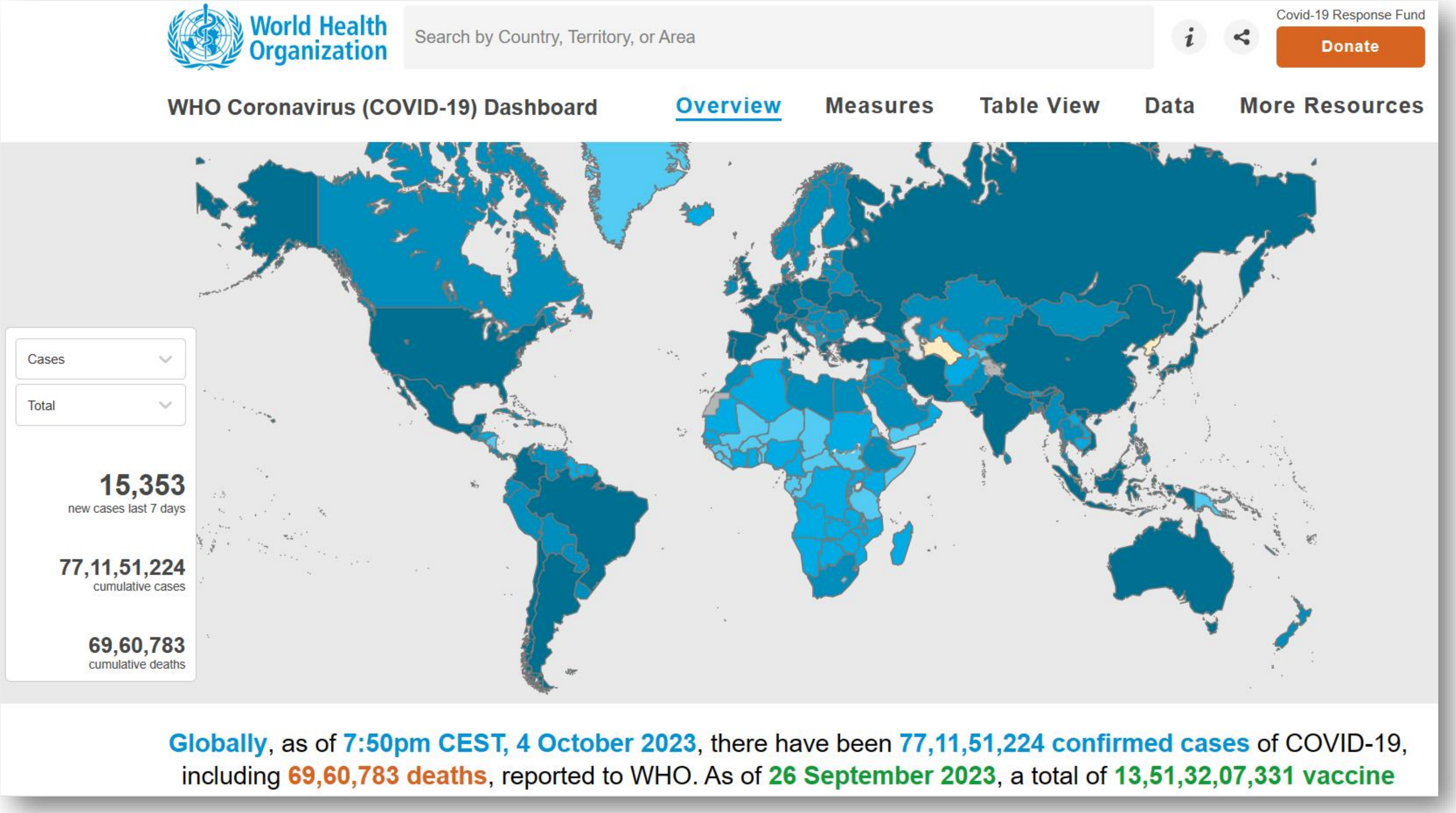
Phase-4

COVID – 19 CASES ANALYSIS

PROBLEM STATEMENT:

The project aim is to continue building the project by performing different activities like feature engineering, model training, evaluation etc as per the instructions in the project.



WHO Coronavirus (COVID-19) Dashboard

Cases

Total

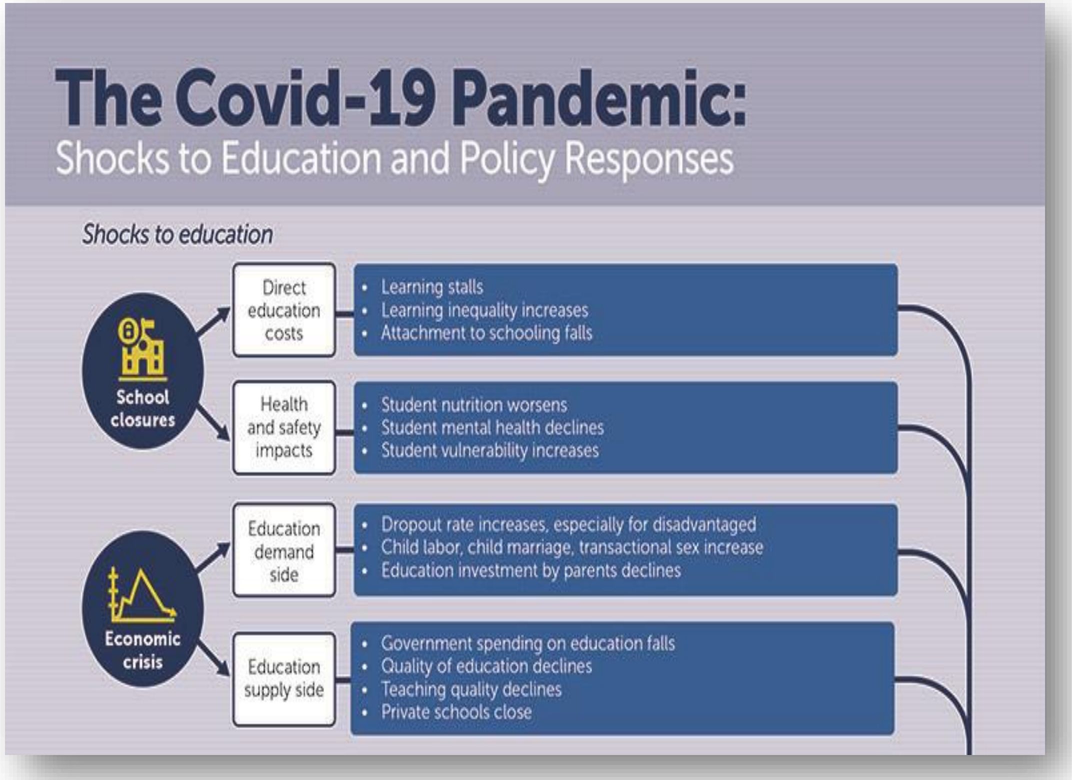
15,353
new cases last 7 days

77,11,51,224
cumulative cases

69,60,783
cumulative deaths

Globally, as of **7:50pm CEST, 4 October 2023**, there have been **77,11,51,224 confirmed cases** of COVID-19, including **69,60,783 deaths**, reported to WHO. As of **26 September 2023**, a total of **13,51,32,07,331 vaccine**

PROJECT OVERVIEW:



The Covid-19 Pandemic:
Shocks to Education and Policy Responses

Shocks to education

- Learning stalls
- Learning inequality increases
- Attachment to schooling falls

- Student nutrition worsens
- Student mental health declines
- Student vulnerability increases

- Dropout rate increases, especially for disadvantaged
- Child labor, child marriage, transactional sex increase
- Education investment by parents declines

- Government spending on education falls
- Quality of education declines
- Teaching quality declines
- Private schools close

This project involve a combination of data modeling, report building, and potentially custom scripting depending on your specific requirements. Below is a simplified example of the process. Note that this is a high-level overview, and actual code would depend on your data source and the specific analysis you wish to perform. Also, IBM Cognos uses its own scripting language called "Cognos Expressions" for report development.

DATA ANALTICS OUTLINE:

Data Connection Setup:

- In Cognos, we can set up a data connection to your COVID-19 dataset. We would define data source details such as the database or file location, connection credentials, and any required SQL queries to retrieve the data.

Data Modeling:

- In IBM Cognos, we can use IBM Framework Manager to create a data model. This involves defining the structure of our data, such as tables, relationships, calculations, and business rules. This is where we specify how COVID-19 data relates to other data sources if necessary.

Report Development:

- Cognos allows you to design reports and dashboards using a drag-and-drop interface. We can create various types of visualizations like tables, charts, and maps to present COVID-19 data. The code would mostly involve creating and formatting these reports.

Here's a simple example of how we might create a report in Cognos to display daily new cases:

- Create a new report

- Drag the "Date" dimension to the Rows section

- Drag "New Cases" measure to the Columns section

- Apply necessary formatting and calculations

- Add titles, labels, and a date range filter

Data Analysis:

- Cognos provides various functions and expressions to perform calculations and aggregations on our data. For instance, we can calculate rolling averages, growth rates, or percentage changes in cases. The code for data analysis will depend on the specific calculations we want to perform.

Geospatial Analysis:

- If we wish to include geographic analysis, Cognos can handle geospatial data. We might write code to create a map visualization showing COVID-19 hotspots or trends in different regions.

Automation and Scheduling:

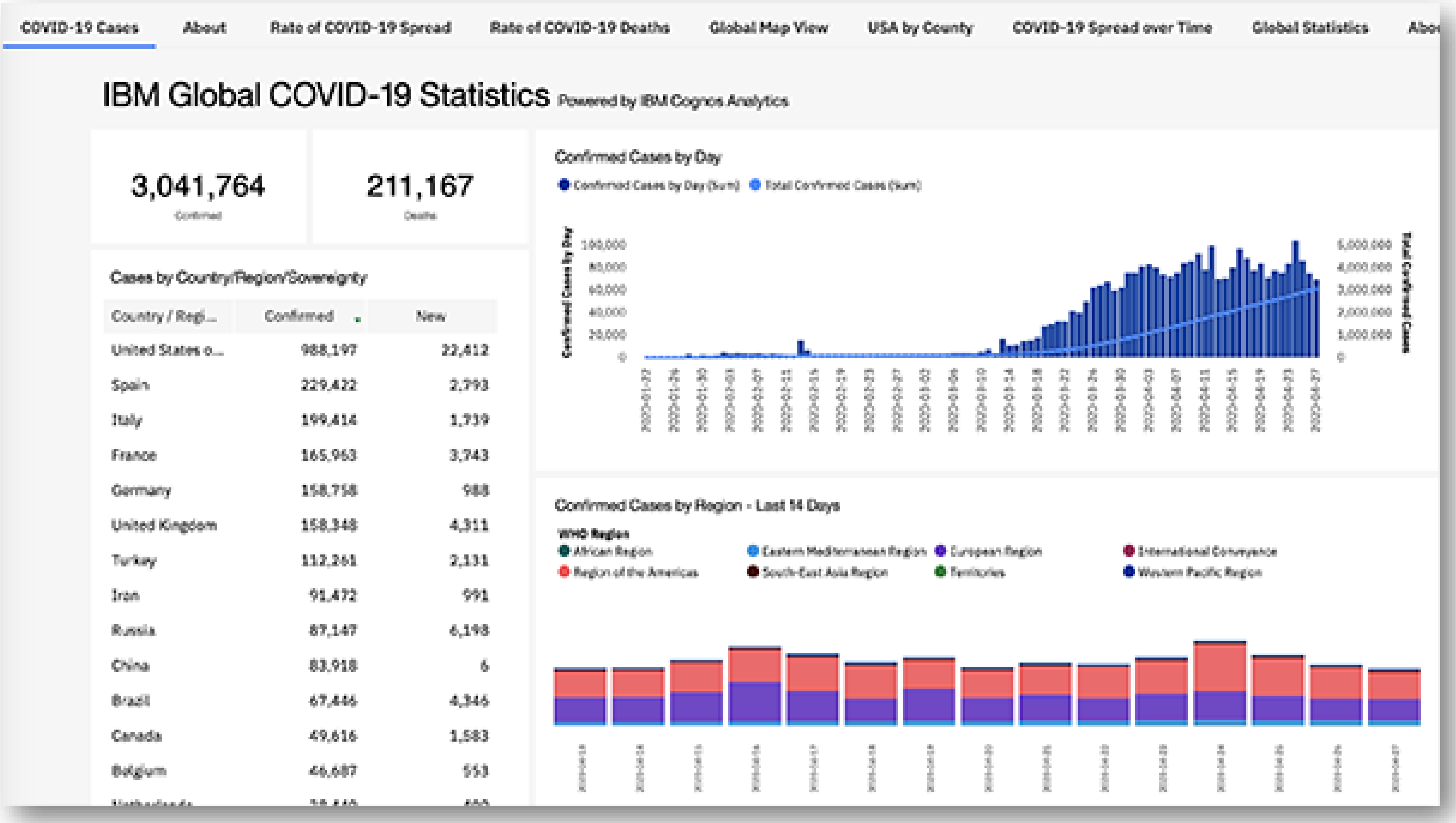
- Cognos allows us to schedule data refreshes and report distribution. We would set up schedules and configurations using the Cognos administration interface.

Custom Scripting:

- Depending on our requirements, we may need to write custom scripts. Cognos uses its own scripting language for expressions, but we can also integrate with external data sources and perform custom scripting in languages like JavaScript if needed.

Security and Compliance:

- In Cognos, we'd configure security settings and data access permissions to ensure that sensitive data is protected.
- Remember that creating a full COVID-19 analysis project in Cognos is a detailed process, and it's important to have a good understanding of Cognos tools and features, as well as a clear project plan to meet our objectives.



IBM Global COVID-19 Statistics

Powered by IBM Cognos Analytics

3,041,764
Confirmed

211,167
Deaths

Cases by Country/Region/Sovereignty

Country / Regi...	Confirmed	New
United States o...	988,197	22,412
Spain	229,422	2,793
Italy	199,414	1,739
France	165,963	3,743
Germany	158,758	988
United Kingdom	158,348	4,311
Turkey	112,261	2,131
Iran	91,472	991
Russia	87,147	6,198
China	83,918	6
Brazil	67,446	4,346
Canada	49,616	1,583
Belgium	46,687	553
Netherlands	[illegible]	[illegible]

PROCEDURE:COVID-19 ANALYSIS IN PYTHON:

Step 1: Install Required Libraries

Eg: pip install pandas matplotlib seaborn scikit-learn

Step 2: Prepare the Data

Obtain COVID-19 data in CSV format.

Save the data file (e.g., covid_data.csv) in the same directory as your Python script

Step 3: Python Script

Use any code editor or integrated development environment (IDE) to create and run the Python script.

Step 4: Running the Script

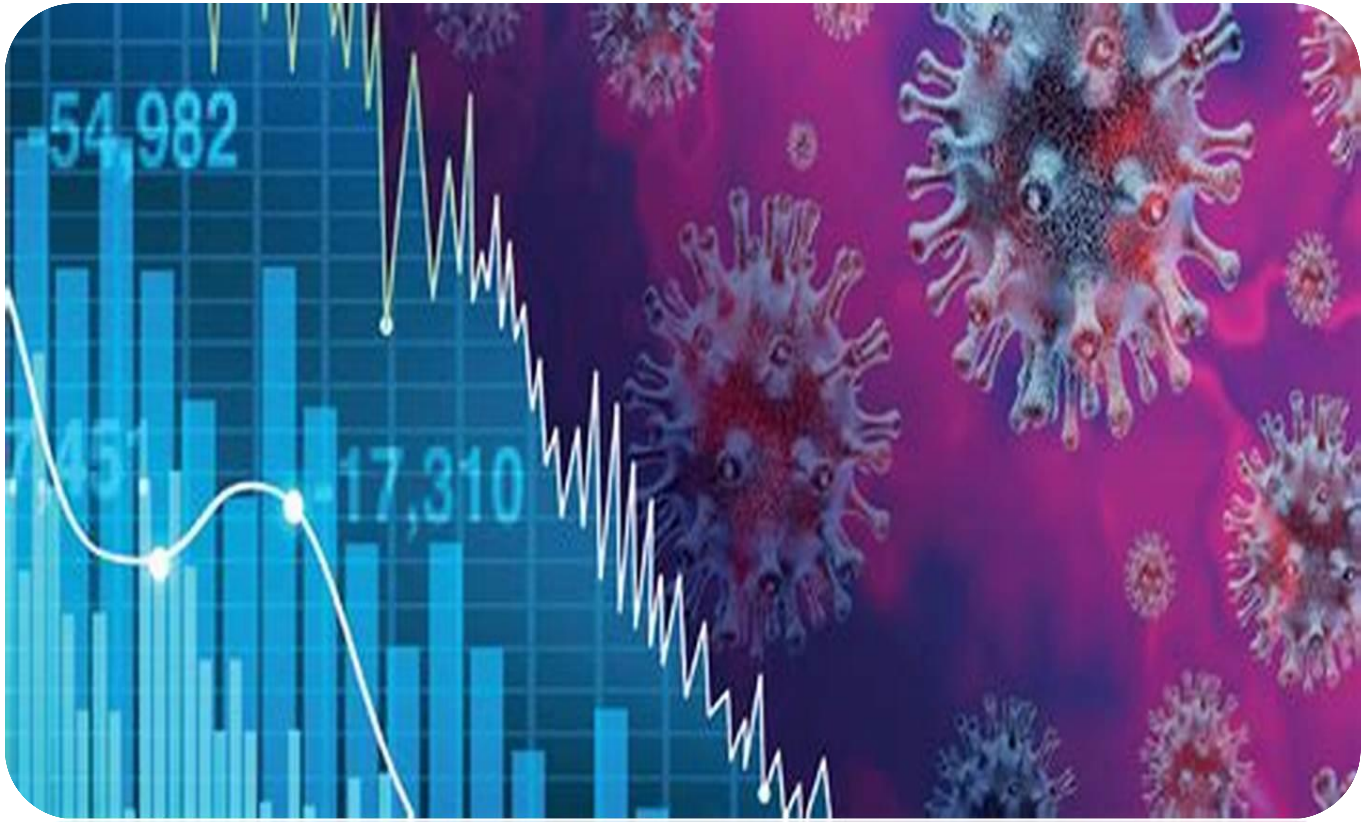
Save and run the script

The script will load the data, perform analysis, and display the visualizations and results as specified in the code.

Review the output and adapt the code as needed for your specific analysis goals.

PYTHON CODE:

To perform COVID-19 data analysis in Python, we can use libraries such as pandas for data manipulation, matplotlib and seaborn for data visualization, and scikit-learn for machine learning.



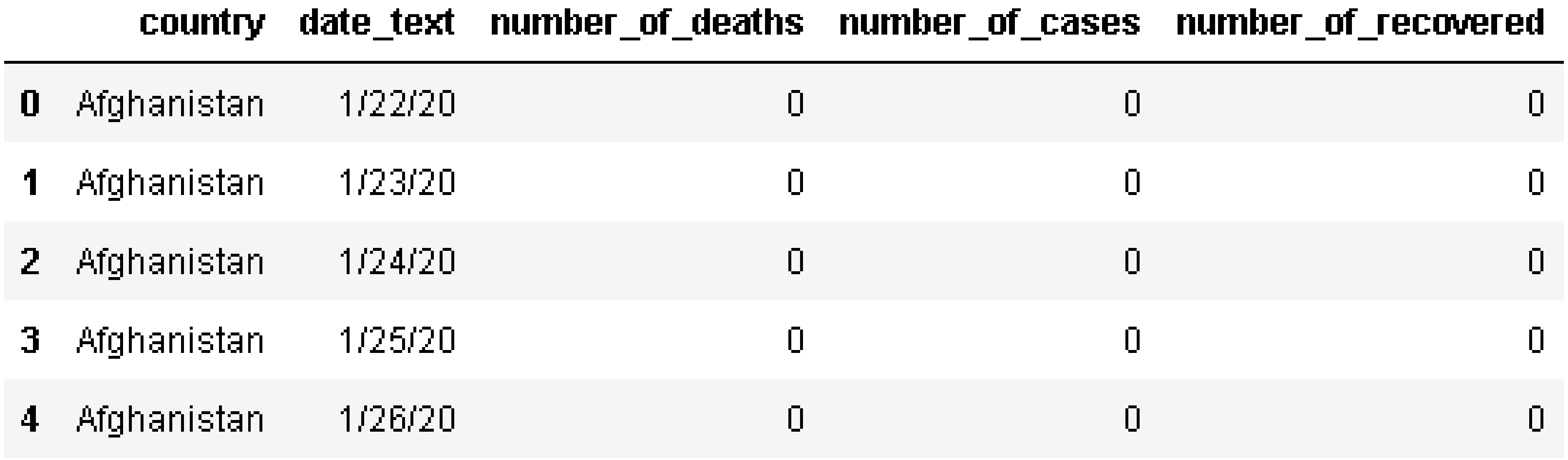
```
import warnings
warnings.simplefilter('ignore')
import requests
cases_request = requests.get('https://coronavirus-tracker-api.herokuapp.com/confirmed')
deaths_request = requests.get('https://coronavirus-tracker-api.herokuapp.com/deaths')
recovered_request = requests.get('https://coronavirus-tracker-api.herokuapp.com/recovered')
cases_json_data = cases_request.json()
deaths_json_data = deaths_request.json()
recovered_json_data = recovered_request.json()
def transform_data(json_data, number_field_name):
complete_list = []
    for country_data in json_data['locations']:
        for history_date, number in country_data['history'].items():
            if history_date == 'latest':
                continue
```

```
day_data = {'country': country_data['country'], 'date_text': history_date, number_field_name:
   number}

           complete_list.append(day_data)

    return complete_list



cases_list = transform_data(cases_json_data,  'number_of_cases')

deaths_list = transform_data(deaths_json_data,  'number_of_deaths')

recovered_list = transform_data(recovered_json_data,  'number_of_recovered')

cases_list[:5]
```

Output:

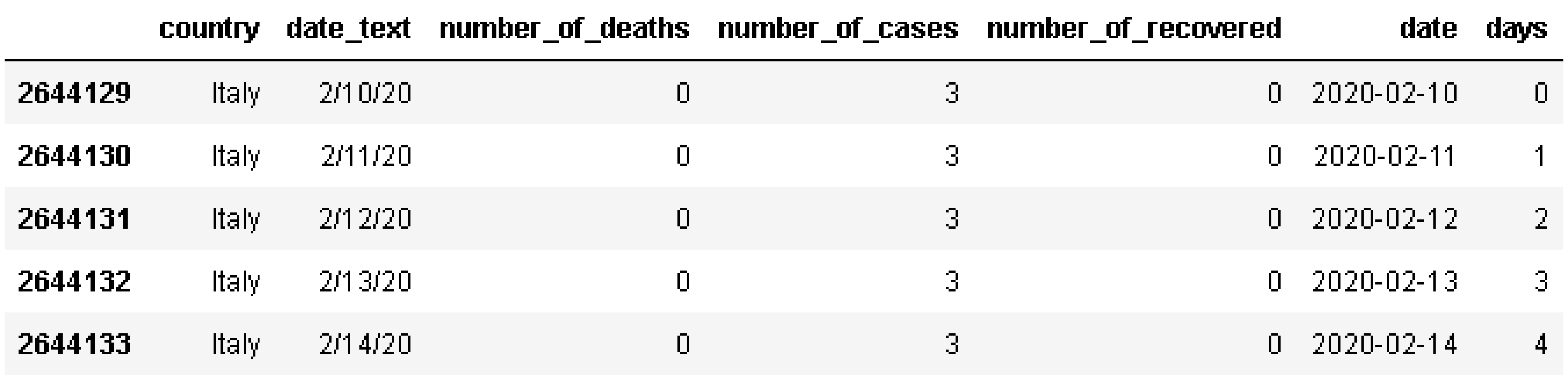
```
[{'country': 'Afghanistan', 'date_text': '1/22/20', 'number_of_cases': 0}, {'country': 'Afghanistan',
   'date_text': '1/23/20', 'number_of_cases': 0}, {'country': 'Afghanistan', 'date_text': '1/24/20',
   'number_of_cases': 0}, {'country': 'Afghanistan', 'date_text': '1/25/20', 'number_of_cases': 0},
   {'country': 'Afghanistan', 'date_text': '1/26/20', 'number_of_cases': 0}]
```

```
import pandas as pd
cases_df = pd.DataFrame(cases_list)
deaths_df = pd.DataFrame(deaths_list)
recovered_df = pd.DataFrame(recovered_list)
complete_df = deaths_df.merge(cases_df, on=['country', 'date_text'])
complete_df = complete_df.merge(recovered_df, on=['country', 'date_text'])
complete_df.head()
```



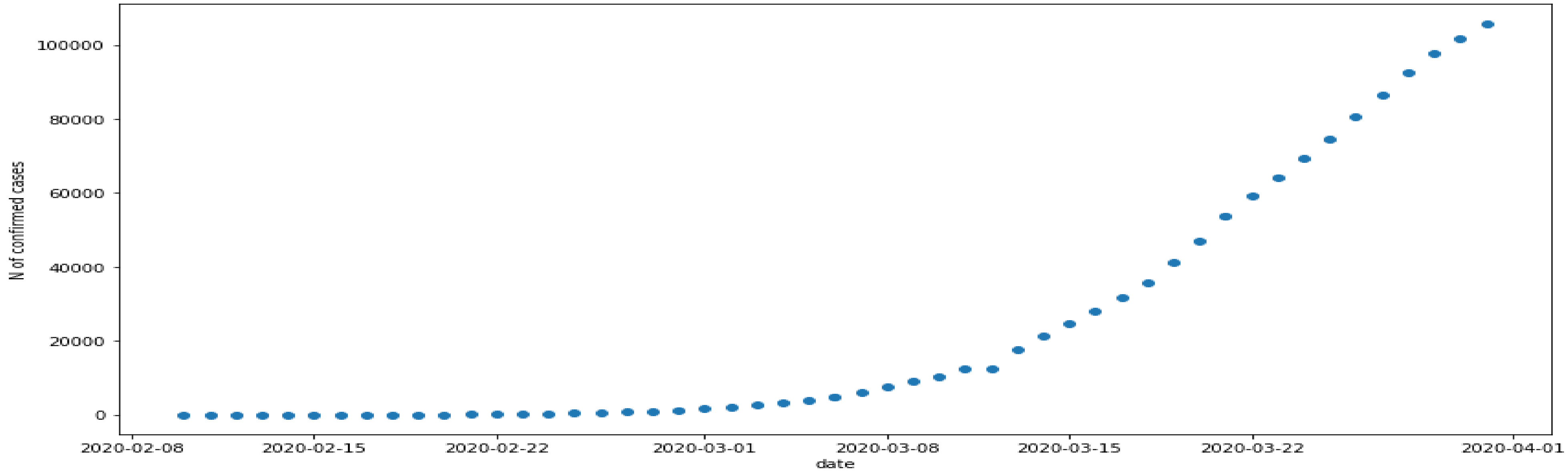
	country	date_text	number_of_deaths	number_of_cases	number_of_recovered
0	Afghanistan	1/22/20	0	0	0
1	Afghanistan	1/23/20	0	0	0
2	Afghanistan	1/24/20	0	0	0
3	Afghanistan	1/25/20	0	0	0
4	Afghanistan	1/26/20	0	0	0

```
from datetime import date, timedelta, datetime

from matplotlib import pyplot as plt

from datetime import date, datetime

country = 'Italy'

start_date = date(2020, 2, 10)

country_df = complete_df[complete_df['country'] == country]

country_df['date'] = country_df['date_text'].apply(lambda x: datetime.strptime(x,
  '%m/%d/%y').date())

country_df['days'] = country_df['date'].apply(lambda x: (x - start_date).days)

country_df = country_df[country_df['days'] >= 0]

country_df = country_df.sort_values(by='days')

country_df.head()
```

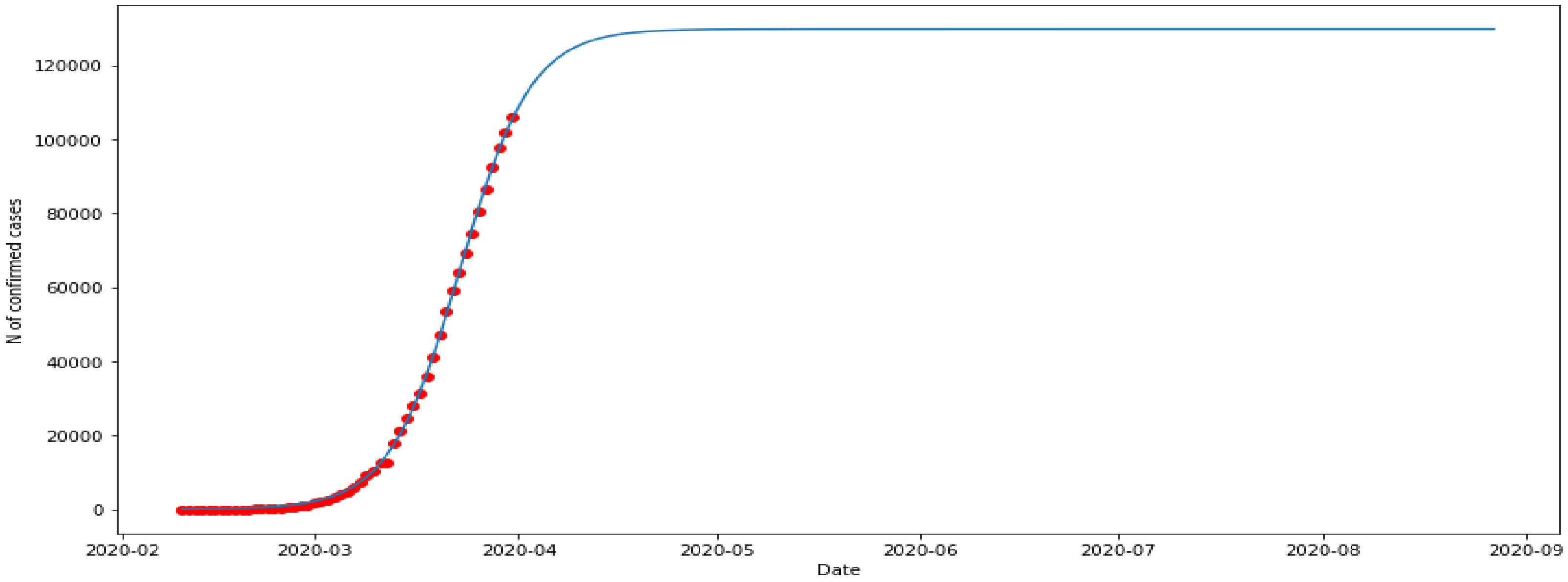


	country	date_text	number_of_deaths	number_of_cases	number_of_recovered	date	days
2644129	Italy	2/10/20	0	3	0	2020-02-10	0
2644130	Italy	2/11/20	0	3	0	2020-02-11	1
2644131	Italy	2/12/20	0	3	0	2020-02-12	2
2644132	Italy	2/13/20	0	3	0	2020-02-13	3
2644133	Italy	2/14/20	0	3	0	2020-02-14	4

```
plt.figure(figsize=(14, 7))
plt.scatter(country_df['date'], country_df['number_of_cases'])
plt.xlabel('date')
plt.ylabel('N of confirmed cases')
plt.show()
```

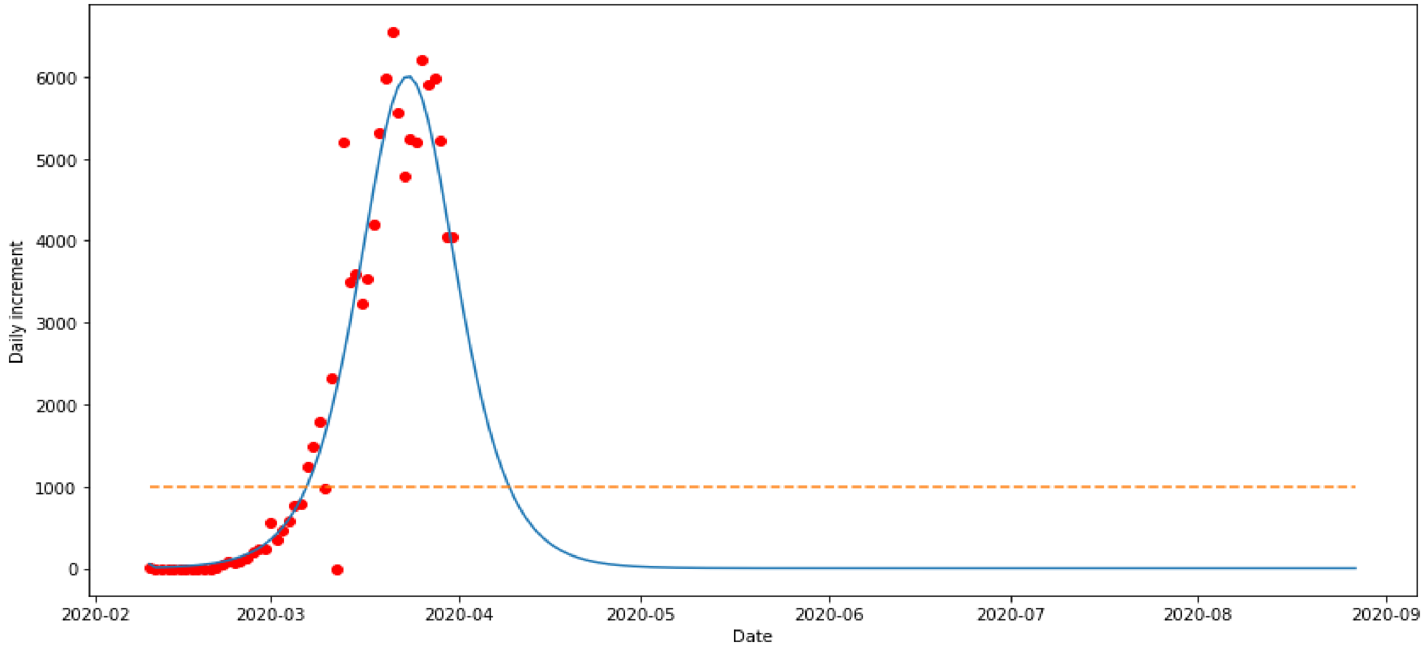


```
from scipy.optimize import curve_fit
import numpy as np
def sigmoid(x, m, k, x0):
    """The standard logistic function."""
    return m / (1 + np.exp(-k * (x - x0)))
x_train = country_df['days'].to_numpy()
y_train = country_df['number_of_cases'].to_numpy()
params, __ = curve_fit(sigmoid, x_train, y_train)
params.tolist()
m, k, x0 = params.tolist()
days_to_predict = np.array(list(range(200)))
prediction_dates_list = [start_date + timedelta(days=i) for i in range(200)]
predicted_cases = sigmoid(days_to_predict, m, k, x0)
hictorical_dates = country_df['date']
plt.figure(figsize=(14,7))
plt.scatter(hictorical_dates, country_df['number_of_cases'], color='r')
plt.plot(prediction_dates_list, predicted_cases)

plt.xlabel('Date')

plt.ylabel('N of confirmed cases')

plt.show()
```

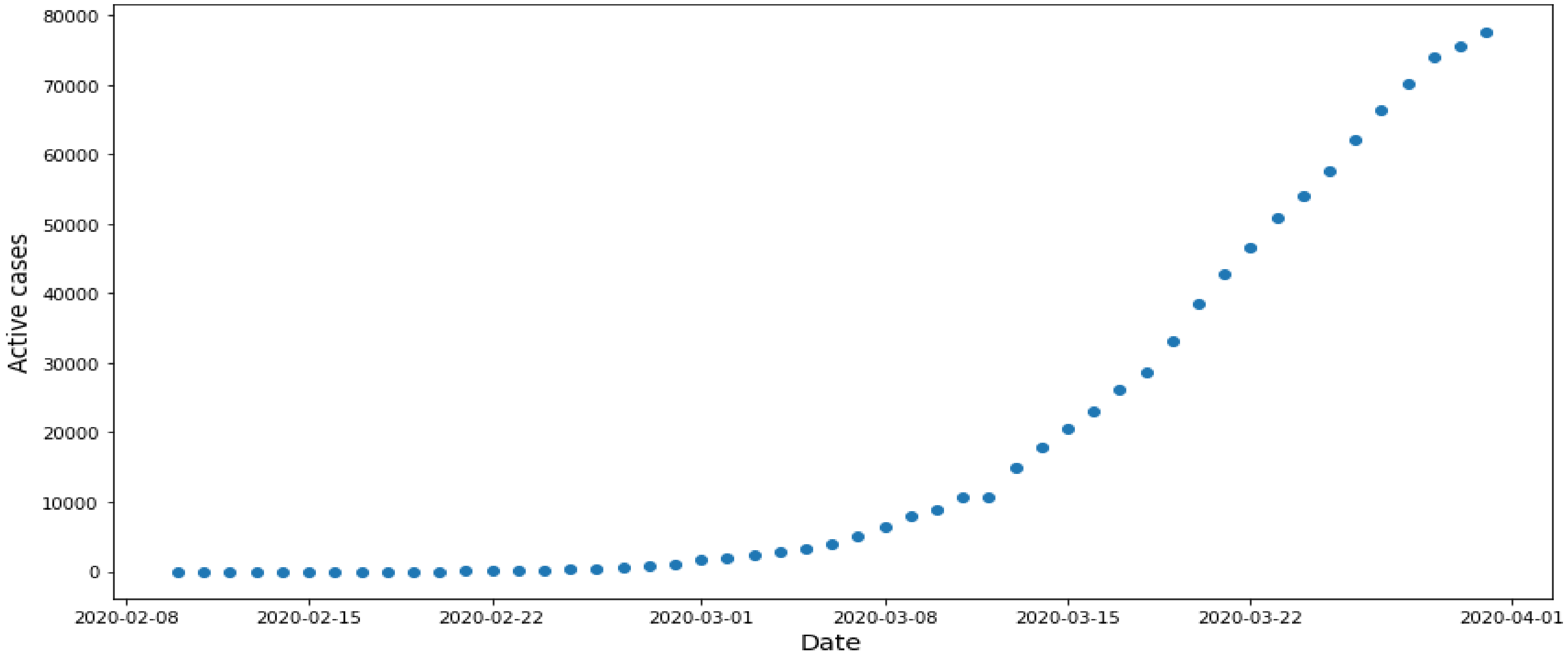


```
def get_new_daily_cases(cumulative_cases_list):
    """The function that transforms a list of
accumulated daily cases to a list of new cases."""
    days_number = len(cumulative_cases_list)
    daily_cases = [cumulative_cases_list[i  + 1] -
 cumulative_cases_list[i]  for i in range(days_number -  1)]
```

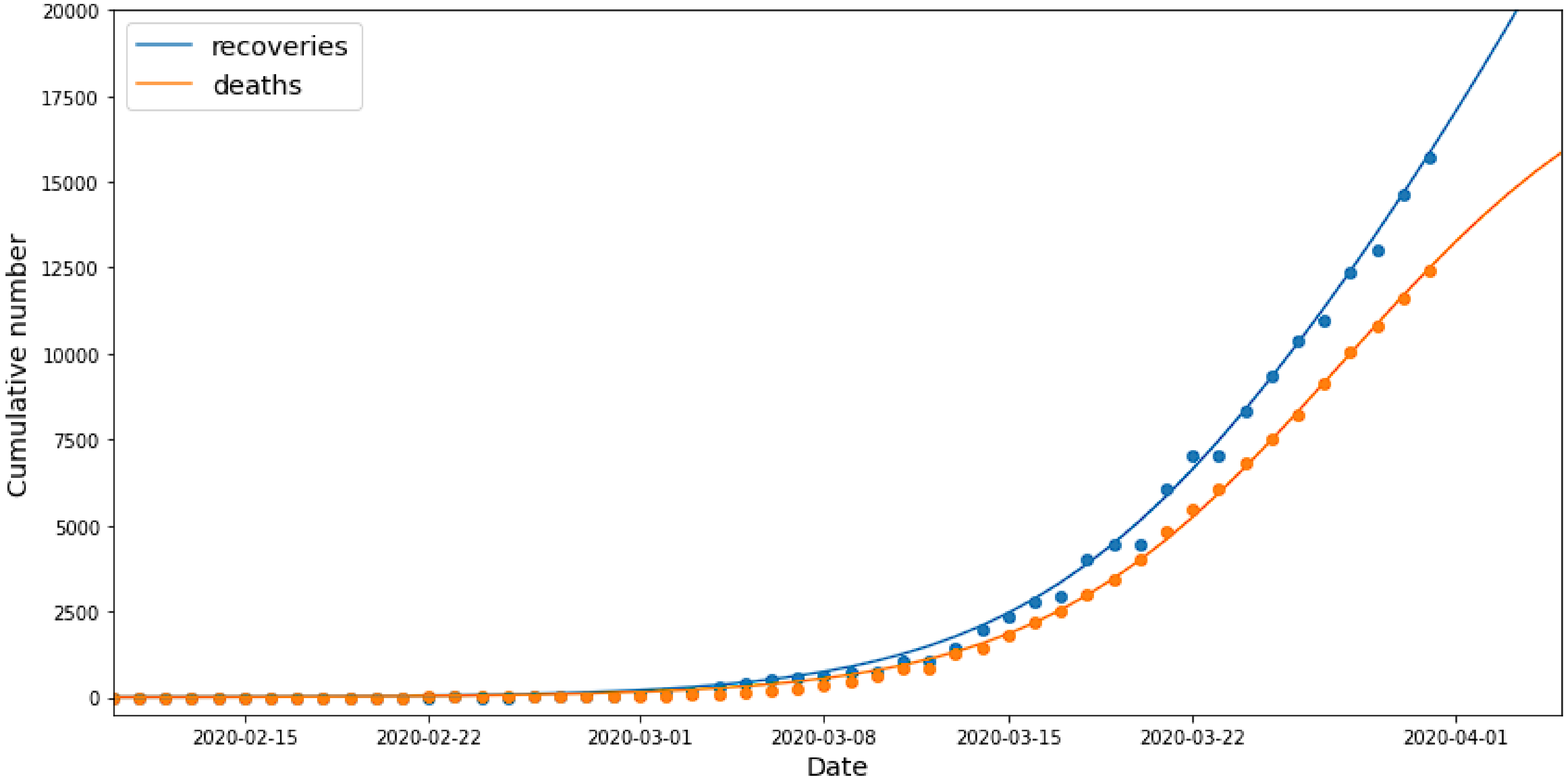
```
predicted_cases_list = predicted_cases.tolist()
predicted_daily_new_cases = get_new_daily_cases(predicted_cases_list)
historical_cases_list  = y_train.tolist()
historical_daily_new_cases = get_new_daily_cases(historical_cases_list)
plt.figure(figsize=(14,7))
plt.scatter(hictorical_dates, historical_daily_new_cases, color='r')
plt.plot(prediction_dates_list, predicted_daily_new_cases)
plt.plot(prediction_dates_list, [1000]*len(prediction_dates_list), '--')    # Trending line
plt.xlabel('Date')
plt.ylabel('Daily increment')
plt.show()
```



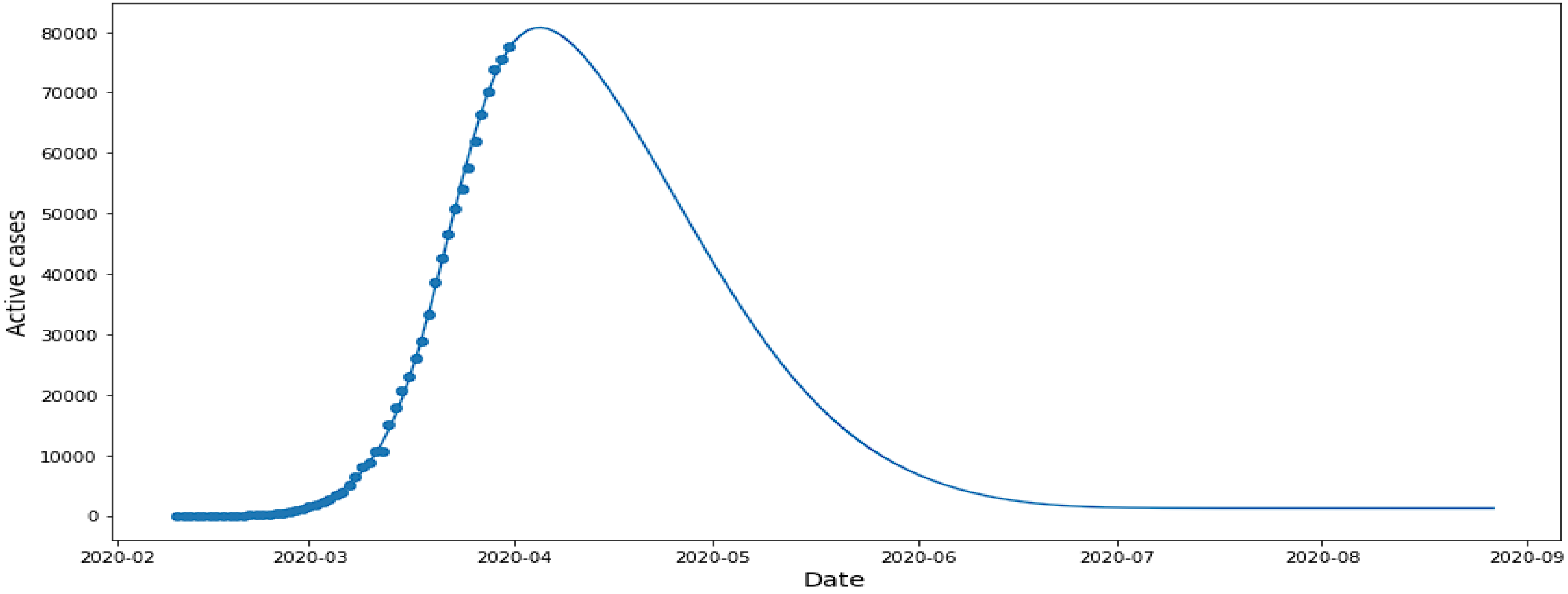
```
country_df['active_cases'] = (country_df['number_of_cases'] -
 country_df['number_of_deaths'] - country_df['number_of_recovered'])
plt.figure(figsize=(14,7))
plt.scatter(country_df['date'], country_df['active_cases'])
plt.xlabel('Date', fontsize='x-large')
plt.ylabel('Active cases', fontsize='x-large')
plt.show()
```

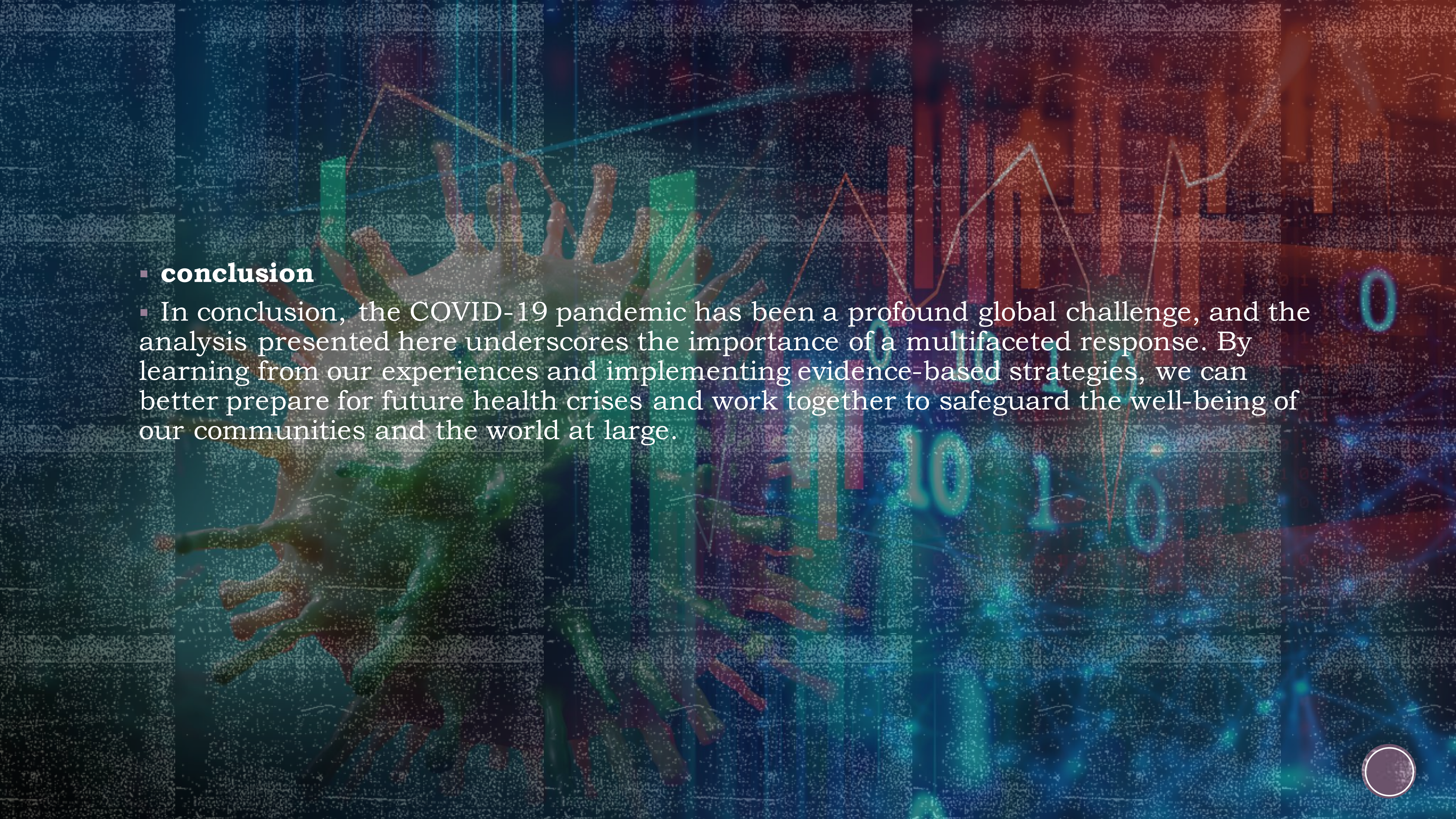


```
deaths_forecast = final_state_function(days_to_predict, m_death, k_death, x0_death,
predicted_daily_new_cases)
recoveries_forecast = final_state_function(days_to_predict, 1 - m_death, k_recovery,
x0_recovery, predicted_daily_new_cases)
plt.figure(figsize=(14,7))
plt.scatter(hictorical_dates, historical_recoveries)
plt.scatter(hictorical_dates, historical_deaths)
plt.plot(prediction_dates_list, recoveries_forecast, label='recoveries')
plt.plot(prediction_dates_list, deaths_forecast, label='deaths')
plt.legend(fontsize='x-large')
plt.xlim(date(2020, 2, 10), date(2020, 4, 5))
plt.ylim(-500, 20000)
plt.xlabel('Date', fontsize='x-large')
plt.ylabel('Cumulative number', fontsize='x-large')
plt.show()
```


recoveries
deaths
Cumulative number
20000
17500
15000
12500
10000
7500
5000
2500
0
2020-02-15
2020-02-22
2020-03-01
2020-03-08
2020-03-15
2020-03-22
2020-04-01
Date

```
active_cases_forecast = np.array(predicted_cases_list) - deaths_forecast - recoveries_forecast
plt.figure(figsize=(14,7))
plt.plot(prediction_dates_list, active_cases_forecast)
plt.scatter(country_df['date'], country_df['active_cases'])
plt.xlabel('Date', fontsize='x-large')
plt.ylabel('Active cases', fontsize='x-large')
plt.show()
```





- **conclusion**
- In conclusion, the COVID-19 pandemic has been a profound global challenge, and the analysis presented here underscores the importance of a multifaceted response. By learning from our experiences and implementing evidence-based strategies, we can better prepare for future health crises and work together to safeguard the well-being of our communities and the world at large.