Assignment 1

September 2, 2022

1 Assignment 1

Matthias Rathbun, Mir Khan, Jay Nagabhairu, Jason Ng, Phoebe Collins 09/02/2022

1.1 Import Libraries and Initialize Notebook

```
[1]: import pandas as pd
     import numpy as np
     import matplotlib.pyplot as plt
     import seaborn as sns
     import statsmodels
     import statsmodels.api as sm
     import statsmodels.formula.api as smf
     import plotly.graph_objects as go
     import plotly.express as px
     pd.set_option('display.float_format', lambda x: '%.3f' % x)
```

1.2 Problem 1.1 (8 Points) Read the EXCLE file "COVID_08312020.csv"

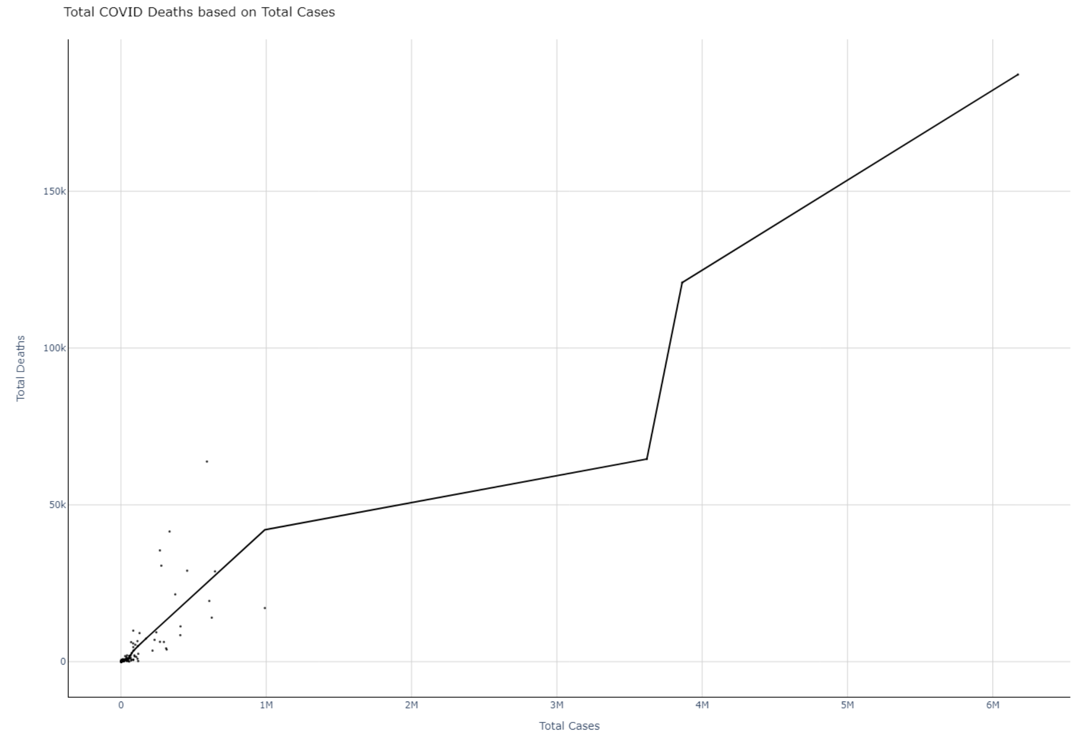
```
[2]: df = pd.read_csv("COVID_08312020.csv")
```

```
[3]: df.head()
```

```
[3]:        Country  Total Cases  Total Deaths  TOTCases_1M  TOTDeath_!M  \
     0  Afghanistan        38162          1402          977           36
     1      Albania         9380           280         3260           97
     2       Angola         2624           107           79            3
     3    Argentina       408426          8457         9023          187
     4      Armenia        43750           877        14760          296

        TotalTested
     0       102598
     1        57618
     2        64747
     3      1242269
     4       205450
```

1.3 Problem 1.2 (8 Points) Produce a scatter plot using "TotalCases" and "TotalDeaths" and impose a loess line on the top of the data.

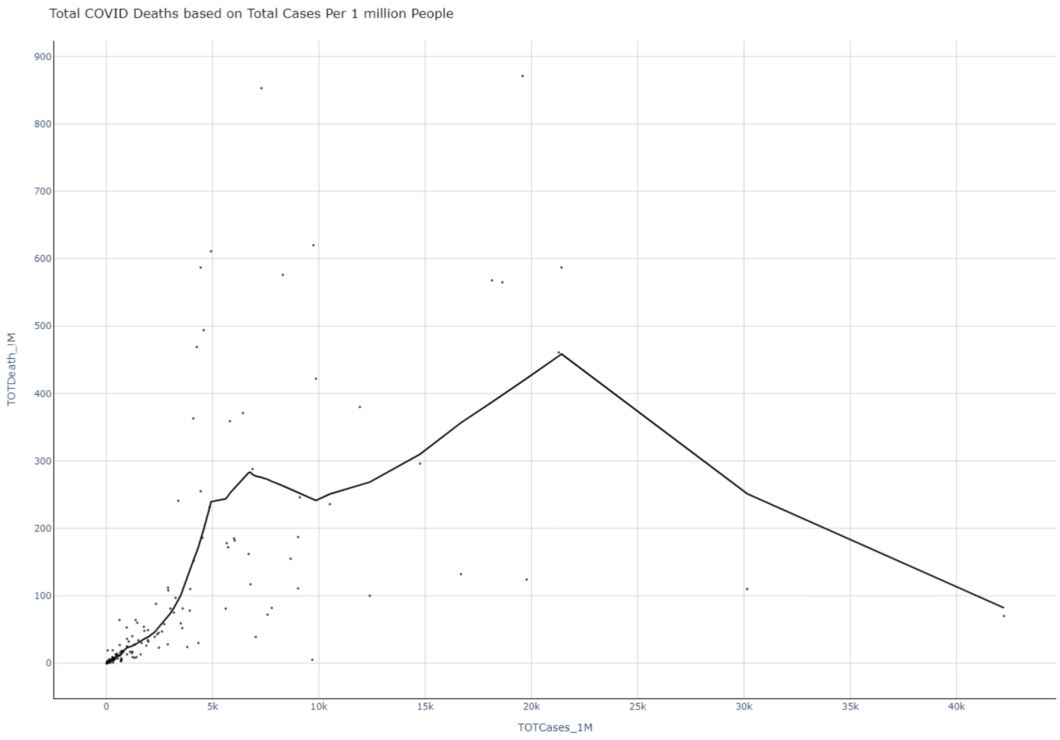
[4]:
```
fig = px.scatter(df, x=df['Total Cases'], y=df['Total Deaths'],
                 opacity=0.8, color_discrete_sequence=['black'], trendline =␣
 ↪"lowess", trendline_options=dict(frac=0.2))

# Change chart background color
fig.update_layout(dict(plot_bgcolor = 'white'))

# Update axes lines
fig.update_xaxes(showgrid=True, gridwidth=1, gridcolor='lightgrey',
                 zeroline=True, zerolinewidth=1, zerolinecolor='lightgrey',
                 showline=True, linewidth=1, linecolor='black')

fig.update_yaxes(showgrid=True, gridwidth=1, gridcolor='lightgrey',
                 zeroline=True, zerolinewidth=1, zerolinecolor='lightgrey',
                 showline=True, linewidth=1, linecolor='black')

# Set figure title
fig.update_layout(title=dict(text="Total COVID Deaths based on Total Cases",
                             font=dict(color='black')))

# Update marker size
fig.update_traces(marker=dict(size=3))
fig.update_layout(
    autosize=True,
    height=1000,)

fig.show()
```



1.4 Problem 1.3 (8 Points) Produce a scatter plot using “ToTCases_1M” and “TotDeath_MPOP” and impose a loess line on the top of the data.

[5]:

```
fig = px.scatter(df, x=df['TOTCases_1M'], y=df['TOTDeath_!M'],
                 opacity=0.8, color_discrete_sequence=['black'], trendline =␣
 ↪"lowess", trendline_options=dict(frac=0.2))

# Change chart background color
fig.update_layout(dict(plot_bgcolor = 'white'))

# Update axes lines
fig.update_xaxes(showgrid=True, gridwidth=1, gridcolor='lightgrey',
                 zeroline=True, zerolinewidth=1, zerolinecolor='lightgrey',
                 showline=True, linewidth=1, linecolor='black')

fig.update_yaxes(showgrid=True, gridwidth=1, gridcolor='lightgrey',
                 zeroline=True, zerolinewidth=1, zerolinecolor='lightgrey',
                 showline=True, linewidth=1, linecolor='black')

# Set figure title
fig.update_layout(title=dict(text="Total COVID Deaths based on Total Cases Per␣
 ↪1 million People",
                             font=dict(color='black')))

# Update marker size
fig.update_traces(marker=dict(size=3))
fig.update_layout(
    autosize=True,
    height=1000,)

fig.show()
```



1.5 Problem 1.4 (8 Points) Produce a table with the following summary statistic including minimum, mean, median, variance, standard deviation, maximum, and skewness for the following five variables "ToTCases_1M", "TotDeath_MPOP", "TotalCases", "TotalDeaths", and "TotalTested". (Note: Display only three decimal place)

[6]:

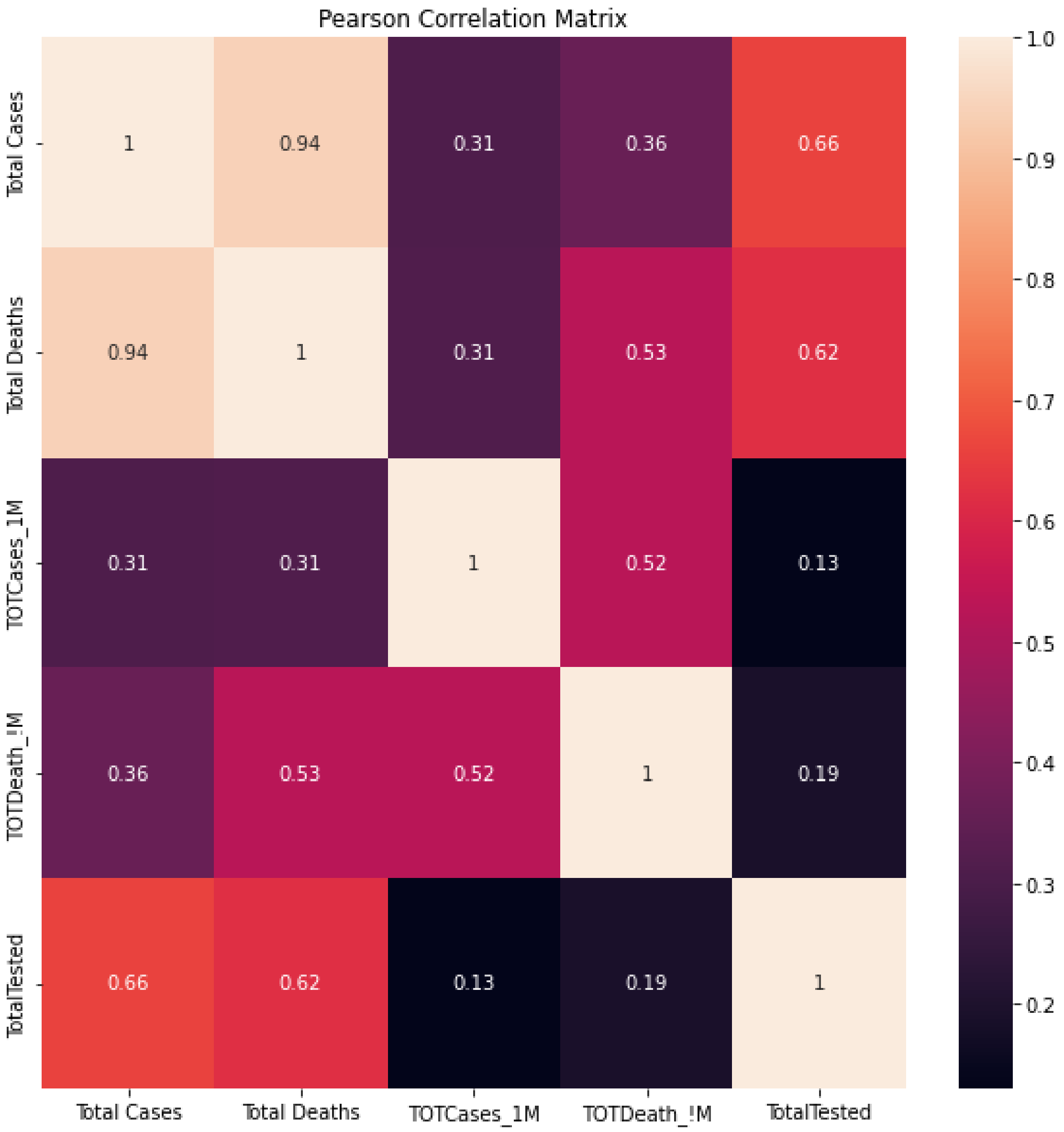
```
df.agg(
    {
        "Total Cases": ["min", "max", "mean", "median", "var", "std","skew"],
        "Total Deaths": ["min", "max", "mean", "median", "var", "std","skew"],
        "TOTCases_1M": ["min", "max", "mean", "median", "var", "std","skew"],
        "TOTDeath_!M": ["min", "max", "mean", "median", "var", "std","skew"],
        "TotalTested": ["min", "max", "mean", "median", "var", "std","skew"]
    }
)
```

```
[6]:             Total Cases  Total Deaths  TOTCases_1M  TOTDeath_!M  \
     min             355.000         1.000       11.000        0.000
     max         6173236.000    187224.000    42230.000      871.000
     mean         181486.137      6091.115     4177.388      115.187
     median        24367.000       411.000     1789.000       34.000
     var    476745369893.148 439344669.045 38146725.660    32155.689
     std          690467.501     20960.550     6176.304      179.320
     skew              6.836         6.343        3.066        2.229

                  TotalTested
     min              120.000
     max         90410000.000
     mean         3141261.633
     median        404944.000
     var    128072560142340.500
     std          11316914.780
     skew               6.328
```

1.6 Problem 1.5 (8 Points) Obtain both the Spearman correlation and the Pearson correlation between the following variables "ToTCases_1M", "TotDeath_MPOP", "TotalCases", "TotalDeaths", and "TotalTested".

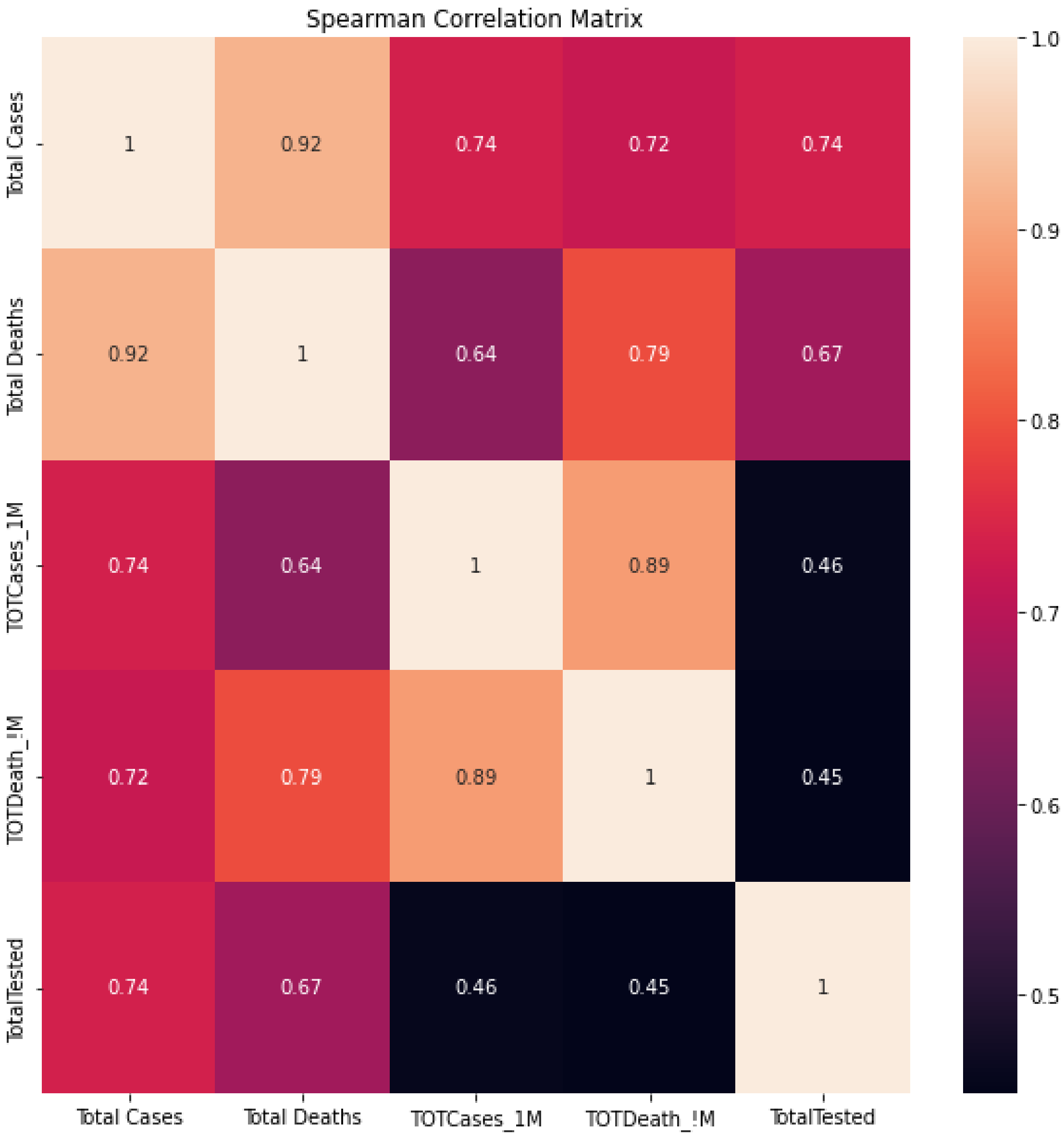
```
[7]: corr = df.corr(method = "pearson")
     plt.figure(figsize = (10,10))
     sns.heatmap(corr, annot = True).set(title = "Pearson Correlation Matrix")
```

```
[7]: [Text(0.5, 1.0, 'Pearson Correlation Matrix')]
```



[8]:
```
corr = df.corr(method = "spearman")
plt.figure(figsize = (10,10))
sns.heatmap(corr, annot = True).set(title = "Spearman Correlation Matrix")
```

[8]:
```
[Text(0.5, 1.0, 'Spearman Correlation Matrix')]
```



1.7 Fill in the blank answers:

1. **6 ; 7**
2. **4**
3. **inner**
4. **regression analysis ; 1000 ; 4**
5. **non-supervised learning**
6. **parametric analysis**
7. **Inference**