SANUSI YUSUF OLADIPUPO

FE/23/61865754

COHORT 2

Capstone Project

Predictive Modelling for COVID-19 in Public Health

CONTENT

INTRODUCTION

Overview

In response to the COVID-19 pandemic, public health organizations have faced immense challenges in predicting the spread of the virus and understanding key factors that influence transmission and patient outcomes.

Purpose of the Report

The purpose of the report is to explain the necessary analytic steps and modeling concepts used to predict the number of possible new cases, so as to prepare the organization to take actionable insights to inform policies, anticipate future outbreaks, and improve health resource allocation.

Overview of the Analysis and Modeling Process

- The data was extracted from Kaggle.
- Necessary Libraries were imported to carry out the analytic process such as Pandas.
- The data was imported into Jupyter Notebook using read_csv.
- Data cleaning was carried out to ensure the data was perfect for analysis.
- Data Normalization followed with the purpose of eliminating high variation within the data set and ensures a more accurate model.
- Feature engineering was carried out to give the model more accuracy.
- Exploratory Data Analysis to give a good insight into the data set.
- Correlation carried out showed the relevant and irrelevant data columns.
- Irrelevant data columns were dropped to prevent over-fitting of our model.
- Necessary libraries were imported for modeling.
- The data was prepared for training ensuring a high correlation within the remaining relevant columns.
- Data split into Train and Test data.
- ARIMA model was used to train.
- Predictions generated and model evaluated with MSE and RMSE equivalent to approx. 0.0053 and 0.073 respectively indicating a highly accurate and reliable model.

DATA DESCRIPTION

The Dataset used for analysis was downloaded through the link provided in the learning material. The link directed to path of download to kaggle where the dataset was extracted from.

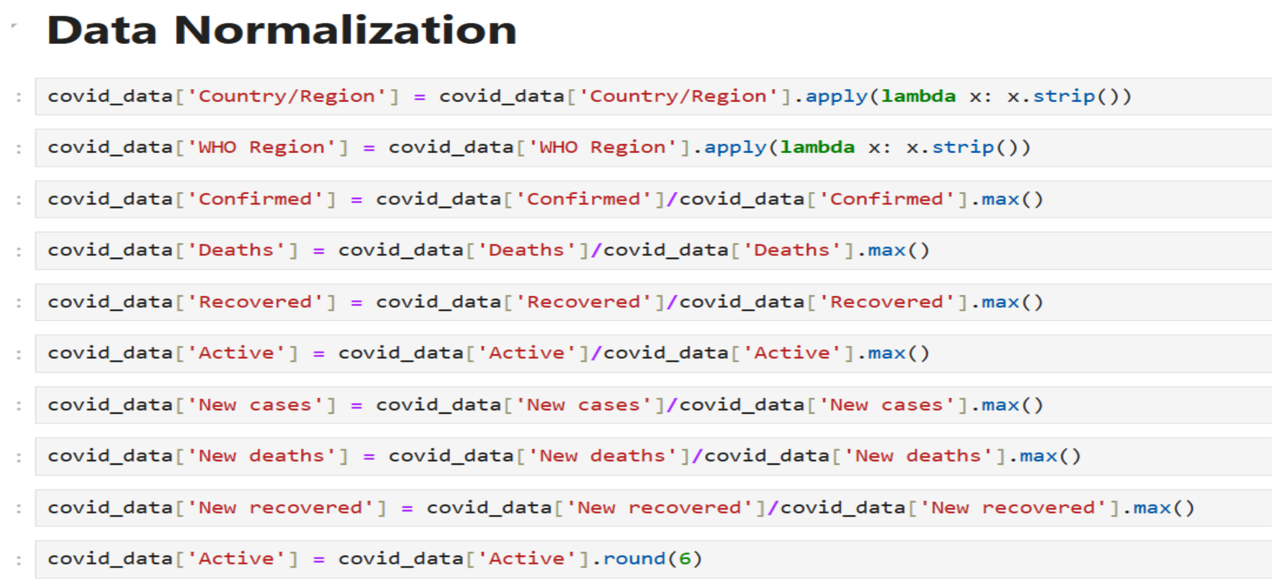
After the download, the file path was copied and pd.read_csv was used to move the dataset into python using Jupyter Notebook.

Observations in the Dataset

- The Dataset consisted of 10 columns.
- There is no missing data or duplicates.
- The Date column datatype was Object.

DATA PREPROCESSING

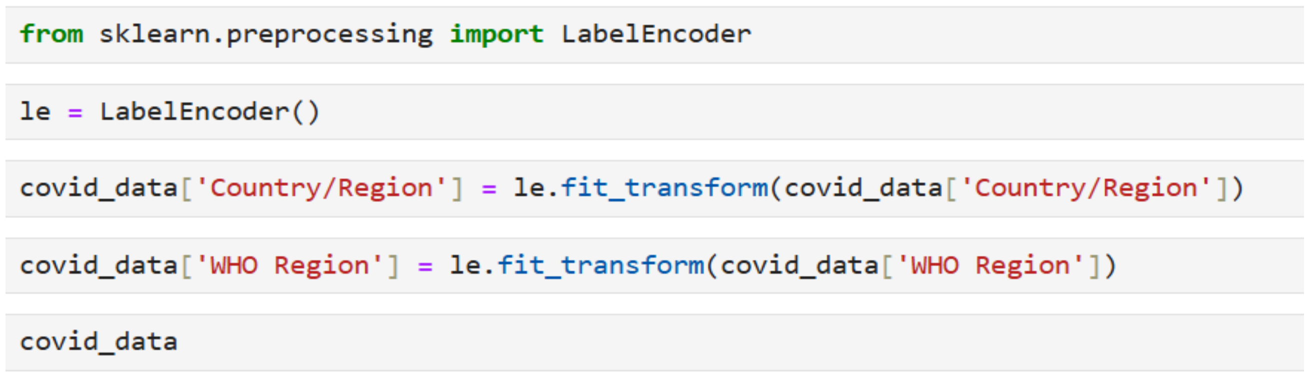
- No missing data or duplicates observed.
- The date column that was initially a string was converted to datetime.
- Data Normalization was done using each value divided by the maximum value to ensure the values were set between 0 and 1 preventing variation and model overfitting.
- The categorical variables were converted to numeric values using LabelEncoder.
- Feature Engineering created more columns to ensure an effective dataset for modeling.



Data Normalization

```
covid_data['Country/Region'] = covid_data['Country/Region'].apply(lambda x: x.strip())
covid_data['WHO Region'] = covid_data['WHO Region'].apply(lambda x: x.strip())
covid_data['Confirmed'] = covid_data['Confirmed']/covid_data['Confirmed'].max()
covid_data['Deaths'] = covid_data['Deaths']/covid_data['Deaths'].max()
covid_data['Recovered'] = covid_data['Recovered']/covid_data['Recovered'].max()
covid_data['Active'] = covid_data['Active']/covid_data['Active'].max()
covid_data['New cases'] = covid_data['New cases']/covid_data['New cases'].max()
covid_data['New deaths'] = covid_data['New deaths']/covid_data['New deaths'].max()
covid_data['New recovered'] = covid_data['New recovered']/covid_data['New recovered'].max()
covid_data['Active'] = covid_data['Active'].round(6)
```

Figure 1- Frequency of Region

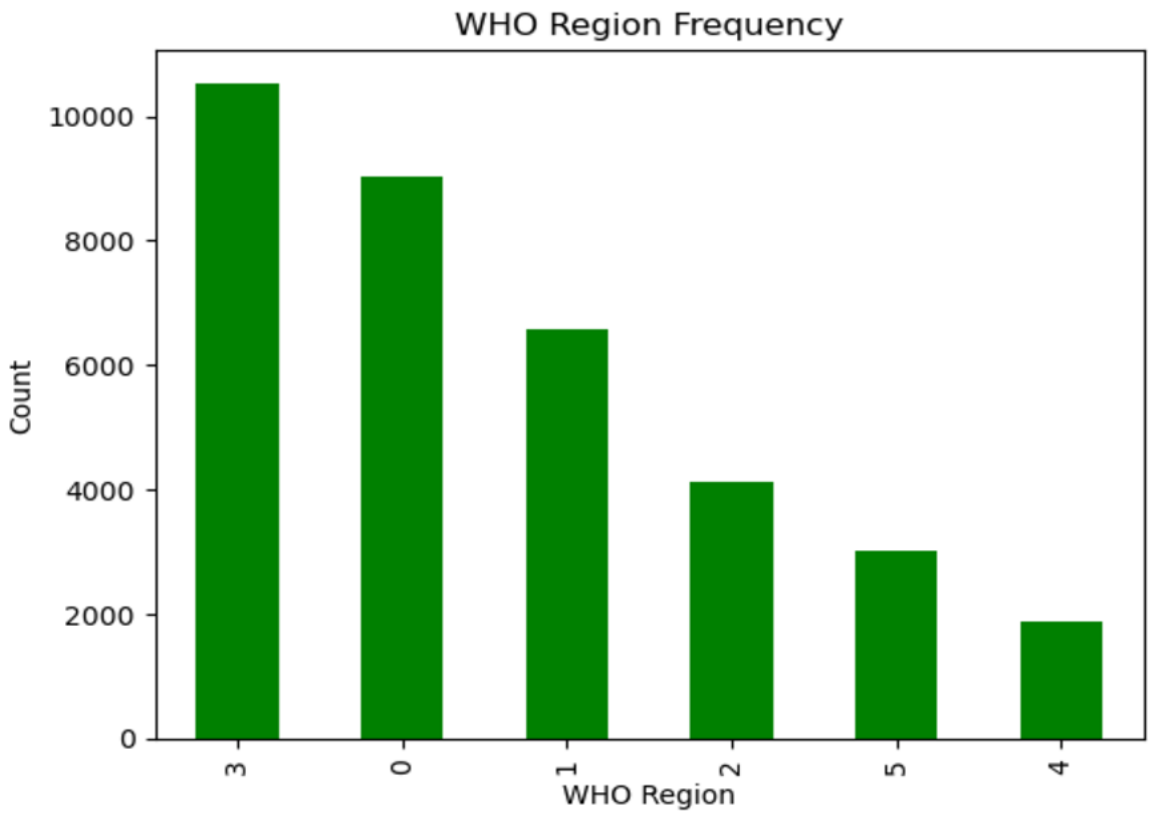


```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
covid_data['Country/Region'] = le.fit_transform(covid_data['Country/Region'])
covid_data['WHO Region'] = le.fit_transform(covid_data['WHO Region'])
covid_data
```

EXPLORATORY DATA ANALYSIS (EDA)

- Bar chat, Scatterplots, and heatmap were used.
- The Bar chat was useful for visualizing the categorical data.
- The Scatter plots gave an idea of the relationship within the continuous variables in the dataset.
- The heatmap clearly showed the relationship within the dataset.
- Before using the heatmap, the scatter plots already showed that there is no correlation between some variables. The function (.corr()) was used to visualize the relationship within the dataset columns.
- Therefore, Columns that showed no correlation with others were dropped.
- The heatmap was then used to ensure that the remaining values in the datasets are highly related.
- The date column showed no correlation with other columns and was not dropped but used as an index to help during prediction.

```
:   Text(0, 0.5, 'Count')
```



```
plt.xlabel('Confirmed')
plt.ylabel('Deaths')
plt.show()
```

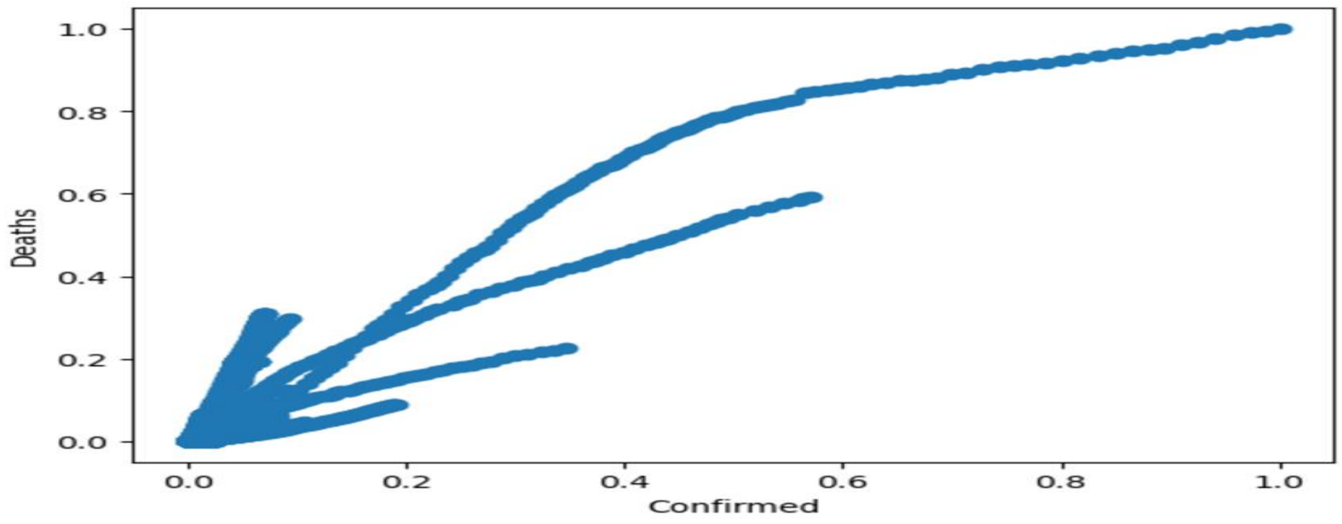


Figure 2 **Relationship between Confirmed and Deaths**

```
plt.scatter(covid_data['Confirmed'], covid_data['Recovered'])
plt.xlabel('Confirmed')
plt.ylabel('Recovered')
plt.show()
```

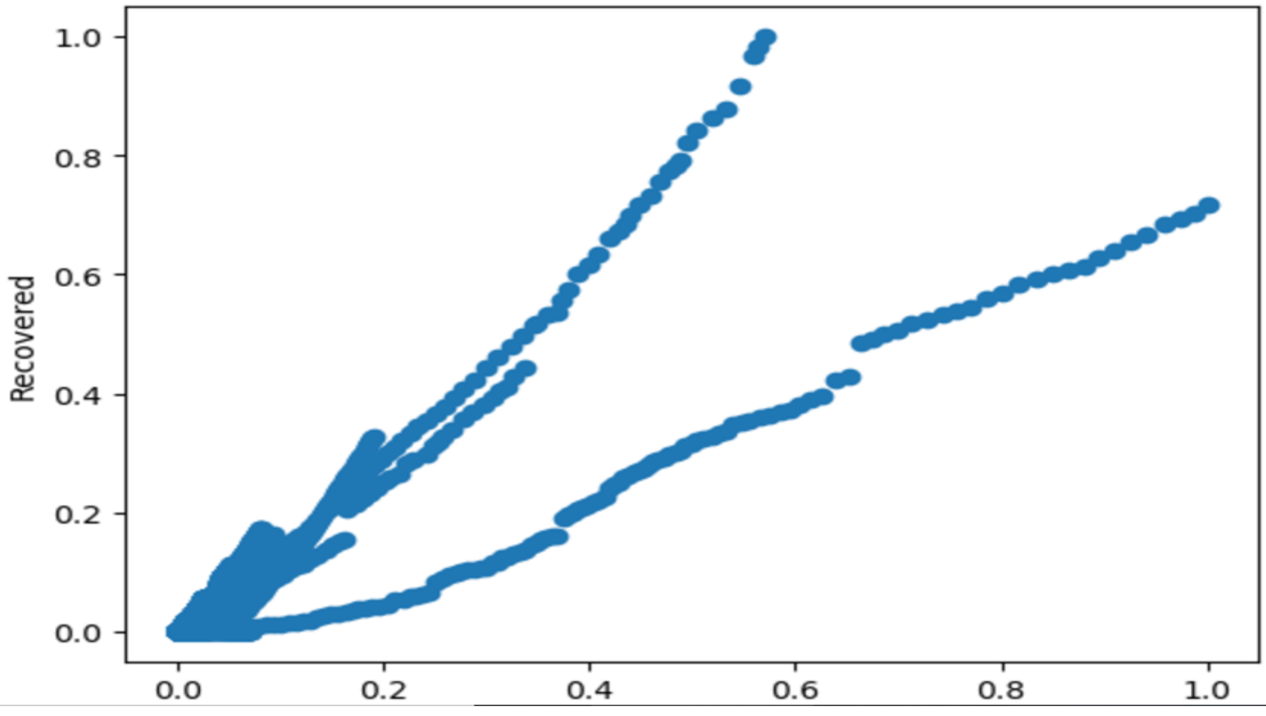


Figure 3: **Relationship between Recovered and Confirmed**

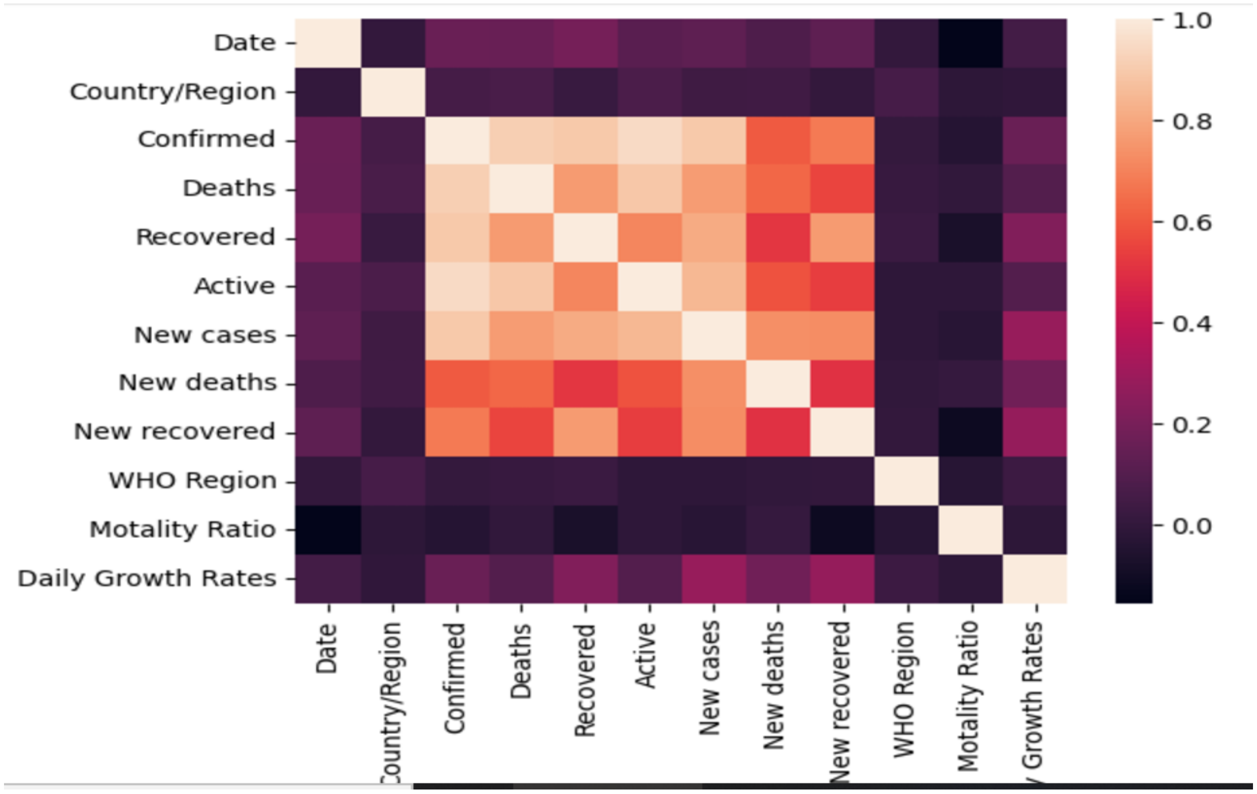


Figure 4: **Heatmap showing correlation in all the datasets**

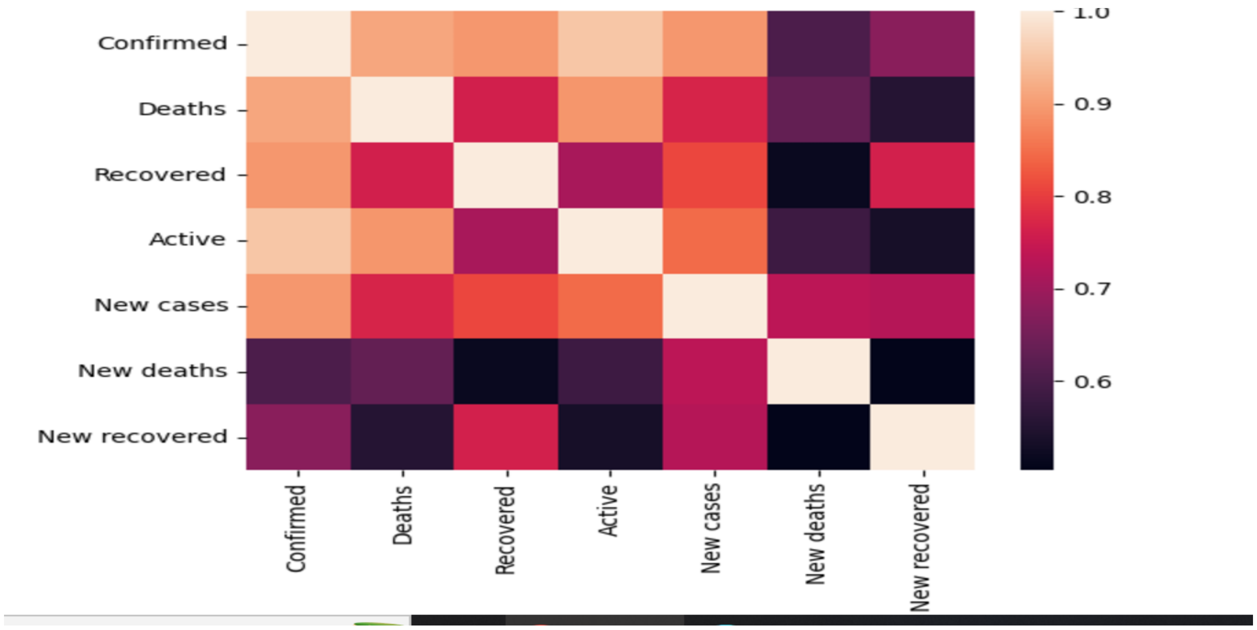
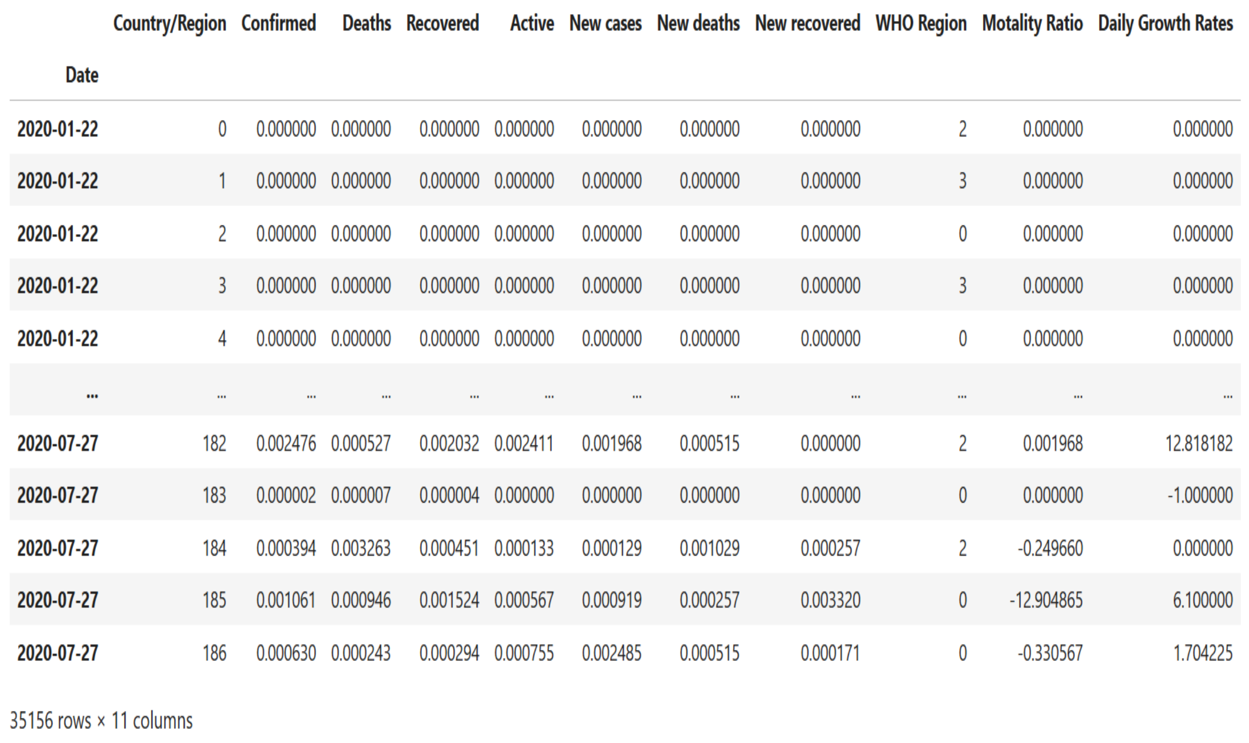


Figure 5: **Heatmap showing correlation after the remover of irrelevant columns**



	Country/Region	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered	WHO Region	Motality Ratio	Daily Growth Rates
Date											
2020-01-22	0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	2	0.000000	0.000000
2020-01-22	1	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	3	0.000000	0.000000
2020-01-22	2	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0	0.000000	0.000000
2020-01-22	3	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	3	0.000000	0.000000
2020-01-22	4	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0	0.000000	0.000000
...	...	...	...	...	...	...	...	...	...	...	...
2020-07-27	182	0.002476	0.000527	0.002032	0.002411	0.001968	0.000515	0.000000	2	0.001968	12.818182
2020-07-27	183	0.000002	0.000007	0.000004	0.000000	0.000000	0.000000	0.000000	0	0.000000	-1.000000
2020-07-27	184	0.000394	0.003263	0.000451	0.000133	0.000129	0.001029	0.000257	2	-0.249660	0.000000
2020-07-27	185	0.001061	0.000946	0.001524	0.000567	0.000919	0.000257	0.003320	0	-12.904865	6.100000
2020-07-27	186	0.000630	0.000243	0.000294	0.000755	0.002485	0.000515	0.000171	0	-0.330567	1.704225

35156 rows × 11 columns

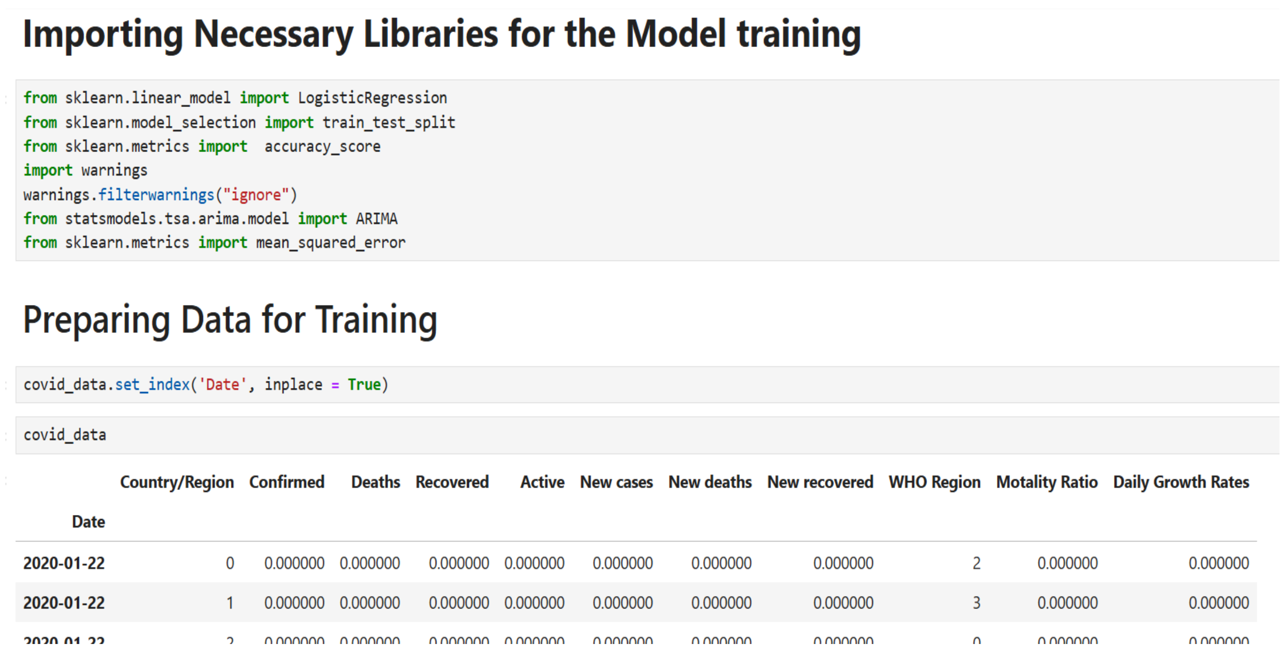
***Figure 6*: Date used as an index to give a more accurate model**

MODEL SELECTION AND TRAINING

- Time series model was used.
- The dataset was taken at regular time interval.
- The model was used to forecast future New-cases in a time series based on past patterns and trends in the covid-19 dataset.
- The time series model used was an Autoregressive (AR) model.
- Auto-Regressive Integrated Moving Average also known as ARIMA was used to build the model.

Procedure

- Data was prepared for training.
- Irrelevant columns were removed.
- Date was set as the Index.
- Data was split into train (80%) and test (20%) data.
- Created and fitted ARIMA model.
- Predictions were generated and the Model was evaluated using Mean Squared Error (MSE).
- The Root Mean Squared Error (RMSE) was used to provide a more interpretable error metric.
- The MSE and RMSE indicated a high-performing, reliable, and accurate model.



Importing Necessary Libraries for the Model training

```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import  accuracy_score
import warnings
warnings.filterwarnings("ignore")
from statsmodels.tsa.arima.model import ARIMA
from sklearn.metrics import mean_squared_error
```

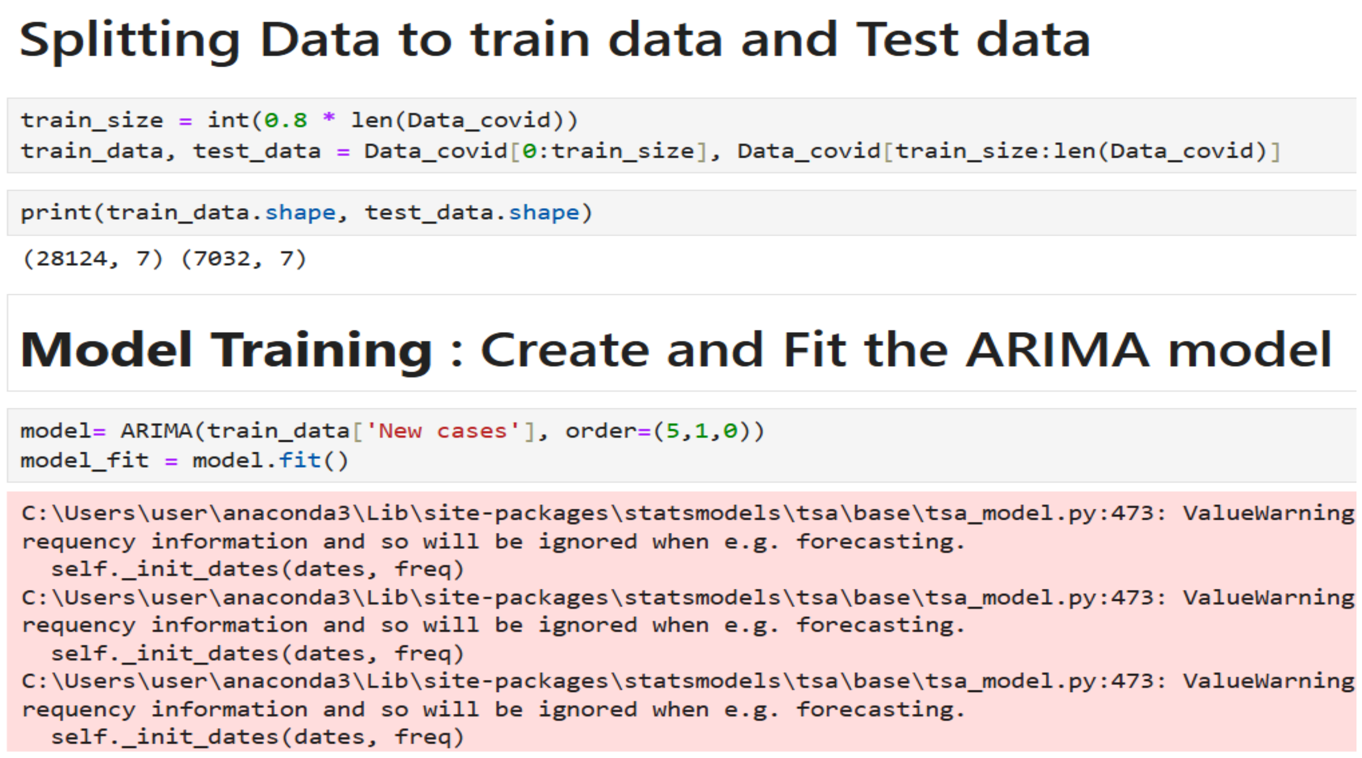
Preparing Data for Training

```
covid_data.set_index('Date', inplace = True)
```

```
covid_data
```

	Country/Region	Confirmed	Deaths	Recovered	Active	New cases	New deaths	New recovered	WHO Region	Motality Ratio	Daily Growth Rates
Date											
2020-01-22	0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	2	0.000000	0.000000
2020-01-22	1	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	3	0.000000	0.000000
2020-01-22	2	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0	0.000000	0.000000

Preparing Data for Training



Splitting Data to train data and Test data

```
train_size = int(0.8 * len(Data_covid))
train_data, test_data = Data_covid[0:train_size], Data_covid[train_size:len(Data_covid)]
```

```
print(train_data.shape, test_data.shape)
```

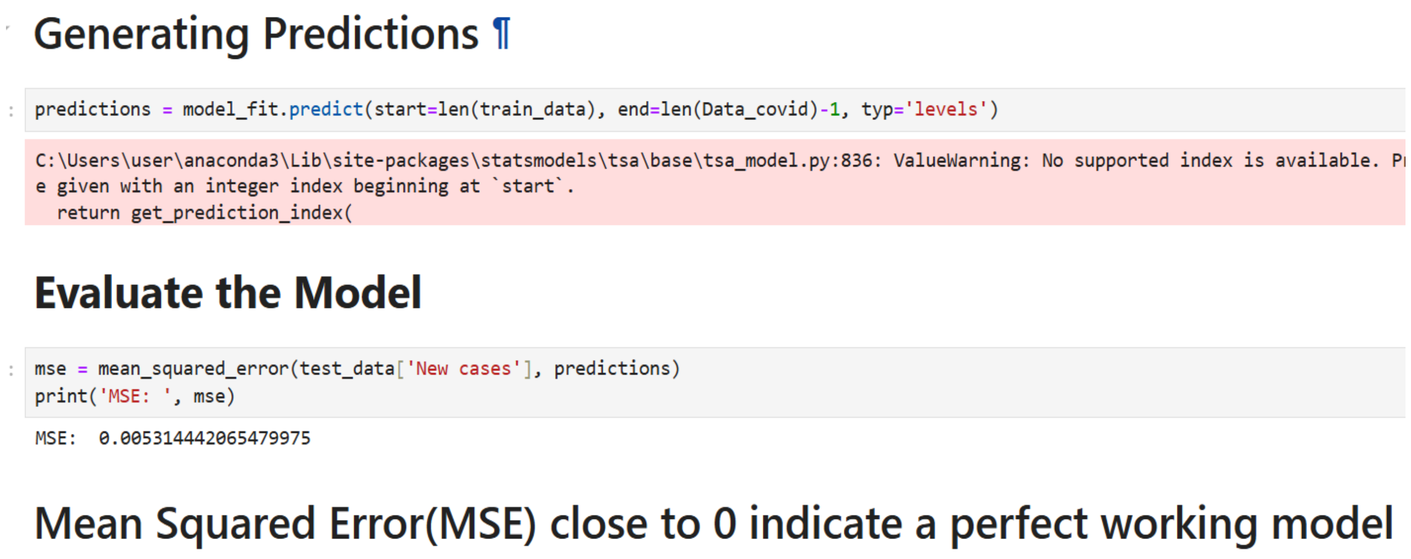
```
(28124, 7) (7032, 7)
```

Model Training : Create and Fit the ARIMA model

```
model= ARIMA(train_data['New cases'], order=(5,1,0))
model_fit = model.fit()
```

```
C:\Users\user\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning
requency information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
C:\Users\user\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning
requency information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
C:\Users\user\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning
requency information and so will be ignored when e.g. forecasting.
  self._init_dates(dates, freq)
```

Splitting and Model Fitting using ARIMA model



Generating Predictions ¶

```
predictions = model_fit.predict(start=len(train_data), end=len(Data_covid)-1, typ='levels')
```

```
C:\Users\user\anaconda3\Lib\site-packages\statsmodels\tsa\base\tsa_model.py:836: ValueWarning: No supported index is available. Pr
e given with an integer index beginning at `start`.
  return get_prediction_index(
```

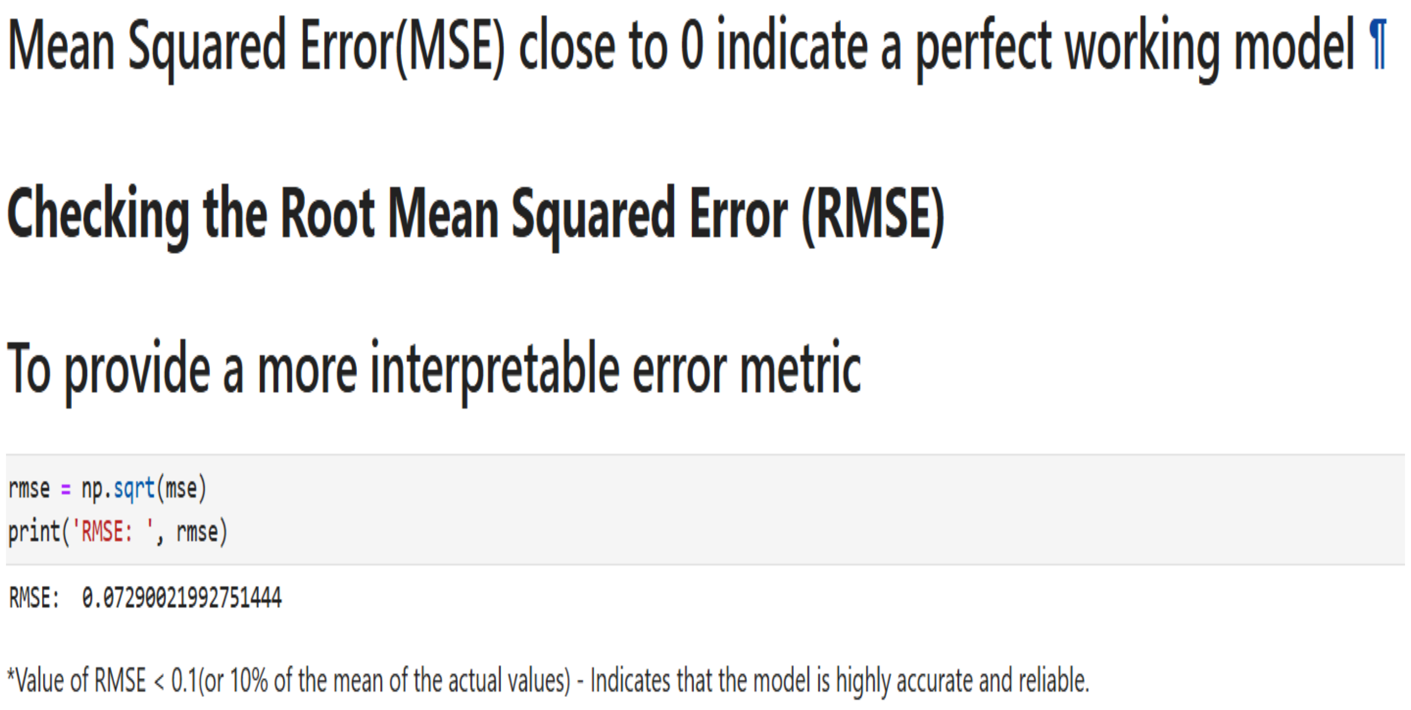
Evaluate the Model

```
mse = mean_squared_error(test_data['New cases'], predictions)
print('MSE: ', mse)
```

```
MSE:  0.005314442065479975
```

Mean Squared Error(MSE) close to 0 indicate a perfect working model

Generating Prediction and Evaluating the MSE



Mean Squared Error(MSE) close to 0 indicate a perfect working model ¶

Checking the Root Mean Squared Error (RMSE)

To provide a more interpretable error metric

```
rmse = np.sqrt(mse)
print('RMSE: ', rmse)
```

```
RMSE:  0.07290021992751444
```

*Value of RMSE < 0.1(or 10% of the mean of the actual values) - Indicates that the model is highly accurate and reliable.

RMSE evaluation

CONCLUSION AND RECOMMENDATIONS

- It was observed that there are high correlations amidst the Number of cases confirmed, deaths recoded, Number recovered, Active population, New cases, New deaths, and New recovered.
- The model was based and built on this observation.
- The result of the accuracy of the model showed that it can be used to predicts almost accurate number of new occurrence of covid-19.
- More data can be collected to future test the model.
- The potential application of the model is to quickly notify the organization of any possible future occurrence.

REFERENCES

- https://www.kaggle.com/datasets/imdevskp/corona-virus-report.