

Submitted in partial fulfillment of the requirement for the award of the degree of

Bachelor of Technology in Computer Science and Engineering

GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Under The Supervision of

Dr. S. Rakesh Kumar

Assistant Prof.

Department of Computer Science and Engineering

Submitted By

19SCSE1010309 – SHRUTI SUMAN

19SCSE1010299 – NISHKARSH MITTAL

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GALGOTIAS UNIVERSITY, GREATER NOIDA

INDIA

December, 2021

A Project Report

on

COVID-19 CASE DETECTION USING ML

Submitted in partial fulfillment of the requirement for the award of the degree of

Bachelor of Technology in Computer Science and Engineering

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Under The Supervision of

Dr. S. Rakesh Kumar

Assistant Prof.

Department of Computer Science and Engineering

Submitted By

19SCSE1010309 – SHRUTI SUMAN

19SCSE1010299 – NISHKARSH MITTAL

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA
INDIA

December, 2021

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA

CANDIDATE'S DECLARATION

I/We hereby certify that the work which is being presented in the project, entitled" COVID-19 CASE DETECTION USING ML : A cross-platform Application " in partial fulfillment of the requirementsfor the award of

BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING

submitted in the School of Computing Science and Engineering of Galgotias University, Greater Noida, is an original work carried out during the period of JULY-2021 to DECEMBER-2021, under the supervision of DR. S RAKESH KUMAR, Assistant Professor, Department of Computer Science and Engineering of School of Computing Science and Engineering , Galgotias University, Greater Noida

The matter presented in the project has not been submitted by me/us for the award of any other degree of this or any other places.

19SCSE1010309 – SHRUTI SUMAN

19SCSE1010299 – NISHKARSH MITTAL

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Supervisor

(DR. S RAKESH KUMAR, Assistant Professor)

CERTIFICATE

The Final Thesis/Project/ Dissertation Viva-Voce examination of 19SCSE1010309 – SHRUTI SUMAN, 19SCSE1010299 – NISHKARSH MITTAL has been held on___________________and his/her work is recommended for the award of BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING.

Signature of Examiner(s) Signature of Supervisor(s)

Signature of Project Coordinator Signature of Dean

Date:

Place:

ABSTRACT

COVID-19 has already been declared as a pandemic by World Health Organization. Huge number of tests are being carried out across the country and there is a large number COVID positive patients. Everyone wants to know that what will be the numbers in future and when will be the peak and when will the number of cases drop. We will predict the number of cases as per the current and past scenario. Currently, COVID cases are being predicted on the basis of previously infected person and number of patients recovered. But, nowadays vaccination is carried out in different parts of the world including our country India. As people are getting vaccinated, the risk of them getting infected is reducing day by day. So, we need to predict taking all the following variables (including number of people vaccinated and number of people infected after vaccination) accurately in future. Also, we have to check the effectiveness of each and every vaccine and then predict the chances of people getting infected. This will help in getting the very accurate data what will be the scenario in future. Three types of vaccine are available in India (COVAXIN, COVISHIELD, SPUTNIK V). All have different effectiveness. So, after collecting all the data as we can predict future number of cases using machine learning algorithm.

Keywords—pandemic, peak, drop, vaccination, infection, effectiveness

TABLE OF CONTENT

CHAPTER 1

INTRODUCTION

Coronavirus disease (COVID-19) is an infectious disease caused by the SARS-CoV-2 virus. It is spreading continuously among the population of a particular locality. This spread of virus has almost stopped the whole world and is also responsible for decrease in economy of our country and many other countries of the world. Lockdown were imposed to contain the spread of this virus. There were a lot of cases increasing on daily basis and fatality rate also increased. The government had taken various measures to protect our population from this pandemic. This has not only affected the whole country economically, but has also caused financial as well as health problems on the individual level. Many lost their jobs and a lot of people have faced losses in business. The overall point is that due to this Corona virus pandemic, people have lost many things. Its affect reduces for sometimes but it again comes back in the form of wave and lefts people losing lives, jobs, business etc. Everyone wants to know the future that when will this pandemic end, when will be the peak phase to get prepared for the next waves to come.

Here is the list of number of infected people in different states of India.

Top 5 most affected states by coronavirus disease 2019:

State	Case Reported	Death	Recovered
Maharashtra	66.3L	1.41L	63.5L
Kerala	51L	37,495	50.4L
Karnataka	29.9L	38,175	29.5L
Tamil Nadu	27.2L	36,375	26.7L
Andhra Pradesh	20.7L	14,426	20.5L

There are a lot of factors on which the spread of Covid 19 (SARS-CoV-2 virus) depends. We will have to find all the factors and then find there dependencies on each other. After finding the dependency, we can classify the factors as dependent and independent variables, which we will further use for predicting the impact of covid 19 on our population and we will be able to predict the cases with accuracy. Some of the factors on the which the future number of new infected people depends are percentage of people infected in a particular area, number of people recovered, number of people who have got both the doses of vaccination etc. With all the details, we will make a machine learning model to predict the future number of cases. This can be helpful for the government as well as people to know when the peak will come and to get alert and take all the safety measures to prevent the spread of Coronavirus disease (SARS-CoV-2 virus).

CHAPTER 2

LITERATURE SURVEY

Coronaviruses are among the main pathogens that predominantly affect the human respiratory system. The focus of the literature review was, therefore, to outline the predominant variables and methodology used in studies related to the spread of the virus. People with prevalent illnesses such as diabetes, hypertension, diabetes, stroke, heart, or kidney failure, as well as elderly people with impaired immune systems, are at an increased risk of infection. Closed areas with low ventilation and airflow may increase the risk of infection. The spread of the virus is believed to occur through respiratory droplets from coughing and sneezing, as with other respiratory viruses, including influenza virus and rhinoviruses. Aerosol transmission is also possible in case of protracted exposure to elevated aerosol concentrations in closed spaces. In the context of the COVID- 19 pandemic, people across the world are using various methods to explore prediction models with the goal of addressing the problems caused by the pandemic. The motivation for this SLR was to help researchers across the world study the various prediction models that have been created by numerous authors from multiple countries by providing information on a comprehensive range of models in one place. A systematic review is a compilation of various studies related to a single topic. It aims to provide a comprehensive and unbiased review of all the relevant studies in a given field. Our SLR was conducted to determine which prediction models are currently available, and the objective of the study was to identify the various methods used to develop different types of prediction models and to conduct an effectiveness or quality assessment of the models, which helps to evaluate their accuracy. It is hoped that this SLR will help healthcare workers and researchers wisely and confidently choose accurate prediction models to facilitate healthcare management by arranging medical facilities and equipment. Researchers or scholars can enhance their research program by using this SLR to obtain up-to date information on the various techniques used in prediction models, as well as their efficiency and accuracy. All currently available prediction models for COVID-19 were systematically reviewed and critically appraised. There are presently a number of individual and prognostic models for COVID-19, all of which show moderate to excellent demarcation. To explore the different vaticination models and find the best-suited model in terms of furnishing high delicacy while minimizing the burden on the healthcare system and perfecting care for cases, both the opinion and prognostic evaluation of conditions need to be bettered. This study will impact decision-makers in colorful aspects.

CHAPTER 3

SYSTEM ANALYSIS

3.1 REQUIRED TOOLS

- **3.1.1 Hardware Requirements**
  - Processor – i3
  - Hard Disk – 5 GB
  - Memory – 1GB RAM

- **3.1.2 Software Requirements**
  - Windows Xp,
  - Windows 7(ultimate, enterprise)
    Visual studio 2010

ABOUT WINDOWS XP - Windows XP is a version of the Microsoft Windows operating system for personal computers. The letters "XP" stand for eXPerience. Microsoft released Windows XP on October 25, 2001. Windows XP replaced Windows 2000 and Windows ME, which helped bring the NT and 9x versions of Windows together.

ABOUT WINDOWS 7 - Windows 7 is a major release of the Windows NT operating system developed by Microsoft. It was released to manufacturing on July 22, 2009, and became generally available on October 22, 2009.

Required tools & Libraries

- Jupyter Notebook - The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and narrative text. Its uses include data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.
- Pandas – 1.2.4 - Pandas is an open-source library that is built on top of NumPy library. It is a Python package that offers various data structures and operations for manipulating numerical data and time series. It is mainly popular for importing and analyzing data much easier.
- Scikit-learn – 0.24.1
- NLTK – 3.6.2

To install above modules please run the following command:

pip install pandas scikit-learn nltk

CHAPTER 4

Problem Formulation

My goal was to develop public models that would reasonably predict the growth of the coronavirus for the next day. This goal can be broken down and formalized into two parts:

1. To build a model that will predict the next day's number of coronavirus cases based on the last n day's number of coronavirus cases and deaths.
2. To build a model that will predict the next day's number of coronavirus deaths based on the last k day's number of coronavirus cases and deaths.
3. While I learned that n and k should be the same value after rigorous testing of my models, when I first defined the problem, I assumed that n and k may be different numbers as to not limit the possibilities of improving my individual models.

CHAPTER 5

Proposed System

The proposed system is grounded on machine literacy frame. Logistic Retrogression algorithm has been used for prognosticating the COVID-19 in someone's body.

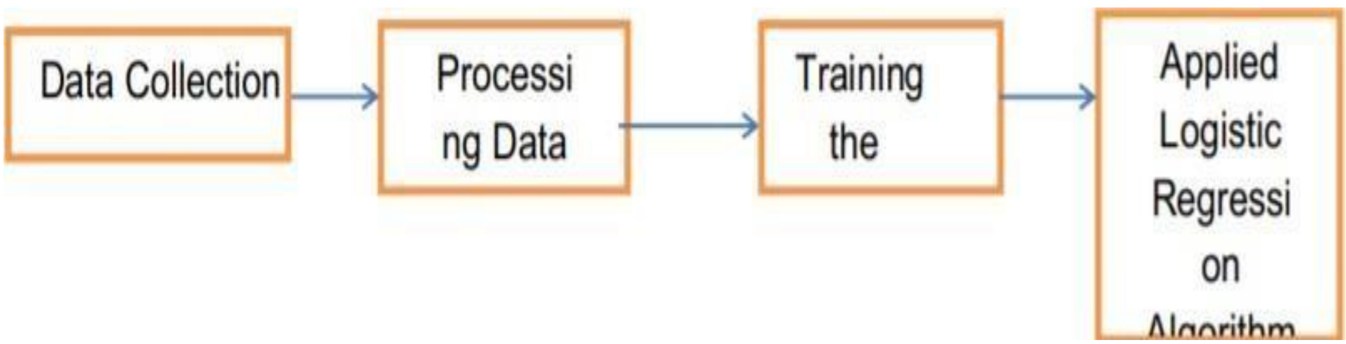


Fig. System Architecture

Implementation

```
import numpy as np

import pandas as pd

import os

for dirname, _, filenames in os.walk('/kaggle/input'):

    for filename in filenames:

        print(os.path.join(dirname, filename))
```

df=pd.read_csv("downloads/covid_19_india.csv")

state_testing = pd.read_csv('downloads/StatewiseTestingDetails.csv')

df_vc = pd.read_csv('downloads/covid_vaccine_statewise.csv/covid_vaccine_statewise.csv')

df.head()

Out[28]:

	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

Fig. head of dataset

df.info()

```
In [29]:  df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18110 entries, 0 to 18109
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Sno                       18110 non-null  int64
 1   Date                      18110 non-null  object
 2   Time                      18110 non-null  object
 3   State/UnionTerritory      18110 non-null  object
 4   ConfirmedIndianNational   18110 non-null  object
 5   ConfirmedForeignNational  18110 non-null  object
 6   Cured                     18110 non-null  int64
 7   Deaths                    18110 non-null  int64
 8   Confirmed                 18110 non-null  int64
dtypes: int64(4), object(5)
memory usage: 1.2+ MB
```

df.isnull().sum()

```
Out[30]: Sno                         0
         Date                        0
         Time                        0
         State/UnionTerritory        0
         ConfirmedIndianNational     0
         ConfirmedForeignNational    0
         Cured                       0
         Deaths                      0
         Confirmed                   0
         dtype: int64
```

df.drop(["Sno","Time","ConfirmedIndianNational","ConfirmedForeignNational"],inplace=True,axis=1) *df['Active_cases']=df['Confirmed']-(df['Cured']+df['Deaths'])*

df.head()

Out[32]:

	Date	State/UnionTerritory	Cured	Deaths	Confirmed	Active_cases
0	2020-01-30	Kerala	0	0	1	1
1	2020-01-31	Kerala	0	0	1	1
2	2020-02-01	Kerala	0	0	2	2
3	2020-02-02	Kerala	0	0	3	3
4	2020-02-03	Kerala	0	0	3	3

df['Confirmed'] = pd.to_numeric(df['Confirmed'], errors='coerce')

df['Confirmed']=df['Confirmed'].fillna(0)

df['Confirmed']=df['Confirmed'].astype('int')

df['Deaths'] = pd.to_numeric(df['Deaths'], errors='coerce')

df['Deaths']=df['Deaths'].fillna(0)

df['Deaths']=df['Deaths'].astype('int')

df['Cured'] = pd.to_numeric(df['Cured'], errors='coerce')

df['Cured']=df['Cured'].fillna(0)

df['Cured']=df['Cured'].astype('int')

df['Deaths'] = pd.to_numeric(df['Deaths'], errors='coerce')

df['Deaths']=df['Deaths'].fillna(0)

df['Deaths']=df['Deaths'].astype('int')

df['Cured'] = pd.to_numeric(df['Cured'], errors='coerce')

df['Cured']=df['Cured'].fillna(0)

df['Cured']=df['Cured'].astype('int')

#statewise analysis

statewise = pd.pivot_table(df, values=['Confirmed','Deaths','Cured'], index='State/UnionTerritory', aggfunc='max')

*statewise['Recovery Rate'] = statewise['Cured']*100 / statewise['Confirmed']*

*statewise['Mortality Rate'] = statewise['Deaths']*100 /statewise['Confirmed']*

statewise = statewise.sort_values(by='Confirmed', ascending= False)

statewise.style.background_gradient(cmap='YlOrRd')

	Confirmed	Cured	Deaths	Recovery Rate
State/UnionTerritory				
Maharashtra	6363442	6159676	134201	96.797865
Maharashtra*	6229596	6000911	130753	96.329056
Kerala	3586693	3396184	18004	94.688450
Karnataka	2921049	2861499	36848	97.961349
Karanataka	2885238	2821491	36197	97.790581
Tamil Nadu	2579130	2524400	34367	97.877967
Andhra Pradesh	1985182	1952736	13564	98.365591
Uttar Pradesh	1708812	1685492	22775	98.635309
West Bengal	1534999	1506532	18252	98.145471
Delhi	1436852	1411280	25068	98.220276

State/UnionTerritory	Confirmed	Cured	Deaths	Recovery Rate
Chhattisgarh	1003356	988189	13544	98.488373
Odisha	988997	972710	6565	98.353180
Rajasthan	953851	944700	8954	99.040626
Gujarat	825085	814802	10077	98.753704
Madhya Pradesh	791980	781330	10514	98.655269
Madhya Pradesh***	791656	780735	10506	98.620487
Haryana	770114	759790	9652	98.659419
Bihar	725279	715352	9646	98.631285
Bihar****	715730	701234	9452	97.974655
Telangana	650353	638410	3831	98.163613
Punjab	599573	582791	16322	97.201008
Assam	576149	559684	5420	97.142232
Telengana	443360	362160	2312	81.685312
Jharkhand	347440	342102	5130	98.463620

State/UnionTerritory	Confirmed	Cured	Deaths	Recovery Rate
Uttarakhand	342462	334650	7368	97.718871
Jammu and Kashmir	322771	317081	4392	98.237140
Himachal Pradesh	208616	202761	3537	97.193408
Himanchal Pradesh	204516	200040	3507	97.811418
Goa	172085	167978	3164	97.613389
Puducherry	121766	119115	1800	97.822873
Manipur	105424	96776	1664	91.796934
Tripura	80660	77811	773	96.467890
Meghalaya	69769	64157	1185	91.956313
Chandigarh	61992	61150	811	98.641760
Arunachal Pradesh	50605	47821	248	94.498567
Mizoram	46320	33722	171	72.802245
Nagaland	28811	26852	585	93.200514

State/UnionTerritory	Confirmed	Cured	Deaths	Recovery Rate
Sikkim	28018	25095	356	89.567421
Ladakh	20411	20130	207	98.623291
Dadra and Nagar Haveli and Daman and Diu	10654	10646	4	99.924911
Dadra and Nagar Haveli	10377	10261	4	98.882143
Lakshadweep	10263	10165	51	99.045114
Cases being reassigned to states	9265	0	0	0.000000
Andaman and Nicobar Islands	7548	7412	129	98.198198
Unassigned	77	0	0	0.000000
Daman & Diu	2	0	0	0.000000

pip install plotly

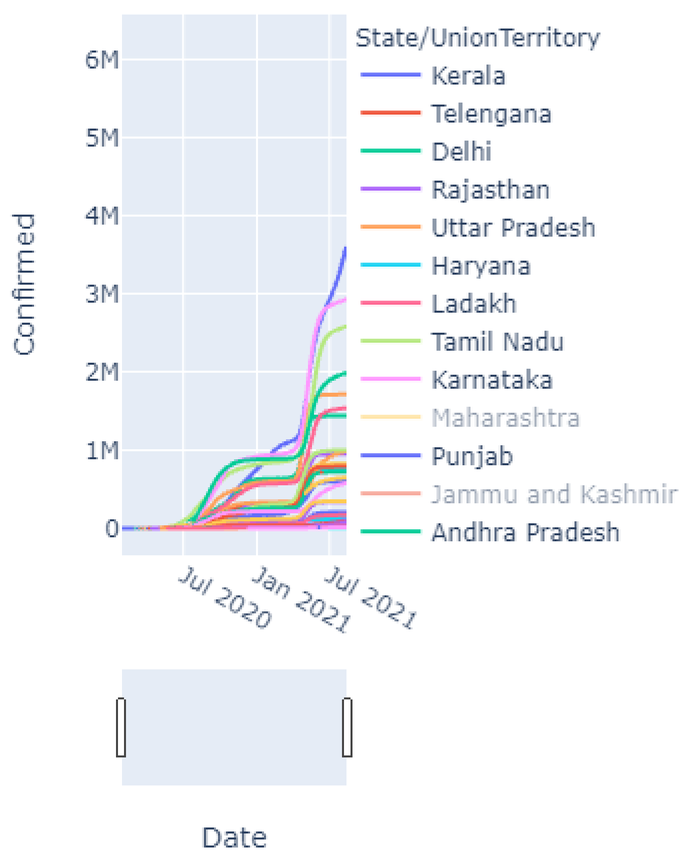
```
Collecting plotly
  Downloading plotly-5.5.0-py2.py3-none-any.whl (26.5 MB)
Collecting tenacity>=6.2.0
  Downloading tenacity-8.0.1-py3-none-any.whl (24 kB)
Requirement already satisfied: six in c:\programdata\anaconda3\lib\site-p
ackages (from plotly) (1.15.0)
Installing collected packages: tenacity, plotly
Successfully installed plotly-5.5.0 tenacity-8.0.1
Note: you may need to restart the kernel to use updated packages.
```

import seaborn as sns

import matplotlib.pyplot as plt

```
%matplotlib inline

import plotly.express as px

import plotly.offline as pyo

import plotly.graph_objs as go

from plotly.subplots import make_subplots

from IPython.display import display, Markdown
```

```
plt.figure(figsize = (18,10))
figure = px.line(df, x='Date', y='Confirmed', color='State/UnionTerritory')
figure.update_xaxes(rangeslider_visible=True)
pyo.iplot(figure)
```




```
locations = {
```

```
"Kerala" : [10.8505,76.2711],
"Maharashtra" : [19.7515,75.7139],
"Karnataka": [15.3173,75.7139],
"Telangana": [18.1124,79.0193],
"Uttar Pradesh": [26.8467,80.9462],
"Rajasthan": [27.0238,74.2179],
"Gujarat":[22.2587,71.1924],
"Delhi" : [28.7041,77.1025],
"Punjab":[31.1471,75.3412],
"Tamil Nadu": [11.1271,78.6569],
"Haryana": [29.0588,76.0856],
"Madhya Pradesh":[22.9734,78.6569],
"Jammu and Kashmir":[33.7782,76.5762],
"Ladakh": [34.1526,77.5770],
"Andhra Pradesh":[15.9129,79.7400],
"West Bengal": [22.9868,87.8550],
"Bihar": [25.0961,85.3131],
"Chhattisgarh":[21.2787,81.8661],
"Chandigarh":[30.7333,76.7794],
"Uttarakhand":[30.0668,79.0193],
"Himachal Pradesh":[31.1048,77.1734],
"Goa": [15.2993,74.1240],
"Odisha":[20.9517,85.0985],
"Andaman and Nicobar Islands": [11.7401,92.6586],
"Puducherry":[11.9416,79.8083],
"Manipur":[24.6637,93.9063],
"Mizoram":[23.1645,92.9376],
"Assam":[26.2006,92.9376],
"Meghalaya":[25.4670,91.3662],
```

```
    "Tripura":[23.9408,91.9882],
    "Arunachal Pradesh":[28.2180,94.7278],
    "Jharkhand" : [23.6102,85.2799],
    "Nagaland": [26.1584,94.5624],
    "Sikkim": [27.5330,88.5122],
    "Dadra and Nagar Haveli":[20.1809,73.0169],
    "Lakshadweep":[10.5667,72.6417],
    "Daman and Diu":[20.4283,72.8397]
    }
statewise["Lat"] = ""
statewise["Long"] = ""

for index in statewise.index :
    if index in locations:
        statewise.loc[statewise.index == index,"Lat"] = locations[index][0]
        statewise.loc[statewise.index == index,"Long"] = locations[index][1]
    else:
        statewise.drop([index],inplace=True)

import folium
from IPython.display import HTML, display
india = folium.Map(location=[23,80],
zoom_start=4,max_zoom=6,min_zoom=4,height=500,width="80%")
for i in range(0,len(statewise[statewise['Confirmed']>0].index)):
    folium.Circle(
        location=[statewise.iloc[i]['Lat'], statewise.iloc[i]['Long']],
        tooltip = "<h5 style='text-align:center;font-weight: bold'>"+statewise.iloc[i].name+"</h5>"+
                    "<hr style='margin:10px;'>"+
```

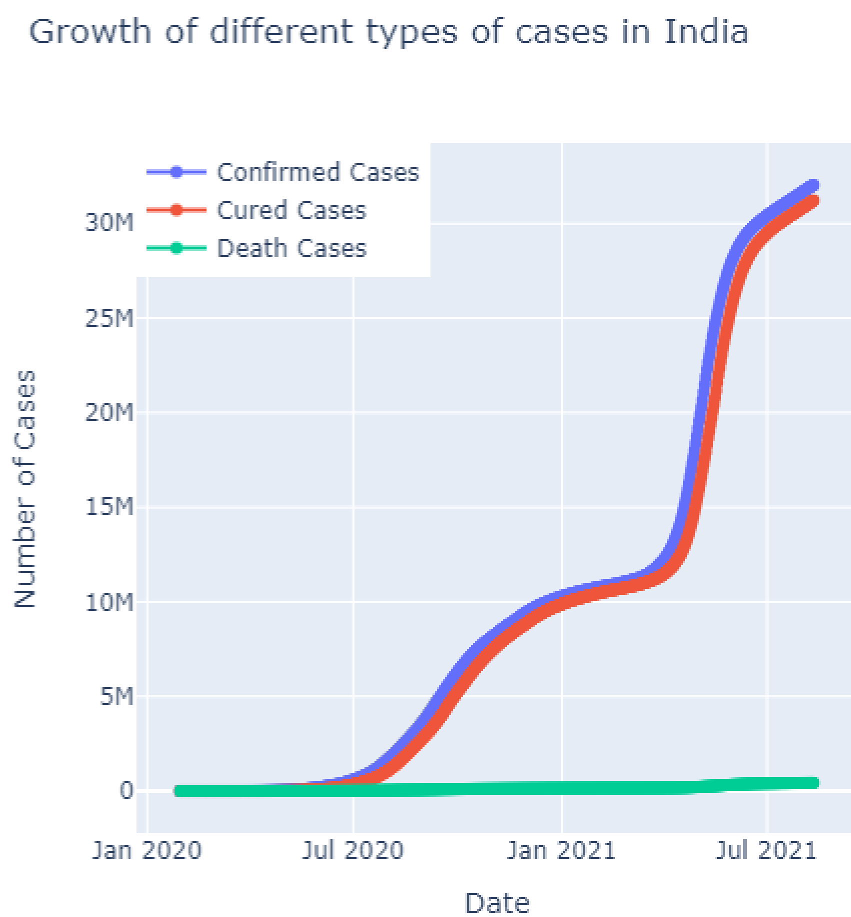
```
                "<ul style='color: #444;list-style-type:circle;align-item:left;padding-left:20px;padding-right:20px'>"+
        "<li>Confirmed: "+str(statewise.iloc[i]['Confirmed'])+"</li>"+"<li>Cured: "+str(statewise.iloc[i]['Cured'])+"</li>"+
        "<li>Deaths: "+str(statewise.iloc[i]['Deaths'])+"</li>"+
        "<li>Recovery  Rate:  "+str(np.round(statewise.iloc[i]['Recovery  Rate'],2))+"</li>"+
        "<li>Mortality  Rate:  "+str(np.round(statewise.iloc[i]['Mortality  Rate'],2))+"</li>"+
        "</ul>"
        ,
        radius=(int(np.log2(statewise.iloc[i]['Confirmed']+1)))*15000,
        color='#ff6600',
        fill_color='#ff8533',
        fill=True).add_to(india)

india
```

```
df1=df.groupby('Date')[['Cured','Deaths','Confirmed']].sum()
df1.tail()
```

Out[19]:

	Cured	Deaths	Confirmed
Date			
2021-08-07	31055861	427371	31895385
2021-08-08	31099771	427862	31934455
2021-08-09	31139457	428309	31969954
2021-08-10	31180968	428682	31998158
2021-08-11	31220981	429179	32036511

```
fig=go.Figure()
fig.add_trace(go.Scatter(x=df1.index, y=df1["Confirmed"],
        mode='lines+markers',
        name='Confirmed Cases'))
fig.add_trace(go.Scatter(x=df1.index, y=df1["Cured"],
        mode='lines+markers',
        name='Cured Cases'))
fig.add_trace(go.Scatter(x=df1.index, y=df1["Deaths"],
        mode='lines+markers',
        name='Death Cases'))
fig.update_layout(title="Growth of different types of cases in India",
       xaxis_title="Date",yaxis_title="Number of Cases",legend=dict(x=0,y=1,traceorder="normal"))
fig.show()
```




```
fig = make_subplots(rows=2, cols=1,
                subplot_titles=("Recovery Rate", "Mortatlity Rate"))
fig.add_trace(
   go.Scatter(x=df1.index, y=(df1["Cured"]/df1["Confirmed"])*100,
          name="Recovery Rate"),
   row=1, col=1
)
fig.add_trace(
   go.Scatter(x=df1.index, y=(df1["Deaths"]/df1["Confirmed"])*100,
          name="Mortality Rate"),
   row=2, col=1
```

```
)
fig.update_layout(height=1000,legend=dict(x=-0.1,y=1.2,traceorder="normal"))
fig.update_xaxes(title_text="Date", row=1, col=1)
fig.update_yaxes(title_text="Recovery Rate", row=1, col=1)
fig.update_xaxes(title_text="Date", row=1, col=2)
fig.update_yaxes(title_text="Mortality Rate", row=1, col=2)
fig.show()
```

-- Recovery Rate
-- Mortality Rate

Mortality rate = (Number of Death Cases / Number of Confirmed Cases) x 100

Recovery Rate= (Number of Recoverd Cases / Number of Confirmed Cases) x 100

Recovery Rate was initially very high when the number of positive (Confirmed) cases were low and showed a drastic drop with increasing number of cases. Increasing Mortality rate and dropped Recovery Rate is worrying sign for India.

Increasing Mortality Rate and very slowly increasing Recovery Rate is conclusive evidence for increase in number of Closed Cases

Recovery Rate is showing an upward trend which is a really good sign. Mortality Rate is showing a slight dips but with occasional upward trends.

```
growth_diff = []


for i in range(1,len(df1['Confirmed'])):
    growth_diff.append(df1['Confirmed'][i] / df1['Confirmed'][i-1])


growth_factor = sum(growth_diff)/len(growth_diff)
print('Average growth factor',growth_factor)
```

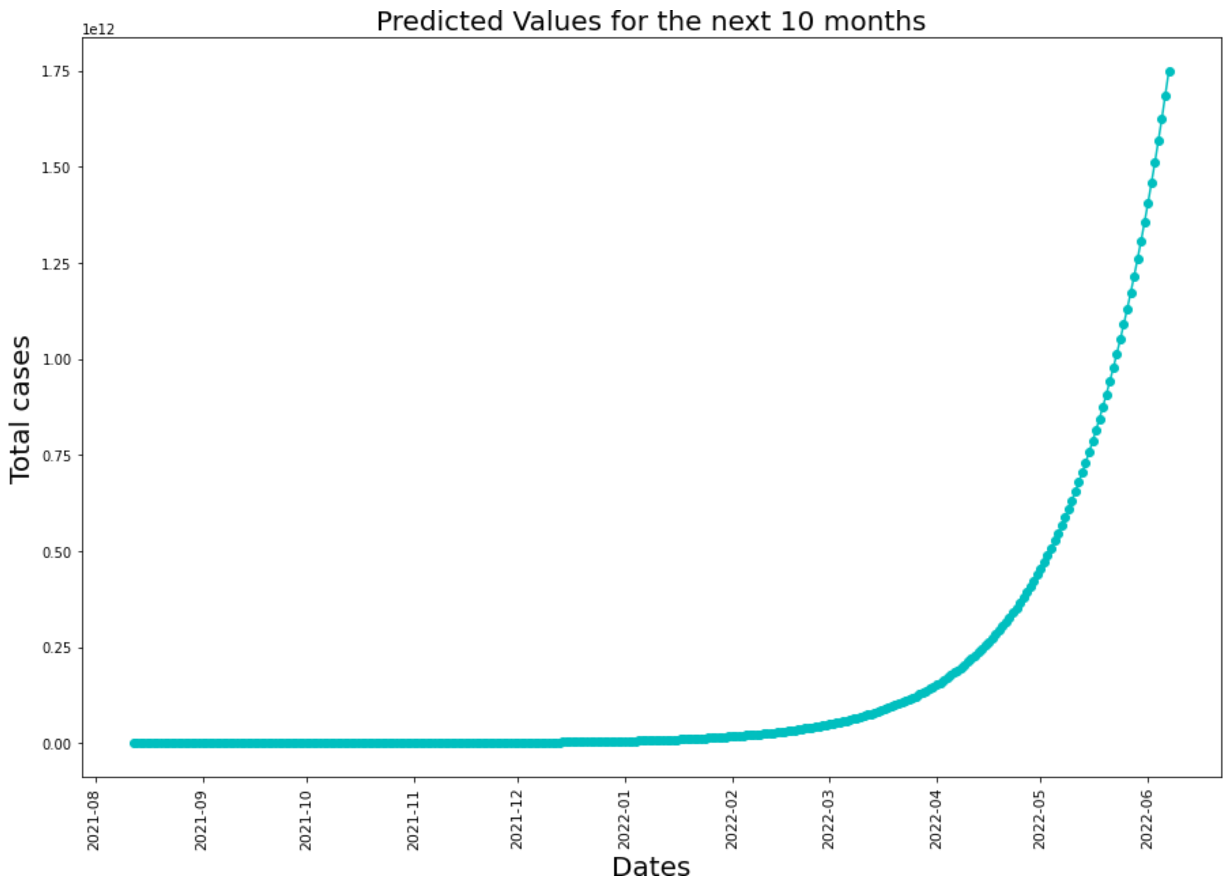
```
Average growth factor 1.0370264842594674
```

```
df1.reset_index(inplace=True)
df1.head()
```

Date	Cured	Deaths	Confirmed	
0	2020-01-30	0	0	1
1	2020-01-31	0	0	1
2	2020-02-01	0	0	2
3	2020-02-02	0	0	3

Date	Cured	Deaths	Confirmed	
4	2020-02-03	0	0	3

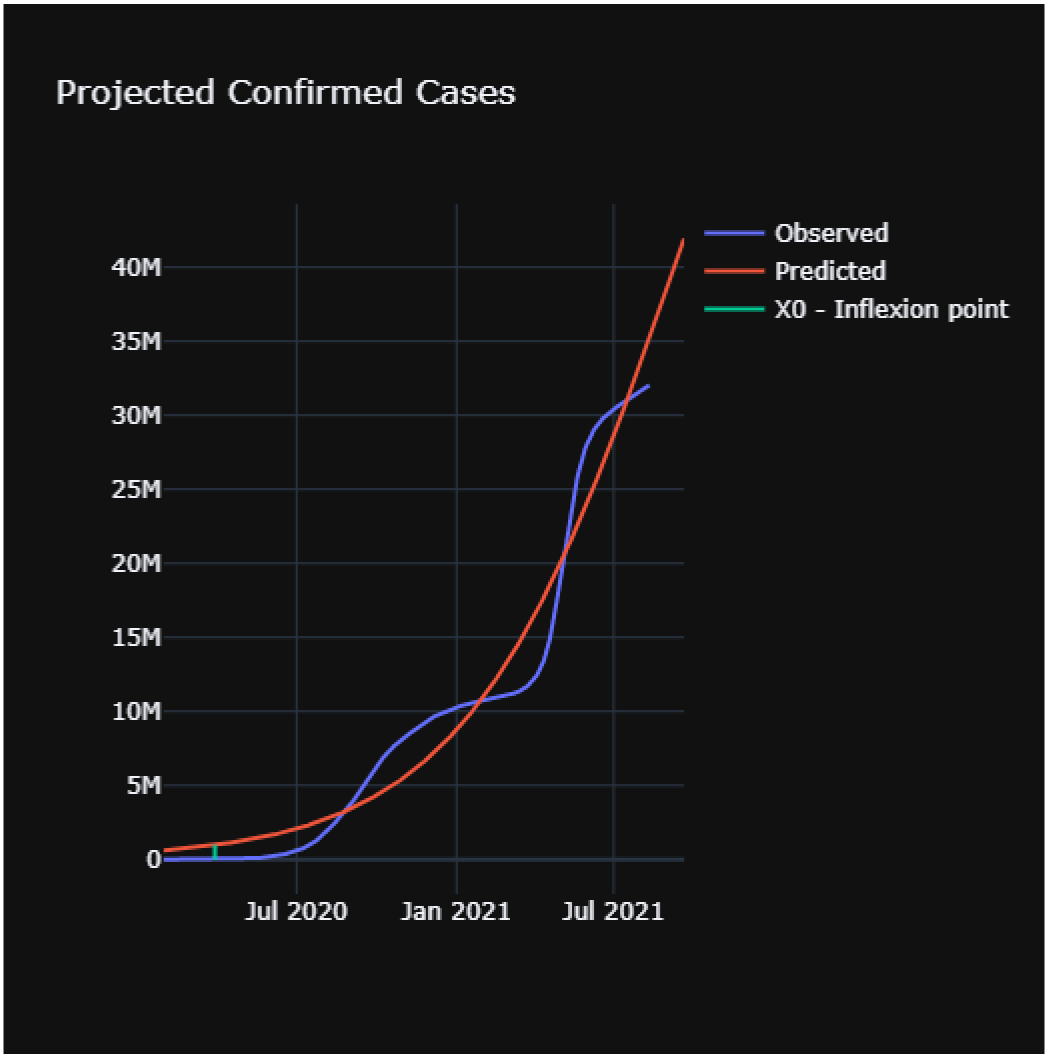
```
prediction_dates = []
from datetime import datetime,timedelta
df1['Date']=pd.to_datetime(df1['Date'], format='%Y-%m-%d')
start_date = df1['Date'][len(df1['Date']) - 1]
for i in range(300):
    date = start_date+timedelta(days=1)
    prediction_dates.append(date)
    start_date = date
previous_day_cases = df1['Confirmed'][len(df1['Date']) - 1]
predicted_cases = []

for i in range(300):
    predicted_value = previous_day_cases * growth_factor
    predicted_cases.append(predicted_value)
    previous_day_cases = predicted_value

plt.figure(figsize= (15,10))
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 10 months" , fontsize = 20)
ax1 = plt.plot_date(y= predicted_cases,x= prediction_dates,linestyle ='-',color = 'c')
```




```
predIndiaDF = df1.copy()

extended_period = 2000

from scipy.optimize import curve_fit

def sigmoid(x, L, k, x0):

   return L / (1 + np.exp(-k * (x - x0))) + 1


popt, pcov = curve_fit(sigmoid, (predIndiaDF.index+1).astype(float), predIndiaDF['Confirmed'],
p0=(0,0,0),maxfev=5000 )


x0 = int(popt[2]/10)

print('\033[1mx0 (point/day of inflexion):\033[0m',int(popt[2]/10))

print('\033[1mL (Maximum no.of cases):\033[0m',int(popt[0]/10000) )

print('\033[1mk (Growth Rate):\033[0m',round(float(popt[1]),2) )

print('\033[1mPCOV: \033[0m\n',pcov )
```

```python
fig = go.Figure()dateRange = pd.date_range(pd.to_datetime(predIndiaDF['Date'].min()),\
                    pd.to_datetime(predIndiaDF['Date'].min())+pd.DateOffset(extended_period))

fig.add_trace(go.Scatter(x = predIndiaDF['Date'],
            y = predIndiaDF['Confirmed'],
            mode = 'lines',
            name = 'Observed'))
fig.add_trace(go.Scatter(x = dateRange[:600],
            y = sigmoid([x for x in range(extended_period)],*popt),
            mode = 'lines',
            name = 'Predicted'))
fig.add_trace(go.Scatter(x = [dateRange[x0], dateRange[x0]],
            y = [0, sigmoid([x for x in range(extended_period)],*popt)[x0]],
            name = 'X0 - Inflexion point', mode = 'lines'))

fig.update_layout(template='plotly_dark',title='Projected Confirmed Cases')
fig.show()
```

```
x0 (point/day of inflexion): 59
L (Maximum no.of cases): 8151
k (Growth Rate): 0.01
PCOV:
 [[ 9.65384796e+13 -2.72822826e+03  2.43080876e+08]
 [-2.72822826e+03  8.91315475e-08 -7.05572390e-03]
 [ 2.43080876e+08 -7.05572390e-03  6.16079131e+02]]
```




```
import tensorflow as tf
from numpy import array
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import mean_squared_error
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.callbacks import ModelCheckpoint, ReduceLROnPlateau, EarlyStopping
from tensorflow.keras.layers import Dropout
from tensorflow.keras.layers import LSTM
```

```
from tensorflow.keras.utils import plot_model

from tensorflow.keras import Input

from tensorflow.keras.layers import BatchNormalizationfrom fbprophet import Prophet

fb_data = df1.copy()

fb_confirm = fb_data[['Date', 'Confirmed']]

fb_confirm = fb_confirm.rename(columns={'Date': 'ds',

                    'Confirmed': 'y'})


fb_confirm.head()
```

ds	Y	
0	2020-01-30	1
1	2020-01-31	1
2	2020-02-01	2
3	2020-02-02	3
4	2020-02-03	3

```
future_dates = my_model.make_future_dataframe(periods=15, freq='MS')

future_dates.tail()
```

ds	
570	2022-07-01

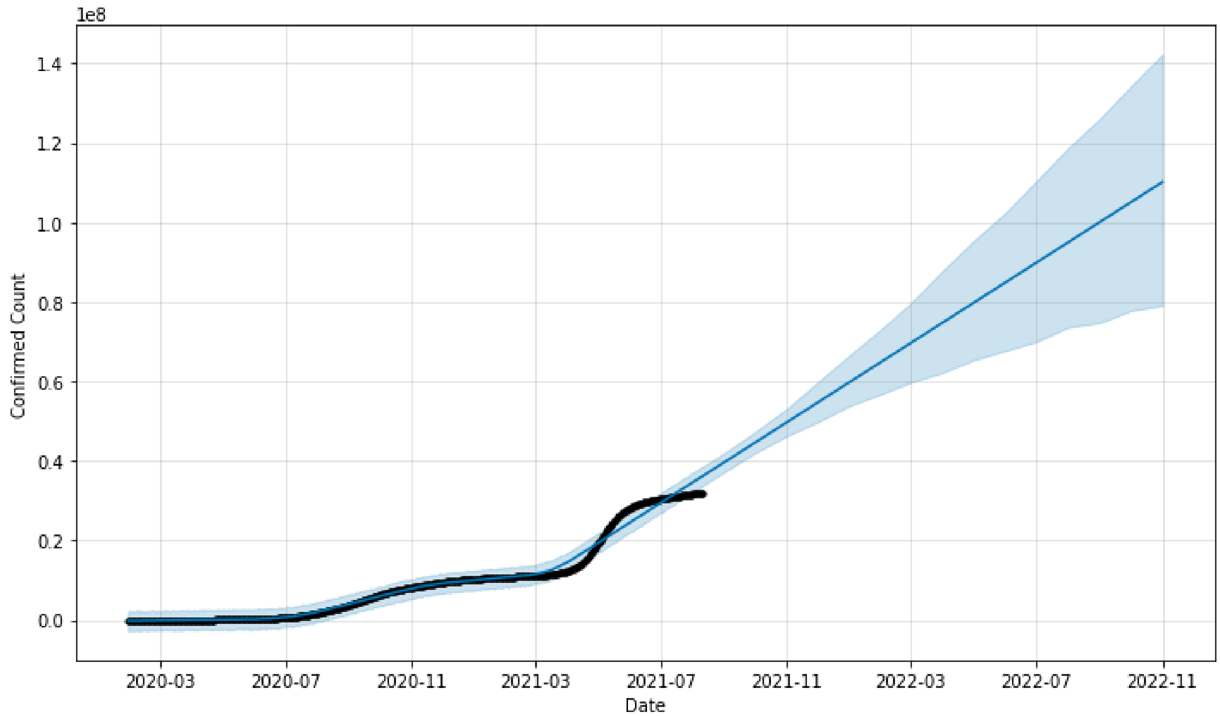
ds	
571	2022-08-01
572	2022-09-01
573	2022-10-01
574	2022-11-01

```
forecast = my_model.predict(future_dates)

forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail()
```

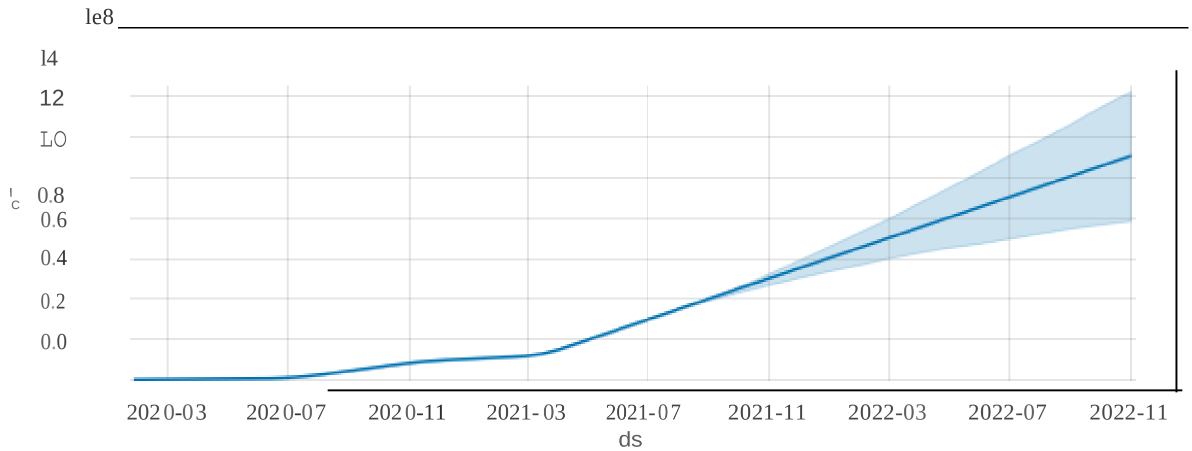
ds	Yhat	yhat_lower	yhat_upper	
570	2022-07-01	8.982125e+07	6.995001e+07	1.103366e+08
571	2022-08-01	9.492832e+07	7.359266e+07	1.186771e+08
572	2022-09-01	1.000809e+08	7.473027e+07	1.260847e+08
573	2022-10-01	1.050324e+08	7.775718e+07	1.342115e+08
574	2022-11-01	1.101297e+08	7.905184e+07	1.422771e+08

```
from fbprophet.plot import plot_plotly, add_changepoints_to_plot

fig = plot_plotly(my_model, forecast)

pyo.iplot(fig)
```

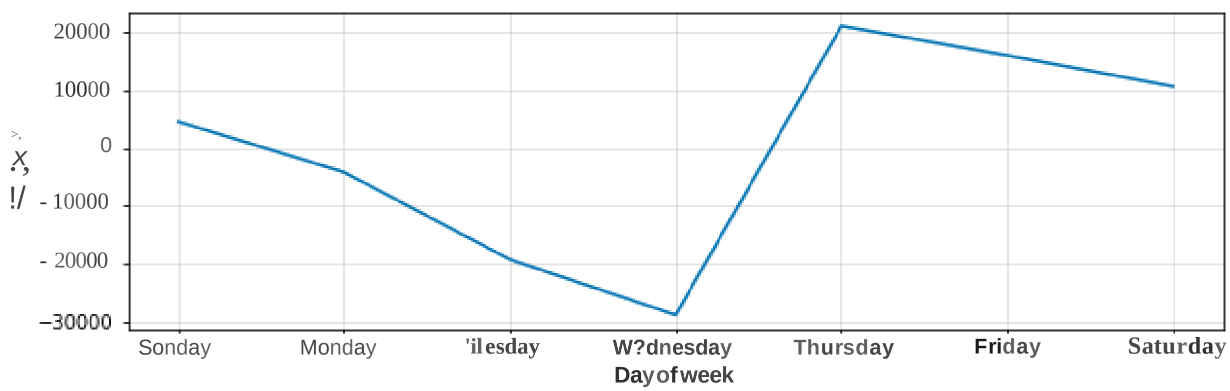
fig = my_model.plot(forecast,xlabel='Date',ylabel='Confirmed Count')



forecast_components=my_model.plot_components(forecast)



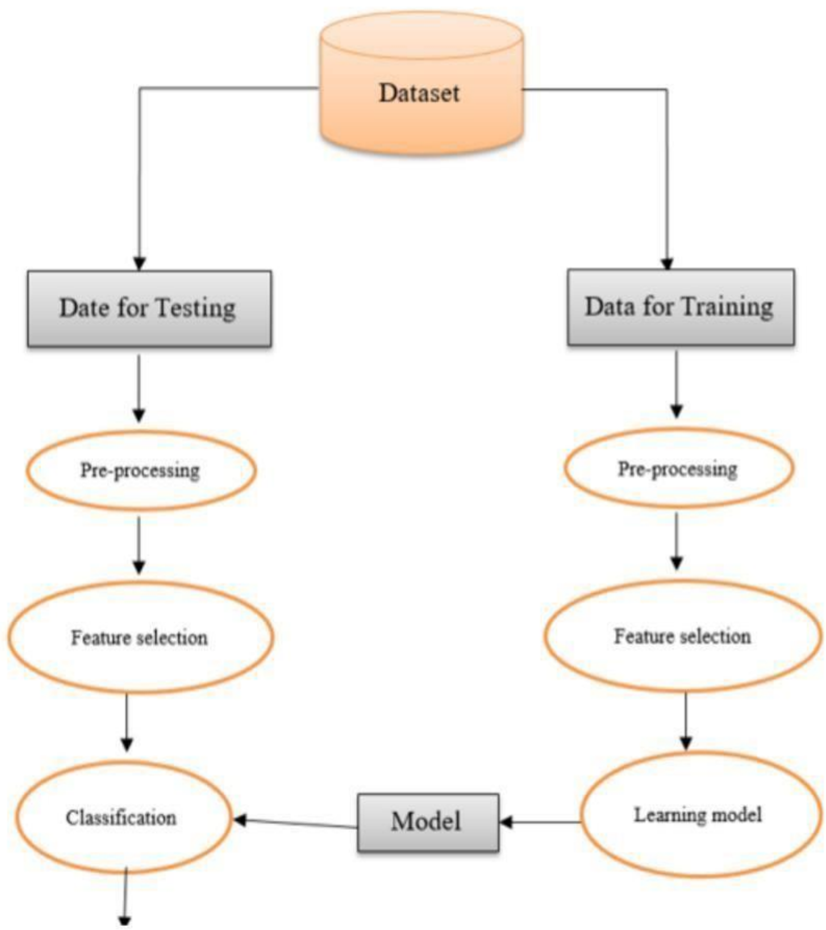
le8
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
2020-03
2020-07
2020-11
2021-03
2021-07
2021-11
2022-03
2022-07
2022-11
ds



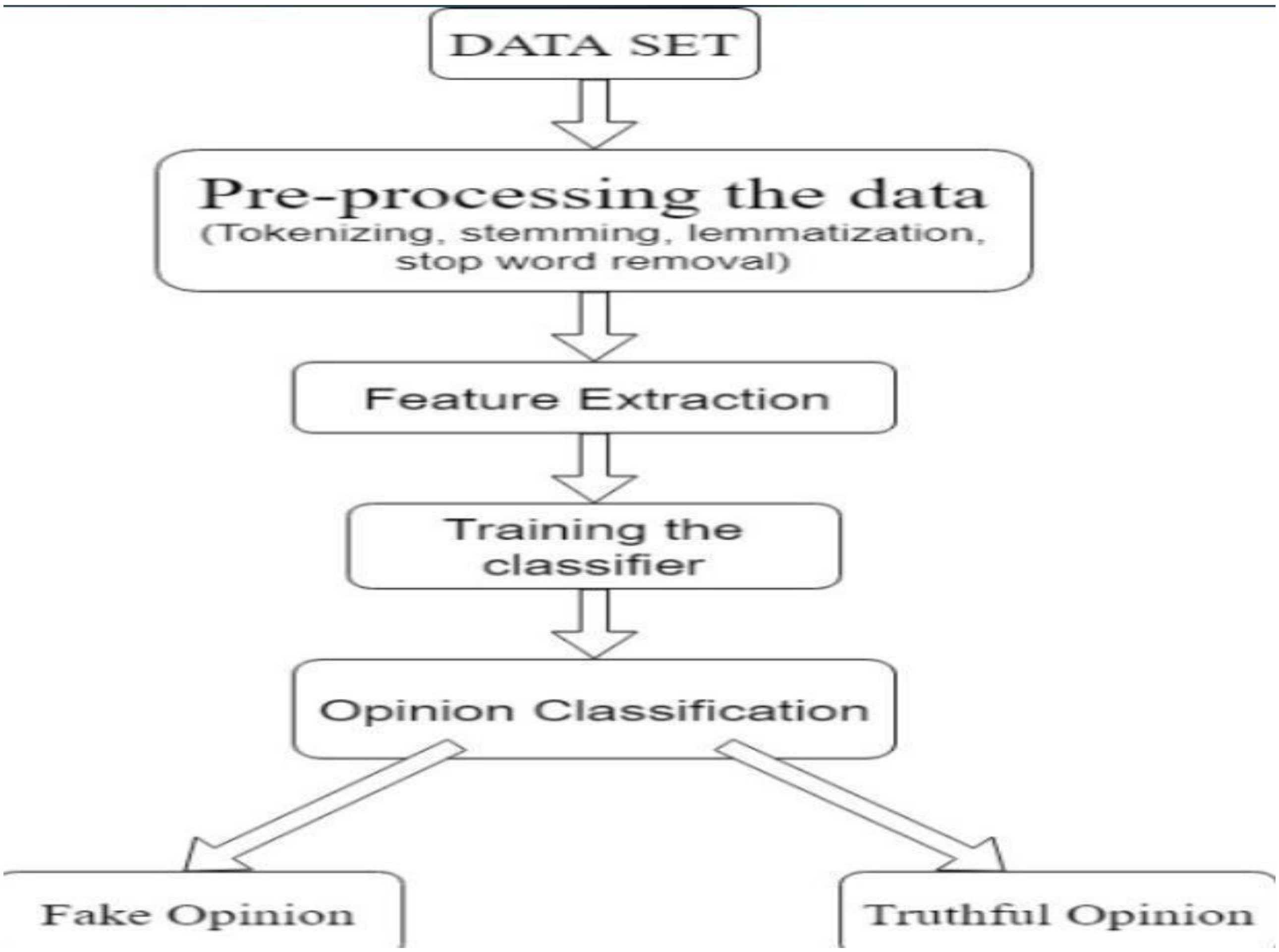
20000
10000
0
-10000
-20000
-30000
Sunday
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Day of week

Chapter -6

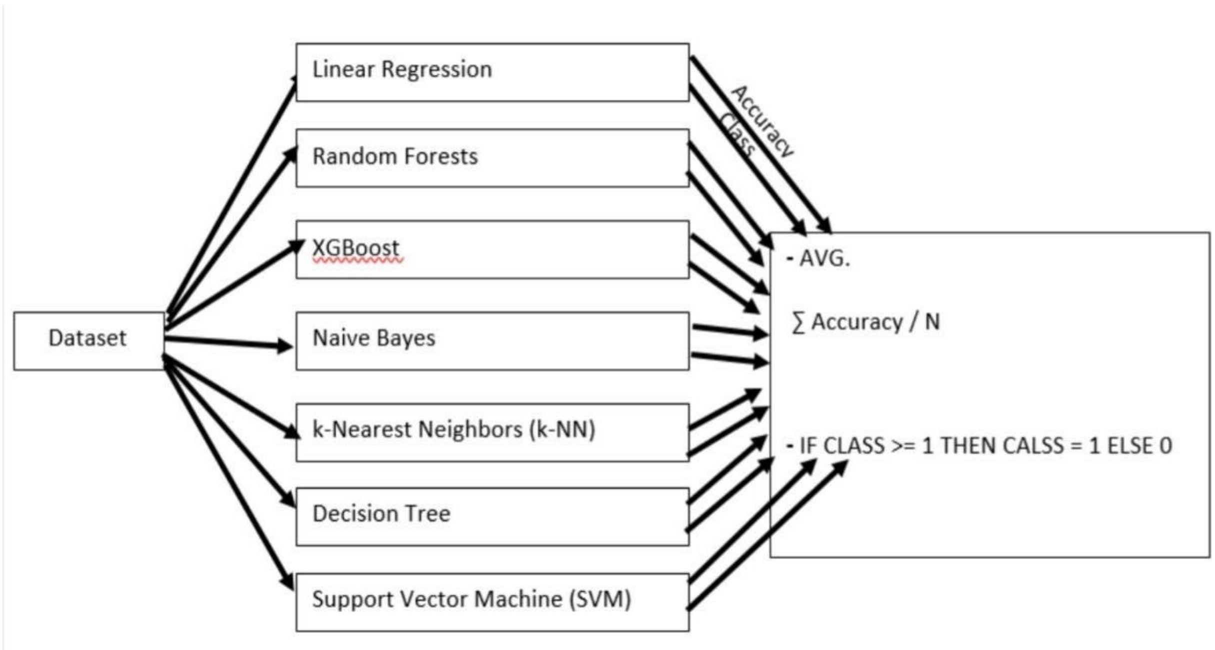
Diagram for proposed System



System Arcitecture Diagram



Algorithms Used



Chapter 7

CONCLUSION

Coronavirus disease (COVID-19) is an infectious disease caused by the SARS-CoV-2 virus. It is spreading continuously among the population of a particular locality. This spread of virus has almost stopped the whole world and is also responsible for decrease in economy of our country and many other countries of the world. Lockdown were imposed to contain the spread of this virus. There were a lot of cases increasing on daily basis and fatality rate also increased. The government had taken various measures to protect our population from this pandemic. This has not only affected the whole country economically, but has also caused financial as well as health problems on the individual level. Many lost their jobs and a lot of people have faced losses in business. The overall point is that due to this Corona virus pandemic, people have lost many things. Its affect reduces for sometimes but it again comes back in the form of wave and lefts people losing lives, jobs, business etc. Everyone wants to know the future that when will this pandemic end, when will be the peak phase to get prepared for the next waves to come.

REFERENCES

A. Douglas, “News consumption and the new electronic media,” The International Journal of Press/Politics, vol. 11, no. 1, pp. 29–52, 2006.

View at: Publisher Site | Google Scholar

J. Wong, “Almost all the traffic to fake news sites is from facebook, new data show,” 2016.

View at: Google Scholar

D. M. J. Lazer, M. A. Baum, Y. Benkler et al., “The science of fake news,” Science, vol. 359, no. 6380, pp. 1094–1096, 2018.

View at: Publisher Site | Google Scholar

S. A. García, G. G. García, M. S. Prieto, A. J. M. Guerrero, and C. R. Jiménez, “The impact of term fake news on the scientific community scientific performance and mapping in web of science,” Social Sciences, vol. 9, no. 5, 2020.

View at: Google ScholarReferences

A. Douglas, “News consumption and the new electronic media,” The International Journal of Press/Politics, vol. 11, no. 1, pp. 29–52, 2006.

View at: Publisher Site | Google Scholar

J. Wong, “Almost all the traffic to fake news sites is from facebook, new data show,” 2016.

View at: Google Scholar

D. M. J. Lazer, M. A. Baum, Y. Benkler et al., “The science of fake news,” Science, vol. 359, no. 6380, pp. 1094–1096, 2018.

View at: Publisher Site | Google Scholar

S. A. García, G. G. García, M. S. Prieto, A. J. M. Guerrero, and C. R. Jiménez, “The impact of term fake news on the scientific community scientific performance and mapping in web of science,” Social Sciences, vol. 9, no. 5, 2020.

View at: Google Scholar