COVID-19 PREDICTOR

By

M N M SASIDHAR **18BEC1198**
ADITYA SIRANGU **18BEC1252**

A project report submitted to

Dr SATHIYA NARAYANAN S

SCHOOL OF ELECTRONICS ENGINEERING

in partial fulfilment of the requirements for the course of

CSE3505 – FOUNDATION OF DATA ANALYTICS

in

B.TECH - ELECTRONICS AND COMMUNICATION ENGINEERING

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

Vandalur – Kelambakkam Road

Chennai – 600127

NOVEMBER 2020

BONAFIDE CERTIFICATE

Certified that this project report entitled "**COVID-19 PREDICTOR"** is a bonafide work of **M N M SASIDHAR (18BEC1198) and ADITYA SIRANGU (18BEC1252)** who carried out the project work under my supervision and guidance for **CSE3505 – FOUNDATION OF DATA ANALYTICS.**

Dr SATHIYA NARAYANAN S

Assistant Professor (Senior Grade 1)

School of Electronics Engineering (SENSE),

Vellore Institute of Technology (VIT Chennai)

Chennai – 600 127.

ABSTRACT

The outbreak of COVID-19 Coronavirus has created a disastrous situation throughout the world. The collective prevalence of COVID-19 is swiftly increasing day by day [6]. Machine Learning can be used very efficiently to forecast progression of the epidemic and design schemes to contain its spread. This project applies regression models to look at and forecast the spread of the virus and increase in the deaths. A Machine Learning based regression model is built to forecast the future. In this project, we have used Data-driven estimation methods like curve fitting for prediction of the number of COVID-19 cases all over the world 30 days ahead. The prediction of the two parameters (number of positive cases, number of deaths) found by the proposed model is accurate and the relevant analysis helps relevant countries, administrators and health officials to make appropriate decisions. We show that exponential fit can be obtained to develop a prediction framework.

ACKNOWLEDGEMENT

We wish to express our sincere thanks and gratitude to our project guide and mentor, Dr. SATHIYA NARAYANAN S. Assistant Professor (Senior Grade-1) School of Electronics Engineering, for his consistent encouragement and valuable guidance offered to us in a pleasant manner throughout the course of the project work.

We thank our parents for supporting us throughout the course of our project.

We thank Dr. JEETASHREE APARAJEETA for giving us good information and a clear concept in Machine Learning which helped us in doing this project.

M N M SASIDHAR **ADITYA SIRANGU**

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND AND MOTIVATION

The Coronavirus disease (COVID-19) is increasing swiftly all over the world. The virus is a highly transmittable and pathogenic viral infection caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2), which emerged in Wuhan, China and spread around the world [6]. The collective frequency of this virus is increasing and has affected 210 countries and territories with USA, India, Russia, Brazil, and France being the most affected**.** Figure 1 describes the origin and the transmission of the virus.

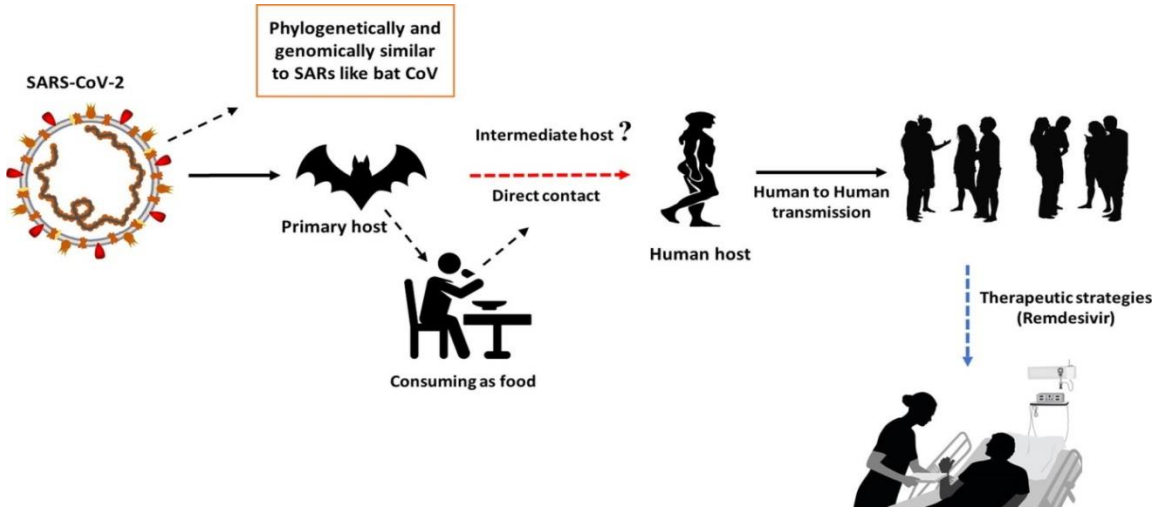

Figure 1: Transmission of COVID 19 [6]

Since the lack of a therapeutic drug, we need to follow the SMS (Sanitizing the surroundings, wearing Masks, maintaining SOCIAL DISTANCE) to control the continuous spread of the virus. This behavior of CoV-2 requires developing strong mathematical models for tracing its spread and forecast the future [6].

1.2 PROBLEM STATEMENT AND OBJECTIVES

PROBLEM STATEMENT

The problem mentioned earlier can be divided into two parts:

1. To build a model that will predict the number of covid cases on a particular day for the next 30 days.
2. To build a model that will predict the number of deaths for the next 30 days.

OBJECTIVES:

There is a need for pioneering results to develop, and analyze the big data on the increasing system of infected people and their data. We developed a GUI which uses Machine Learning to predict where and when, the disease is likely to spread, and alert those countries to keep up with the requirements.

1.3 DATASET

- The dataset has been taken from an online website which collects the data from the World Health Organization and updates it daily.
- https://opendata.ecdc.europa.eu/covid19/casedistribution/csv
- The main advantage of this dataset is that the dataset gets updated daily which makes implementation easier as there is no requirement to download the dataset every time we work on it.

CHAPTER 2

2.1 PROPOSED/IMPLEMENT METHOD

ML MODEL:

As per experimental assessments and previous datasets on SARS-CoV-1 virus pandemic, many sources have revealed that data corresponding to new cases with time may or may not follow a standard distribution like Gaussian or Exponential [2]. The regression curves are drawn by comparing the r2_score and the rmse values of the Linear fit, Polynomial fit and the Exponential fit.

The given functions are used to fit the data:

- $f(x) = ax+b$ [1]
- $f(x) = ax^4+bx^3+cx^2+dx+e$ [1]
- $f(x) = \exp(ax)+b$ [1]

Here, f(x) denotes the number of cases with x, where $x > 0$ is the time in number of days from the first case, and a, b, c, d and e are parameters of each model dependently [2]. Now, we can find the appropriate values of the parameters and to minimize the error between the predicted cases (f(x)) and the actual cases (ý). This can be done using the popular Machine Learning techniques of curve fitting.

Once we fit the curves with the dataset chosen we calculate the metrics namely, RMSE (Root Mean Square Error), r2_score and Mean absolute error for all the fits and select the best fit curve [1].

Using the best fit curve we predict the number of cases/deaths on a given particular day.

We have developed our own algorithm for the exponential fit.

CONTRIBUTION TOWARDS ALGORITHM DEPLOYED:

- We consider the following exponential model:

$$y = ae^{bx} \quad \text{(eq-1)}$$

- Applying natural log on both LHS and RHS of the equation, we have the following corresponding equation:

$$\ln y = bx + \ln a \quad \text{(eq-2)}$$

- The obtained equation (2) is in the form of a linear regression model:

$$y' = a' + bx \quad \text{(eq-3)}$$

- A model of the form of (2) is referred to as a **log-level regression** model. Hence this model can be stated as an exponential regression model of form $y = ae^{bx}$ by setting $a = e^{a'}$.
- Now, we perform the linear regression fit on the model and find out the coefficients a' and b.
- We know that $a = e^{a'}$. Using the linear fit we find out the parameters of the exponential model.
- We now fit the training data to the exponential model to predict the output of the training data.

2.2 ADVANTAGES

- The main advantage of this GUI is that we need not use new dataset every time we start our GUI.
- The main aim of our project is to predict the positive cases or the deaths in a country of a given date.
- Hence the user has the flexibility of selecting the country, predictor (Positive cases or Deaths) and the date.
- The developed GUI displays the predicted number of Positive cases/Deaths on the selected date.
- The prediction accuracy which is used to select the best fit curve is also displayed.
- The trend which is being observed by the selected country is displayed with a graph justifying it, which is helpful for the governments to take precautionary measurements and act accordingly.
- After the execution the user can easily restart the GUI or exit, this helps the user to make easy predictions continuously.

2.3 CHALLENGES FACED

- The dataset we have taken collected the data since January 1st, 2020 and during this time most of the countries haven't registered a single case.

- Since there were no cases registered and the dataset starts form 1st of January there were many zeros in both positive cases column and deaths column.

- Hence, we faced difficulty in training the exponential regression, as ln(0) is undefined and there were many zeros in the dataset.

- To train the dataset for exponential regression we had to either replace all zeroes with some value or drop all the rows containing the value zero.

- From the columns Positive and Deaths, dropping all the rows having zero cases will cause a loss in data and output gets tampered due to insufficiency in data.

- Hence, to overcome this difficulty, firstly we considered the starting value of our dataset, when the country has registered it first covid case.
- This helps us in reducing the zeroes by a huge count which does not affect the output.

- Even after doing the above process, zeroes still exist, hence we ran a loop for ((length of column/2) : (length of column)) if the length is "even" and for((length of column+1/2) : (length of column)) if length is "odd". and replaced all zeroes with value "1".

- Now we ran the loop for the other half of the dataset and dropped remaining rows containing the zero.

- All the above process has been carried down during the data-preprocessing and the final dataset is given as input to all the regression models.

2.4 CODE

```
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import preprocessing
import tkinter as tk
from tkinter import ttk
from tkinter import *
from tkinter import simpledialog
from PIL import Image, ImageTk


root = Tk()
root.title("DATE SELECTOR")
root.geometry("5000x2000")
root.configure(background="black")


class Example(Frame):
    def __init__(self, master, *pargs):
        Frame.__init__(self, master, *pargs)
        self.image = Image.open("pred.png")
        self.img_copy= self.image.copy()
```

```
        self.background_image = ImageTk.PhotoImage(self.image)

        self.background = Label(self, image=self.background_image)
        self.background.pack(fill=BOTH, expand=YES)
        self.background.bind('<Configure>', self._resize_image)

    def _resize_image(self,event):

        new_width = event.width
        new_height = event.height

        self.image = self.img_copy.resize((new_width, new_height))

        self.background_image = ImageTk.PhotoImage(self.image)
        self.background.configure(image =  self.background_image)



e = Example(root)
e.pack(fill=BOTH, expand=YES)

def chosingNumbers():
    global string
    string = mynumber.get()

label = tk.Label(root, text = "Choose A Country" , bg ="grey4" , fg = "white")
label.place(relx = 0.45,rely = 0.75)
```

```
mynumber = tk.StringVar()

combobox = ttk.Combobox(root, width = 15 , textvariable = mynumber)

combobox['values'] = ("India","Australia","Canada", "China", "Argentina","Brazil", "Egypt",
"France", "Indonesia", "Italy", "Nigeria", "Philippines", "Poland", "Russia", "Saudi_Arabia",
"South_Africa", "Spain", "United_States_of_America")

combobox.place(relx=0.53, rely=0.75)


def chosingNumbers1(event):

    global string1

    string1 = mynumber1.get()


label7 = tk.Label(root, text = "SELECT THE PREDICTOR" , bg ="grey4" , fg = "white")

label7.place(relx = 0.42,rely = 0.70)


mynumber1 = tk.StringVar()

combobox1 = ttk.Combobox(root, width = 15 , textvariable = mynumber1)

combobox1['values'] = ("POSITIVE","DEATHS")

combobox1.bind("<<ComboboxSelected>>",chosingNumbers1)

combobox1.place(relx=0.53, rely=0.70)


img = ImageTk.PhotoImage(Image.open("pred.png"))

button = Button(root, text = "Click Here To Confirm", command = chosingNumbers,bg='midnight
blue',fg = 'White')

button.place(relx=0.45, rely=0.80)

def input():

    global answer

    answer = simpledialog.askstring("PLEASE ENTER THE DATE", "FORMAT : DD/MM/YYYY",

                                    parent=root)
```

```
button1 = Button(root,text = "CLICK HERE TO PREDICT THE NUMBER OF COVID CASES"
,command=input,bg='cornflower blue')

button1.place(relx=0.4, rely=0.85)

label2 = tk.Label(root, text = "PLEASE QUIT AFTER ENETRING DETAILS",bg = "yellow2")

label2.place(relx=0.43, rely=0.90)

root.mainloop()

print(string1)

data = pd.read_csv("https://opendata.ecdc.europa.eu/covid19/casedistribution/csv" )

print(data)

CNTY = ['Afghanistan', 'Albania', 'Algeria', 'Andorra','Angola','Anguilla','Antigua_and_Barbuda',
'Armenia' 'Aruba', 'Austria','Azerbaijan', 'Bahamas', 'Bahrain','Bangladesh', 'Barbados',
'Belarus','Belgium', 'Belize', 'Benin', 'Bermuda','Bhutan', 'Bolivia', 'Bonaire, Saint Eustatius and Saba',
'Bosnia_and_Herzegovina', 'Botswana', 'British_Virgin_Islands', 'Brunei_Darussalam', 'Bulgaria',
'Burkina_Faso', 'Burundi', 'Cambodia', 'Cameroon', 'Cape_Verde',
'Cases_on_an_international_conveyance_Japan', 'Cayman_Islands', 'Central_African_Republic',
'Chad', 'Chile', 'Colombia', 'Comoros', 'Congo', 'Costa_Rica', 'Cote_dIvoire', 'Croatia','Cuba', 'Curaçao',
'Cyprus', 'Czechia', 'Democratic_Republic_of_the_Congo', 'Denmark', 'Djibouti', 'Dominica',
'Dominican_Republic', 'Ecuador', 'El_Salvador', 'Equatorial_Guinea', 'Eritrea', 'Estonia', 'Eswatini',
'Ethiopia', 'Falkland_Islands_(Malvinas)', 'Faroe_Islands', 'Fiji','Finland', 'French_Polynesia', 'Gabon',
'Gambia', 'Georgia', 'Germany', 'Ghana', 'Gibraltar', 'Greece', 'Greenland', 'Grenada', 'Guam',
'Guatemala', 'Guernsey', 'Guinea', 'Guinea_Bissau', 'Guyana', 'Haiti', 'Holy_See', 'Honduras',
'Hungary', 'Iceland', 'Iran',

'Iraq','Ireland', 'Isle_of_Man','Israel', 'Jamaica', 'Japan','Jersey', 'Jordan', 'Kazakhstan', 'Kenya',
'Kosovo', 'Kuwait', 'Kyrgyzstan', 'Laos', 'Latvia', 'Lebanon', 'Lesotho', 'Liberia', 'Libya', 'Liechtenstein',
'Lithuania', 'Luxembourg', 'Madagascar', 'Malawi', 'Malaysia', 'Maldives', 'Mali', 'Malta', 'Mauritania',
'Mauritius', 'Mexico', 'Moldova', 'Monaco', 'Mongolia', 'Montenegro', 'Montserrat', 'Morocco',
'Mozambique',

'Myanmar','Namibia', 'Nepal', 'Netherlands', 'New_Caledonia', 'New_Zealand', 'Nicaragua', 'Niger',
'North_Macedonia', 'Northern_Mariana_Islands', 'Norway', 'Oman', 'Pakistan', 'Palestine',

 'Panama', 'Papua_New_Guinea', 'Paraguay', 'Peru', 'Portugal', 'Puerto_Rico', 'Qatar', 'Romania',
'Rwanda', 'Saint_Kitts_and_Nevis', 'Saint_Lucia', 'Saint_Vincent_and_the_Grenadines', 'San_Marino',
'Sao_Tome_and_Principe', 'Senegal', 'Serbia','Seychelles', 'Sierra_Leone', 'Singapore', 'Sint_Maarten',
'Slovakia', 'Slovenia', 'Somalia', 'South_Korea', 'South_Sudan', 'Sri_Lanka', 'Sudan', 'Suriname',
'Sweden', 'Switzerland', 'Syria', 'Taiwan', 'Tajikistan', 'Thailand','Timor_Leste', 'Togo',
'Trinidad_and_Tobago', 'Tunisia', 'Turkey', 'Turks_and_Caicos_islands', 'Uganda', 'Ukraine',
'United_Arab_Emirates', 'United_Kingdom', 'United_Republic_of_Tanzania',
'United_States_Virgin_Islands', 'Uruguay', 'Uzbekistan', 'Venezuela', 'Vietnam', 'Western_Sahara',
'Yemen', 'Zambia', 'Zimbabwe']

for i in range(len(CNTY)):
```

```
    data.drop(data[data.countriesAndTerritories == CNTY[i]].index,inplace = True)
print(data)
data.reset_index(drop=True)
total_rows=len(data.axes[0])
C = data['countriesAndTerritories'].reset_index(drop=True)
D = data['dateRep'].reset_index(drop=True)
E = data['cases'].reset_index(drop=True)
F = data['deaths'].reset_index(drop=True)
data_new = pd.DataFrame(columns = ["DATE","POSITIVE","DEATHS","COUNTRY"] )
for i in range(total_rows-1,0,-1):
    if(string == C[i]):
        data_new = data_new.append({'DATE' : D[i] ,"POSITIVE" : E[i],"DEATHS" :
F[i],"COUNTRY" : C[i]} , ignore_index=True)
            print(data_new)
data = data_new. dropna()
data = data[data[string1] != 0]
data.head()
days = []
for i in range(1,len(data)+1):
    days.append(i)
data.insert(0, "DAY", days, True)
data[string1] = data[string1].replace(to_replace=0, method='ffill')
print(data)
data[string1] = data[string1].astype(float)
data[string1] = data[string1].replace(to_replace=0, method='ffill')
X = data[string1].values.reshape(-1,1)
logarithm = "log"+string1
data[logarithm] = np.log(X)
data[logarithm] = data[logarithm].fillna(0)
```

```
data[logarithm] = data[logarithm].replace(to_replace=-(np.inf), method='ffill')

data.isnull().values.any()

data.head()

data.to_csv(string+"_cleansed_data_final.csv")

DAT = data['DATE'][data.index[-1]]

from datetime import datetime, timedelta


x = DAT

future_dates = []

for i in range(30):

    future_dates.append((datetime.strptime(x, '%d/%m/%Y') +
timedelta(days=1)).strftime('%d/%m/%Y'))

    x = future_dates[i]

DAT1 = data['DATE']

DAT2 = DAT1.values.tolist()

for i in range(len(future_dates)):

    DAT2.append(future_dates[i])

from pandas import DataFrame

future_dates = DataFrame(DAT2,columns=['ds'])

days = []

for i in range(1,len(future_dates)+1):

    days.append(i)

future_dates.insert(0, "DAY", days, True)

future_dates

a = future_dates['ds'].values.reshape(-1,1)

b = []

import re

for i in range(len(a)):

    b.append(a[i][0])
```

```
    def change_date_format(dt):
        return re.sub(r'(\d{4})-(\d{1,2})-(\d{1,2})', '\\3-\\2-\\1', dt)
    b[i] = change_date_format(b[i])
c = future_dates['DAY'].values.reshape(-1,1)
d = []
for i in range(len(c)):
    d.append(int(c[i][0]))
inp = str(answer)
for i in range(len(b)):
    if(inp == b[i]):
        n = d[i]
        break
X = data['DAY'].values.reshape(-1,1)
y = data[string1].values.reshape(-1,1)
reg = LinearRegression()
reg.fit(X, y)
y_pred = reg.predict(X)
print("The linear model is: Y = {:.5} + {:.5}X".format(reg.intercept_[0], reg.coef_[0][0]))
plt.figure(figsize=(16, 8))
plt.scatter(
    X,
    y,
    c='black',label="DATA"
)
plt.plot(
    X,
    y_pred,
    c='blue',
    linewidth=2,label="OPTIMIZED DATA"
```

```
)
plt.xlabel("DAYS")
plt.ylabel("POSITIVE")
plt.legend()
plt.show()
from sklearn.metrics import r2_score
r2_1 = r2_score(y ,y_pred )
from sklearn.metrics import mean_squared_error
from math import sqrt
rms1 = sqrt(mean_squared_error(y, y_pred))
lin_reg_pred = reg.predict([[n]])
poly = PolynomialFeatures(degree =4)
X_poly = poly.fit_transform(X)

poly.fit(X_poly, y)
lin2 = LinearRegression()
lin2.fit(X_poly, y)
pred = lin2.predict(X_poly)
X, y_pred1 = zip(*sorted(zip(X, pred)))

plt.figure(figsize=(16, 8))
plt.scatter(
    X,
    y,
    c='black',label="DATA"
)
plt.plot(
    X, y_pred1,
    c='blue',label="OPTIMIZED DATA"
```

```
)
plt.xlabel("DAYS")
plt.ylabel("POSITIVE")
plt.legend()
plt.show()
from sklearn.metrics import r2_score
r2_2 = r2_score(y ,y_pred1)
from sklearn.metrics import mean_squared_error
from math import sqrt
rms2 = sqrt(mean_squared_error(y, y_pred))
poly_pred = lin2.predict(poly.fit_transform([[n]]))
poly_pred
import numpy as np


# curve-fit() function imported from scipy
from scipy.optimize import curve_fit


from matplotlib import pyplot as plt
X = data['DAY'].values.reshape(-1,1)
k = data[logarithm].values.reshape(-1,1)
reg = LinearRegression()
reg.fit(X, k)
predictions = reg.predict(X)
print("The linear model to find the coeffecients of the exponential regression is: Y = {:.5} +
{:.5}X".format(reg.intercept_[0], reg.coef_[0][0]))
print("alpha = {:.5}".format(reg.intercept_[0]))
alpha=reg.intercept_[0]
print("beta = {:.5}".format(reg.coef_[0][0]))
beta = reg.coef_[0][0]
```

```
plt.figure(figsize=(16, 8))
plt.scatter(
    X,
    k,
    c='black'
)
plt.plot(
    X,
    predictions,
    c='blue',
    linewidth=2
)
plt.xlabel("DAYS")
plt.ylabel("POSITIVE")
plt.show()
ans=[]
l=np.exp(alpha)
print(l)
k = beta
print(k)
for i in range(len(X)):
    ans.append(l*(np.exp(k*X[i])))
plt.figure(figsize=(16, 8))
plt.plot(X, y, 'o', color ='red', label ="data")
plt.plot(X, ans, '--', color ='blue', label ="optimized data")
plt.legend()
plt.show()
print("EQUATION  = ",l,"exp(",k,")")
expo_pred = l*(np.exp(k*n))
```

```
from sklearn.metrics import r2_score
r2_3 = r2_score(y ,ans)
from sklearn.metrics import mean_squared_error
from math import sqrt
rms3 = sqrt(mean_squared_error(y, ans))
pred = []
pred.append(lin_reg_pred[0][0])
pred.append(poly_pred[0][0])
pred.append(expo_pred)
max1 = max(r2_1,r2_2,r2_3)
pred_cases = 0
r2 = 0
X_PRED = []
Y_PRED = []
if (max1 == r2_1):
  pred_cases = pred[0]
  r2 = r2_1
  reg = "LINEAR"
  X_PRED = X
  Y_PRED = y_pred
  rmse = rms1
elif (max1 == r2_2):
  pred_cases = pred[1]
  r2 = r2_2
  reg = "POLYNOMIAL"
  X_PRED = X
  Y_PRED = y_pred1
  rmse = rms1
else:
```

```
 pred_cases = pred[2]
 r2 = r2_3
 reg = "EXPONENTIAL"
 X_PRED = X
 Y_PRED = ans
 rmse = rms1
import pandas as pd
import math
df = pd.DataFrame({'Country':[string],
                   'Regression Type':[reg],
                   'r2_SCORE':[r2],
                   'rmse':[rmse],
                   'Date':[inp],
                   'Predicted no of cases':[math.ceil(pred_cases)]})
Print(df)
import tkinter as tk
from tkinter import ttk
from tkinter import *
from tkinter import simpledialog
from PIL import Image, ImageTk
import tkinter.font as font
import matplotlib
matplotlib.use('TkAgg')
import numpy as np
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
from matplotlib.figure import Figure
from IPython.display import HTML, Javascript, display
import IPython
```

```
root = Tk()
root.title("FINAL OUTPUT")
root.geometry("5000x3000")
root.configure(background="black")

class Example(Frame):
    def __init__(self, master, *pargs):
        Frame.__init__(self, master, *pargs)




        self.image = Image.open("pred_op1.png")
        self.img_copy= self.image.copy()



        self.background_image = ImageTk.PhotoImage(self.image)


        self.background = Label(self, image=self.background_image)
        self.background.pack(fill=BOTH, expand=YES)
        self.background.bind('<Configure>', self._resize_image)


    def _resize_image(self,event):


        new_width = event.width
        new_height = event.height


        self.image = self.img_copy.resize((new_width, new_height))
```

```
        self.background_image = ImageTk.PhotoImage(self.image)
        self.background.configure(image =  self.background_image)


text1 = "SELECTED COUNTRY :"+string
text2 = "SELECTED DATE :"+inp
text3 = string1+" PREDICTION : "+str(math.ceil(pred_cases))
text4 = "TREND BEING FOLLOWED : "+reg
text5 = "PREDICTION ACCURACY : "+str(round(r2*100,2))+"%"


e = Example(root)
e.pack(fill=BOTH, expand=YES)


myFont = font.Font(family = "Cambria",size=15)
myFont1 = font.Font(size = 20)


img = PhotoImage("pred_op3.png")


w = tk.Label(root, text="WELCOME TO THE COVID-19 PREDICTOR",bg = "grey12",fg =
"white")
w['font'] = myFont1
w.place(relx = 0.32 , rely = 0.01)


w1 = tk.Label(root, text=text1,bg = "sky blue",fg = "black")
w1['font'] = myFont
w1.place(relx = 0.1 , rely = 0.1)


w2 = tk.Label(root, text=text2,bg = "sky blue",fg = "black")
w2['font'] = myFont
w2.place(relx = 0.1 , rely = 0.2)
```

```
w3 = tk.Label(root, text=text3,bg = "sky blue",fg = "black")
w3['font'] = myFont
w3.place(relx = 0.7 , rely = 0.1)

w5 = tk.Label(root, text=text5,bg = "sky blue",fg = "black")
w5['font'] = myFont
w5.place(relx = 0.7 , rely = 0.2)

w4 = tk.Label(root, text=text4,bg = "sky blue",fg = "black")
w4['font'] = myFont
w4.place(relx = 0.7 , rely = 0.3)

def initialize():
    display(HTML(
        '''
            <script>
                code_show = false;

                    IPython.notebook.kernel.restart();

                        IPython.notebook.execute_all_cells();


            </script>
        '''
    ))
```

```
button3 = Button(root, text = "CLICK HERE TO RESTART", command = initialize,bg='sky blue',fg = 'black')

button3['font'] = myFont

button3.place(relx=0.7, rely=0.4)

def close_window():

   root.destroy()

button5 = Button(root)

button5['text'] ="CLICK HERE TO QUIT THE WINDOW."

button5['command'] = close_window

button5['font'] = myFont

button5['bg'] = 'sky blue'

button5.place(relx = 0.7,rely = 0.8)



class mclass:

   def __init__(self,  root):

      self.root = root

      self.button = Button (root, text="DISPLAY GRAPH", command=self.plot,bg = "sky blue",fg = "black")

      self.button['font'] = myFont

      self.button.place(relx = 0.1,rely = 0.3)


   def plot (self):

      x= X_PRED

      v= y

      p= []

      for i in range(len(Y_PRED)):

         p.append(Y_PRED[i][0])
```

```
        q = np.array(p)
        fig = Figure(figsize=(7,5))
        a = fig.add_subplot(111)
        a.scatter(x,v,color='red',label = "DATA")
        a.plot(x, q,color='blue',label  = "PREDICTION CURVE")

        a.set_title (reg, fontsize=16)
        a.set_ylabel("Y", fontsize=14)
        a.set_xlabel("X", fontsize=14)
        a.legend()
        canvas = FigureCanvasTkAgg(fig, master=self.root)
        canvas.get_tk_widget().place(relx = 0.05,rely = 0.4)
        canvas.draw()

start= mclass (root)
root.mainloop()
```

CHAPTER 3

3.1 MAIN RESULTS

ANALYSIS OF DATA:

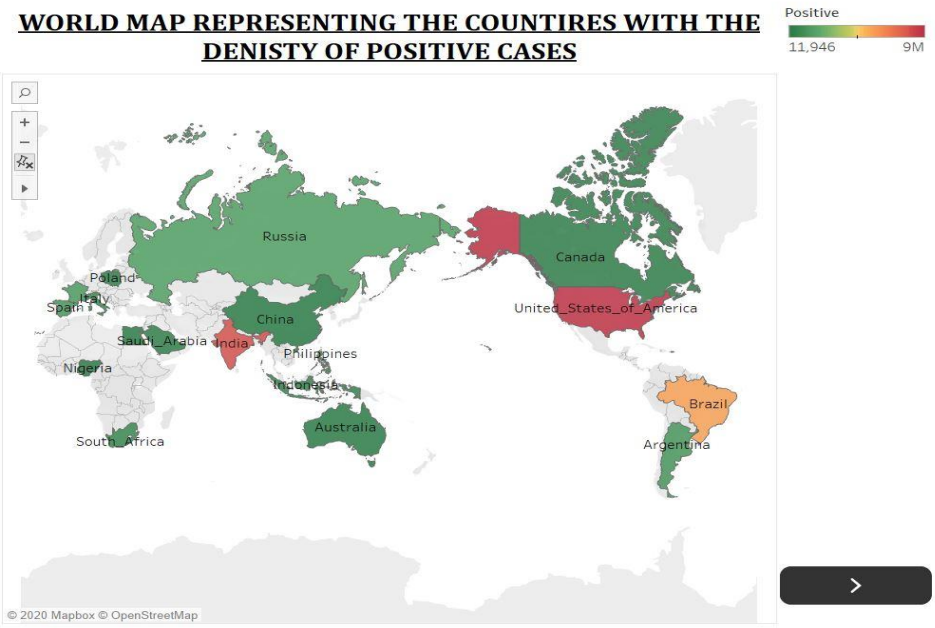

Fig.2: represents the density of Positive cases across the globe for selected countries as on 31th October.

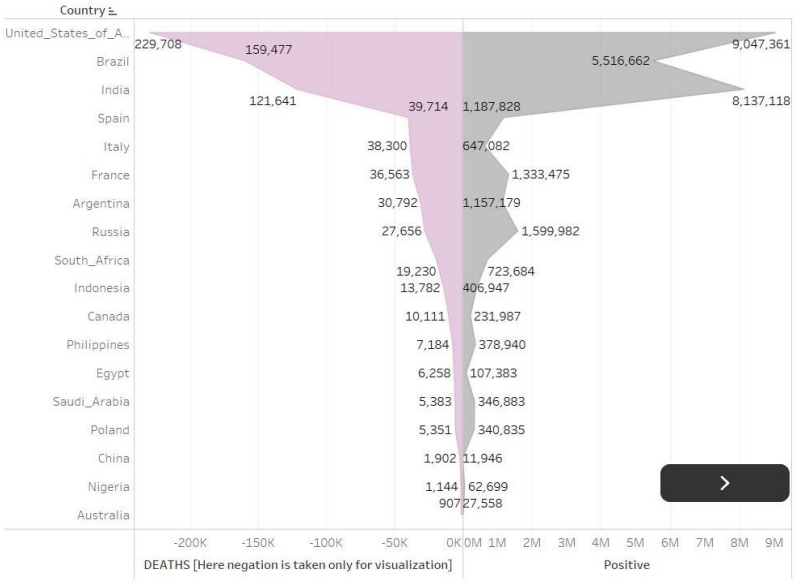

Fig.3 Cumulative representation of Positive cases and Deaths in selected countries

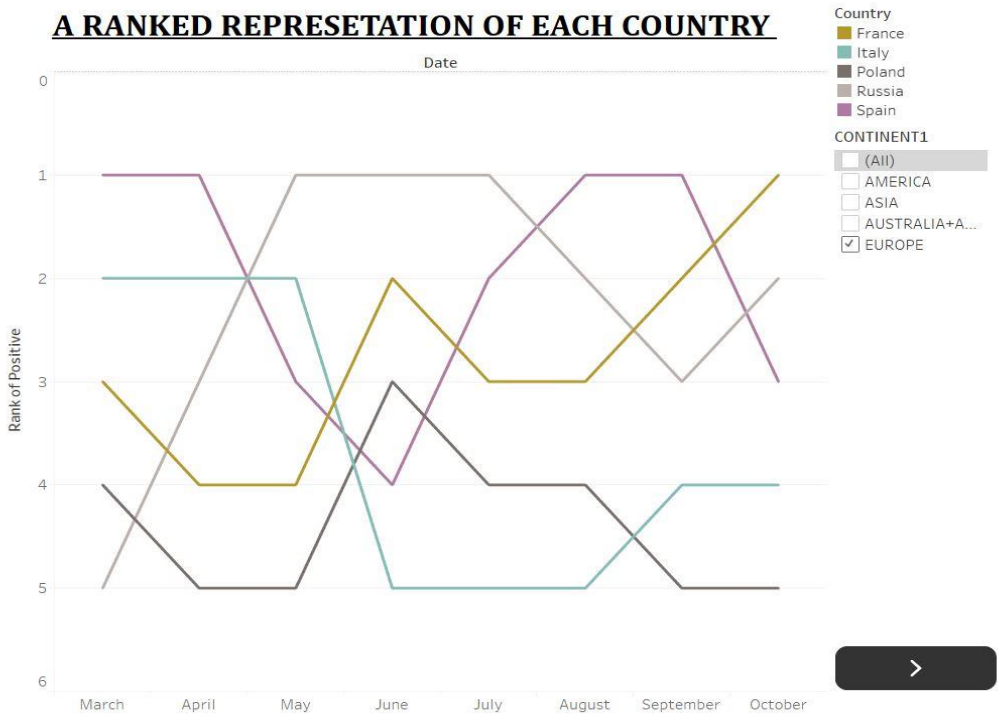

Figure 4: Represents the ranks of different countries in their continents

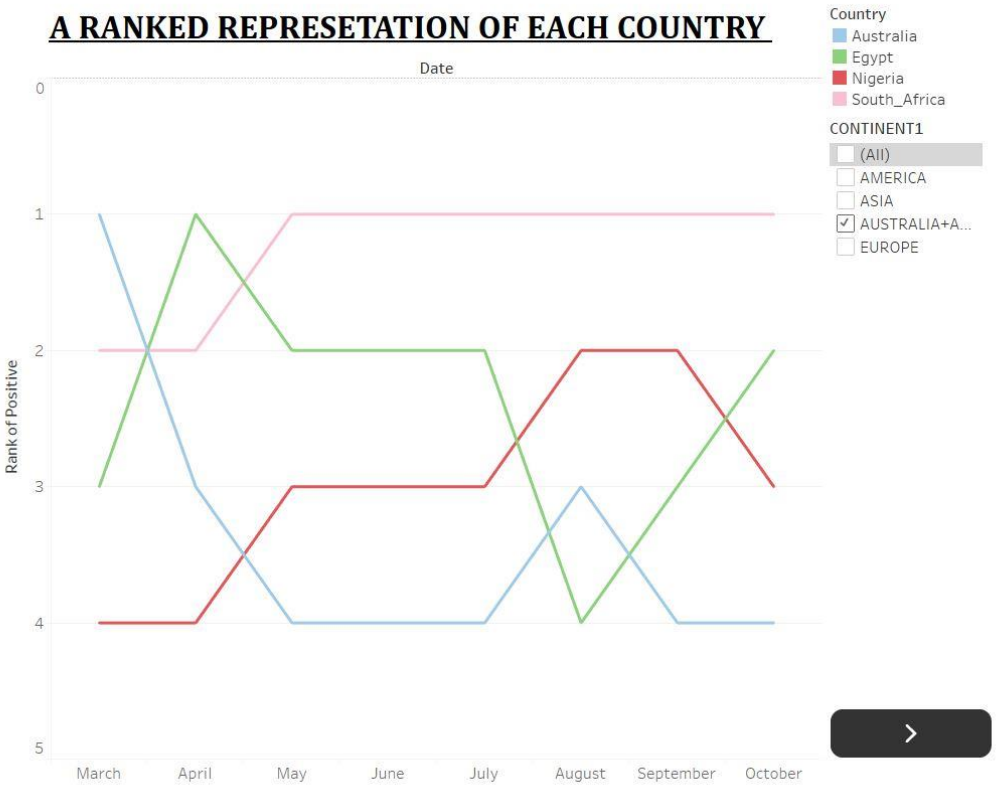

Figure 5: Represents the ranks of different countries in their continents

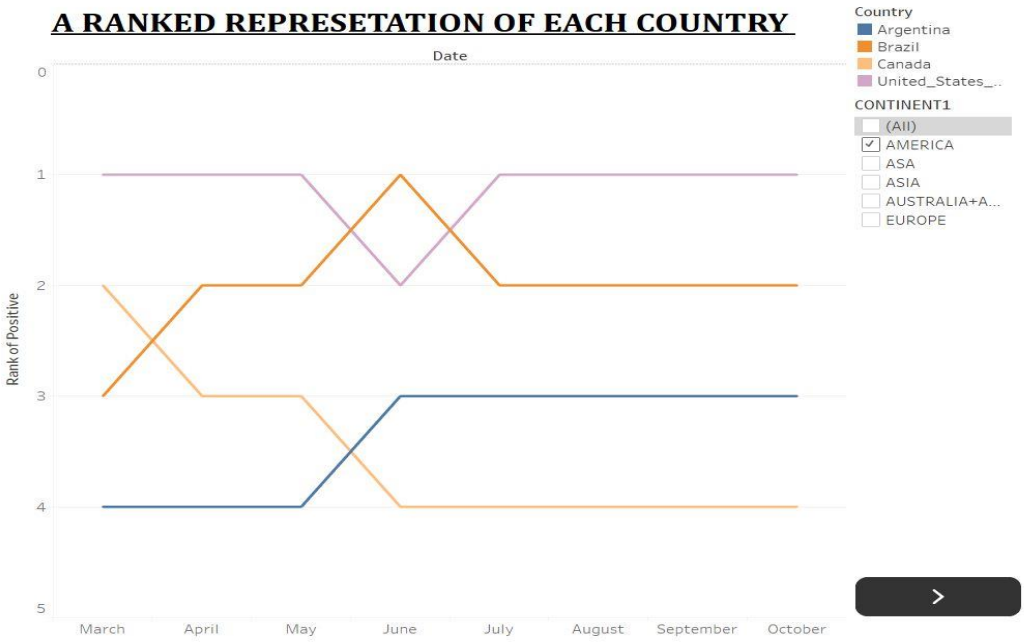

Figure 6: Represents the ranks of different countries in their continents

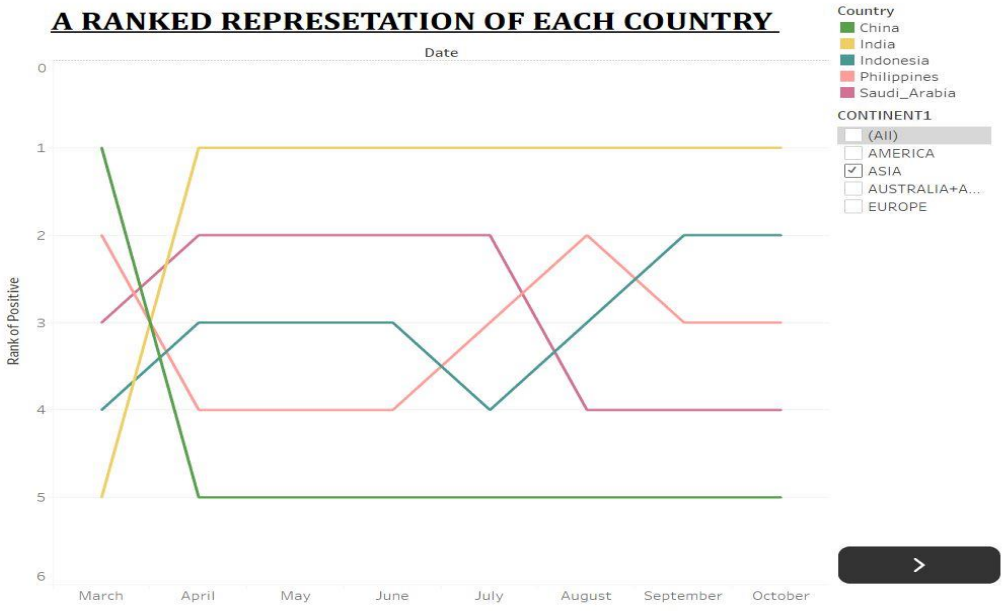

Figure 7: Represents the ranks of different countries in their continents

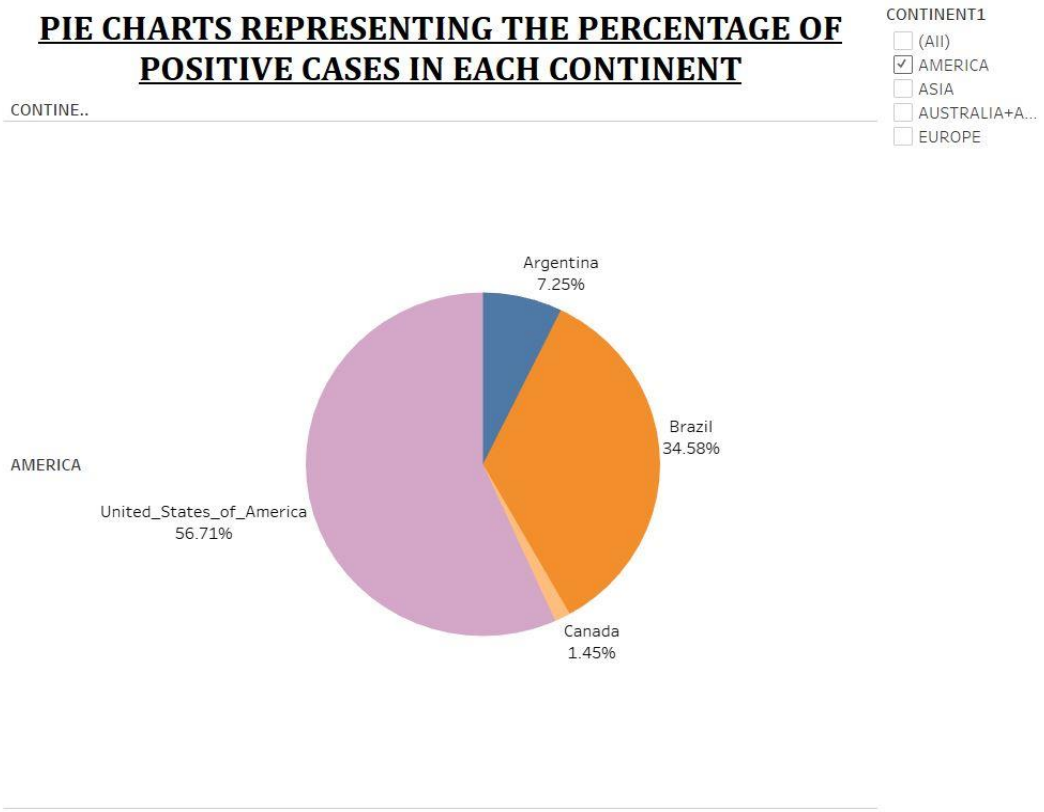

Figure 8:Pie chart representing percentage of positive cases in a country w.r.t the continent

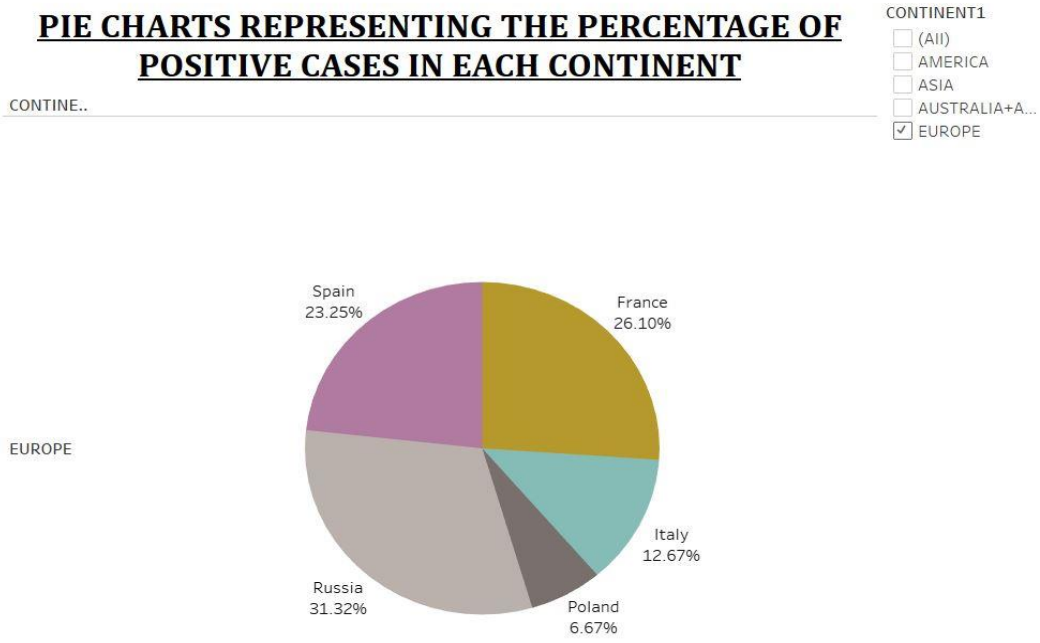

Figure 9: Pie chart representing percentage of positive cases in a country w.r.t the continent

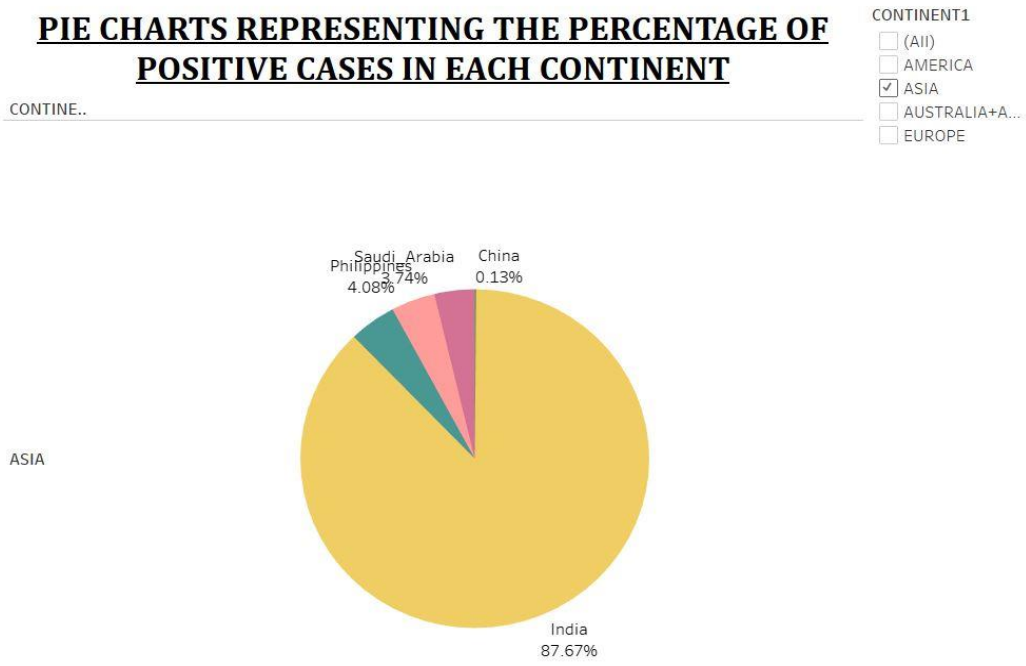

Figure 10: Pie chart representing percentage of positive cases in a country w.r.t the continent

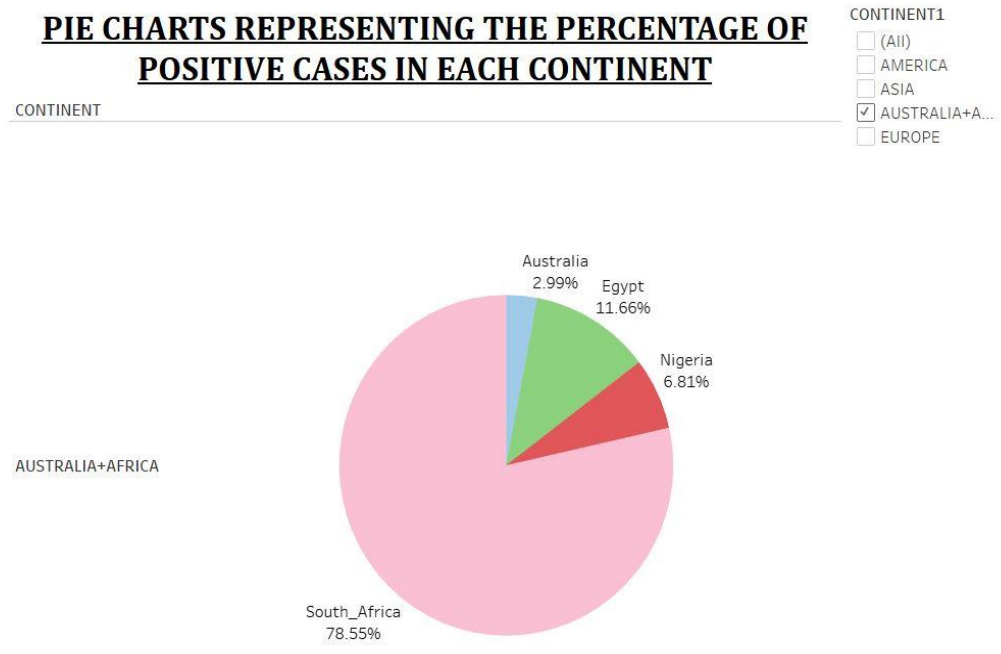

Figure 11: Pie chart representing percentage of positive cases in a country w.r.t the continent

A WORD MAP TO DETERMINE THE NUMBER OF POSITIVE CASES:

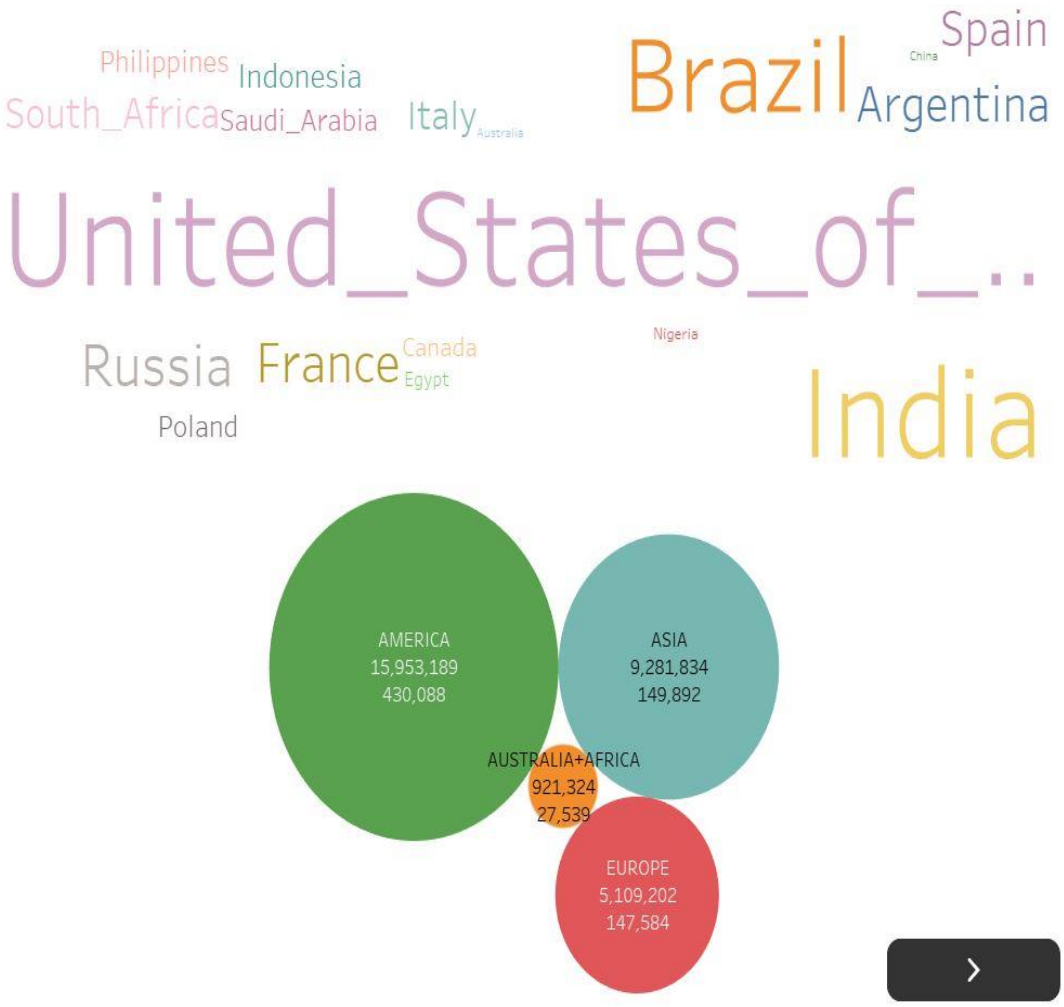

Figure 12: Word map and size map representing the number of cases in countries and continents w.r.t other countries and continents

PREDITOR GUI:

INPUT GUI FOR POSITIVE CASES:

Figure 13: A picture displaying input GUI

LINEAR REGRESSION FIT FOR INDIA ON 01/11/2020:

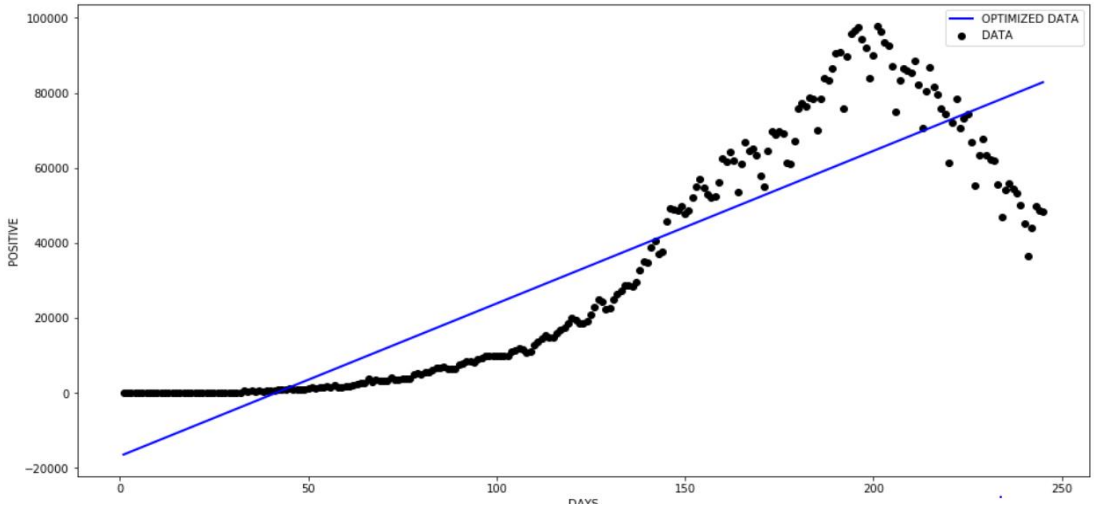

Figure 14: A statistical linear model for India

POLYNOMIAL REGREESION FIT FOR INDIA ON 01/11/2020:

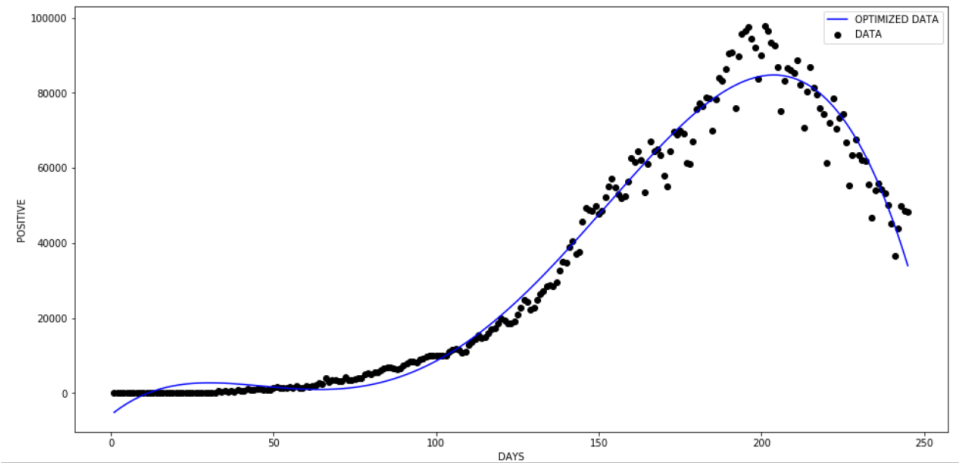

Figure 15: A statistical polynomial model for India

EXPONENTIAL REGRESSION FOR INDIA ON 01/11/2020:

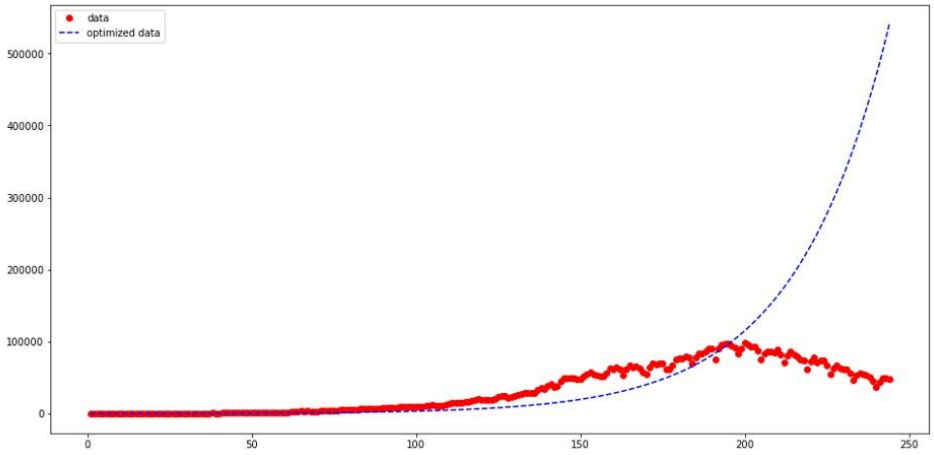

Figure 16: A statistical exponential model for India

FINAL GUI OUTPUT FOR POSITIVE CASES:

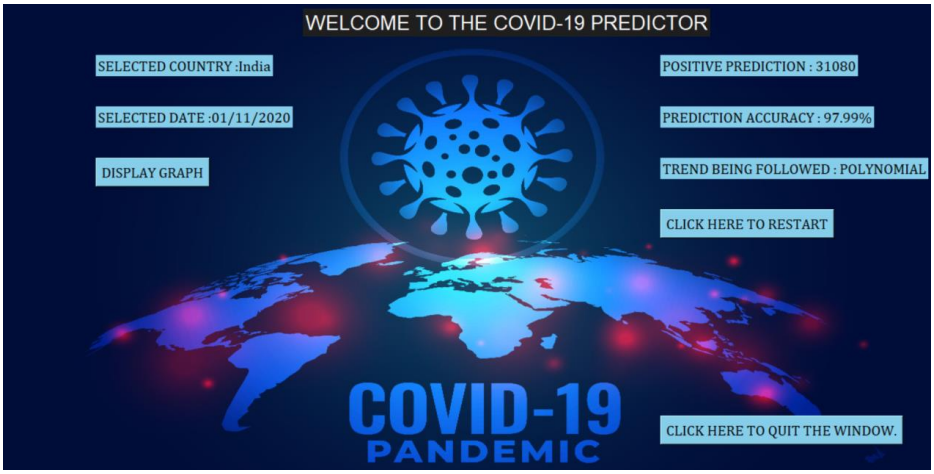

Figure 17: A picture of output GUI

TABLE REPRESENTING THE COMPARISION OF METRICS:

COUNTRY	RMSE			R2_SCORE			MEAN_ABSOLUTE _ERROR		
	LINEAR	POLYNOMIAL	EXPONENTIAL	LINEAR	POLYNOMIAL	EXPONENTIAL	LINEAR	POLYNOMIAL	EXPONENTIAL
INDIA	12271	4262	99393	0.85	0.98	0.81	10520	3090	40220
FRANCE	2921	1597	2611	0.38	0.82	0.38	2353	1056	1704
RUSSIA	2786	1159	5303	0.16	0.85	0.32	2356	829	4363
U.S.A	12777	10673	66089	0.57	0.71	0.44	10639	8739	39219
ITALY	1429	730	1639	0.11	0.77	0.15	1184	555	1109

Footer Area 7

Table 1: Preliminary Evaluation of different models.

TABLE SHOWING PREDICTED AND ACTUAL VALUES

	DATE : 08/10/2020			
COUNTRY	POSITIVE		DEATHS	
	PREDICTED	ACTUAL	PREDICTED	ACTUAL
INDIA	76373	75809	994	971
FRANCE	14348	18129	51	76
RUSSIA	11569	11493	186	191
U.S.A	34359	56137	458	928
ITALY	1868	4458	20	22

October 31, 2020 Footer Area 8

Table 2: Comparison between total predicted cases and actual cases

3.2 INFERENCES

- Figure 2 represents a heat map of the world

- Figure 3 represent the cumulative positive cases and deaths of all selected countries.

- Figure 4,5,6 and 7 give us the rank of each country in its continent based on the number of positive cases

- Figure 8,9,10 and 11 display a pie chart showing the percentage of cases in a country with respect to its own continent

- Figure 12 represents a word map and a size chart :
  In the word map, the size of the country name represents the density of positive cases in that country.

  In the size chart we can observe that the size of the circle represents the total number of cases in that continent that has been described in the picture.

- Figure 14 represents the linear regression fit, we can clearly see that the residuals are not random and following a certain pattern, hence we can infer that linear regression is not an ideal fit for predicting the covid cases.

- Figure 15 represents the polynomial regression fit; we have fixed the degree as "4" for the regression model. If we increase the degree of the model the training set accuracy increases but the test set decreases drastically.

- Figure 16 represents the exponential fit, we have observed that at the mid covid times (July-august) India as show the signs of following the exponential regression but along the course of time it is following more of polynomial regression than exponential regression.

- Table-1 shows the comparison of the metrics that have been considered for the evaluation of the best fit curve

- The Table-2 shows the model forecast of the spread of the COVID-19 for every major country for which sufficient data was available and model fits had R2_score>0.5 using the proposed model.

CHAPTER 4

CONCLUSION AND RECOMMENDATION FOR FUTURE WORK

4.1 CONCLUSION

This study has discussed how improved regression modeling and Machine Learning can help to forecast the progress of the epidemic proactively. Using the proposed Polynomial model based on regression, we show that the developed model is able to make statistically better predictions as compared to the baseline models like Gaussian and Exponential.

The GUI developed can be efficiently used by the government to forecast the upcoming cases and take up necessary precautionary measures to control the spread of the virus.

4.2 RECOMMNDATIONS FOR FUTURE WORK

- There is good scope for the project to turn it up into an app which makes it easier for the user to have this project in mobile phone and predictions become very easy.

- Some important parameters like density of population at a particular region, age distribution of infected people, transportation of migrant workers, availability of health care facilities, etc., can be included as factors effecting the number of cases or deaths to make predictions.

- Models like Weibull fit function can be used for further time series examination and forecasts [4].

- The prediction of the contribution of numerous socio-economic variables that define the weakness, dispersion and evolution of the epidemic can be done by developing appropriate algorithms [4].

REFERENCES

[1] Machine Learning An Algorithmic Perspective 2nd Edition by STEPHEN MARSLAND

[2] Introduction to Machine Learning, Second Edition (Adaptive Computation and Machine Learning) by ALPAYDIN ETHEM

[3] KAI LIU, YING CHEN, RUZHENG LIN, AND KUNYUAN HAN. Clinical features of covid-19 in elderly patients: A comparison with young and middle-aged patients. Journal of Infection, 2020.

[4] MARY RAYGOZA. Covid-19, exponential growth, and the power of showing up in social solidarity: The math behind the virus. 2020.

[5] Mathematics for machine learning. Cambridge: Cambridge University Press, 2019 by DEISENROTH, MARC PETER, A. ALDO FAISAL, AND CHENG SOON ONG.

[6] COVID-19 infection: Origin, transmission, and characteristics of human coronaviruses by MUHAMMAD ADNANSHEREEN, SULIMANKHAN, ABEERKAZMI, NADIABASHIR, RABEEASIDDIQUE.