PHASE 5

PROJECT OVERVIEW:

"In the wake of the global COVID-19 pandemic, our data analytics project using IBM Cognos aims to provide valuable insights into the spread and impact of the virus. Leveraging a diverse dataset that includes infection rates, vaccination data, hospitalizations, and demographic information, we seek to uncover trends, patterns, and correlations that can inform public health decisions and policy recommendations. Our project will involve data cleaning, visualization, and advanced analytics techniques to produce meaningful reports and dashboards that not only track the progression of the pandemic but also facilitate data-driven decision-making for healthcare authorities and policymakers. Through this project, we aspire to contribute to a better understanding of the COVID-19 crisis and support efforts to mitigate its effects on society."

DESIGN THINKING:

ANALYSIS OBJECTIVES:

1. Comparing Mean Values and Standard Deviations:

- Utilize IBM Cognos to calculate the mean (average) number of COVID-19 cases and deaths over a specified time period. This provides a central measure of the data distribution.
- Calculate the standard deviation using Cognos to measure the spread or dispersion of the data points. A higher standard deviation indicates greater variability in cases and deaths.

2. Identifying Trends and Outliers:

- Create line charts or time series visualizations in Cognos to identify trends and fluctuations in COVID-19 cases and deaths over time. This can help in spotting surges or declines in specific periods.
- Use Cognos to identify outliers or anomalies in the data, such as unusually high or low numbers of cases and deaths. This can be crucial for detecting potential data reporting errors or significant events.

3. Comparing Regional or Demographic Data:

- Break down the analysis by regions or demographics using Cognos' filtering and grouping capabilities. Compare mean values and standard deviations across different geographic areas or demographic groups to identify disparities.
- Create bar charts or heat maps in Cognos to visualize these variations, making it easier to understand and communicate the differences in COVID-19 statistics.

4. Forecasting and Predictive Modelling:

- Utilize Cognos for time series forecasting and predictive modelling to estimate future COVID-19 cases and deaths based on historical data. This can help in resource planning and decision-making.
- Evaluate the accuracy of your forecasts by comparing them with actual data over time, allowing for adjustments to response strategies.

Data Collection Considerations:

- Regular Data Updates: COVID-19 data evolves rapidly. It's crucial to collect and update data frequently to reflect the latest information.
- Data Consistency: Ensure that data sources follow consistent reporting formats. Inconsistencies can lead to errors in analysis.
- Data Completeness: Monitor data for missing values and seek ways to address or impute missing data if necessary.
- Data Transparency: The project should maintain transparency about the sources, methodologies, and processes used for data collection.

Data quality assurance:

- Data Validation: Implement validation checks to identify outliers or erroneous data points that may skew analysis results.
- Data Cleaning: Conduct data cleaning processes to remove duplicates, address missing values, and correct data anomalies.
- Data Integrity: Maintain data integrity by verifying that data collected is accurate, reliable, and representative of the pandemic's actual situation.
- Ethical Considerations: Adhere to legal and ethical guidelines regarding data privacy and confidentiality, particularly when dealing with sensitive health data.

Documentation:

Document data sources, collection procedures, and any data cleaning or preprocessing steps undertaken. Comprehensive documentation ensures the transparency and reproducibility of the analysis.

Data Quality is Fundamental: Data quality is critical for the success of a COVID-19 analysis project. High-quality data ensures that the insights and conclusions drawn from the analysis are meaningful and reliable. Regular updates, validation checks, and thorough documentation are key practices to maintain data quality throughout the project.

By following these data collection and quality principles, a data analyst can lay a strong foundation for their COVID-19 analysis project, ultimately leading to more accurate and informative results.

Here we are using the covid 19 cases dataset

Step 1: Data Preprocessing:

Data Collection:

Start by gathering the data you want to analyze. Ensure that your dataset is in a suitable format for analysis, such as CSV, Excel, or a supported database.Here we are using the COVID 19 cases dataset which is in .csv format.

Firstly we have to upload the dataset.

.

CC

```
from google.colab import files

uploaded = files.upload()
```

```
Choose files  Covid_19.csv
• Covid_19.csv(text/csv) - 101231 bytes, last modified: 19/10/2023 - 100% done
Saving Covid_19.csv to Covid_19.csv
```

One of the key steps in the data analysis process is data preprocessing. This involves cleaning, transforming, and preparing the data for analysis. Here's an example of data preprocessing in Python using a sample dataset and some common libraries like Pandas and NumPy:
python
Copy code
# Import necessary libraries
import pandas as pd
import numpy as np

# Load your dataset (replace 'your_dataset.csv' with your data file)
data = pd.read_csv('your_dataset.csv')

# Explore the first few rows of the dataset
print(data.head())

```
[6] df=pd.read_csv("Covid_19.csv")
    df.head()
```

	dateRep	day	month	year	cases	deaths	countriesAndTerritories
0	31-05-2021	31	5	2021	366	5	Austria
1	30-05-2021	30	5	2021	570	6	Austria
2	29-05-2021	29	5	2021	538	11	Austria
3	28-05-2021	28	5	2021	639	4	Austria
4	27-05-2021	27	5	2021	405	19	Austria

```
[7] df.columns

Index(['dateRep', 'day', 'month', 'year', 'cases', 'deaths',
       'countriesAndTerritories'],
      dtype='object')
```

```
[8] df.shape

(2730, 7)
```

```
[9] df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2730 entries, 0 to 2729
Data columns (total 7 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   dateRep                 2730 non-null   object
 1   day                     2730 non-null   int64
 2   month                   2730 non-null   int64
 3   year                    2730 non-null   int64
 4   cases                   2730 non-null   int64
 5   deaths                  2730 non-null   int64
 6   countriesAndTerritories 2730 non-null   object
dtypes: int64(5), object(2)
memory usage: 149.4+ KB
```

# Check for missing values
print(data.isnull().sum())

```
[10] df.isnull().sum()

dateRep                   0
day                       0
month                     0
year                      0
cases                     0
deaths                    0
countriesAndTerritories   0
dtype: int64
```

```
df.describe()
```

	day	month	year	cases	deaths
count	2730.000000	2730.000000	2730.0	2730.000000	2730.000000
mean	16.000000	4.010989	2021.0	3661.010989	65.291941
std	8.765919	0.818813	0.0	6490.510073	113.956634
min	1.000000	3.000000	2021.0	-2001.000000	-3.000000
25%	8.000000	3.000000	2021.0	361.250000	2.000000
50%	16.000000	4.000000	2021.0	926.500000	14.500000
75%	24.000000	5.000000	2021.0	3916.250000	72.000000
max	31.000000	5.000000	2021.0	53843.000000	956.000000

STEP 2: Data Cleaning

Perform data cleaning to handle missing values, remove duplicates, and address any inconsistencies or errors in the data. This can be done using data preparation tools in Cognos or external tools like Excel or Python.

Handle missing values (if any)
Example: Replace missing values in a column with the mean of that column
data['column_name_with_missing_values'].fillna(data['column_name_with_missing_values'].mean(), inplace=True)

```
df.fillna(df.mean(), inplace=True)
```

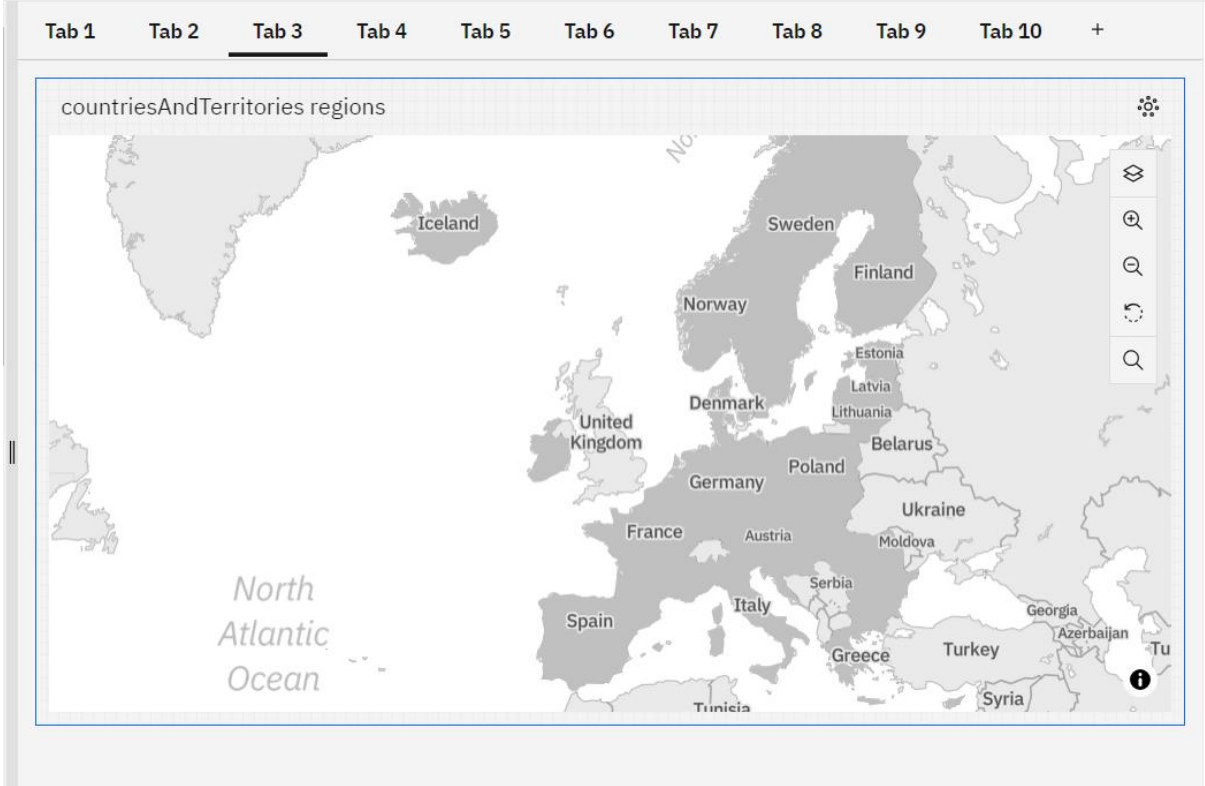
# Remove duplicates (if any)
data.drop_duplicates(inplace=True)

```
data.drop_duplicates(inplace=True)
```

Step 3: Data Visualization:

VISUALIZATION BY USING IBM COGNOS ANALYTICS:

Create Dashboards: In Cognos, you can design interactive dashboards that allow users to interact with data. Dashboards often include various visualizations, such as charts, graphs, and maps.
Here by using IBM COGNOS ANALYTICS we can visualize the map of COVID 19 death cases in the Europian countries.

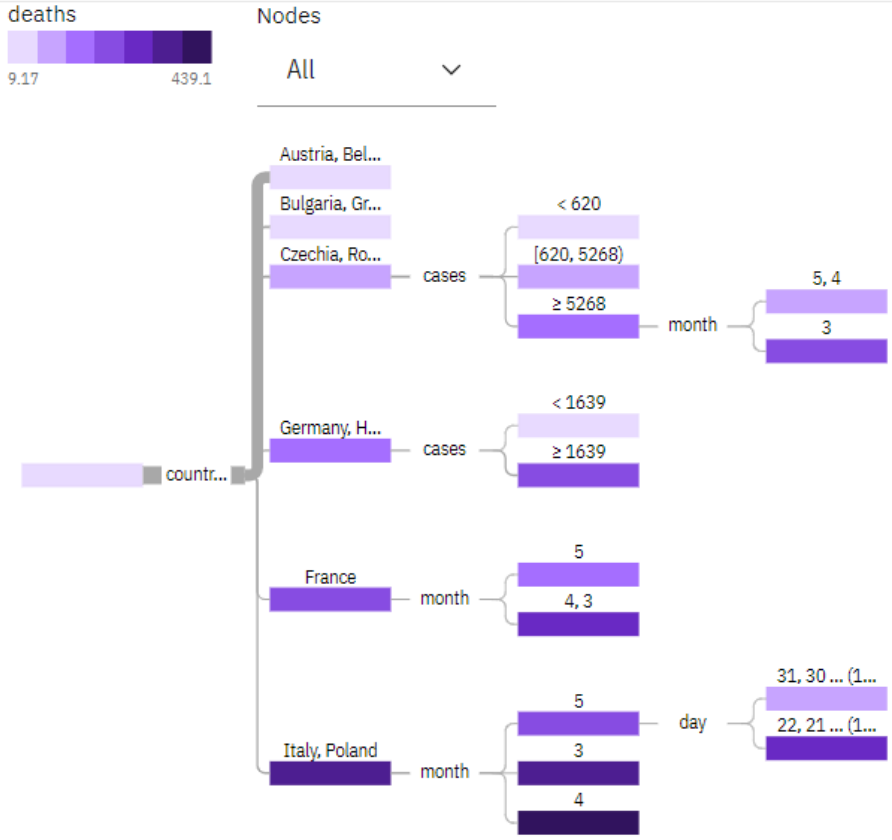


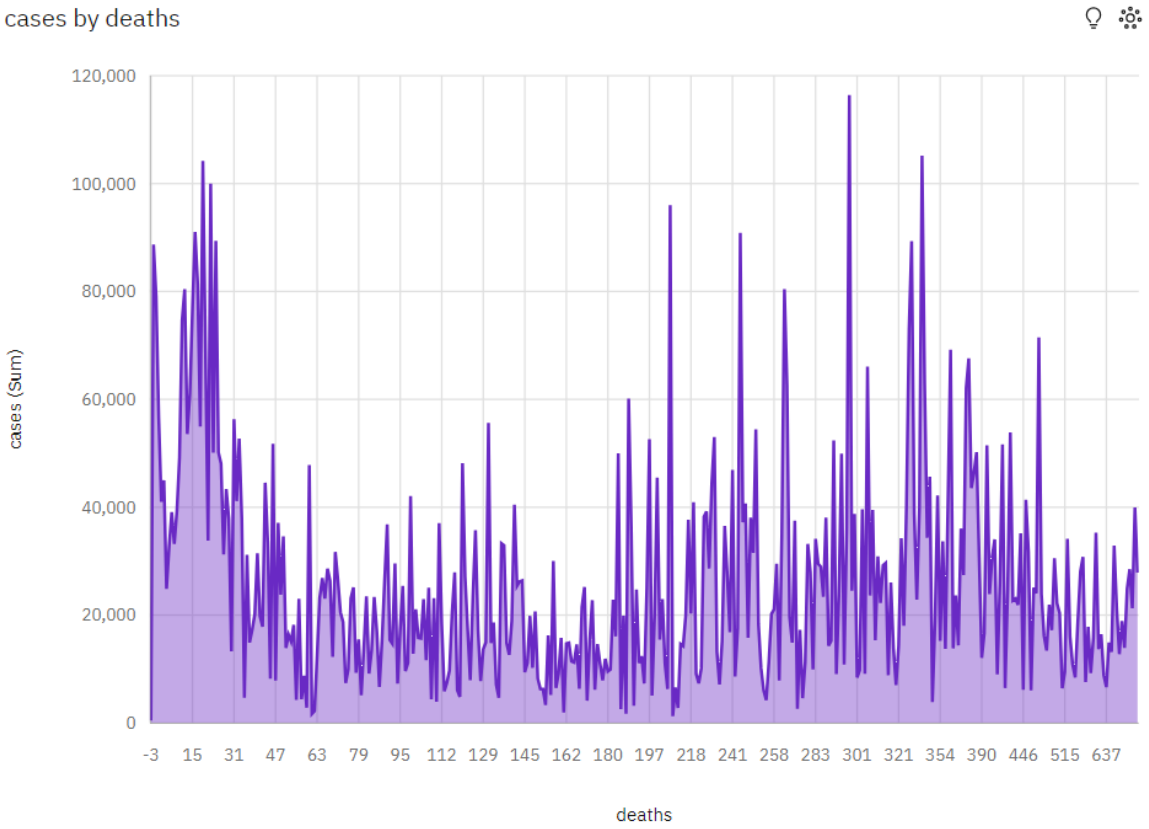
Apply Filters and Interactivity: Add filters and parameters to make your dashboards more interactive. Users should be able to drill down into the data and customize their views. We can even customize the visualization in the ibm cognos analytics.
Apply Filters and Interactivity: Add filters and parameters to make your dashboards more interactive. Users should be able to drill down into the data and customize their views. We can even customize the visualization in the ibm cognos analytics.
Here we are using the decision tree to identify the death rates in the europian country;

Decision tree

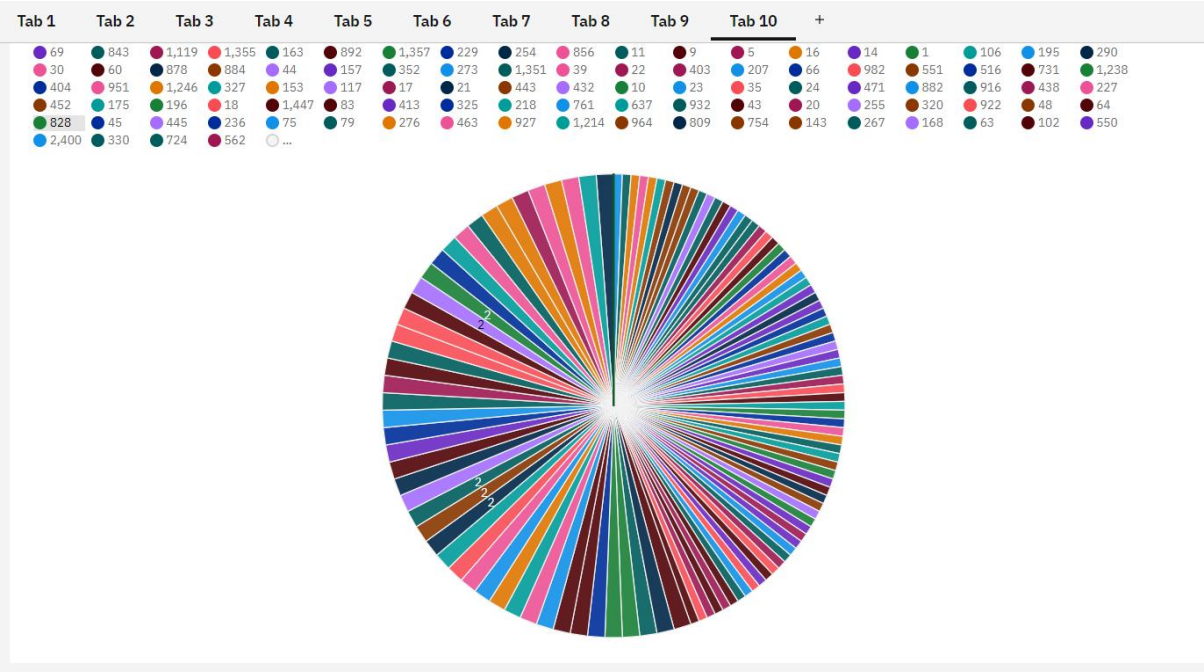
deaths

Tree sunburst | Tree diagram | Rules





PIE CHART FOR CASES BY DEATHS:

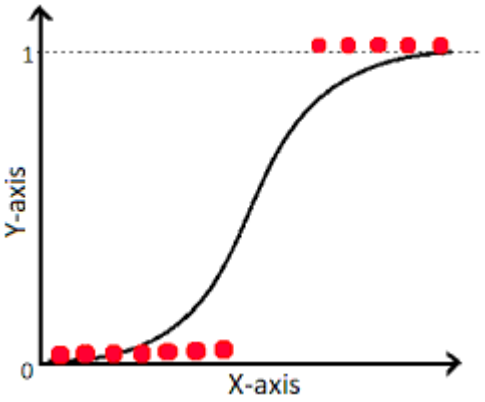


BAR CHART FOR CASES BY MONTH:

ALGORITHM:

1. LOGISTIC REGRESSION

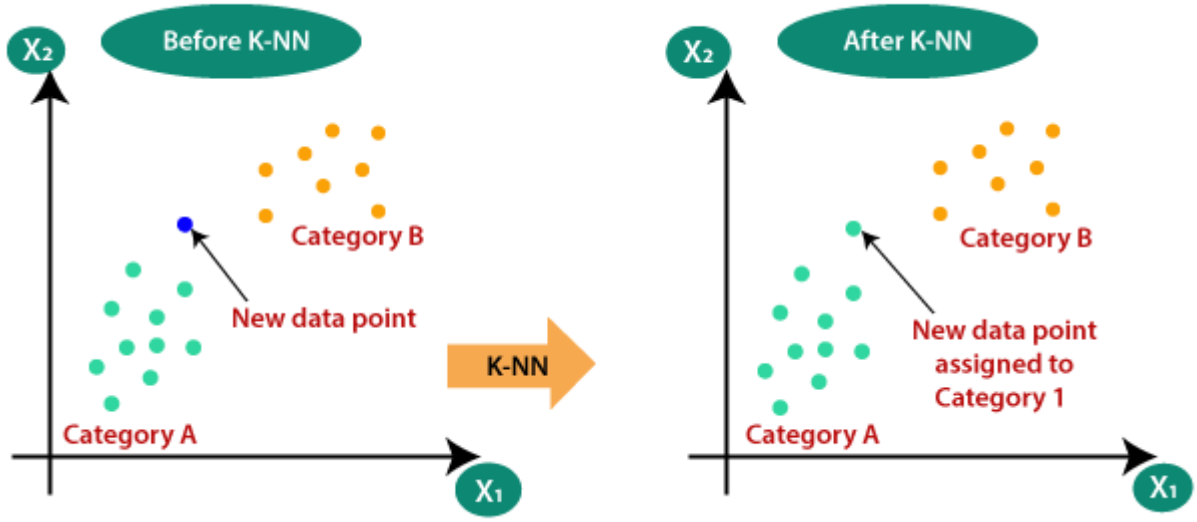
regression is a supervised machine learning algorithm mainly used for classification Logistic tasks where the goal is to predict the probability that an instance of belonging to a given class. It is used for classification algorithms its name is logistic regression. it's referred to as regression because it takes the output of the linear regression function as input and uses a sigmoid function to estimate the probability for the given class.



2. KNN

K-Nearest Neighbours is one of the most basic yet essential classification algorithms in Machine Learning. It belongs to the supervised learning domain and finds intense application in pattern recognition, data mining, and intrusion detection.

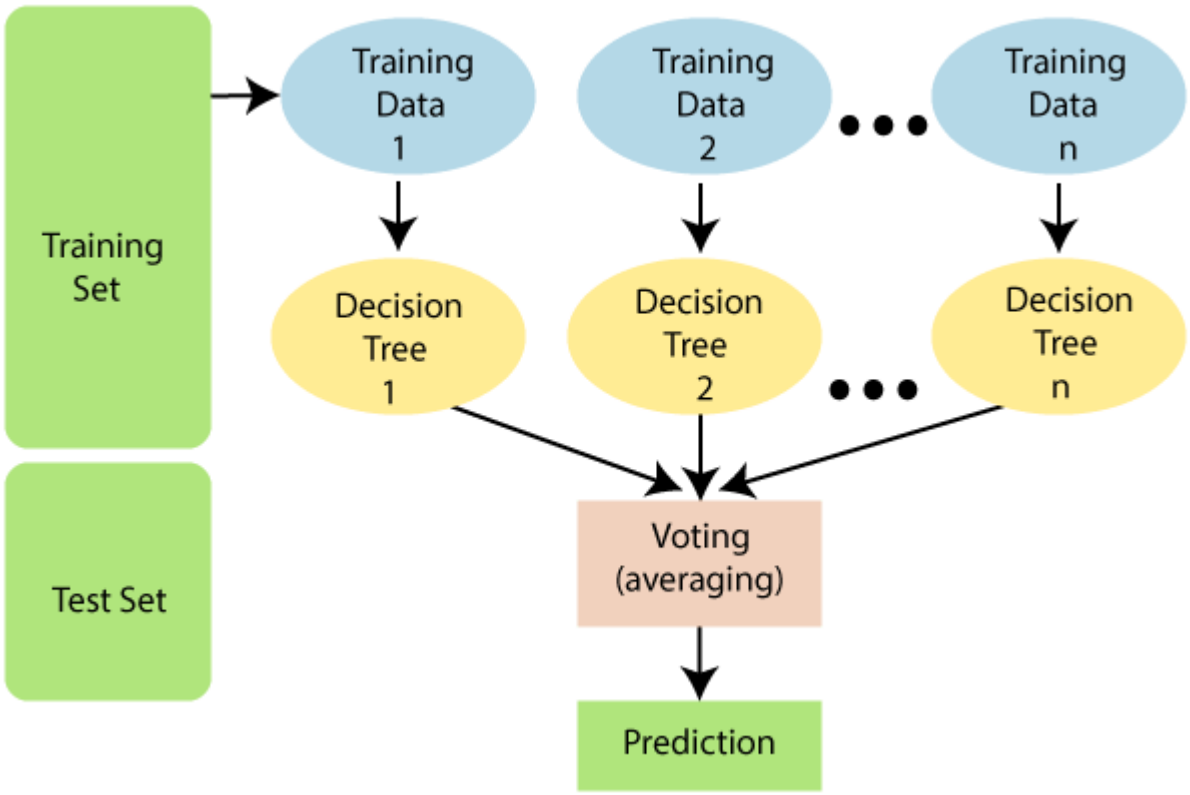
It is widely disposable in real-life scenarios since it is non-parametric, meaning, it does not make any underlying assumptions about the distribution of data (as opposed to other algorithms such as GMM, which assume a Gaussian distribution of the given data). We are given some prior data (also called training data), which classifies coordinates into groups identified by an attribute.



3. RANDOM FOREST CLASSIFIER

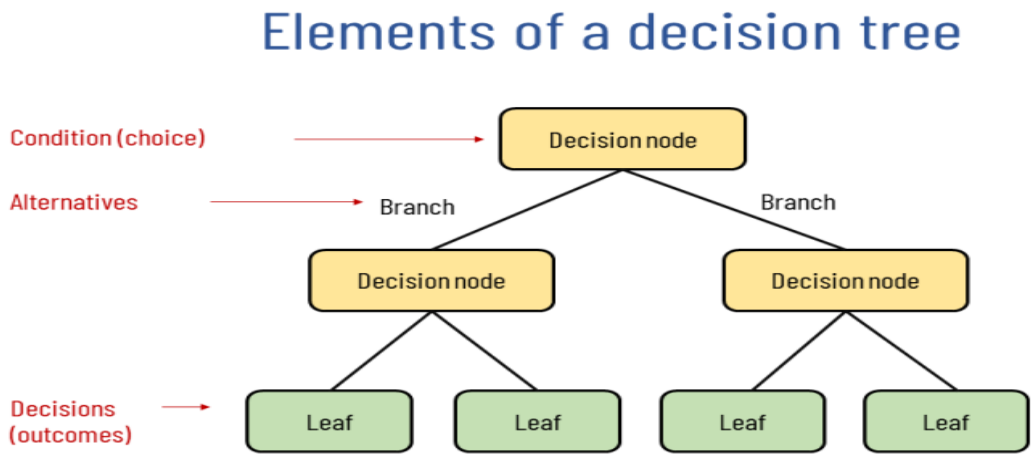
Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.



4.DECISION TREE

A decision tree is a flowchart-like tree structure where each internal node denotes the feature, branches denote the rules and the leaf nodes denote the result of the algorithm. It is a versatile supervised machine-learning algorithm, which is used for both classification and regression problems. It is one of the very powerful algorithms. And it is also used in Random Forest to train on different subsets of training data, which makes random forest one of the most powerful algorithms in machine learning.



RESULTS:

```
from google.colab import files

uploaded = files.upload()
```

Choose files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
```

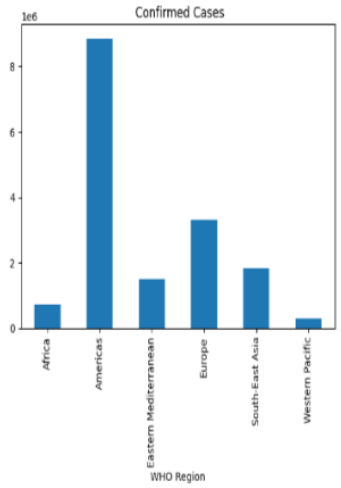
```
[ ] import os
```

```
[ ] data.drop_duplicates(inplace=True)
```

	Country/Region	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered	Deaths / 100 Cases	Recovered / 100 Cases	Deaths / 100 Recovered	Confirmed last week	1 week change	1 week % increase	WHO Region
0	Afghanistan	36263	1269	25198	9796	106	10	18	3.50	69.49	5.04	35526	737	2.07	Eastern Mediterranean
1	Albania	4880	144	2745	1991	117	6	63	2.95	56.25	5.25	4171	709	17.00	Europe
2	Algeria	27973	1163	18837	7973	616	8	749	4.16	67.34	6.17	23691	4282	18.07	Africa
3	Andorra	907	52	803	52	10	0	0	5.73	88.53	6.48	884	23	2.60	Europe
4	Angola	950	41	242	667	18	1	0	4.32	25.47	16.94	749	201	26.84	Africa

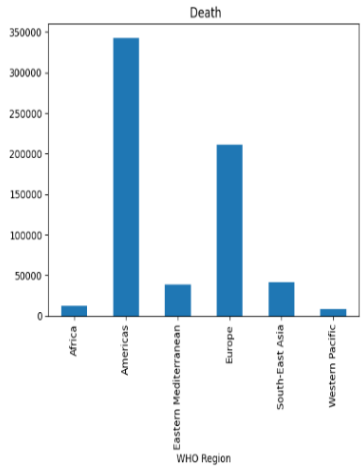
```
Region = df.groupby(by='WHO Region').sum()
Region
Region['Confirmed'].plot(kind='bar', )
plt.title(' Confirmed Cases')
```

```
<ipython-input-13-b291c8b3f079>:1: FutureWarning: The default value of numeric_only in DataFrameGroupBy.sum is deprecated. In a future version, numeric_only will default to False. Either specify numeric_only or select only co
  Region = df.groupby(by='WHO Region').sum()
Text(0.5, 1.0, ' Confirmed Cases')
```



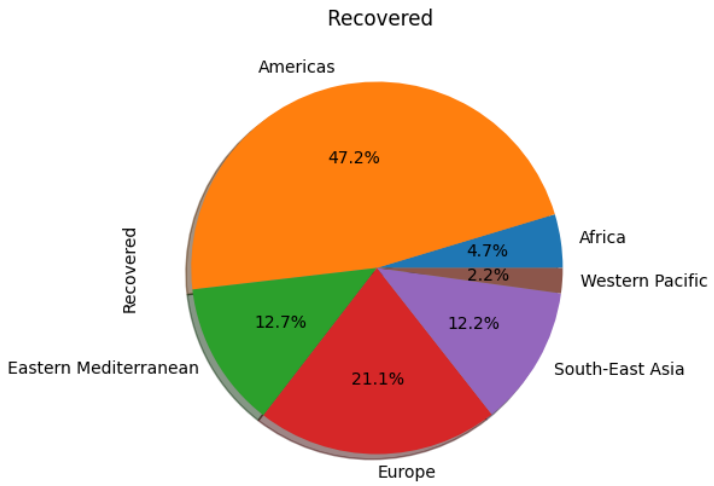

```
Region['Deaths'].plot(kind='bar', )
plt.title(' Death')
```

```
Text(0.5, 1.0, ' Death')
```

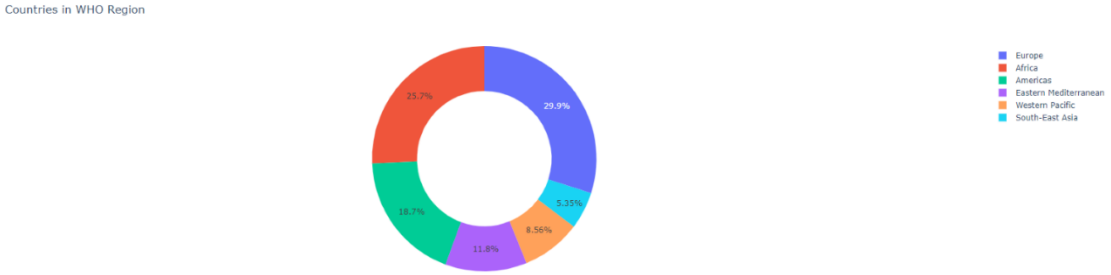


```
Region['Recovered'].plot(kind='pie',  figsize=(10,5), shadow=True, autopct='%1.1f%%')
plt.title(' Recovered')
```

```
Text(0.5, 1.0, ' Recovered')
```




```
[ ] fig = px.pie(whoRegionCountries_df, values='Total Countries',names=whoRegionCountries_df.index, title='Countries in WHO Region', hole= 0.6)
    fig.show()
```

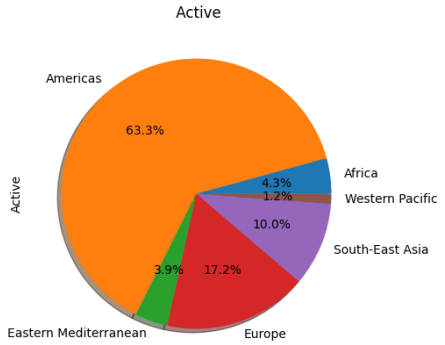



```
whoRegionCountriesActive_df = countrywise_df.groupby(['WHO Region'])[['Active','Deaths','Recovered']].sum()
whoRegionCountriesActive_df.sort_values(['Active'], ascending=False, inplace=True)
whoRegionCountriesActive_df
```

WHO Region	Active	Deaths	Recovered
Americas	4027938	342732	4468616
Europe	1094656	211144	1993723
South-East Asia	637015	41349	1156933
Africa	270339	12223	440645
Eastern Mediterranean	251005	38339	1201400
Western Pacific	77409	8249	206770

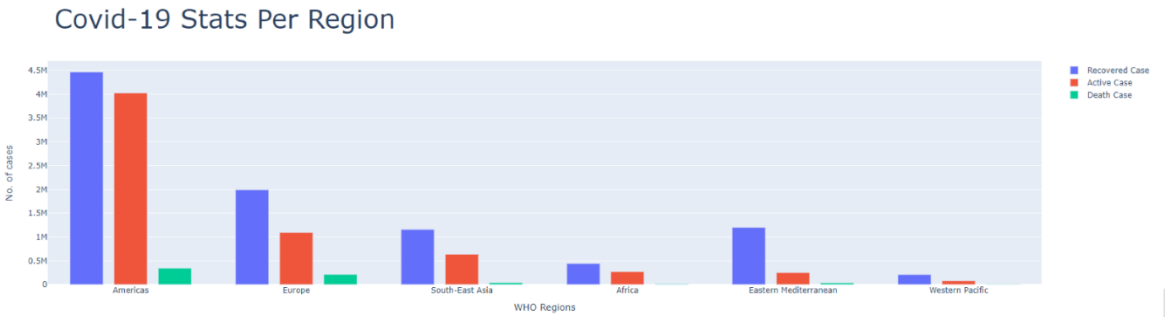
```
Region['Active'].plot(kind='pie', figsize=(10,5), shadow=True, autopct='%1.1f%%') # autopct create % )
plt.title(' Active')
```

```
Text(0.5, 1.0, ' Active')
```

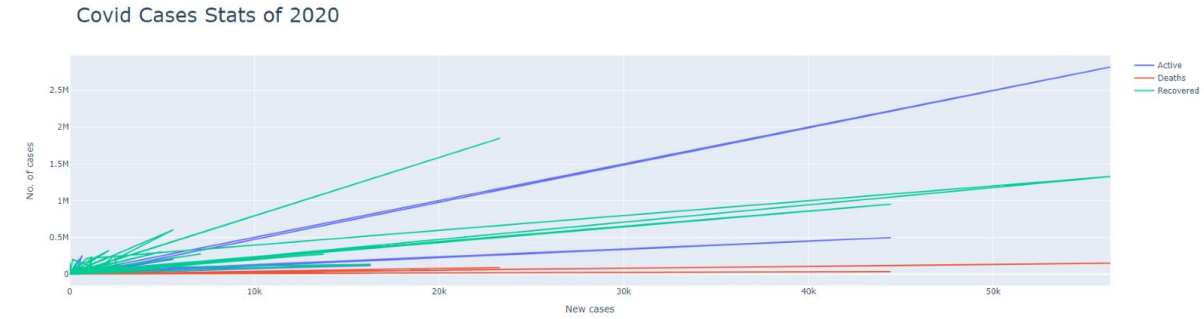



```
anchos = [0.2] * 6
fig = go.Figure()
fig.add_trace(go.Bar(x = whoRegionCountriesActive_df.index,
                     y = whoRegionCountriesActive_df['Recovered'],
                     width = anchos, name = 'Recovered Case'))
fig.add_trace(go.Bar(x = whoRegionCountriesActive_df.index,
                     y = whoRegionCountriesActive_df['Active'],
                     width = anchos, name = 'Active Case'))
fig.add_trace(go.Bar(x = whoRegionCountriesActive_df.index,
                     y = whoRegionCountriesActive_df['Deaths'],
                     width = anchos, name = 'Death Case'))


fig.update_layout(title =  "Covid-19 Stats Per Region",
                  barmode = 'group', title_font_size = 40)
fig.update_xaxes(title_text = 'WHO Regions')
fig.update_yaxes(title_text = "No. of cases")
# fig.write_image(path + "figclust1.png")
fig.show()
```



```
fig = go.Figure()
fig.add_trace(go.Scatter(x=daywise_df['New cases'], y=daywise_df['Active'],
                    mode='lines',
#                     text=casesInMonths_df['Active'],
                    name='Active'))
fig.add_trace(go.Scatter(x=daywise_df['New cases'], y=daywise_df['Deaths'],
                    mode='lines',
#                     text=casesInMonths_df['Deaths'],
                    name='Deaths'))
fig.add_trace(go.Scatter(x=daywise_df['New cases'], y=daywise_df['Recovered'],
                    mode='lines',
#                     text=casesInMonths_df['Recovered'],
                    name='Recovered'))
fig.update_layout(title =  "Covid Cases Stats of 2020",
                  barmode = 'group', title_font_size = 30)
fig.update_xaxes(title_text = 'New cases')
fig.update_yaxes(title_text = "No. of cases")
```



The screenshots above show the code and results of the various phases of the Data Analysis done by us on our Covid-19 dataset negative based on these symptoms.

CONCLUSION:

The Covid - 19 Pandemic is a huge struggle for all of us. The project we are making will seek to find the answers to the most pertinent questions as to what is it that makes the covid 19 such a tragedy and what all people are the ones who are most affected by it. It will seek to find the appropriate response which can be mounted by the authorities concerned and we can reach to a place of proper discussion about the problem and solve it in the best possible manner out there. It will also lead to a solution to any medical condition we might encounter later on in our lives where we can apply data sciences for medical diagnostics. This project saves on the already limited resources that India have and prevents the spread as people can use it to get an idea that they should go and get tested . It also helps unhealthy and infected people to isolate themselves. Using this system we can effectively and efficiently mitigate the burden on our healthcare system which is completely stressed out.