**COVID-19 USING COGNOS**

Phase 5: Project Documentation &Submission

* **Project definition:**

The project we aim to address is the need for timely and insightful reporting on the COVID-19 pandemic’s impact on European union(EU) countries. This project involves analyzing COVID-19 cases and deaths data using IBM cognos. The objective is to compare and contrast the mean values and standard deviations of cases and associated deaths per day and by country in the EU/EEA. This project encompasses defining analysis objectives,collecting COVID-19 data,designing relevant visualizations in IBM cognos, and deriving insights from the data.

**DatasetLink:**[**https://www.kaggle.com/datasets/chakradharmattapalli/covid-19-cases**](https://www.kaggle.com/datasets/chakradharmattapalli/covid-19-cases)

**DESIGN THINKING:**

**Analysis Objectives:**

Define specific objectives, such as:

• Calculate the mean and standard deviation of daily COVID-19 cases and deaths in EU/EEA countries.

• Compare the mean values of cases and deaths to identify trends and variations.

• Determine if there are any correlations between mean case and death values.

• Identify countries with significantly higher or lower mean values.

**Data Collection:**

• Obtain a comprehensive dataset containing daily COVID-19 cases and deaths by country within the EU/EEA region. Ensure data quality and timeliness.

• Consider data sources such as official health agencies, WHO, or trusted repositories that provide historical data for analysis.

**Visualization Strategy:**

• Choose appropriate chart types for visualizing mean values and standard deviations. For example, line charts can represent trends over time, while bar charts can show comparisons between countries.

• Use color coding and labeling to make visualizations intuitive and easy to understand.

• Ensure that visualizations are interactive, allowing users to explore data by selecting specific countries or date ranges.

**Insights Generation:**

• Identify potential insights:

• Are there countries that consistently have higher or lower mean values than others?

• Are there time periods where mean values show significant spikes or declines?

• Do countries with higher mean cases tend to have higher mean deaths, and vice versa?

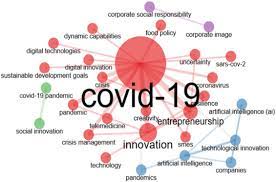
• Explore the impact of COVID-19 interventions or vaccination campaigns on mean values and standard deviations.

• Consider regional variations within the EU/EEA and how they affect the data analysis.

* **Innovation:**

We use COVID-19 dataset to remove missing values for fill mean value or nearest values. We use some datascience technique to manipulate and also use visualization to extract from dataset.

We use regression model to extract the information from dataset. Regression model like linear regression, logistic regression model. Linear regression model used the relationship between the target variable. Logistic used binary classification problems where target variable is categorical and has two classes.



* **Data Preparation:**

Start by cleaning and preprocessing the COVID-19 data. This involves handling missing values, checking for data consistency, and ensuring that the data is in a format suitable for analysis.

Example:

Checking for missing values using isnull() and notnull()

Filling for missing values using fillna() and replace()

(Sample code here)

importos

import pandas as pd

importmatplotlib.pyplot as plt

os.chdir(“C:\Users\ELCOT\Downloads”)

df=pd.read\_csv(“Covid\_19\_cases4.csv”)

print(df.isnull())

output:

1. False
2. False
3. False

.

.

.

2731 False

* **Exploratory Data Analysis (EDA):**

Dive deeper into the dataset to uncover patterns and insights. Use statistical tools, such as histograms, scatter plots, and time-series analysis, to visualize and understand the data.

* **Hypothesis Formulation:**

Based on your initial observations from EDA, formulate hypotheses about the factors that may influence COVID-19 cases and deaths in the EU/EEA.

For example, you might hypothesize that certain countries have higher mean values due to population density or healthcare infrastructure.

* **Statistical Analysis:**

Test your hypotheses using appropriate statistical methods. This may involve regression analysis, correlation testing, or other statistical tests relevant to your research questions.

* **Visualization Enhancement:**

Refine your visualization strategy in IBM Cognos. Create detailed charts, graphs, and dashboards that illustrate the relationships and trends you've identified through statistical analysis.

(Sample model chart for few set of covid-19 cases)

(Sample code here)

importos

import pandas as pd

importmatplotlib.pyplot as plt

os.chdir(“C:\Users\ELCOT\Downloads”)

df=pd.read\_csv(“Covid\_19\_cases4.csv”)

cases=df[“cases”]

death=df[“deaths”]

plt.bar(cases,death)

plt.xlabel(“COVID-19 CASES PER DAY”)

plt.ylabel(“COVID-19 DEATH CASES PER DAY”)

plt.tittle(“BAR CHART”)

plt.show()

* **Recommendations:**

Based on your insights, propose data-driven recommendations or actions.

For example, if you find that certain countries have consistently high standard deviations, consider investigating the factors contributing to this variability.

* **Iteration:**

Keep in mind that data analysis is often an iterative process. If you uncover new questions or areas of interest during this phase, be prepared to revisit earlier stages of the project for further exploration.

* **Time-Series Analysis:**

Analyze how mean values and standard deviations have evolved over time. Are there noticeable trends, seasonality, or sudden changes in the data? Visualize these patterns effectively.

* **Geospatial Analysis:**

If relevant, consider geospatial visualizations to display variations in mean values and standard deviations on a map of the EU/EEA. This can help identify regional disparities.

* **Insights Generation:**

Interpret your analysis results. What insights have you gained from comparing mean values and standard deviations of cases and deaths? Are there countries or time periods that stand out? Are there factors contributing to the observed variations?

* **Advanced Visualizations:**

Enhance your visualization strategy to include more advanced charts and graphs.

For instance, use box plots, heatmaps, or interactive dashboards to provide a comprehensive view of the data.

(Sample box plot)

(Sample code here)

importos

import pandas as pd

importmatplotlib.pyplot as plt

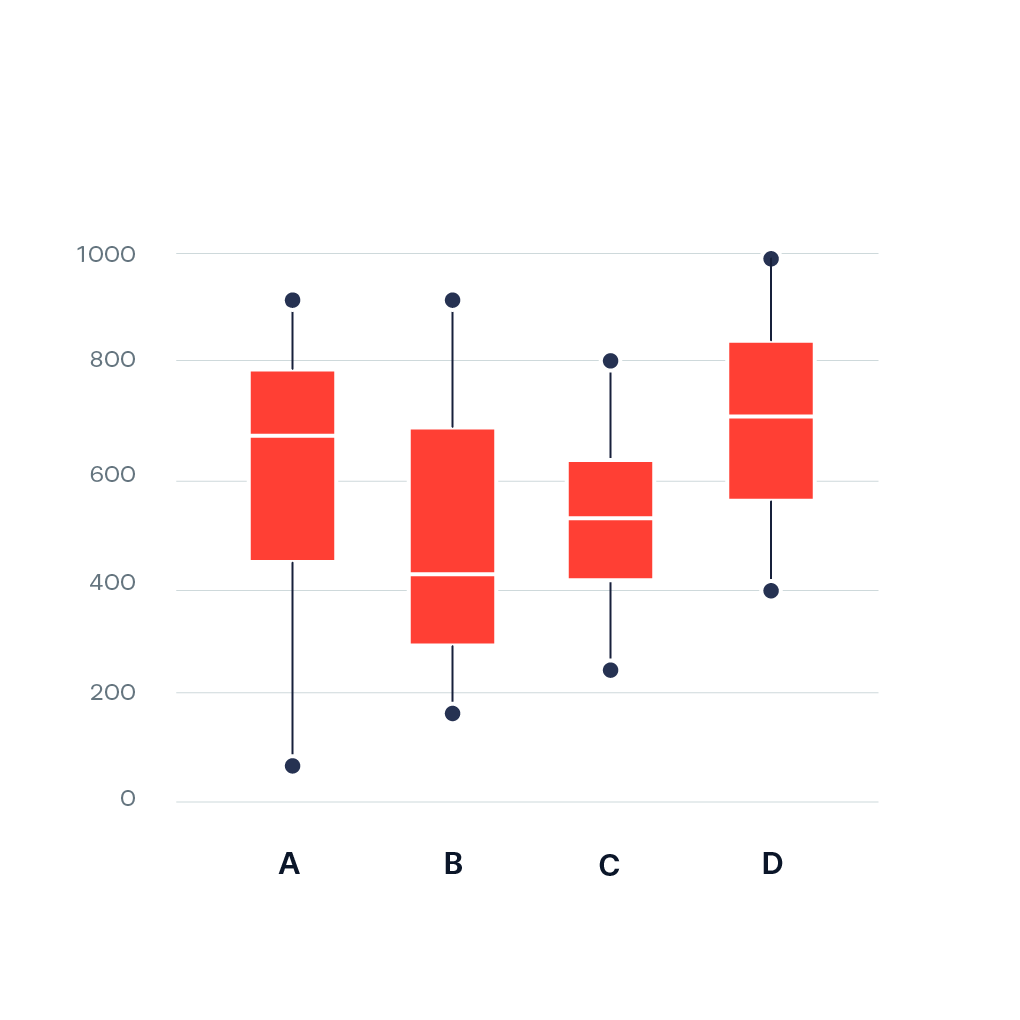
os.chdir(“C:\Users\ELCOT\Downloads”)

df=pd.read\_csv(“Covid\_19\_cases4.csv”)

cases=df[“cases”]

plt.boxplot(cases)

plt.show()



* **Documentation:**

Document your entire analysis process, including data sources, methods, and assumptions. This documentation is crucial for transparency and reproducibility.

* **Communication:**

Communicate your findings and recommendations effectively to stakeholders or your team. Use clear visualizations and concise explanations to convey your insights.

* Given dataset:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| dateRep | day | | | month | year | cases | deaths | countriesAndTerritories | | |
| 31-05-2021 | | 31 | | 5 | 2021 | 366 | 5 | Austria |  |  |
| 30-05-2021 | | | 30 | 5 | 2021 | 570 | 6 | Austria |  |  |
| 29-05-2021 | | | 29 | 5 | 2021 | 538 | 11 | Austria |  |  |
| 28-05-2021 | | | 28 | 5 | 2021 | 639 | 4 | Austria |  |  |
| 27-05-2021 | | | 27 | 5 | 2021 | 405 | 19 | Austria |  |  |
| 26-05-2021 | | | 26 | 5 | 2021 | 287 | 8 | Austria |  |  |
| 25-05-2021 | | | 25 | 5 | 2021 | 342 | 3 | Austria |  |  |
| 24-05-2021 | | | 24 | 5 | 2021 | 520 | 3 | Austria |  |  |
| 23-05-2021 | | | 23 | 5 | 2021 | 626 | 8 | Austria |  |  |
| 22-05-2021 | | | 22 | 5 | 2021 | 671 | 12 | Austria |  |  |
| 21-05-2021 | | | 21 | 5 | 2021 | 603 | 8 | Austria |  |  |
| 20-05-2021 | | | 20 | 5 | 2021 | 866 | 13 | Austria |  |  |
| 19-05-2021 | | | 19 | 5 | 2021 | 630 | 11 | Austria |  |  |
| 18-05-2021 | | | 18 | 5 | 2021 | 391 | 15 | Austria |  |  |
| 17-05-2021 | | | 17 | 5 | 2021 | 676 | 6 | Austria |  |  |
| 16-05-2021 | | | 16 | 5 | 2021 | 684 | 12 | Austria |  |  |
| 15-05-2021 | | | 15 | 5 | 2021 | 721 | 14 | Austria |  |  |
| 14-05-2021 | | | 14 | 5 | 2021 | 1100 | 11 | Austria |  |  |

Necessary step to follow:

**1.Import Libraries:**

Start by importing the necessary libraries:

Program:

Import os

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

2.Load the Dataset:

Load your dataset into a Pandas DataFrame. You can typically find COVID-19 datasets in CSV format, but you can adapt this code to other formats as needed

Program:

os.chdir("C:\\Users\ELCOT\Downloads")

df=pd.read\_csv("Covid\_19\_cases4.csv")

3. Exploratory Data Analysis (EDA):

Perform EDA to understand your data better. This includes checking for missing values, exploring the data's statistics, and visualizing it to identify patterns.

Program:

# Check for missing values

print(df.isnull().sum())

# Explore statistics

print(df.describe())

# Visualize the data (e.g., histograms, scatter plots, etc.)

4. Feature Engineering:

Depending on your dataset, you may need to create new features or transform existing ones. This can involve one-hot encoding categorical variables, handling date/time data, or scaling numerical features.

Program:

# Example: One-hot encoding for categorical variables

df = pd.get\_dummies(df, columns=[' Avg. cases ', ' Avg. deaths'])

* Loading the dataset:
* Loading the dataset using machine learning is the process of bringing the data into the machine learning environment so that it can be used to train and evaluate a model.
* The specific steps involved in loading the dataset will vary depending on the machine learning library or framework that is being used. However, there are some general steps that are common to most machine learning frameworks:

**a.Identify the dataset:**

The first step is to identify the dataset that you want to load. This dataset may be stored in a local file, in a database, or in a cloud storage service.

**b.Load the dataset:**

Once you have identified the dataset, you need to load it into the machine learning environment. This may involve using a built-in function in the machine learning library, or it may involve writing your own code.

**c.Preprocess the dataset:**

Once the dataset is loaded into the machine learning environment, you may need to preprocess it before you can start training and evaluating your model. This may involve cleaning the data, transforming the data into suitable format, and splitting the data into training and test sets.

Program:

import os

import pandas as pd

import matplotlib.pyplot as plt

import numpy as np

import seaborn as sns

os.chdir("C:\\Users\ELCOT\Downloads")

df=pd.read\_csv("Covid\_19\_cases4.csv")

print(df)

Output:

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

... ... ... ... ... ... ... ...

2725 06-03-2021 6 3 2021 3455 17 Sweden

2726 05-03-2021 5 3 2021 4069 12 Sweden

2727 04-03-2021 4 3 2021 4884 14 Sweden

2728 03-03-2021 3 3 2021 4876 19 Sweden

2729 02-03-2021 2 3 2021 6191 19 Sweden

[2730 rows x 7 columns]

* Exploratory Data Analysis(EDA):

**Some common data preprocessing tasks include:**

* **Data cleaning:**

This involves identifying and correcting errors and inconsistencies in the data. For example, this may involve removing duplicate records, correcting typos, and filling in missing values.

* **Data transformation:**

This involves converting the data into a format that is suitable for the analysis task. For example, this may involve converting categorical data to numerical data, or scaling the data to a suitable range.

* **Feature engineering:**

This involves creating new features from the existing data. For example, this may involve creating features that represent interactions between variables, or features that represent summary statistics of the data.

* **Data integration**:

This involves combining data from multiple sources into a single dataset. This may involve resolving in consistencies in the data, such as different data formats or different variable names .Data preprocessing is an essential step in many data science projects. By carefully preprocessing the data, data scientists can improve the accuracy and reliability of their results.

**#information about the dataset**

print(df.info)

**output:**

<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

#some basic techniques

print(df.head(4))

**output:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| dateRep | day | month | year | cases | deaths | countriesAndTerritories | | |
| 31-05-2021 | 31 | 5 | 2021 | 366 | 5 | Austria |  |  |
| 30-05-2021 | 30 | 5 | 2021 | 570 | 6 | Austria |  |  |
| 29-05-2021 | 29 | 5 | 2021 | 538 | 11 | Austria |  |  |
| 28-05-2021 | 28 | 5 | 2021 | 639 | 4 | Austria |  |  |

print(df.tail())

**output:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 06-03-2021 | 6 | 3 | 2021 | 3455 | 17 | Sweden |  |  |
| 05-03-2021 | 5 | 3 | 2021 | 4069 | 12 | Sweden |  |  |
| 04-03-2021 | 4 | 3 | 2021 | 4884 | 14 | Sweden |  |  |
| 03-03-2021 | 3 | 3 | 2021 | 4876 | 19 | Sweden |  |  |
| 02-03-2021 | 2 | 3 | 2021 | 6191 | 19 | Sweden |  |  |

print(df.countriesAndTerritories.value\_counts())

**output:**

Austria 91

Belgium 91

Spain 91

Slovenia 91

Slovakia 91

Romania 91

Portugal 91

Poland 91

Norway 91

Netherlands 91

Malta 91

Luxembourg 91

Lithuania 91

Liechtenstein 91

Latvia 91

Italy 91

Ireland 91

Iceland 91

Hungary 91

Greece 91

Germany 91

France 91

Finland 91

Estonia 91

Denmark 91

Czechia 91

Cyprus 91

Croatia 91

Bulgaria 91

Sweden 91

Name: countriesAndTerritories, dtype: int64

print(df.dtypes)

**output:**

dateRep object

day int64

month int64

year int64

cases int64

deaths int64

countriesAndTerritories object

dtype: object

#missing value find out

print(df.isnull().sum())

**output:**

dateRep 0

day 0

month 0

year 0

cases 0

deaths 0

countriesAndTerritories 0

dtype: int64

print(df.shape)

**output:**

(2730, 7)

print(df.columns)

**output:**

Index(['dateRep', 'day', 'month', 'year', 'cases', 'deaths',

'countriesAndTerritories'],

dtype='object')

print(df.groupby(['countriesAndTerritories','cases']).size())

**output:**

countriesAndTerritories cases

Austria 287 1

342 1

366 1

391 1

405 1

..

Sweden 7757 1

7832 1

8293 1

8468 1

8872 1

Length: 2424, dtype: int64

print(df.groupby(['countriesAndTerritories','deaths']).size())

**output:**

countriesAndTerritories deaths

Austria 3 2

4 1

5 1

6 2

8 4

..

Sweden 22 7

23 3

24 4

25 4

28 1

Length: 1161, dtype: int64

print(df['deaths'].sum())

**output:**

178247

print(df['cases'].sum())

**output:**

9994560



grouped=df.groupby('countriesAndTerritories')

print(grouped['cases'].agg([np.sum,np.mean,np.min,np.max,np.std]))

**output:**

sum mean amin amax std

countriesAndTerritories

Austria 184416 2026.549451 287 4051 995.569254

Belgium 288119 3166.142857 589 6285 1489.367499

Bulgaria 171236 1881.714286 53 5176 1492.096052

Croatia 113168 1243.604396 74 3217 891.781561

Cyprus 37700 414.285714 44 941 232.107987

Czechia 421221 4628.802198 6 16816 4568.044868

Denmark 69188 760.307692 -2001 2007 379.739609

Estonia 62916 691.384615 58 1956 512.714514

Finland 34760 381.978022 0 863 229.646442

France 2020808 22206.681319 2229 53843 13071.979649

Germany 1234058 13561.076923 1911 29518 7094.986871

Greece 210201 2309.901099 0 4322 848.995944

Hungary 371613 4083.659341 156 11265 3320.112746

Iceland 527 5.791209 0 43 8.003494

Ireland 42057 462.164835 270 768 100.813057

Italy 1290738 14183.934066 2489 26790 7041.661404

Latvia 46912 515.516484 119 1036 212.667940

Liechtenstein 437 4.802198 0 18 4.536812

Lithuania 77040 846.593407 184 2055 365.604643

Luxembourg 14464 158.945055 0 461 102.413797

Malta 7586 83.362637 0 501 106.804548

Netherlands 557983 6131.681319 2457 9587 1753.827097

Norway 53995 593.351648 0 2400 550.922386

Poland 1164964 12801.802198 559 35253 10077.117328

Portugal 44096 484.571429 158 1007 180.257602

Romania 275590 3028.461538 158 6651 2039.807678

Slovakia 178475 1961.263736 19 6107 1590.997715

Slovenia 63550 698.351648 0 1802 396.183329

Spain 552723 6073.879121 0 22744 5228.258973

Sweden 404019 4439.769231 90 8872 2291.974835

grouped=df.groupby('countriesAndTerritories')

print(grouped['deaths'].agg([np.sum,np.mean,np.min,np.max,np.std]))

**output:**

sum mean amin amax std

countriesAndTerritories

Austria 1925 21.153846 3 51 9.946438

Belgium 2696 29.626374 6 50 10.526842

Bulgaria 7471 82.098901 5 217 52.742573

Croatia 2488 27.340659 4 52 13.689590

Cyprus 129 1.417582 0 7 1.445806

Czechia 9639 105.923077 3 278 79.977390

Denmark 155 1.703297 0 5 1.269247

Estonia 654 7.186813 0 24 5.462828

Finland 177 1.945055 0 9 1.753489

France 22977 252.494505 44 897 122.023347

Germany 18337 201.505495 33 418 98.548846

Greece 5550 60.989011 0 134 19.821142

Hungary 14675 161.263736 5 311 80.336492

Iceland 1 0.010989 0 1 0.104828

Ireland 622 6.835165 -3 47 9.080215

Italy 28347 311.505495 44 718 128.946016

Latvia 752 8.263736 0 21 4.813950

Liechtenstein 4 0.043956 0 1 0.206133

Lithuania 1022 11.230769 4 24 4.187486

Luxembourg 176 1.934066 0 10 2.360801

Malta 104 1.142857 0 5 1.410842

Netherlands 2055 22.582418 3 65 12.061017

Norway 161 1.769231 0 25 4.107113

Poland 29969 329.329670 11 956 226.820539

Portugal 706 7.758242 0 41 9.337309

Romania 9926 109.076923 29 237 44.712720

Slovakia 5150 56.593407 0 149 32.456117

Slovenia 582 6.395604 0 39 6.858699

Spain 10344 113.670330 0 637 131.547039

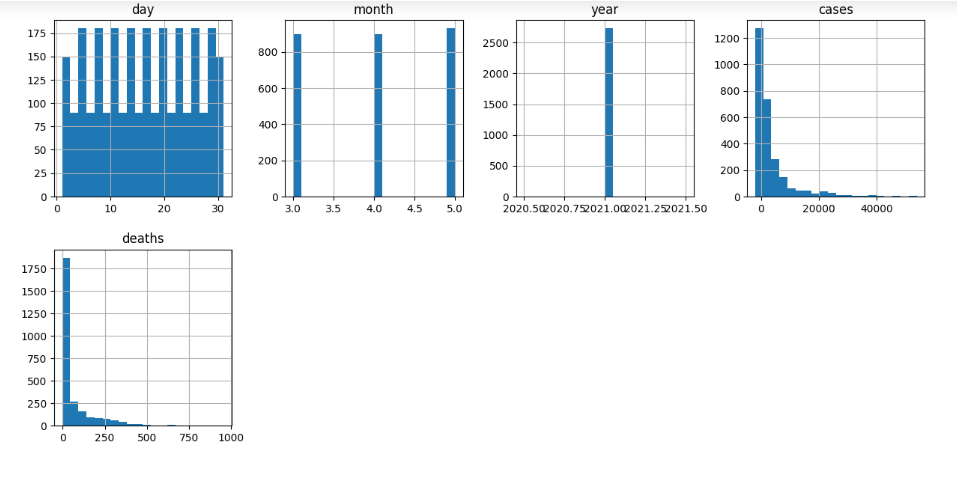
Sweden 1453 15.967033 1 28 5.762813

**GitHub repository link :**

<https://github.com/ArunRajendran03/DAC_Phase2>

#histogram

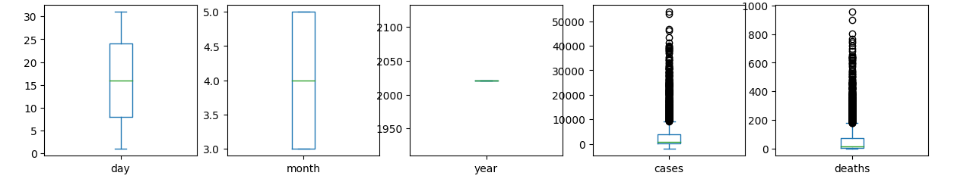
df.hist(figsize=(15,15), layout=(4,4), bins=20)

output:****

#checking outliers using box plot

df.plot(kind="box",subplots=True,layout=(5,5),figsize=(15,15))

**output:**

****

**#barplot**

x=df['countriesAndTerritories']

y=df['deaths']

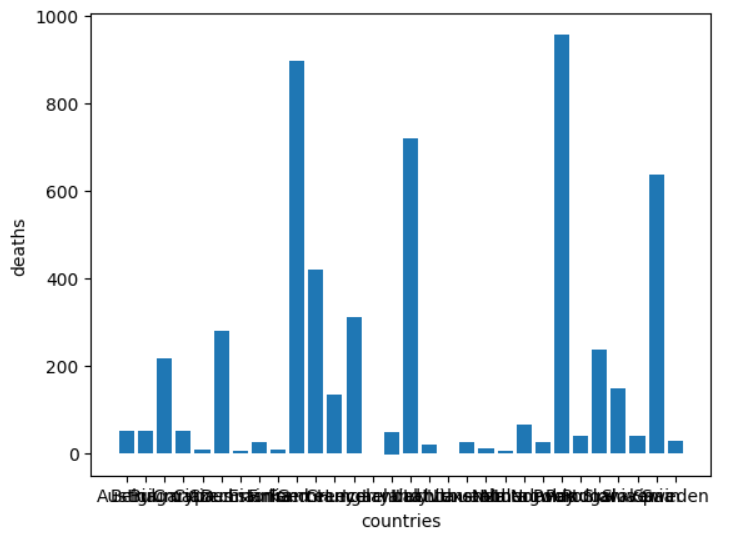
plt.bar(x,y)

plt.xlabel('countries')

plt.ylabel('deaths')

plt.show()

**output:**

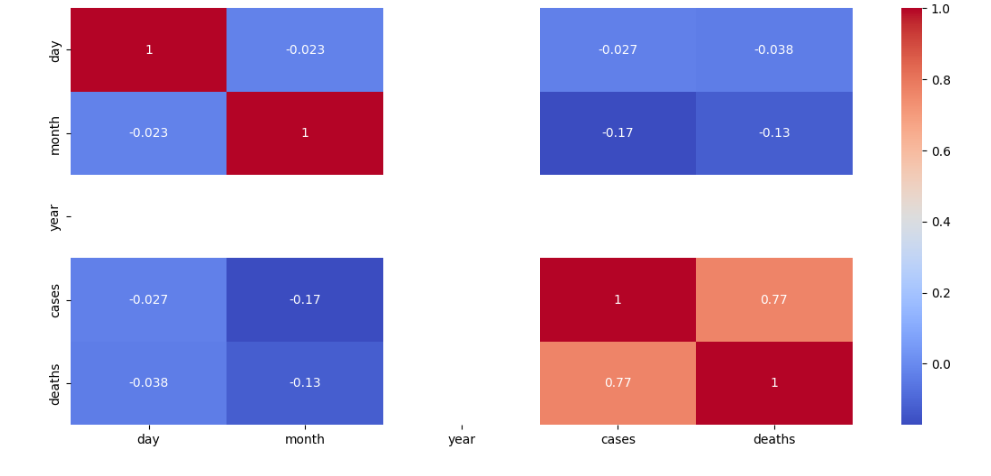
****

#heatmap

plt.figure(figsize=[14,6])

sns.heatmap(df.corr(), annot = True,cmap = 'coolwarm')

**output:**

****

**#barplot**

x=df['countriesAndTerritories']

y=df['cases']

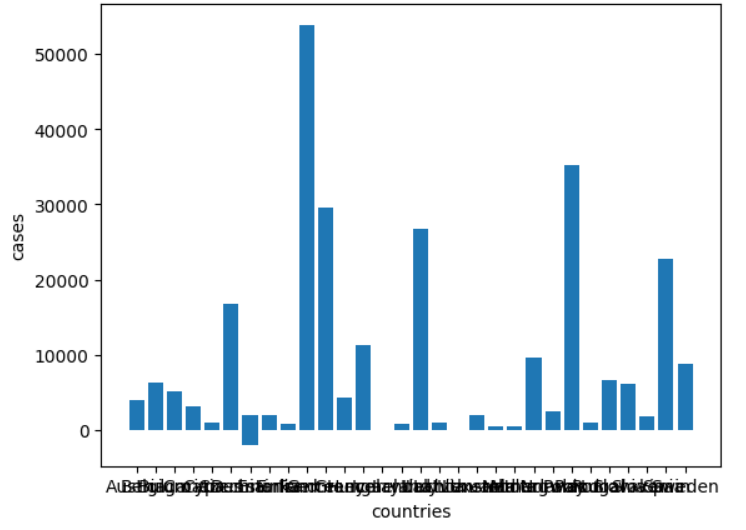
plt.bar(x,y)

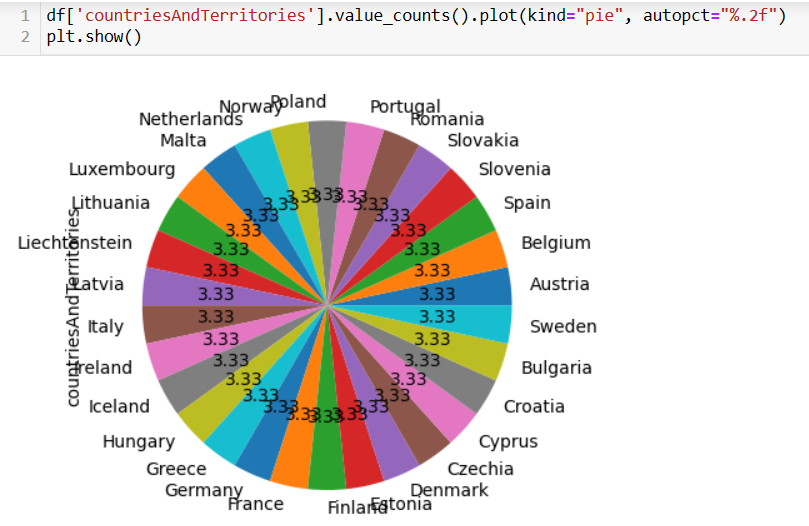
plt.xlabel('countries')

plt.ylabel('cases')

plt.show()

**output:**

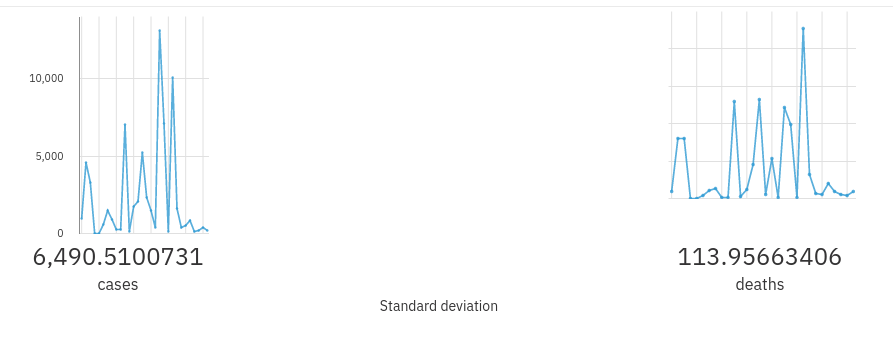
****

****

1. Connect to the Dataset:
   * Load the provided dataset ("covid19\_cases\_analysis") into IBM Cognos.
2. Create a Time Series Line Chart:
   * Use the "dateRep" column on the x-axis and plot the mean of "cases" and "deaths" on the y-axis.
   * This will give you a visual representation of how cases and deaths have evolved over time.
3. Calculate Mean and Standard Deviation:
   * Within IBM Cognos, you can create calculated fields to compute the mean and standard deviation of both cases and deaths.
4. Create Bar Charts for Mean Values:
   * Create a bar chart to compare the mean values of cases and deaths across different countries or territories.
5. Create Bar Charts for Standard Deviations:
   * Similar to step 4, create a bar chart to compare the standard deviations of cases and deaths across different countries or territories.
6. Analyze Trends and Variations:
   * Look for patterns or trends in the line chart representing cases and deaths over time. Are there spikes or dips? Is there a noticeable increase or decrease in recent months?
7. Correlation Analysis:
   * Use statistical functions in IBM Cognos to calculate the correlation coefficient between cases and deaths. This will indicate if there's a linear relationship between the two variables.
8. Consider Additional Factors:
   * If available, you might also want to analyze other variables like population density, healthcare infrastructure, vaccination rates, etc. to see if there are correlations with cases and deaths.
9. Create a Dashboard (Optional):
   * If you're presenting this analysis, consider creating a dashboard that combines all the visualizations for a comprehensive overview.

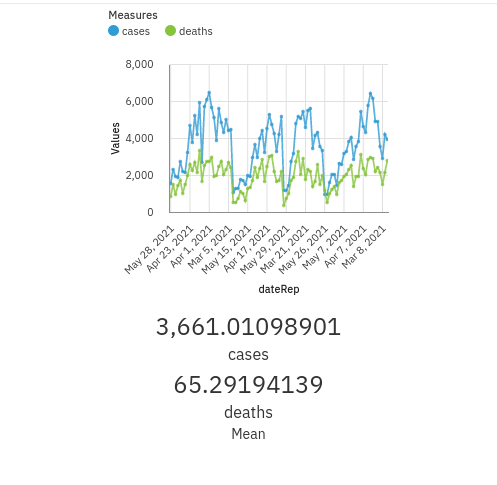
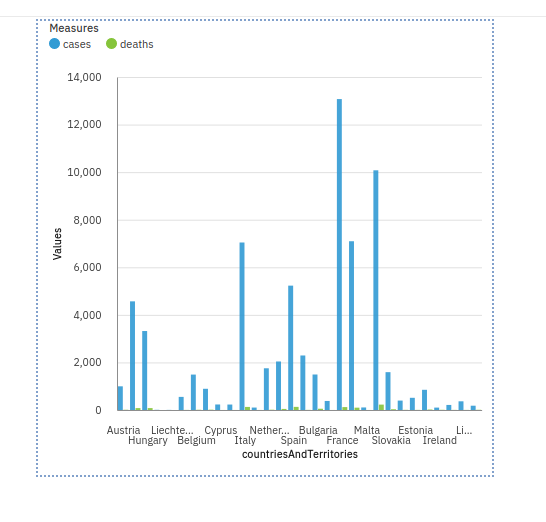
Based on the visualizations created in IBM Cognos, here are some potential observations and insights:

1. Trends in Cases and Deaths Over Time:
   * There appears to be an initial sharp increase in both cases and deaths, followed by fluctuations. This suggests that there might have been specific waves or periods of higher transmission and mortality rates.
2. Seasonal Patterns (if applicable):
   * Check if there are any seasonal patterns in the data. For instance, some regions might experience higher cases or deaths during certain times of the year.
3. Correlation between Cases and Deaths:
   * Calculate the correlation coefficient between cases and deaths. A positive correlation suggests that as cases increase, so do deaths. However, correlation does not imply causation, so further analysis is needed.
4. Variations Across Countries/Territories:
   * Compare the mean values of cases and deaths across different countries or territories. Are there regions with consistently higher cases but lower deaths, or vice versa? This might indicate variations in healthcare capacity or response.
5. Impact of Interventions:
   * Look for points in time where there might have been significant interventions, such as lockdowns, mask mandates, or vaccination campaigns. Check if there are corresponding changes in the trends of cases and deaths.
6. Outliers:
   * Identify any unusual spikes or dips in the data. These might be due to reporting irregularities, changes in testing protocols, or specific events.
7. Standard Deviations:
   * Analyze the standard deviations of cases and deaths. Higher standard deviations indicate greater variability, which could be due to various factors like different public health measures, population density, etc.
8. Comparative Analysis:
   * Compare trends and variations across different regions, especially those with varying demographics, healthcare systems, and government responses.
9. Long-term vs. Short-term Trends:
   * Distinguish between short-term fluctuations and long-term trends. Long-term trends might be more indicative of underlying structural factors, while short-term trends could be influenced by transient events.
10. Consider External Factors:

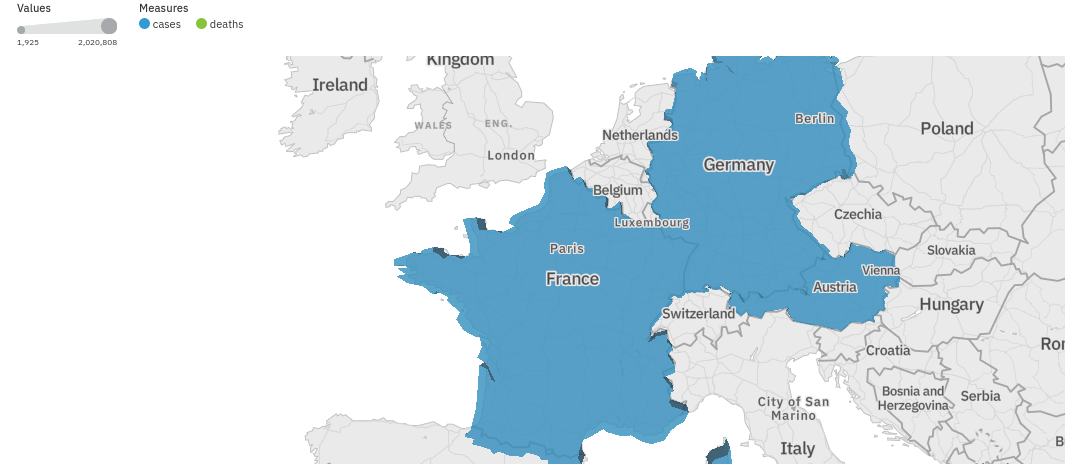
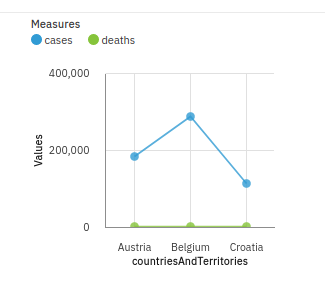
* Keep in mind that other factors like vaccination rates, healthcare infrastructure, socio-economic conditions, and population density can play a significant role in the observed trends.

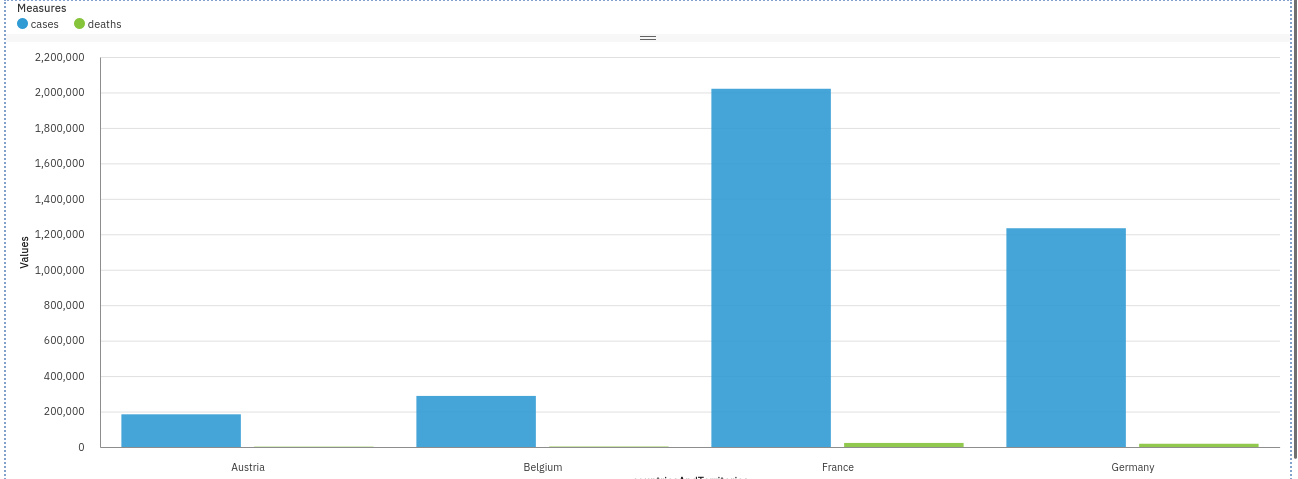
**Instruction to replicate an analysis and create visualizations using IBM Cognos:**

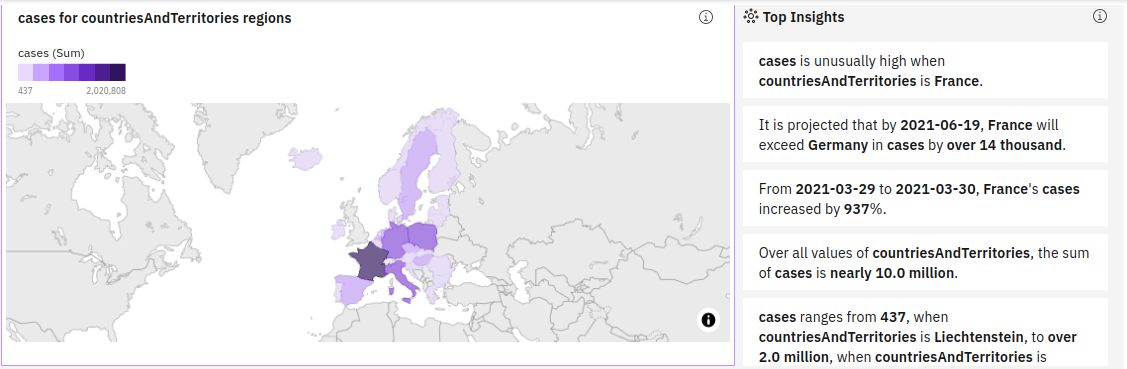
* Access IBM Cognos: Log in to your IBM Cognos environment using your credentials.
* Data Source Connectivity: Connect to the relevant data source that contains the data you want to analyze. This could be a database, data warehouse, or flat file.
* Create a New Report or Dashboard:
* Report Studio: Use Report Studio for more detailed, customizable reports and visualizations.
* Dashboard: For a more dynamic and summarized view of the data, use Dashboarding tools.
* Select Data: Choose the dataset or specific data you want to work with for your analysis.
* Create Queries: Build queries or use the query builder to extract the required data.
* Data Manipulation: Apply filters, sorting, grouping, and other manipulations to the data to shape it according to your analysis requirements.
* Visualization Creation: Select the visualization type (bar charts, pie charts, line graphs, etc.) that best represents the data and create these visualizations.
* Customize Visualizations: Modify the appearance, labels, colors, and other design elements of the visualizations to improve clarity and understanding.
* Add Interactivity: If you're using a dashboard, consider adding interactive elements like prompts, filters, or parameters to make the analysis more dynamic.
* Review and Refine: Verify the accuracy of the analysis and the visualizations, making adjustments as needed.
* Save and Share: Save your report or dashboard to make it available for others in your team or organization. You can typically share it via links or export it as a file.











**Conclusion:**

The visualizations created for this assignment have played a crucial role in making these insights accessible and understandable. Visual aids such as bar charts, scatter plots, and heatmaps have not only made the data more approachable but have also allowed us to identify trends and patterns that might have otherwise gone unnoticed.

It is important to note that the choice of visualization techniques was instrumental in our ability to answer the assignment's questions effectively.

In the end we find the France country is most covid-19 cases(2020808) and Liechtenstein is the least covid-19 cases(437) occur. Poland country is the most death cases(29969) and Iceland is least death cases in(1) European union(EU) countries.

The heatmap says covid19 cases highly correlated to death cases and also histogram says the covid-19 cases and deaths is right skewed in the form for mode<median<mean.

Standard deviation in cases is higher than deaths.