DATA ANALYTICS PORTFOLIO

Case Study By Stefan Rieß

Careerfoundry Graduate

05/2022

Project 06
Covid-19 Evolution

Objective:

- Visualization and evolution of the current ongoing Global Covid-19 Pandemic. How did different tactics in living and fighting the Covid-19 Pandemic affect the World?
- To understand and Visualize the Evolution of Covid-19 several analytical techniques have been used.
- Tools that have been used :
  - Python, Tableau, MS Office, and Machine Learning. To perform consistency checks, create Matrix and Heatmaps, usage of Supervised Regression machine learning and Clustering Time Series.
- The Data was obtained from our World in Data which can be found **here.**
- The whole presentation is based on Tableau **here.**
- All Coding and descriptions can be found under Github **here.**

How did i approach this Project?

- As the whole Topic of Covid-19 is fascinating, the first point was for me to find an official and reliable Source of Data for Covid-19 to be explored and Analysed. After some research, it seemed that Our World in Data is an official Source of Data that collects Data from official government resources. I planned to finish the whole Analysis, cleaning, and preparing the Data to be done within two weeks of the course.
- **Our World in Data**
- I started to ask myself questions about the Pandemic. I found this very interesting because I live in a multinational family myself (German/Chinese) living in France I found this very interesting.
- In my Analysis, I wanted to look at how each country approached and reacted to the Virus, how Vaccines and restrictions affect daily life, and which strategies worked best for each country from 0-Covid to Herd-immunity.

Challenges

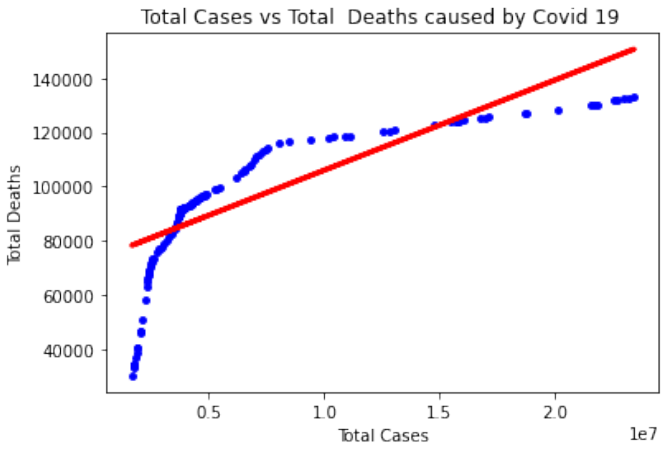
- The worldwide Dataset on the Covid-19 Pandemic is tremendously significant, and running Scripts took quite some time.

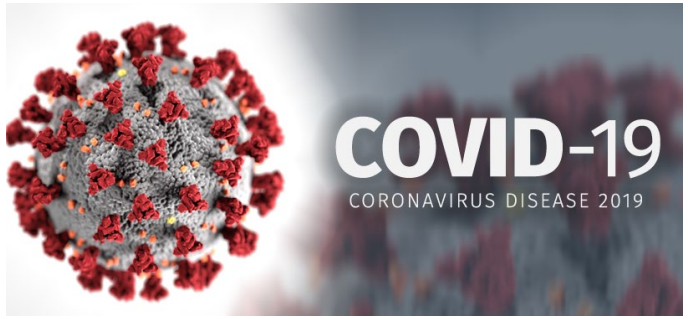
After consulting with my Tutor, we decided to mainly focus on the Top Affected European Countries and only specific submitted Data in the Source, such as Death, Vaccinations, and more which decreased the size of the Dataset to work and Train with.

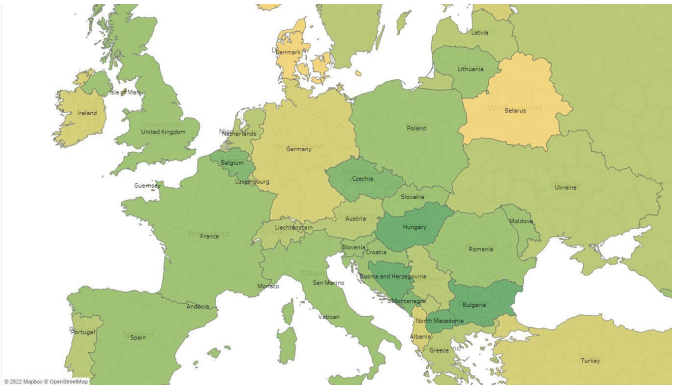
- Some Analytical Processes couldn‘t be easily performed as a beginner, such as time Series Analysis

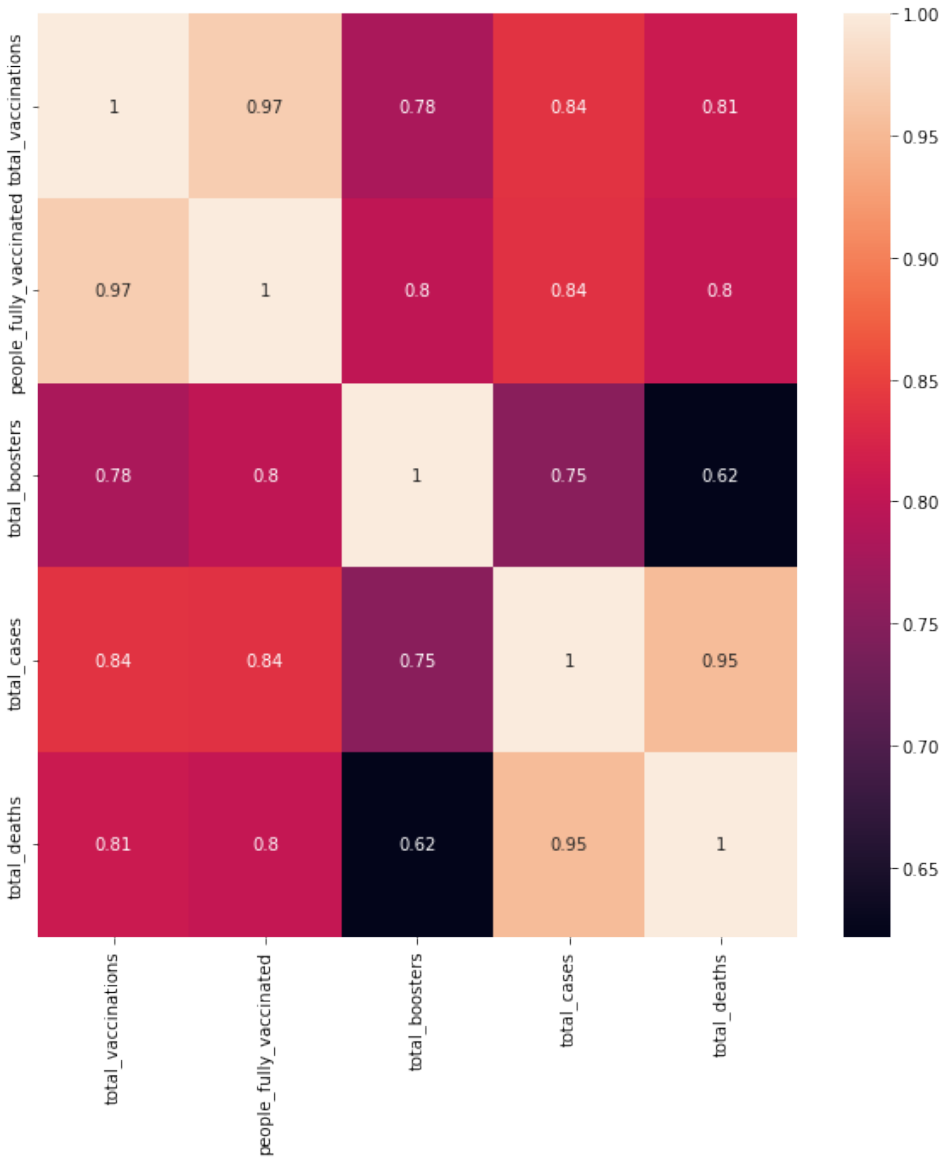
After reconsulting and confirming with my Tutor, the Covid-19 Dataset I started to work with wasn‘t the best fit for Timeseries and Cluster Analysis. We went the extra mile and used different datasets for these Exercises, which gave us good practice in cleaning and preparing data. (All the Data and Scripts can be found via Github on page 1)

CODING AND VISUALISATION

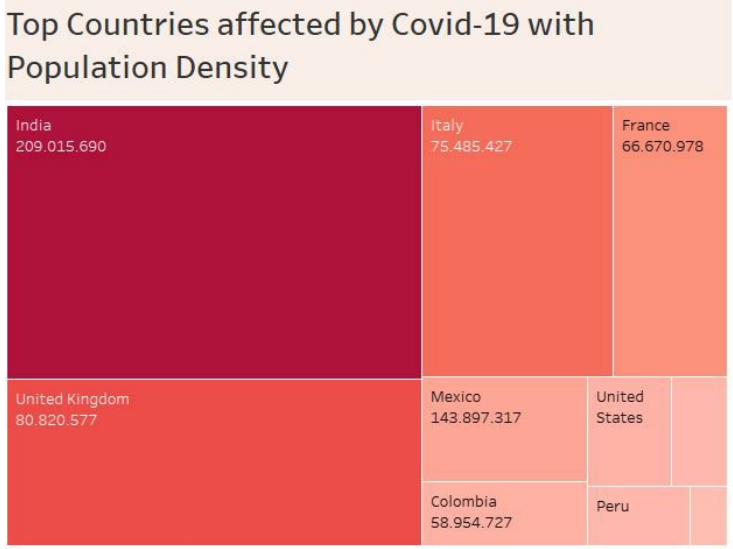






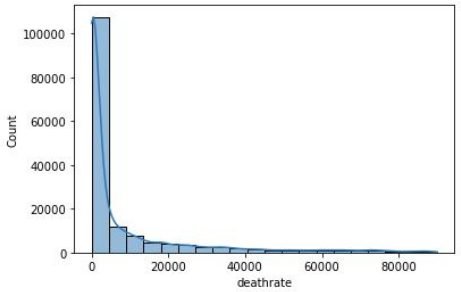


CODING AND VISUALISATION



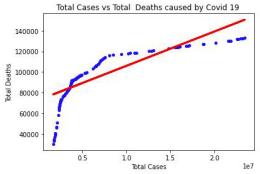
CODING AND VISUALISATION

```
In [20]: # Check the rating variable

         sns.histplot(total_death['deathrate'], bins=20, kde = True)

Out[20]: <AxesSubplot:xlabel='deathrate', ylabel='Count'>
```




```
In [21]: # Select only entries with a non-zero deathrate

         total_death = total_death[total_death['deathrate'] >= 1]
```

```
In [77]: # Create a plot that shows the regression line from the model on the test set.

         plot_test = plt
         plot_test.scatter(X_test, y_test, color='blue', s = 15)
         plot_test.plot(X_test, y_predicted, color='red', linewidth =3)
         plot_test.title('Total Cases vs Total  Deaths caused by Covid 19 ')
         plot_test.xlabel('Total Cases')
         plot_test.ylabel('Total Deaths')
         plot_test.show()
```

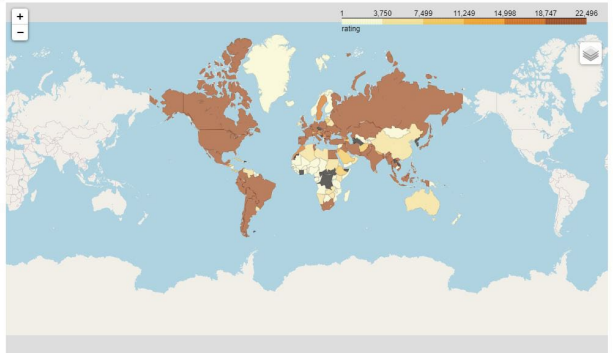



```
In [77]: # Create objects that contain the model summary statistics.

         rmse = mean_squared_error(y_test, y_predicted) # This is the mean squared error
         r2 = r2_score(y_test, y_predicted) # This is the R2 score.
```

```
In [78]: # Print the model summary statistics. This is where you evaluate the performance of the model.

         print('Slope:' ,regression.coef_)
         print('Mean squared error: ', rmse)
         print("R2 score: ", r2)

         Slope: [[0.00333098]]
         Mean squared error:  235231497.7844783
         R2 score:  0.6128432918907266
```

An R2 score of 0.61 shows that this model is a poor/medium fit

```
In [23]: # Setup a folium map at a high-level zoom
         map = folium.Map(location = [100, 0], zoom_start = 1.5)

         # Choropleth maps bind Pandas Data Frames and json geometries.This allows us to quickly visualize data combinations
         folium.Choropleth(
             geo_data = country_geo,
             data = data_to_plot,
             columns = ['location', 'total_deaths'],
             key_on = 'feature.properties.name',
             fill_color = 'YlOrBr', fill_opacity=0.6, line_opacity=0.1,
             legend_name = "rating").add_to(map)
         folium.LayerControl().add_to(map)

         map

Out[23]:
```



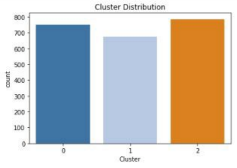
CODING AND VISUALISATION

Clusters Count Verification

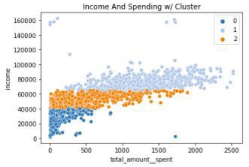
In [53]:
```
df['clusters'].value_counts()
```

Out[53]:
```
2    786
0    750
1    676
Name: clusters, dtype: int64
```

In [54]:
```
# we check the cluster counts and distribution visualized

sns.countplot(data=df, x="clusters", palette="tab20")
plt.title("Cluster Distribution")
plt.xlabel("Cluster")
plt.show()
```



In [55]:
```
pl = sns.scatterplot(data = df,x=df["total_amount__spent"], y=df["income"],hue=df["clusters"], palette="tab20")
pl.set_title("Income And Spending w/ Cluster")
plt.legend()
plt.show()
```



Outlier removal

In [29]:
```
# We saw in initial tests already some outliers. Removing them now
```

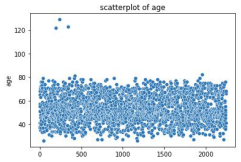
In [30]:
```
df.head()
```

Out[30]:

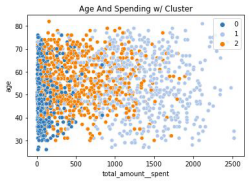
	year_birth	income	kidhome	teenhome	dt_customer	recency	mntwines	mntfruits	mntmeatproducts	mntfishproducts	mntsweetproducts	mntgoldprods	n
0	1957	58138.0	0	0	2012-04-09	58	635	88	546	172	88	88	
1	1954	46344.0	1	1	2014-08-03	38	11	1	6	2	1	6	
2	1965	71613.0	0	0	2013-08-21	26	426	49	127	111	21	42	
3	1984	26646.0	1	0	2014-10-02	26	11	4	20	10	3	5	
4	1981	58293.0	1	0	2014-01-19	94	173	43	118	46	27	15	

In [31]:
```
# Creating Age Column for easier outlier removal

df['age'] = 2022 -df['year_birth']
```

In [32]:
```
sns.scatterplot(data =df['age'])
plt.title('scatterplot of age');
```

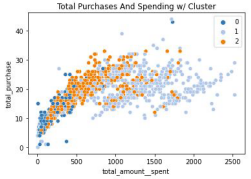


Here we can see several points which are cleary outliers. For removal we can set Age 100 as maximum to cover all



Here we have a clear distribution of Age Groups compared with total amount Spent confirming our initial Cluster

```
pl = sns.scatterplot(data = df,x=df["total_amount__spent"], y=df["total_purchase"],hue=df["clusters"], palette="tab20")
pl.set_title("Total Purchases And Spending w/ Cluster")
plt.legend()
plt.show()
```



And another Cluster showing the distribution of total amount spended and total purchases

```
df.columns
```

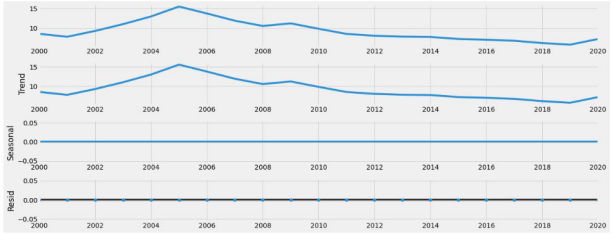
CODING AND VISUALISATION

3. Time series analysis: decomposition

In [19]:
```
# Decomposing the time series using an additive model

decomposing = sm.tsa.seasonal_decompose(data_sub, model='additive')
```

In [20]:
```
from pylab import rcParams # This will define a fixed size for all special charts.

rcParams['figure.figsize'] = 18, 7
```

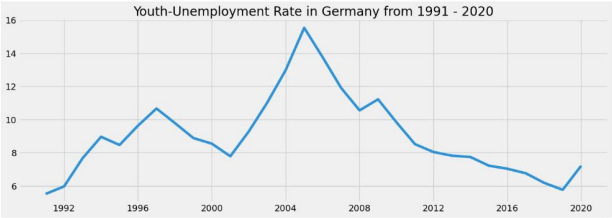
In [21]:
```
# Plotting the separate components

decomposing.plot()
plt.show()
```



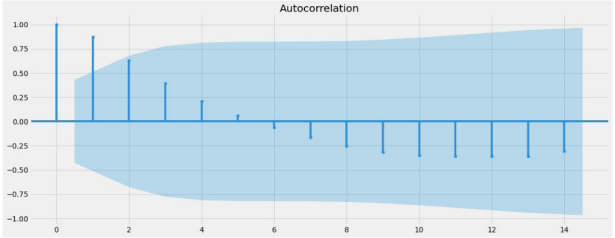
While there is no seasonal pattern, we can however relate the uprising unemployment starting 2001 until 2005 due to economics in that time, and the starting rize in 2019 is more likely related to the Covid disease. As younger Adults just finishing studies/school are still having trouble to find jobs because of the Restrictions which are just now started to be lifted. Once the Data is updated, i expect the Trend to go downwards again.

In [8]:
```
# Plot the data using matplotlib.

plt.figure(figsize=(15,5), dpi=100) # The dpi argument controls the quality of the visualization here.
# When it's set to 100, it will produce lower-than-standard quality, which is useful if, similar to this notebook,
# you'll have a lot of plots. A large number of plots will increase the size of the notebook,
# which could take more time to load and eat up a lot of RAM!
plt.title('Youth-Unemployment Rate in Germany from 1991 - 2020')
plt.plot(data)
```

Out[8]:
```
[<matplotlib.lines.Line2D at 0x1e8d593f790>]
```



In [23]:
```
# Checking out autocorrelations with a plot

from statsmodels.graphics.tsaplots import plot_acf, plot_pacf # Import autocorrelation and partial correlation plots.

plot_acf(data_sub)
plot_pacf
plt.show()
```



The vertical lines represent the lags in the series, while the blue area represents the confidence interval. When lines go above the blue edge of the confidence interval, this means there are lags that are significantly correlated with each other. As visibly most vertical lines are within the confidence interval, it can be deduced that the data is stationary. From now on and with updated Datainput, we could use this Trends in Time Series to directly see how , for example a covid Disease, Disasters can have effects on the Population and for example the unemployment Rate. This Analysis can be used for every Country we have Data from.

Evolution of Covid-19

- Since the Covid-19 Disease started to spread around the Globe in 2019, we can see the Evolution of the Disease in Every Country. From China beginning with a 0 Covid Tolerance, bringing a Lockdown to whole Cities with Millions of People, to Countries that denied the Existence of Covid contemplating a Conspiracy to other countries who have been entirely "relaxed."

- Overall, the World concluded that only together this can be defeated as even North Korea recently started to confirm and report Cases.

- However, despite the differences in Politics, worldwide, it has been shown the efficiency in Vaccinations that it is possible to Live with Covid.

- The whole presentation can be reviewed under **Tableau** and the Coding with Python on **Github**

Overall we can see that no matter how each country fights Covid-19, each has its unique way of success, and people learn to live with the virus.