VARUVAN VADIVELAN INSTITUTE OF TECHNOLOGY

NAAN MUDHALAVAN : IBM

PHASE : 5

TECHNOLOGY : DATA SCIENCE

PROJECT TITLE: COVID 19 VACCINE ANALYSIS

INTRODUCTION

Introducing the COVID-19 vaccine, a beacon of hope in our global battle against the pandemic. Developed through rigorous scientific endeavours this vaccine represents a triumph of human innovation and collaboration. Designed to stimulate the immune system's response to the virus, the vaccine offers a shield against severe illness and, ultimately, paves the way for a safer, healthier world. As we navigate these challenging times, the vaccine stands as a powerful tool, reminding us that, together, we can overcome adversity and protect the well-being of individuals and communities worldwide.

PROBLEM STATEMENT

Tracking the progress of the Covid-19 vaccine in Nigeria in comparison to Africa and the world.

DATA

The data used in this analysis was collected from data . This dataset contains the data from all countries of the world with the following features:

Country- this is the country for which the vaccination information is provided;

Country ISO Code – ISO code for the country;

Date – date for the data entry; for some of the dates we have only the daily vaccinations, for others, only the (cumulative) total;

Total number of vaccinations – this is the absolute number of total immunizations in the country;

Total number of people vaccinated – a person, depending on the immunization scheme, will receive one or more (typically 2) vaccines; at a certain moment, the number of vaccination might be larger than the number of people;

Total number of people fully vaccinated – this is the number of people that received the entire set of immunization according to the immunization scheme (typically 2); at a certain moment in time, there might be a certain number of people that received one vaccine and another number (smaller) of people that received all vaccines in the scheme;

Daily vaccinations (raw) – for a certain data entry, the number of vaccination for that date/country;

Daily vaccinations – for a certain data entry, the number of vaccination for that date/country;

Total vaccinations per hundred – ratio (in percent) between vaccination number and total population up to the date in the country;

Total number of people vaccinated per hundred – ratio (in percent) between population immunized and total population up to the date in the country;

Total number of people fully vaccinated per hundred – ratio (in percent) between population fully immunized and total population up to the date in the country;

Number of vaccinations per day – number of daily vaccination for that day and country;

Daily vaccinations per million – ratio (in ppm) between vaccination number and total population for the current date in the country;

Vaccines used in the country – total number of vaccines used in the country (up to date);

Source name – source of the information (national authority, international organization, local organization etc.);

Source website – website of the source of information;

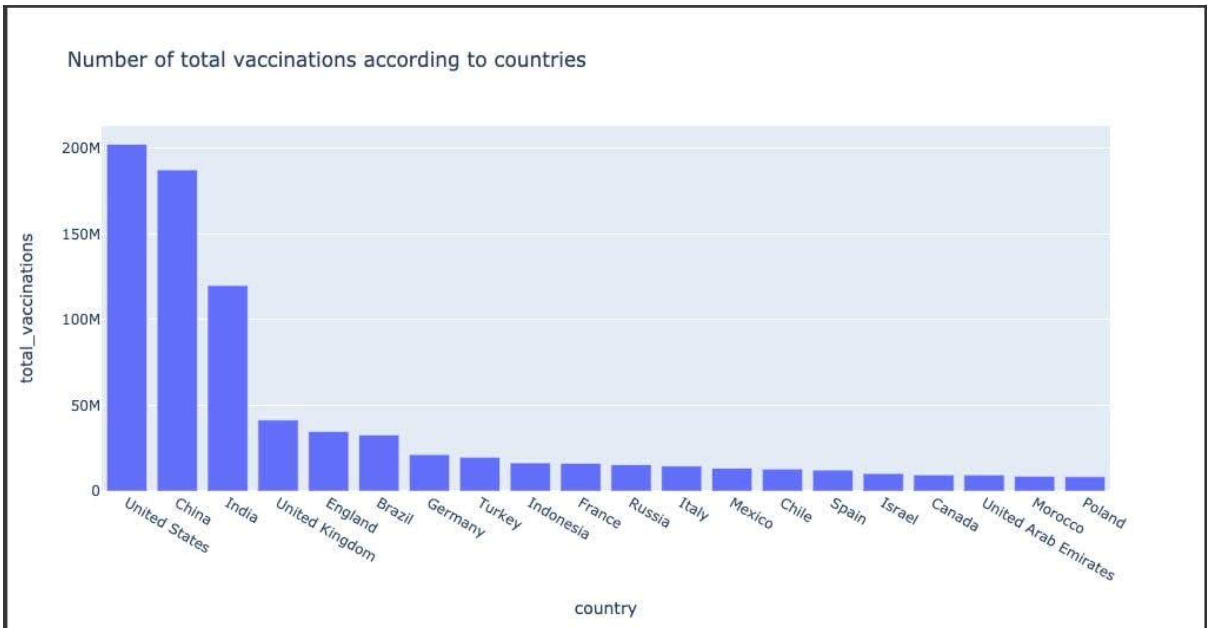
PREPROCESSING

After I imported the needed libraries, loaded the

dataset and viewed the information therein, I created a list of all African countries and stored them in a variable which I named Africa. Then I created a new data frame that was grouped by the following columns: country iso code vaccines total vaccinations people vaccinated people fully vaccinated daily vaccinations total vaccinations per hundred people vaccinated per hundred people fully vaccinated per hundred daily vaccinations per million.

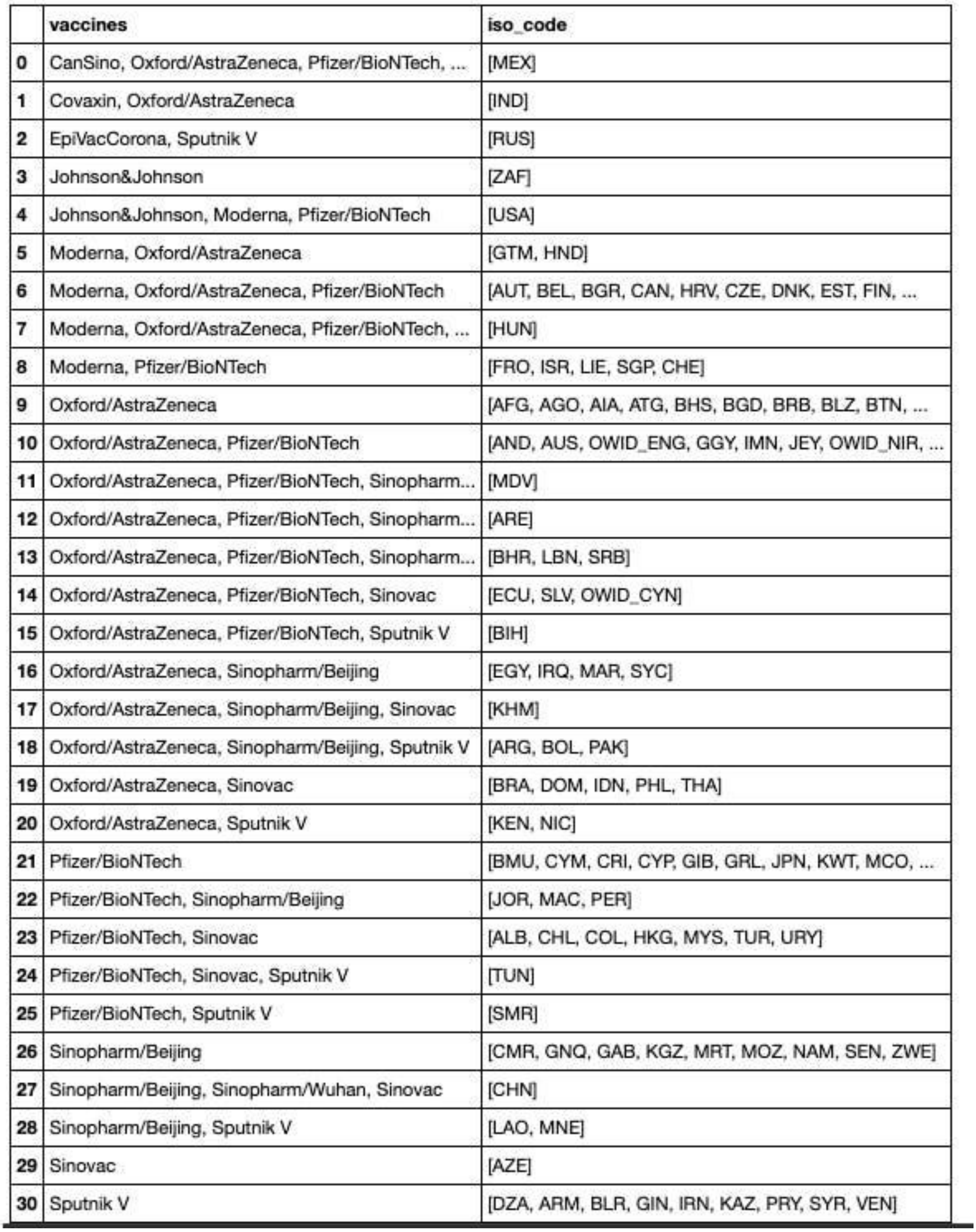
WORLD FOCUS

I analyzed total number of vaccinations around the world by ranking of the first 20 countries



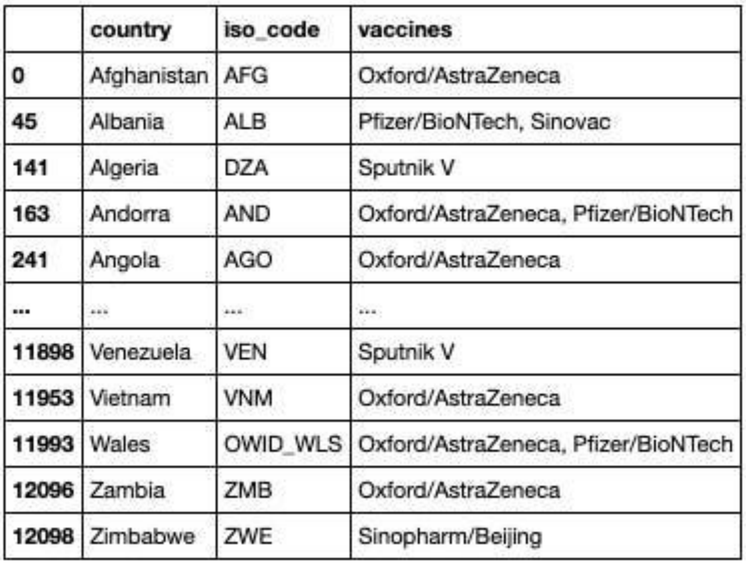
From the above graph, it shows the United States as the number one in the world with total number of vaccinations done while Bangladesh ranked the least.

I visualized the types of vaccines available to the countries which it is supplied to



	vaccines	iso_code
0	CanSino, Oxford/AstraZeneca, Pfizer/BioNTech, ...	[MEX]
1	Covaxin, Oxford/AstraZeneca	[IND]
2	EpiVacCorona, Sputnik V	[RUS]
3	Johnson&Johnson	[ZAF]
4	Johnson&Johnson, Moderna, Pfizer/BioNTech	[USA]
5	Moderna, Oxford/AstraZeneca	[GTM, HND]
6	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	[AUT, BEL, BGR, CAN, HRV, CZE, DNK, EST, FIN, ...
7	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech, ...	[HUN]
8	Moderna, Pfizer/BioNTech	[FRO, ISR, LIE, SGP, CHE]
9	Oxford/AstraZeneca	[AFG, AGO, AIA, ATG, BHS, BGD, BRB, BLZ, BTN, ...
10	Oxford/AstraZeneca, Pfizer/BioNTech	[AND, AUS, OWID_ENG, GGY, IMN, JEY, OWID_NIR, ...
11	Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm...	[MDV]
12	Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm...	[ARE]
13	Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm...	[BHR, LBN, SRB]
14	Oxford/AstraZeneca, Pfizer/BioNTech, Sinovac	[ECU, SLV, OWID_CYN]
15	Oxford/AstraZeneca, Pfizer/BioNTech, Sputnik V	[BIH]
16	Oxford/AstraZeneca, Sinopharm/Beijing	[EGY, IRQ, MAR, SYC]
17	Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac	[KHM]
18	Oxford/AstraZeneca, Sinopharm/Beijing, Sputnik V	[ARG, BOL, PAK]
19	Oxford/AstraZeneca, Sinovac	[BRA, DOM, IDN, PHL, THA]
20	Oxford/AstraZeneca, Sputnik V	[KEN, NIC]
21	Pfizer/BioNTech	[BMU, CYM, CRI, CYP, GIB, GRL, JPN, KWT, MCO, ...
22	Pfizer/BioNTech, Sinopharm/Beijing	[JOR, MAC, PER]
23	Pfizer/BioNTech, Sinovac	[ALB, CHL, COL, HKG, MYS, TUR, URY]
24	Pfizer/BioNTech, Sinovac, Sputnik V	[TUN]
25	Pfizer/BioNTech, Sputnik V	[SMR]
26	Sinopharm/Beijing	[CMR, GNQ, GAB, KGZ, MRT, MOZ, NAM, SEN, ZWE]
27	Sinopharm/Beijing, Sinopharm/Wuhan, Sinovac	[CHN]
28	Sinopharm/Beijing, Sputnik V	[LAO, MNE]
29	Sinovac	[AZE]
30	Sputnik V	[DZA, ARM, BLR, GIN, IRN, KAZ, PRY, SYR, VEN]

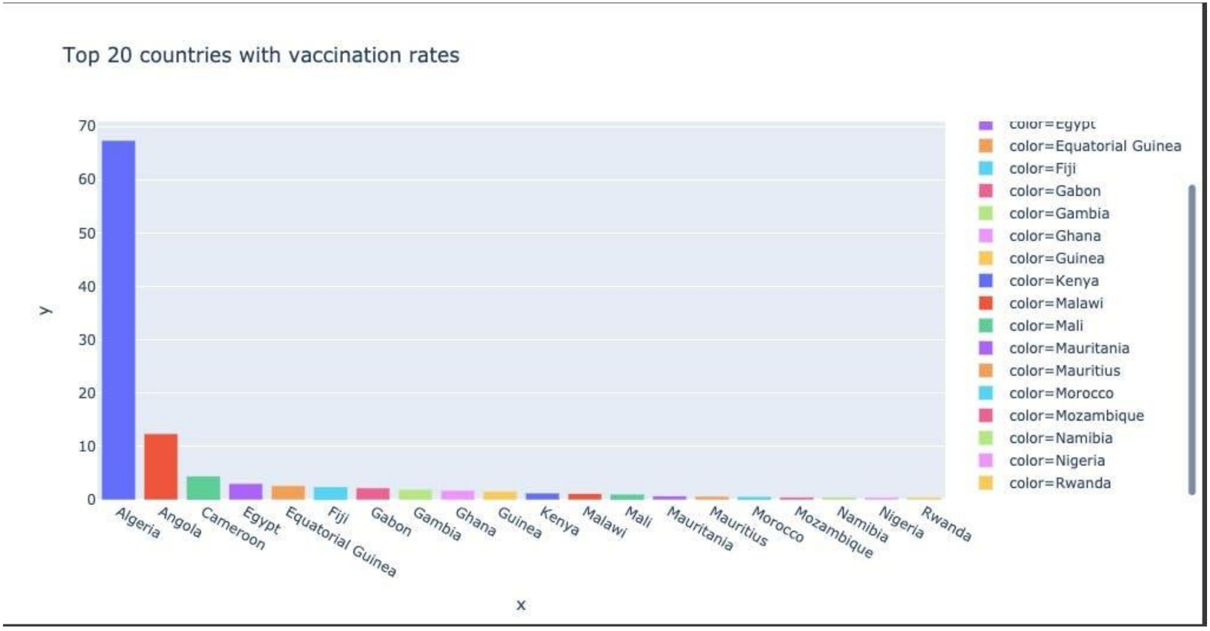
Then the total distribution so far around the world



	country	iso_code	vaccines
0	Afghanistan	AFG	Oxford/AstraZeneca
45	Albania	ALB	Pfizer/BioNTech, Sinovac
141	Algeria	DZA	Sputnik V
163	Andorra	AND	Oxford/AstraZeneca, Pfizer/BioNTech
241	Angola	AGO	Oxford/AstraZeneca
...	...	...	...
11898	Venezuela	VEN	Sputnik V
11953	Vietnam	VNM	Oxford/AstraZeneca
11993	Wales	OWID_WLS	Oxford/AstraZeneca, Pfizer/BioNTech
12096	Zambia	ZMB	Oxford/AstraZeneca
12098	Zimbabwe	ZWE	Sinopharm/Beijing

Focus On Africa:

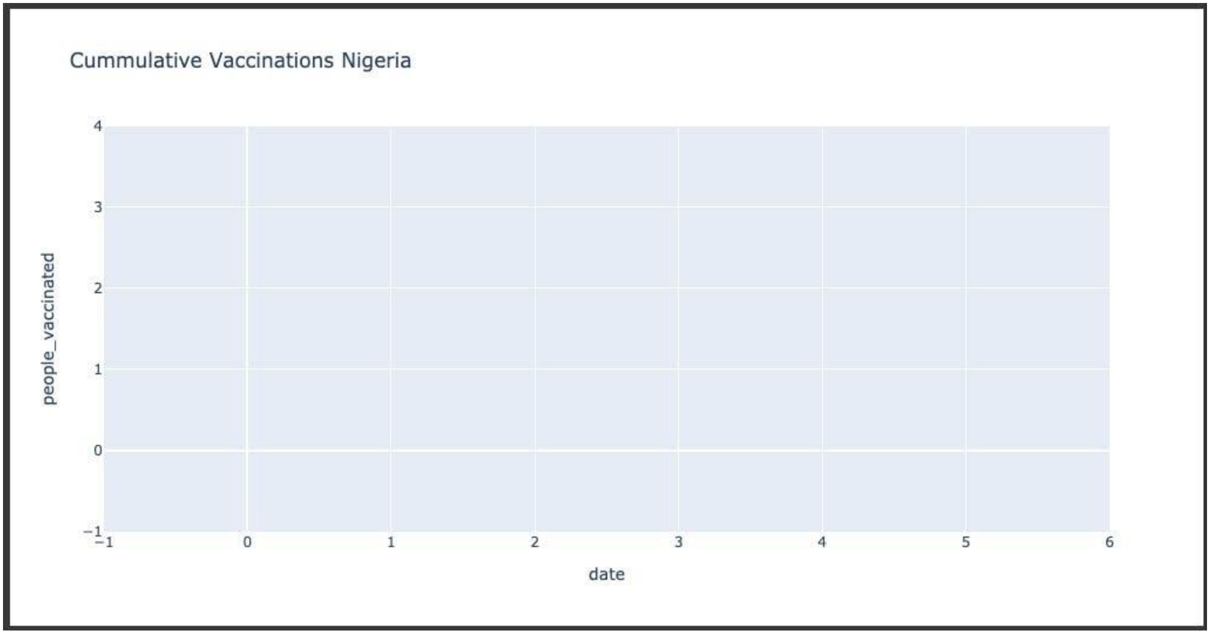
I analyzed the top 20 countries in Africa in terms of their vaccination rate



The above graph shows that Algeria has the highest number of vaccination rate with Mozambique, Nigeria, Namibia and Rwanda occupying the least vaccination rates.

Focus on Nigeria:

I tried analyzing the cumulative vaccination in Nigeria and came up with an empty graph



vaccination in Nigeria stems from the lack of daily data update and this is a problem that needs to be corrected.

VACCINE DESIGN

COVID-19, Coronavirus Disease-19; SARS-CoV-2, Severe Acute Respiratory Syndrome Coronavirus-2; RSV, Respiratory Syncytial Virus; MERS, Middle East Respiratory Syndrome; RBD, Receptor Binding Domain; ACE2, Angiotensin Converting Enzyme 2, Coalition for Epidemic Preparedness Innovations; EUA, Emergency Use Authorization; CDC, Centers for Disease Control and Prevention; SAGE, Strategic Advisory Group of Experts on Immunization; ASHP, American Society of Health-System Pharmacists; PRR, Pattern Recognition Receptors; PAMP, Pathogen-Associated Molecular Pattern; TLR, Toll-Like Receptor; VLP, Virus-Like Particle; I.M., Intramuscular; S.C., Subcutaneous; APC, Antigen-Presenting Cell; WCA, Whole-Cell Antigen; LAV, Live-Attenuated Virus;

Ad, Adenovirus; GMV, Genetically Modified Viruses; MVA, Modified Vaccinia Ankara; LMH, Live Modified Horsepox; VEE, Venezuelan Equine Encephalitis; Chimpanzee Adenoviral Vector 1, ChAdOx1; VRP, Virion Replicon Particle; LNP, Lipid Nanoparticles

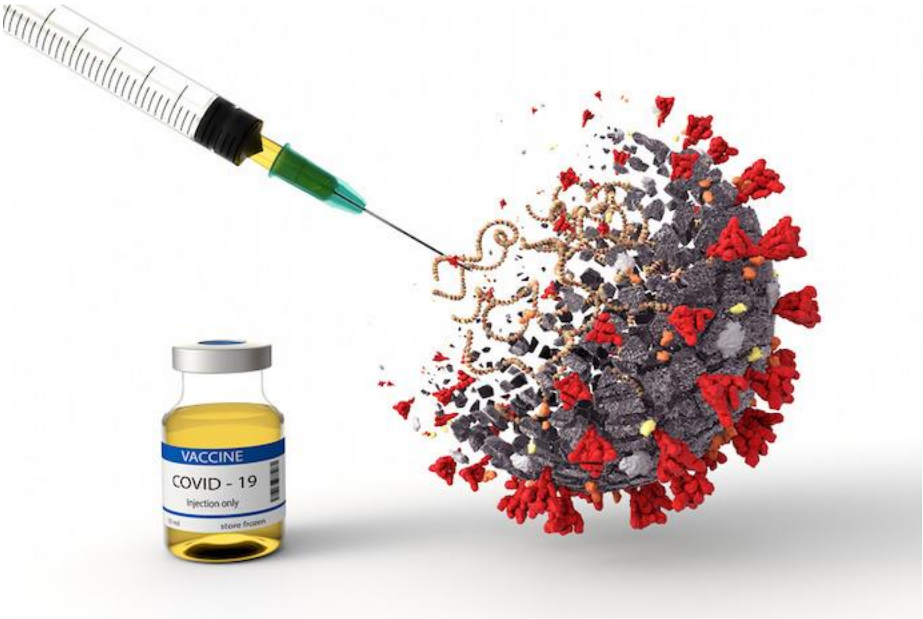
DISTRIBUTION:

An ethical, equitable, and efficient distribution of COVID-19 vaccines is imperative to minimize the risk of new variants of the virus and protect the population from SARS-CoV-2

The board of the ASHP (American Society of Health-System Pharmacists) approved some principles for distributing the COVID-19 vaccine on August 25, 2020. ASHP offers effective communication, cooperation, and coordination with local and international public health organizations, regulatory agencies, and health departments to implement a framework for the ethical and equitable global distribution of COVID-19 vaccines. For the effective distribution of COVID-19 vaccines, the organization also proposes the best practices for proper storage methods and the use of vaccines with maximum shelf life, minimized decadence, and no wastage by temperature control, from distribution to administration .

Using Data Analytics to Accelerate COVID-19 Vaccine Development

In the midst of a global COVID-19 pandemic, a top priority for many pharma and biopharma companies is to get a vaccine developed, produced and delivered to the public as quickly as possible. Ushering a vaccine through rigorous testing protocols and regulatory approvals is not an easy (or quick) effort, but incorporating advanced data analytics could help accelerate the process. Data analytics has proven effective in speeding vaccine development both by enabling more efficient Design of Experiments (DOE) and by creating rapid-scale production rollout processes.



The key to accelerating vaccine development in the face of COVID-19 could be to involve data analytics, both using DOE for a more efficient QbD approach and MVDA for faster data insights.

COVID-19 has presented the biopharma industry with a number of unique challenges. First and foremost, as a novel coronavirus, it's a new pathogen in humans which has not only made it a challenge to diagnose and treat, but also makes it difficult to develop a vaccine with demonstrated efficacy. In addition, many of the technologies being pursued for treatment, such as RNA vaccines, are relatively novel themselves. So, while vaccine developers are moving as quickly as possible, the process still must be very thorough. That is where data analytics comes in to play.

Some of the major vaccine challenges that can be addressed with data analytics include:

- accelerating process development
- developing robust scale-up and tech transfer methods
- ensuring manufacturing success improving manufacturing processes

Process Development

When it comes to accelerating process development, Design of Experiments (or DOE) is a tool that allows for a systematic approach to process development studies – ultimately reducing the number of experiments needed, and in the long run, also reducing the overall cost of experimentation. We also can't overlook the importance that acceleration of process development can have on gaining a competitive edge through speed to market. Afterall, there are more than 150 companies engaged in the race to develop a COVID-19 vaccine candidate.

Support Scale-Up and Tech Transfer

The second challenge that data analytics can support is scale-up and tech transfer. When companies need to quickly and efficiently produce hundreds of millions of doses globally, it's important to have an efficient and organized way to manage scale-up and technology transfer. For any manufacturer, commercial success depends on being able to increase drug substance production volume quickly and effectively and to move to production freely. The time and financial cost of failure can be significant.

The expectation to scale-up to manufacturing in six to 12 months to address COVID-19 is an unprecedented speed. However, by using data analytics tools like MVDA (multivariate data analysis), the number of total batches needed to prove robustness can be less. Therefore, it's important to use the best data you have for single or multiple batch runs during both process development and GMP manufacturing.

Continuously Improve Manufacturing

The last challenge is being able to continuously improve the manufacturing process. Using real-time analytics to monitor and control manufacturing processes has been a proven tool to ensure both process robustness as well as the product quality – even in such expedited timelines.

A Closer Look at Accelerating Process Development

During vaccine process development, product components, in-process materials, final product specifications and manufacturing processes are all defined. And at the same time researchers are having to decide on and consider things like safety, efficacy, costs, transport, storage, administration, doses, and immunity.

A Quality by Design (QbD) approach to Design of Experiments (DOE) enables vaccine developers to systematically determine the individual and interactive effects of various factors that influence the results of experiments.

For vaccine development, DOE can be broken down into three different investigational objectives:

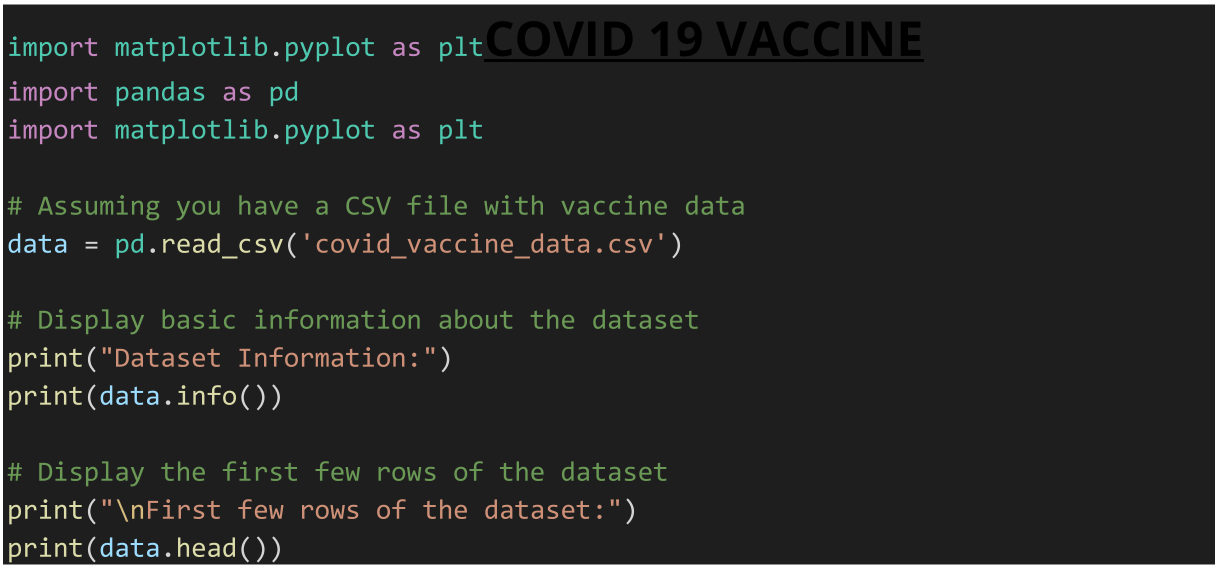
1. Accelerating screening
2. Supporting optimization
3. Ensuring robust characterization

Accelerating screening means that vaccine developers can investigate many process parameters at the same time, enabling faster time to market, reduced costs for experimentation, and overall maximized knowledge. One example of accelerated screening can be done during clone screening processes.

The second area where DOE supports accelerated vaccine development is in **process optimization.** That means developers can use DOE to help determine factors, ranges and inputs needed to achieve specific process goals, for example, to ensure high-quality, high-performing and safe products. One example of this could be media optimization or formula optimization.

The last point is **ensuring robust characterization.** DOE can be used to analyze each unit operation's design space, and then to calculate the extent of all the ICH Q8 guidelines. Doing proper bioprocess characterization ensures product stability, robustness and scalability, as well as staying in compliance with regulatory requirements.

COVID 19 VACCINE

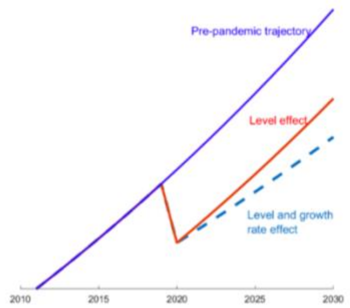


COVID 19 VACCINE

```
import matplotlib.pyplot as plt
import pandas as pd
import matplotlib.pyplot as plt

# Assuming you have a CSV file with vaccine data
data = pd.read_csv('covid_vaccine_data.csv')

# Display basic information about the dataset
print("Dataset Information:")
print(data.info())

# Display the first few rows of the dataset
print("\nFirst few rows of the dataset:")
print(data.head())
```

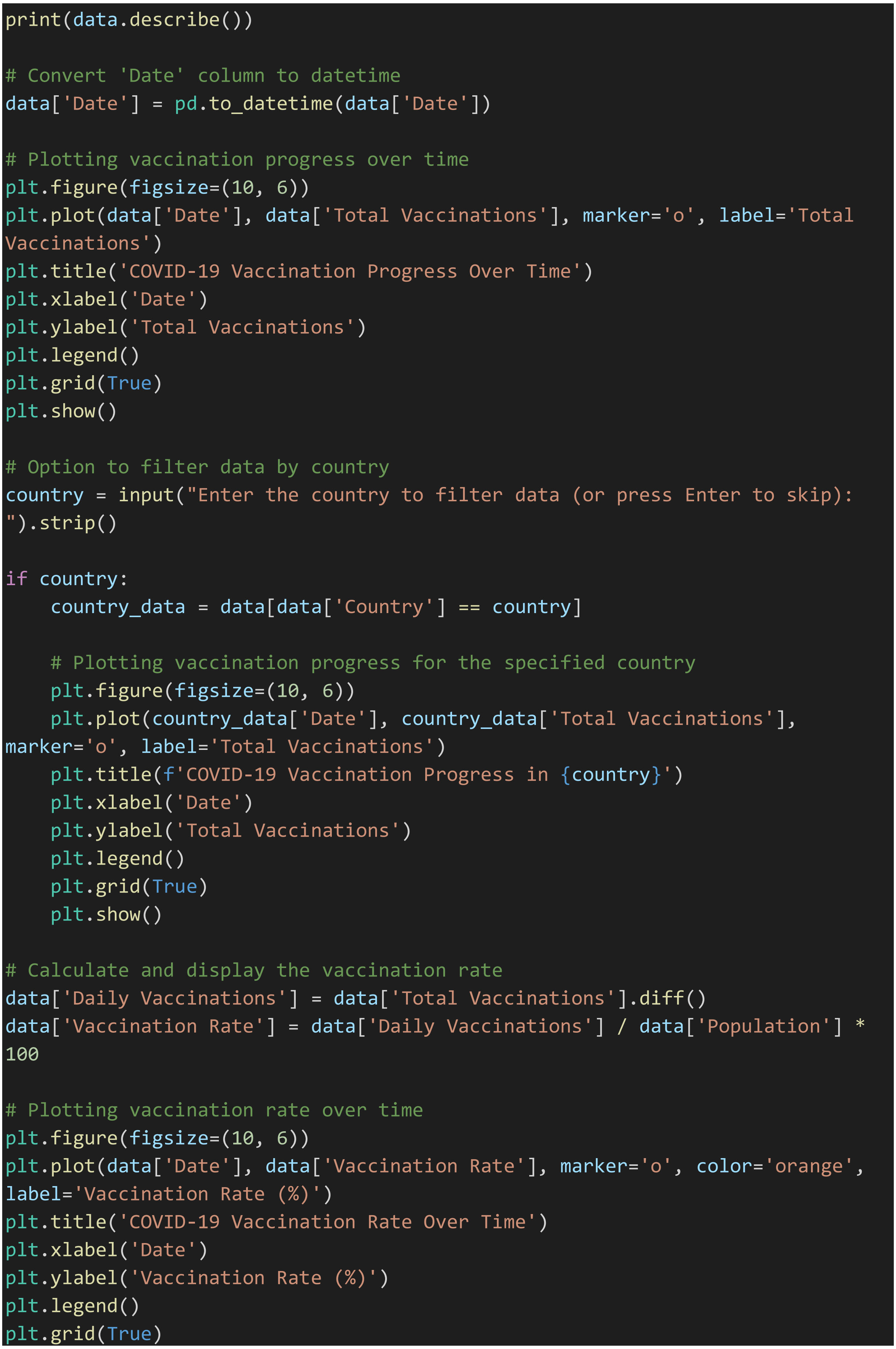
```
# Perform some basic statistics
print("\nBasic Statistics:")
print(data.describe())

# Plotting vaccination progress over time
data['Date'] = pd.to_datetime(data['Date'])
plt.figure(figsize=(10, 6))
plt.plot(data['Date'], data['Total Vaccinations'], marker='o')
plt.title('COVID-19 Vaccination Progress Over Time')
plt.xlabel('Date')
plt.ylabel('Total Vaccinations')
plt.grid(True)
plt.show()
```

OUTPUT



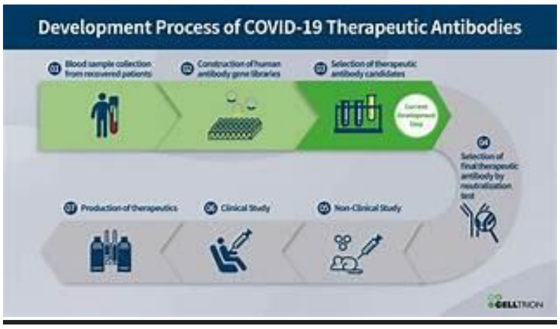
DEVELOPMENT PROGRAM

```
import pandas as pd
import matplotlib.pyplot as plt

# Load the dataset
data = pd.read_csv('covid_vaccine_data.csv')

# Display basic information about the dataset
print("Dataset Information:")
print(data.info())

# Display the first few rows of the dataset
print("\nFirst few rows of the dataset:")
print(data.head())

# Perform some basic statistics
print("\nBasic Statistics:")
```



```python
print(data.describe())

# Convert 'Date' column to datetime
data['Date'] = pd.to_datetime(data['Date'])

# Plotting vaccination progress over time
plt.figure(figsize=(10, 6))
plt.plot(data['Date'], data['Total Vaccinations'], marker='o', label='Total Vaccinations')
plt.title('COVID-19 Vaccination Progress Over Time')
plt.xlabel('Date')
plt.ylabel('Total Vaccinations')
plt.legend()
plt.grid(True)
plt.show()

# Option to filter data by country
country = input("Enter the country to filter data (or press Enter to skip): ").strip()

if country:
    country_data = data[data['Country'] == country]

    # Plotting vaccination progress for the specified country
    plt.figure(figsize=(10, 6))
    plt.plot(country_data['Date'], country_data['Total Vaccinations'], marker='o', label='Total Vaccinations')
    plt.title(f'COVID-19 Vaccination Progress in {country}')
    plt.xlabel('Date')
    plt.ylabel('Total Vaccinations')
    plt.legend()
    plt.grid(True)
    plt.show()

# Calculate and display the vaccination rate
data['Daily Vaccinations'] = data['Total Vaccinations'].diff()
data['Vaccination Rate'] = data['Daily Vaccinations'] / data['Population'] * 100

# Plotting vaccination rate over time
plt.figure(figsize=(10, 6))
plt.plot(data['Date'], data['Vaccination Rate'], marker='o', color='orange', label='Vaccination Rate (%)')
plt.title('COVID-19 Vaccination Rate Over Time')
plt.xlabel('Date')
plt.ylabel('Vaccination Rate (%)')
plt.legend()
plt.grid(True)
plt.show()
```

OUTPUT


Development Process of COVID-19 Therapeutic Antibodies
01 Blood sample collection from recovered patients
02 Construction of human antibody gene libraries
03 Selection of therapeutic antibody candidates
Current development step
04 Selection of final therapeutic antibody by neutralization test
05 Non-Clinical Study
06 Clinical Study
07 Production of therapeutics
CELLTRION

SOME MACHINE LEARNING METHODS

ML algorithms can be divided into supervised or unsupervised learning:

> Supervised ML algorithms is a type of ML technique that can be applied according to what was previously learned to get new data using labeled data and to predict future events or labels. In this type of learning, supervisor (labels) is present to guide or correct. For this first analysis, the known training set and then the output values are predicted using the learning algorithm. The output defined by the learning system can be compared with the actual output; if errors are identified, they can be rectified and the model can be modified accordingly.
>
> Unsupervised ML algorithms: In this type, there is no supervisor to guide or correct. This type of learning algorithm is used when unlabeled or unclassified information is present to train the system. The system does not define the correct output, but it explores the data in such a way that it can draw inferences (rules) from datasets and can describe hidden structures from unlabelled data Semisupervised ML algorithms are algorithms that are between the category of supervised and unsupervised learning. Thus, this type of learning algorithm uses both unlabelled and labelled data for training purposes, generally a small amount of labelled data and a large amount of un labelled data. This type of method is used to improve the accuracy of learning .

> Reinforcement ML algorithms is a type of learning method that gives rewards or punishment on the basis of the work performed by the system. If we train the system to perform a certain task and it fails to do that, the system might be punished; if it performs perfectly, it will be rewarded. It typically works on 0 and 1, in which 0 indicates a punishment and 1 indicates a reward.

It works on the principle in which, if we train a bird or a dog to do
VACCINE

Use of machine learning in COVID-19

ML is used in various fields, including medicine to predict disease and forecast its outcome. In medicine, the right diagnosis and the right time are the keys to successful treatment. If the treatment has a high error rate, it may cause several deaths. Therefore, researchers have started using artificial intelligence applications for medical treatment. The task is complicated because the researchers have to choose the right tool: it is a matter of life or death.

For this task, ML achieved a milestone in the field of health care. ML techniques are used to interpret and analyze large datasets and predict their output. These ML tools were used to identify the symptoms of disease and classify samples into treatment groups. ML helps hospitals to maintain administrative processes and treat infectious disease.

ML techniques were previously used to treat cancer, pneumonia, diabetes, Parkinson disease, arthritis, neuromuscular disorders, and many more diseases; they give more than 90% accurate results in prediction and forecasting.

The pandemic disease known as COVID-19 is a deadly virus that has cost the lives of many people all over the world. There is no treatment for this virus. ML techniques have been used to predict whether patients are infected by the virus based on symptoms defined by WHO and CDC .

ML is also used to diagnose the disease based on x-ray images. For instance, chest images of patients can be used to detect whether a patient is infected with COVID-19 .

Moreover, social distancing can be monitored by ML; with the help of this approach, we can keep ourselves safe from COVID-19 .

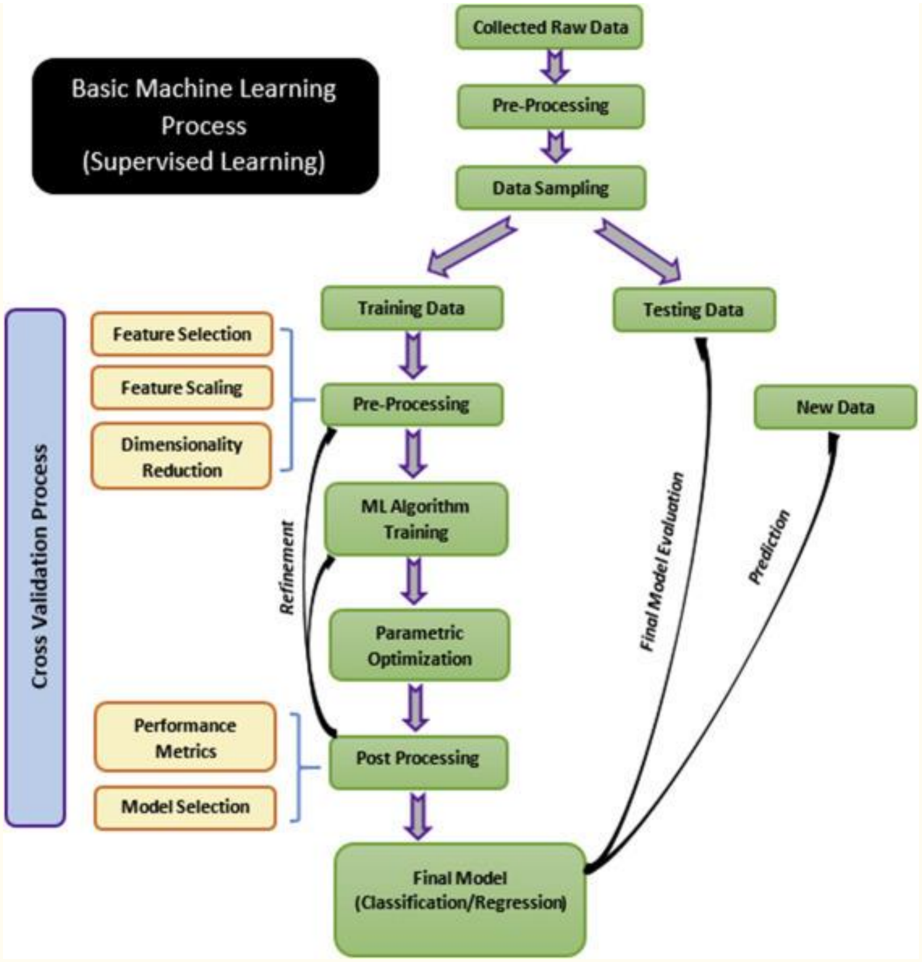
Different techniques for prediction and forecasting

Various ML techniques are used to predict and forecast future events. Some ML techniques used for prediction are support vector machine, linear regression, logistic regression, naive Bayes, decision trees (random forest and ETC), K-nearest neighbor, and neural networks (multilayer perceptron) .

Similarly, some ML techniques used to forecast future events are naive approach, moving average, simple exponential smoothing, Holt's linear trend model, Holt-Winters model, Seasonal Autoregressive Integrated Moving Average Exxogenous Model (SARIMAX) and Autoregressive Integrated Moving Average Model (ARIMA).

Each technique has unique features and is used differently based on the accuracy results. The model with the best accuracy during the model evaluation process is chosen for prediction or forecasting. In the same way, we identified and used the ETC for the symptom-based prediction of COVID-19 and the ARIMA forecasting model to forecast the number of confirmed cases of COVID-19 in India, because they had the best accuracy results among all classifier and forecasting methods we used when we evaluated model performance.

It defines how data are collected and preprocessed, and then are divided into a training dataset and test dataset for training and performance evaluation.



Proposed method for prediction

A symptom-based predictive model was proposed to predict COVID-19 based on symptoms defined by the WHO and CDC.

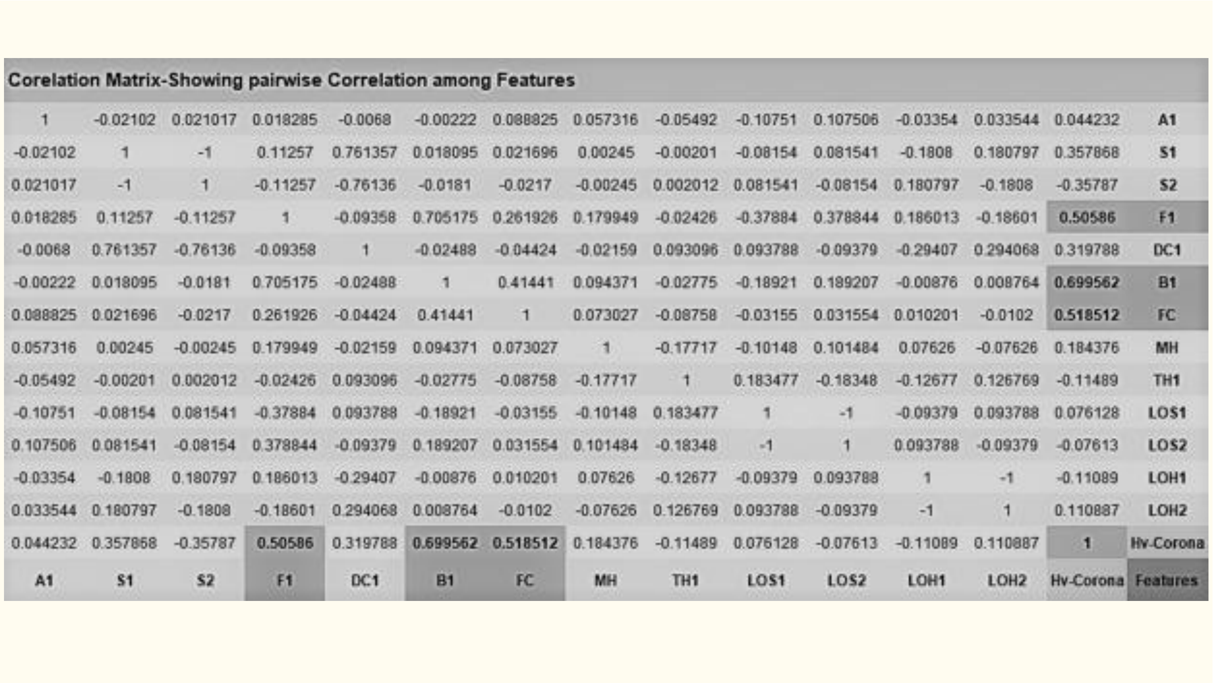
Because there is no proper description of symptoms declared by the WHO, based on some existing symptoms, we defined a model used to predict the disease according to the accuracy given by the model.

We created a symptom database in which rules were created and used as input. Then, these data were used as raw data. Then, feature selection took place as part of preprocessing data. The data were divided into training data (80% of data) and test data (20% of data), usually known as the train-test split process. This split is generally done in a stratified or random manner so that population distribution in both groups consists of shuffled data, which leads minimized bias or skewness in the data. Training data were used to train the ML classifier that we used in the model, and test data were used to test that classifier in terms of accuracy received over a predefined unseen portion of the dataset.

In our work, the symptoms and patient's class dataset was defined on the basis of symptoms such as fever, cough, and sneezing, whether the patient had traveled to an infected place, age, and whether the patient had a history of disease that could increase the possibly of being infected by the virus.

This dataset was then further divided into two sets (training set and testing set) using the test-train split method. The system was trained on the basis of training set data and the accuracy of the ML classifier, and then evaluated over the testing set. Finally, the model was used to predict the probability of infection from the disease using new patient data in terms of positive or negative.

A correlation matrix, which is a tool for the feature selection process, is table used to define correlation coefficients among variables or features. Every cell in the matrix defines a correlation between two variables. It is used to summarize a large dataset and also to identify the most highly correlated features.



Corelation Matrix-Showing pairwise Correlation among Features

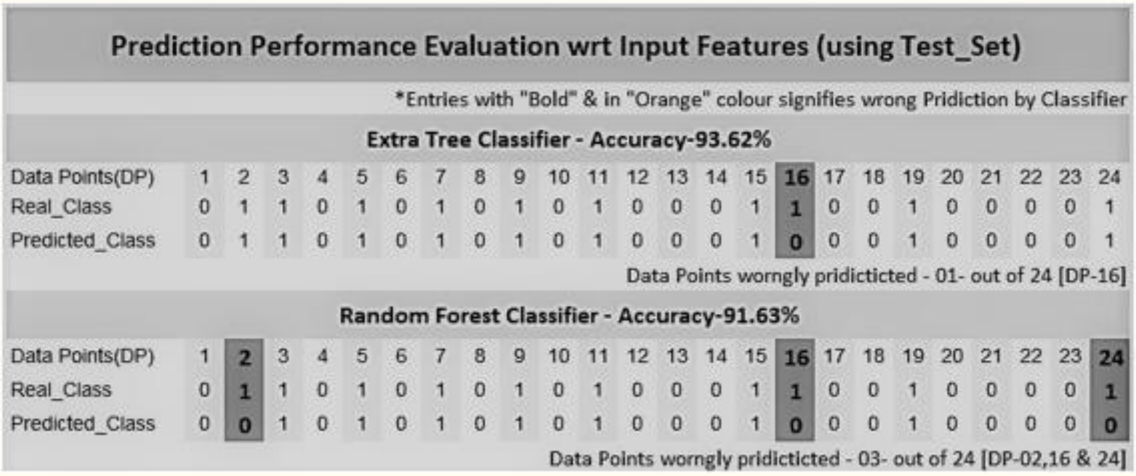
1	-0.02102	0.021017	0.018285	-0.0068	-0.00222	0.088825	0.057316	-0.05492	-0.10751	0.107506	-0.03354	0.033544	0.044232	A1
-0.02102	1	-1	0.11257	0.761357	0.018095	0.021696	0.00245	-0.00201	-0.08154	0.081541	-0.1808	0.180797	0.357868	S1
0.021017	-1	1	-0.11257	-0.76136	-0.0181	-0.0217	-0.00245	0.002012	0.081541	-0.08154	0.180797	-0.1808	-0.35787	S2
0.018285	0.11257	-0.11257	1	-0.09358	0.705175	0.261926	0.179949	-0.02426	-0.37884	0.378844	0.186013	-0.18601	0.50586	F1
-0.0068	0.761357	-0.76136	-0.09358	1	-0.02488	-0.04424	-0.02159	0.093096	0.093788	-0.09379	-0.29407	0.294068	0.319788	DC1
-0.00222	0.018095	-0.0181	0.705175	-0.02488	1	0.41441	0.094371	-0.02775	-0.18921	0.189207	-0.00876	0.008764	0.699562	B1
0.088825	0.021696	-0.0217	0.261926	-0.04424	0.41441	1	0.073027	-0.08758	-0.03155	0.031554	0.010201	-0.0102	0.518512	FC
0.057316	0.00245	-0.00245	0.179949	-0.02159	0.094371	0.073027	1	-0.17717	-0.10148	0.101484	0.07626	-0.07626	0.184376	MH
-0.05492	-0.00201	0.002012	-0.02426	0.093096	-0.02775	-0.08758	-0.17717	1	0.183477	-0.18348	-0.12677	0.126769	-0.11489	TH1
-0.10751	-0.08154	0.081541	-0.37884	0.093788	-0.18921	-0.03155	-0.10148	0.183477	1	-1	-0.09379	0.093788	0.076128	LOS1
0.107506	0.081541	-0.08154	0.378844	-0.09379	0.189207	0.031554	0.101484	-0.18348	-1	1	0.093788	-0.09379	-0.07613	LOS2
-0.03354	-0.1808	0.180797	0.186013	-0.29407	-0.00876	0.010201	0.07626	-0.12677	-0.09379	0.093788	1	-1	-0.11089	LOH1
0.033544	0.180797	-0.1808	-0.18601	0.294068	0.008764	-0.0102	-0.07626	0.126769	0.093788	-0.09379	-1	1	0.110887	LOH2
0.044232	0.357868	-0.35787	0.50586	0.319788	0.699562	0.518512	0.184376	-0.11489	0.076128	-0.07613	-0.11089	0.110887	1	Hv-Corona
A1	S1	S2	F1	DC1	B1	FC	MH	TH1	LOS1	LOS2	LOH1	LOH2	Hv-Corona	Features

The correlation coefficient's value near 1 signifies that features participating in correlation are highly correlated to each other; on the other hand, the correlation coefficient's value near 0 signifies that features are less correlated to each other. Generally, correlation could be of two types: positive and negative. A positive correlation states that an increase or decrease in one feature's value results in an increase or decrease in the other feature's value; in contrast, a negative correlation has a reverse relation between the two features, so an increase in one feature's value results in the decreased value of the other feature.

Rows and columns in the correlation matrix represent each feature's name. Each cell in a table containing the correlation coefficient calculated between features corresponds to the respective row and column of that particular cell it .

It another form of representation of a correlation matrix using a heat map. Heat maps are a popular way to visualize the interrelation between two or more variables or features, because it is easy for the human mind to distinguish between an attribute's ranks by visualizing color coding rather than checking and searching for the best value in a given list of numerical values, as One can easily identify and choose the most correlated feature using heat map visualization, in which the light-colored cell defines the most correlated features and the dark-colored cell defines the least correlated features.

the prediction performance based on two classifiers, random forest and ETC. The ETC gives one wrong prediction (dark gray colored column) out of 14 data points and the random forest classifier (RFC) gives three wrong predictions out of 24 data points.



Prediction Performance Evaluation wrt Input Features (using Test_Set)

*Entries with "Bold" & in "Orange" colour signifies wrong Pridiction by Classifier

Extra Tree Classifier - Accuracy-93.62%

Data Points(DP)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	**16**	17	18	19	20	21	22	23	24
Real_Class	0	1	1	0	1	0	1	0	1	0	1	0	0	0	1	**1**	0	0	1	0	0	0	0	1
Predicted_Class	0	1	1	0	1	0	1	0	1	0	1	0	0	0	1	**0**	0	0	1	0	0	0	0	1

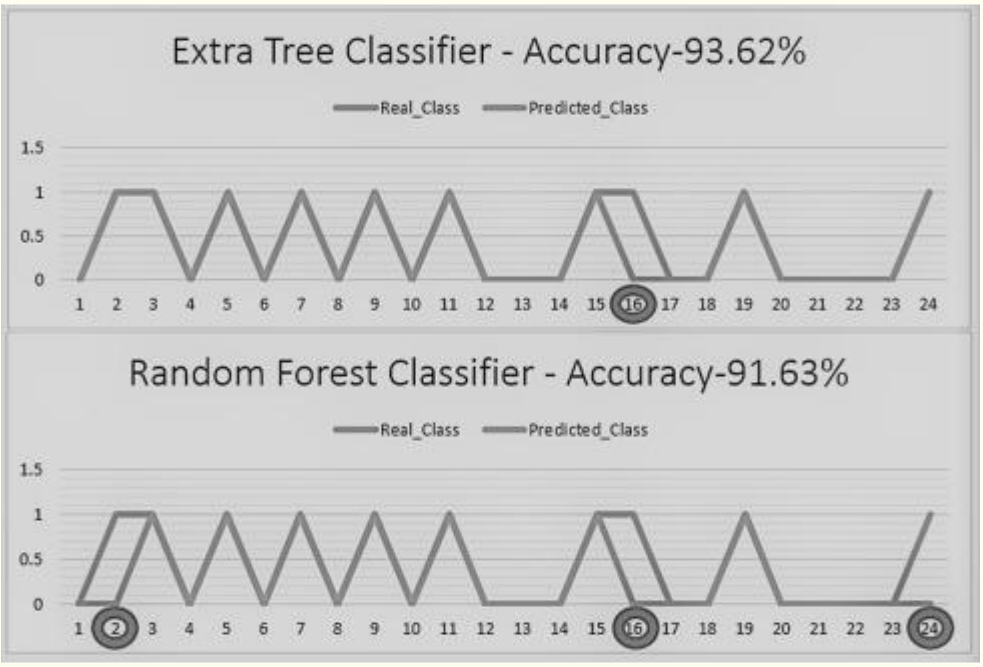
Data Points worngly pridicticted - 01- out of 24 [DP-16]

Random Forest Classifier - Accuracy-91.63%

Data Points(DP)	1	**2**	3	4	5	6	7	8	9	10	11	12	13	14	15	**16**	17	18	19	20	21	22	23	**24**
Real_Class	0	**1**	1	0	1	0	1	0	1	0	1	0	0	0	1	**1**	0	0	1	0	0	0	0	**1**
Predicted_Class	0	**0**	1	0	1	0	1	0	1	0	1	0	0	0	1	**0**	0	0	1	0	0	0	0	**0**

Data Points worngly pridictícted - 03- out of 24 [DP-02,16 & 24]

It compares two classifier outputs using line graphs: ETC and RFC. The figure shows 24 data points, on the basis of which the accuracy of these classifiers is described.



The ETC misclassified at point 16, whereas the RFC misclassified at three points: 2, 16, and 24. This means the ETC is more accurate than the RFC. This comparison is shown for synthetic data. thus, if real data are used, based on those data, training of the classifier is performed and then the classifier is tested for accuracy. The classifier that gives the best accuracy can be used for prediction.

Forecasting

For forecasting through ML, time series analysis may be used, which is an important part of ML. It is a univariate type of regression in which the target feature (dependent feature) is forecast using only one input feature (independent feature), which is time.

It is used to forecast future event values, it has an important role for forecasting the existence of respiratory diseases such as COVID-19. Positive cases are increasing daily, so it is necessary to forecast whether the ratio by which the number is increasing is continuing based on prior observations. It is helpful for the government, because based on the forecast, it can plan for resources to control the spread of disease and act for the future so that the growth rate of the infection decreases without affecting more people.

Forecasts depend completely on past trends, so forecast values cannot be guaranteed. However, this forecasted approximation of events may help authorities to assess forthcoming resource planning to compete with any pandemic situation such as

DATA VISUALIAZATION OF COVID 19

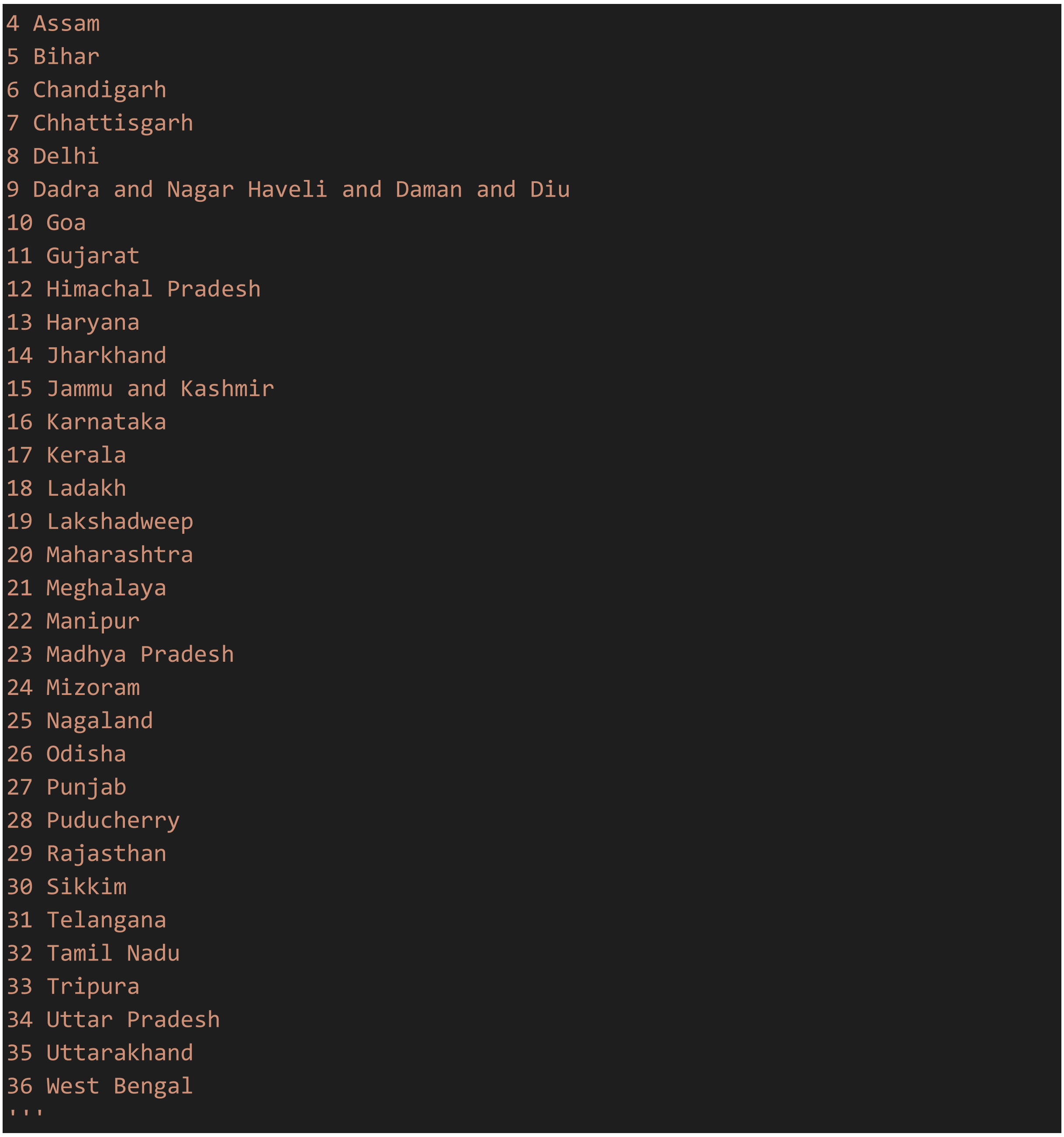
```
#importing modules
import json
import requests
import pandas as pd
import matplotlib.pyplot as plt
```

Function for getting JSON data from API and Visualization of Data

```
#storing the url in the form of string
url="https://api.covid19india.org/state_district_wise.json"

#function to get data from api
def casesData():
    #getting the json data by calling api
    data = ((requests.get(url)).json())
    states = []
```

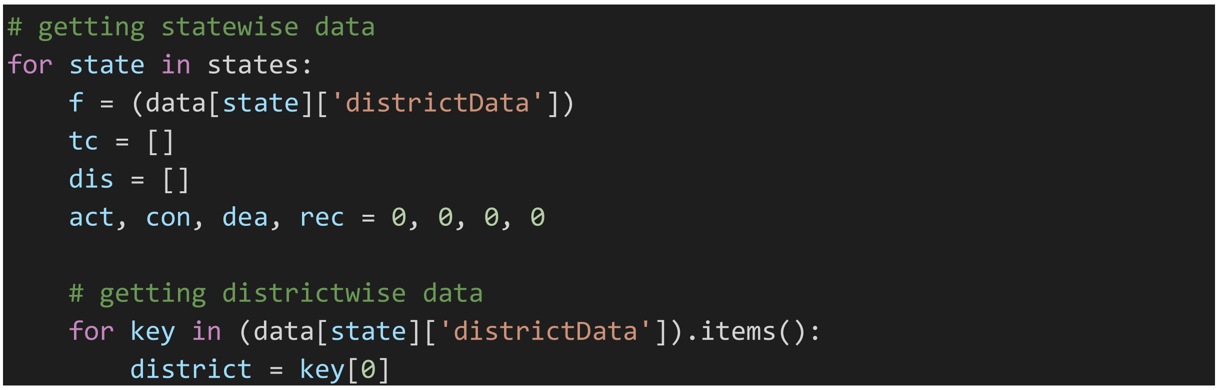
Getting State Names available in JSON Data

```
# getting statewise data
for state in states:
        f = (data[state]['districtData'])
# states data available in JSON Data
'''
0 State Unassigned
1 Andaman and Nicobar Islands
2 Andhra Pradesh
3 Arunachal Pradesh
```



```
4 Assam
5 Bihar
6 Chandigarh
7 Chhattisgarh
8 Delhi
9 Dadra and Nagar Haveli and Daman and Diu
10 Goa
11 Gujarat
12 Himachal Pradesh
13 Haryana
14 Jharkhand
15 Jammu and Kashmir
16 Karnataka
17 Kerala
18 Ladakh
19 Lakshadweep
20 Maharashtra
21 Meghalaya
22 Manipur
23 Madhya Pradesh
24 Mizoram
25 Nagaland
26 Odisha
27 Punjab
28 Puducherry
29 Rajasthan
30 Sikkim
31 Telangana
32 Tamil Nadu
33 Tripura
34 Uttar Pradesh
35 Uttarakhand
36 West Bengal
'''
```

Getting the state-wise Data



```python
# getting statewise data
for state in states:
    f = (data[state]['districtData'])
    tc = []
    dis = []
    act, con, dea, rec = 0, 0, 0, 0

    # getting districtwise data
    for key in (data[state]['districtData']).items():
        district = key[0]
        dis.append(district)
        active = data[state]['districtData'][district]['active']
        confirmed = data[state]['districtData'][district]['confirmed']
        deaths = data[state]['districtData'][district]['deceased']
        recovered = data[state]['districtData'][district]['recovered']
        if district == 'Unknown':
            active, confirmed, deaths, recovered = 0, 0, 0, 0
        tc.append([active, confirmed, deaths, recovered])
        act = act + active
        con = con + confirmed
        dea = dea + deaths
        rec = rec + recovered
    tc.append([act, con, dea, rec])
    dis.append('Total')
    parameters = ['Active', 'Confirmed', 'Deaths', 'Recovered']
```

CREATE A DATAFRAME USING PANDAS

```
# creating a dataframe
df = pd.DataFrame(tc, dis, parameters)
print('COVID - 19', state, 'District Wise Data')
print(df)
```

DATA VISUALIAZATION USING MATPLOTLIB

```
# plotting of data
plt.bar(dis, df['Active'], width=0.5, align='center')
fig = plt.gcf()
fig.set_size_inches(18.5, 10.5)
plt.xticks(rotation=75)
plt.show()
print('*'*100)
```

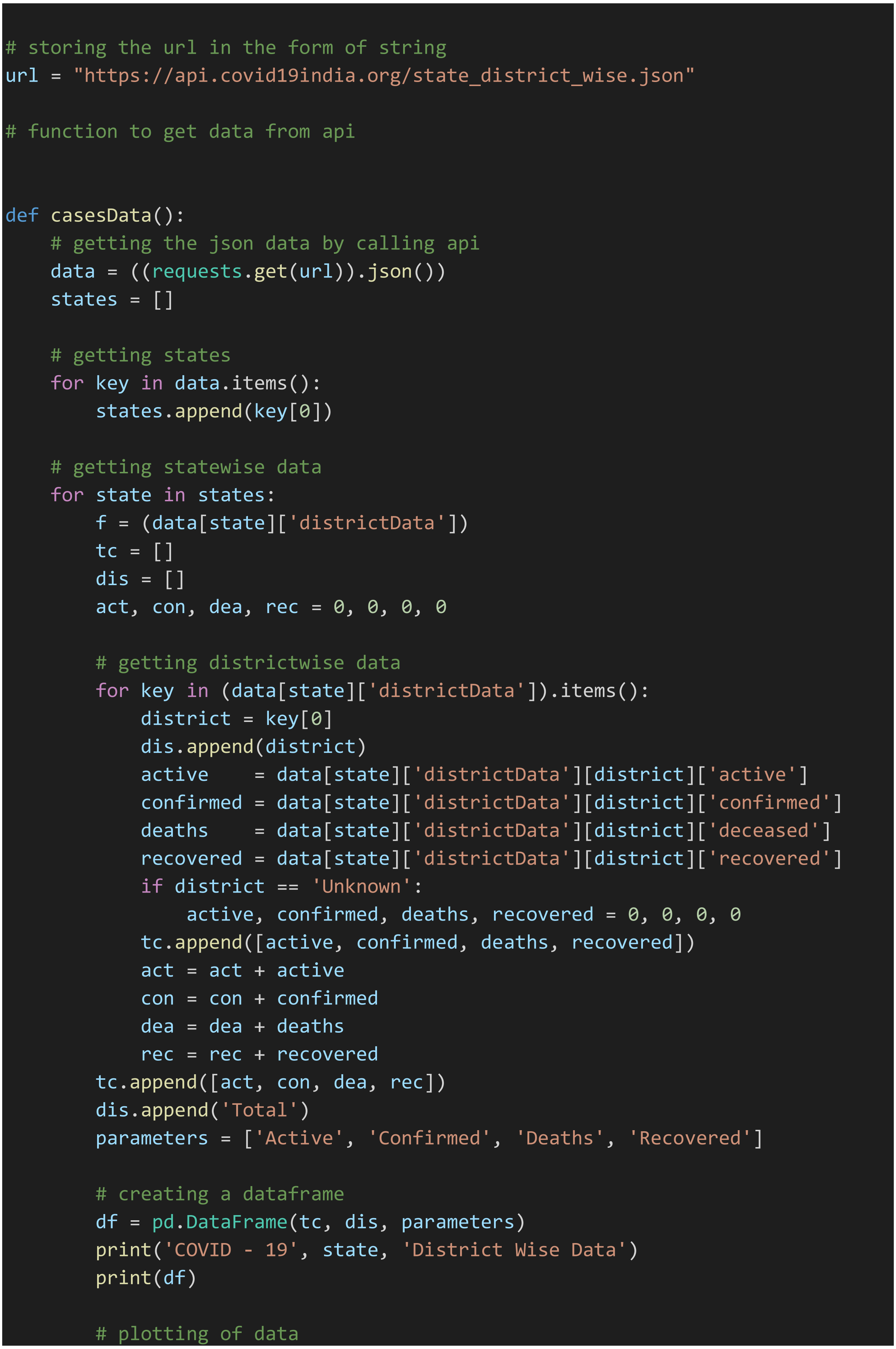
FINAL CASES() FUNCTION CODE

```
# function to get data from api
def casesData():
    # getting the json data by calling api
    data = ((requests.get(url)).json())
    states = []

    # getting states
    for key in data.items():
        states.append(key[0])
```

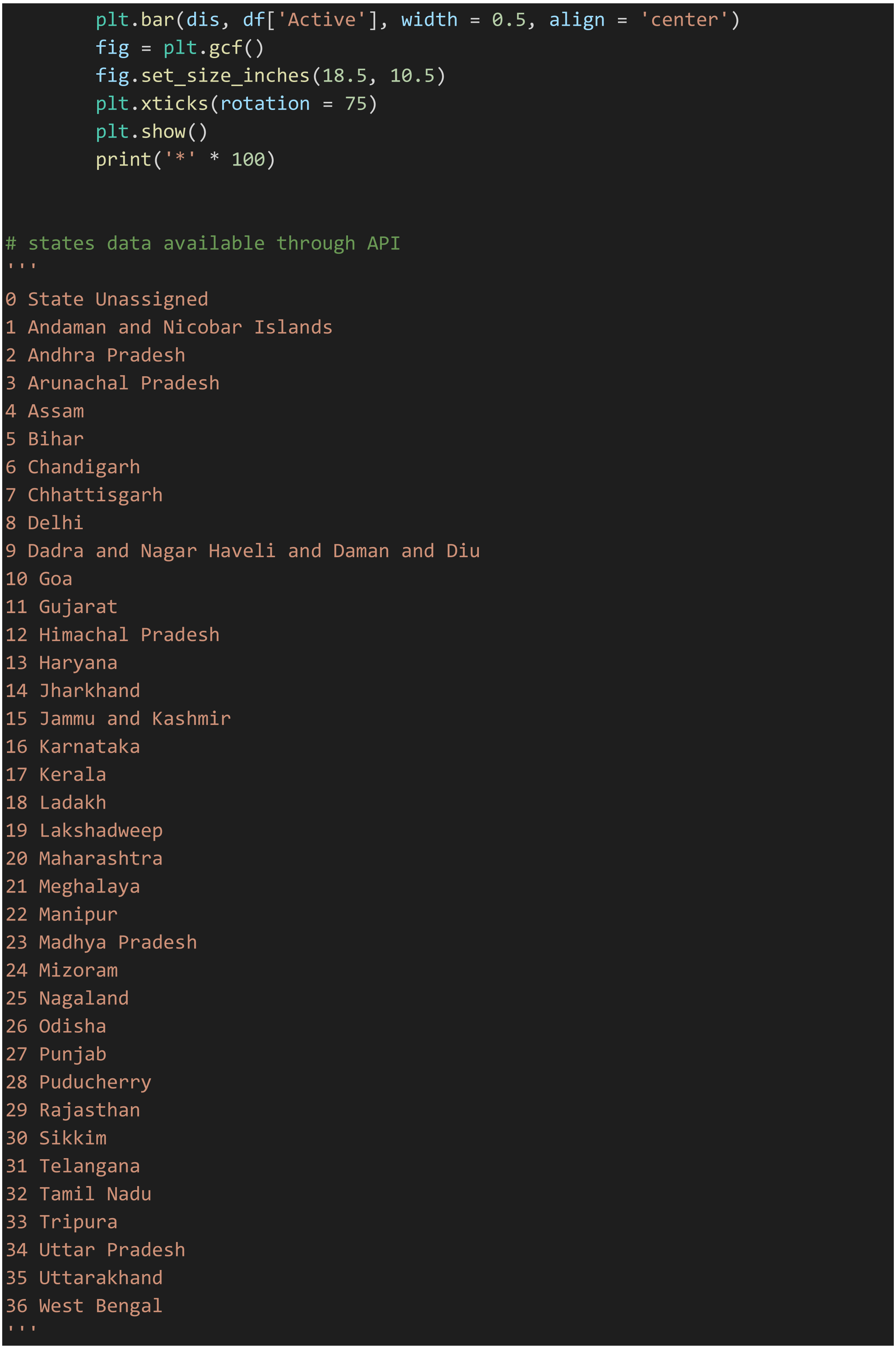
```python

    # getting statewise data
    for state in states:
        f = (data[state]['districtData'])
        tc = []
        dis = []
        act, con, dea, rec = 0, 0, 0, 0

        # getting districtwise data
        for key in (data[state]['districtData']).items():
            district = key[0]
            dis.append(district)
            active = data[state]['districtData'][district]['active']
            confirmed = data[state]['districtData'][district]['confirmed']
            deaths = data[state]['districtData'][district]['deceased']
            recovered = data[state]['districtData'][district]['recovered']
            if district == 'Unknown':
                active, confirmed, deaths, recovered = 0, 0, 0, 0
            tc.append([active, confirmed, deaths, recovered])
            act = act + active
            con = con + confirmed
            dea = dea + deaths
            rec = rec + recovered
        tc.append([act, con, dea, rec])
        dis.append('Total')
        parameters = ['Active', 'Confirmed', 'Deaths', 'Recovered']

        # creating a dataframe
        df = pd.DataFrame(tc, dis, parameters)
        print('COVID - 19', state, 'District Wise Data')
        print(df)

        # plotting of data
        plt.bar(dis, df['Active'], width=0.5, align='center')
        fig = plt.gcf()
        fig.set_size_inches(18.5, 10.5)
        plt.xticks(rotation = 75)
        plt.show()
        print('*' * 100)
```

FINAL IMPLEMENTATION

```python
# importing modules
import json
import requests
import pandas as pd
import matplotlib.pyplot as plt
```



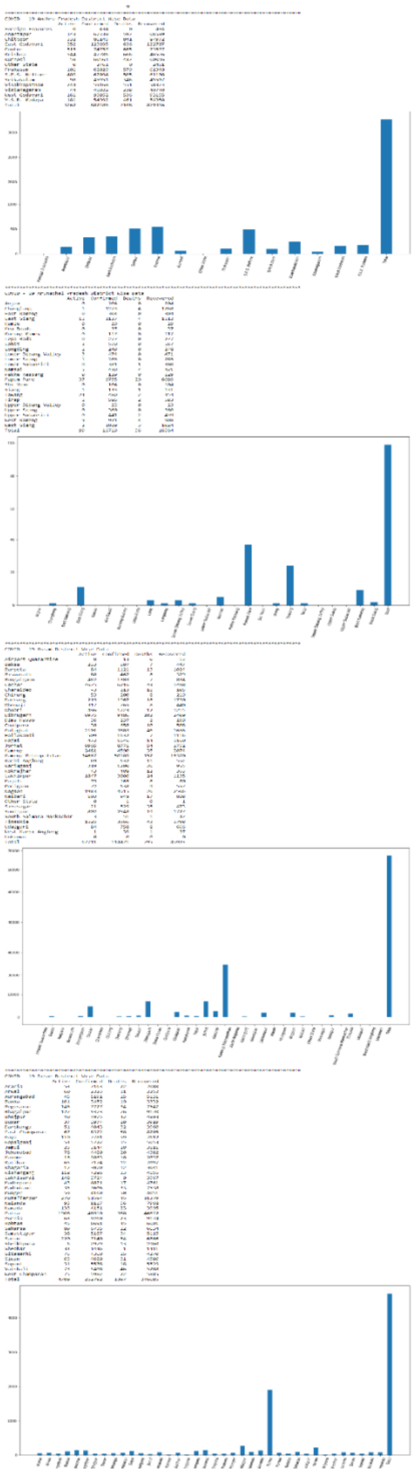
```
# storing the url in the form of string
url = "https://api.covid19india.org/state_district_wise.json"

# function to get data from api


def casesData():
    # getting the json data by calling api
    data = ((requests.get(url)).json())
    states = []

    # getting states
    for key in data.items():
        states.append(key[0])

    # getting statewise data
    for state in states:
        f = (data[state]['districtData'])
        tc = []
        dis = []
        act, con, dea, rec = 0, 0, 0, 0

        # getting districtwise data
        for key in (data[state]['districtData']).items():
            district = key[0]
            dis.append(district)
            active    = data[state]['districtData'][district]['active']
            confirmed = data[state]['districtData'][district]['confirmed']
            deaths    = data[state]['districtData'][district]['deceased']
            recovered = data[state]['districtData'][district]['recovered']
            if district == 'Unknown':
                active, confirmed, deaths, recovered = 0, 0, 0, 0
            tc.append([active, confirmed, deaths, recovered])
            act = act + active
            con = con + confirmed
            dea = dea + deaths
            rec = rec + recovered
        tc.append([act, con, dea, rec])
        dis.append('Total')
        parameters = ['Active', 'Confirmed', 'Deaths', 'Recovered']

        # creating a dataframe
        df = pd.DataFrame(tc, dis, parameters)
        print('COVID - 19', state, 'District Wise Data')
        print(df)

        # plotting of data
```



```
        plt.bar(dis, df['Active'], width = 0.5, align = 'center')
        fig = plt.gcf()
        fig.set_size_inches(18.5, 10.5)
        plt.xticks(rotation = 75)
        plt.show()
        print('*' * 100)


# states data available through API
'''
0 State Unassigned
1 Andaman and Nicobar Islands
2 Andhra Pradesh
3 Arunachal Pradesh
4 Assam
5 Bihar
6 Chandigarh
7 Chhattisgarh
8 Delhi
9 Dadra and Nagar Haveli and Daman and Diu
10 Goa
11 Gujarat
12 Himachal Pradesh
13 Haryana
14 Jharkhand
15 Jammu and Kashmir
16 Karnataka
17 Kerala
18 Ladakh
19 Lakshadweep
20 Maharashtra
21 Meghalaya
22 Manipur
23 Madhya Pradesh
24 Mizoram
25 Nagaland
26 Odisha
27 Punjab
28 Puducherry
29 Rajasthan
30 Sikkim
31 Telangana
32 Tamil Nadu
33 Tripura
34 Uttar Pradesh
35 Uttarakhand
36 West Bengal
'''
```

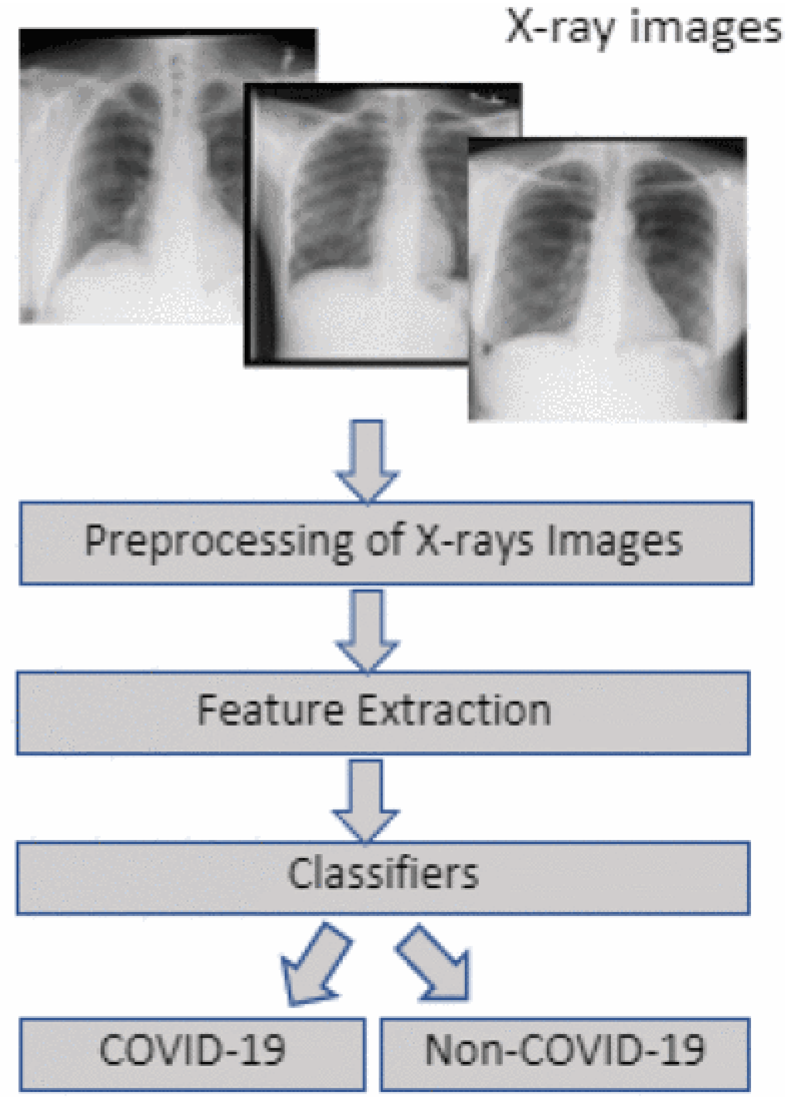
```
#Driver Code
casesData()
```

OUTPUT



METHODOLOGY

The schematic block diagram of the proposed system for identifying COVID-19 is shown in Fig. 1. It has mainly three steps namely, preprocessing, feature extraction, and classification. All the chest X-rays images of the COVIDx dataset are not in the same size. Thus, all the images are resized to 224 × 224 using bi-linear interpolation in the preprocessing step. In the second step, all the above-mentioned feature extraction techniques in Section II are employed to fetch features from the resized images. Finally, extracted features are fed into the above-stated classifiers separately in order to identify COVID-19 patients.



Recommendation:

The Federal Ministry of Health should have a central database that is updated daily with the accurate figures. This will go a long way in tracking both the vaccination progress and the new cases of the virus and will give a clue on how to mitigate the spread in the country.

Conclusion:

The aim of this analysis is to answer the following questions.

1. Vaccines used around the world
2. Situation in Africa
3. Situation in Nigeria.

And also to strengthen my skills on data analysis using visualizations.