DATA ANALYSIS REPORT (TEAM 5)

Technical Team Challenge

Mohamed Mohamed, Joel Aderemi, Taiye Lawal, Nonyelum Anigbo, Anndior Boateng and Muhsen Hussein

27.07.2021
HDR UK Black Internship

INTRODUCTION

For our project we investigated the global effect of the COVID-19 pandemic. Since its emergence in late 2019, the COVID-19 virus has had an unprecedented effect on many people' lives across the world and cost over 4 million lives. Our dataset contains figures such as the number of cases from COVID-19, numbers of deaths from COVID-19 and recovery rates, organised by country and continent. Our aim was to create a data model that provided the user with a simple way to view COVID-19 cases and deaths per country.

DATA EXPLORATION

How The Data Was Collected

The dataset was collected from Kaggle Datasets. This dataset contains summary COVI-19 related cases for each of the 220 countries, as of 30th of June 2021. It was downloaded into a desktop folder which was later loaded into pandas dataframe for analysis.

Features Identified for Analysis

The features identified for the analysis are total confirmed cases, total recovered cases, total deaths, continents, and population. The reason for choosing these features is simply because we want to know how COVID-19 has affected various parts of the world by comparing confirmed cases with death rate and how people are recovering using their populations.

Screenshots of Pandas-Profiling Reports

```
In [1]: # Libraries/Packages
        import os
        import pandas as pd
        import numpy as np
        from matplotlib import pyplot as plt
        import seaborn as sns
        %matplotlib inline

In [12]: sns.set_style("whitegrid")

In [3]: dataset = pd.read_csv("C:\\Users\\joela\\Desktop\\COVID-19\\worldometer_coronavirus_summary_
        data.csv")

In [4]: dataset

Out[4]:
```

	Country	Continent	Total_Confirmed	Total_Deaths	Total_Recovered	Active_Cases	Serious_or_Critical	Total_Ca
0	Afghanistan	Asia	120216	4962	71012	44242	1124.0	
1	Albania	Europe	132521	2456	130009	56	3.0	
2	Algeria	Africa	139626	3716	97089	38821	32.0	
3	Andorra	Europe	13911	127	13720	64	3.0	
4	Angola	Africa	38849	900	33242	4707	11.0	
...	...	...	...	...	...	...	...	
215	Wallis And Futuna Islands	Australia/Oceania	445	7	438	0	0.0	
216	Western Sahara	Africa	10	1	8	1	0.0	
217	Yemen	Asia	6920	1361	4068	1491	23.0	
218	Zambia	Africa	154948	2199	131792	20957	1342.0	
219	Zimbabwe	Africa	49864	1789	39121	8954	10.0	

220 rows × 12 columns

Fig 1: Loading the Dataset in Pandas Dataframe

```
In [29]: # Line Plot to Investigate Population by Continent
         ax = sns.lineplot(x = 'Continent', y = 'Population', data = dataset, ci = None, estimator =
         sum)
         dataset[['Population','Continent']].groupby('Continent').agg({'Population':'sum'})
```

Out[29]:

Continent	Population
Africa	1372380171
Asia	4643241629
Australia/Oceania	42789204
Europe	748080258
North America	593698920
South America	434293623

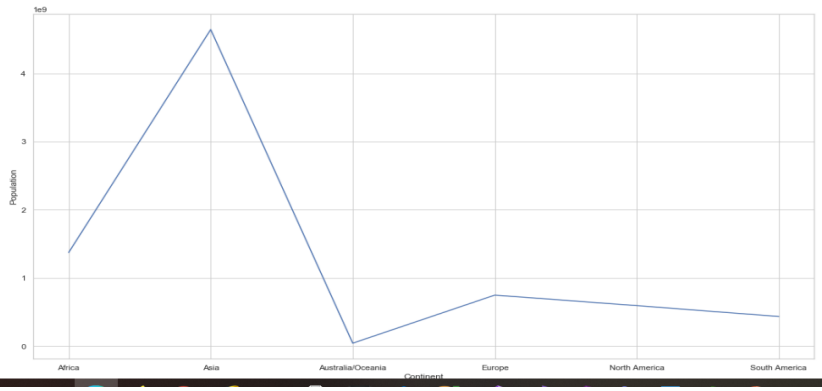


Fig 2: Line Plot Showing Population Distributions

```
In [26]: # Bar Plot to Investigate Total Confirmed Cases by Continents
         bx = sns.barplot(x = 'Continent', y = 'Total_Confirmed', data = dataset, estimator = sum)
         dataset[['Total_Confirmed','Continent']].groupby('Continent').agg({'Total_Confirmed':'sum'})
```

Out[26]:

Continent	Total_Confirmed
Africa	5554718
Asia	55851497
Australia/Oceania	74944
Europe	47982001
North America	40614193
South America	32911345

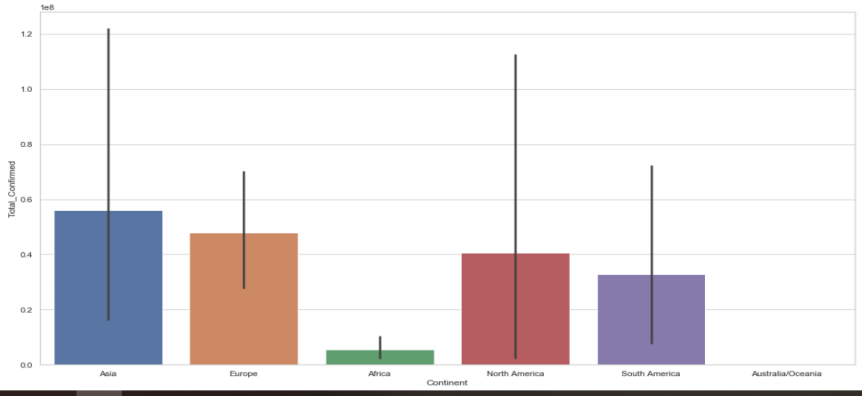


Fig 3: Bar Plot Showing Total Confirmed Cases by Continents

```
In [43]: # Bar Plot to Investigate Total Deaths by Continents
         fx = sns.barplot(x = 'Continent', y = 'Total_Deaths', data = dataset, estimator = sum)
         dataset[['Total_Deaths','Continent']].groupby('Continent').agg({'Total_Deaths':'sum'})
```

Out[43]:

Continent	Total_Deaths
Africa	143360
Asia	792039
Australia/Oceania	1283
Europe	1102752
North America	918502
South America	1005040

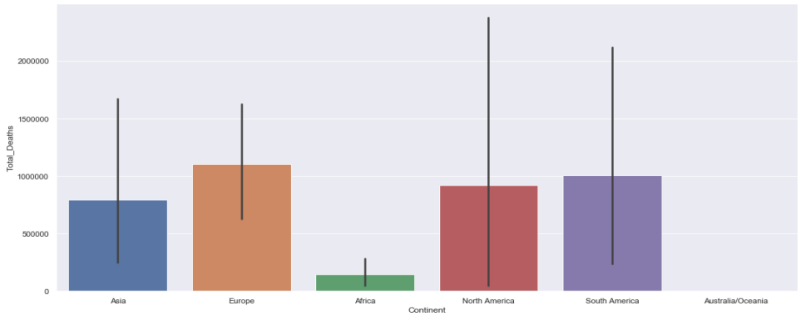


Fig 4: Bar Plot Showing Total Deaths by Continents

```
In [44]: # Bar Plot to Investigate Total Recovery Cases by Continents
         cx = sns.barplot(x = 'Continent', y = 'Total_Recovered', data = dataset, estimator = sum)
         dataset[['Total_Recovered','Continent']].groupby('Continent').agg({'Total_Recovered':'sum'})
```

Out[44]:

Continent	Total_Recovered
Africa	4846702
Asia	53234734
Australia/Oceania	68857
Europe	45429563
North America	34103594
South America	28169452

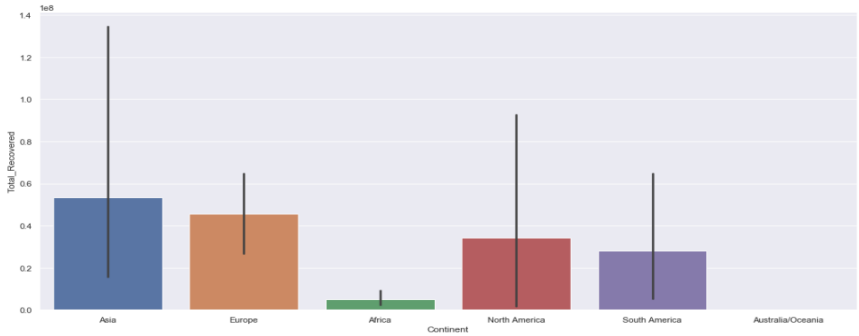


Fig 5: Bar Plot Showing Total Recovery Cases by Continents

```
In [32]: # Mean of Total Deaths
         mean = dataset['Total_Deaths'].mean()
         mean

Out[32]: 18013.52727272727

In [34]: hx = dataset['Total_Deaths'].values
         sns.distplot(hx)
         mean = dataset['Total_Deaths'].mean()
         mean
         plt.axvline(mean,0,1, color='red')

Out[34]: <matplotlib.lines.Line2D at 0x224e72e3c48>
```

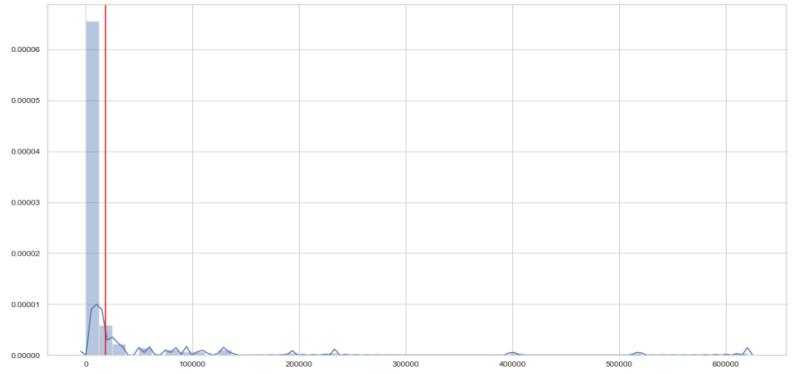


Fig 6: Histogram Showing Mean Total Deaths Distributions

Visualisations Generated Using Power BI

We also generated visualisations with the help of Power BI to compare the analysis with the one obtained from pandas-profiling reports.

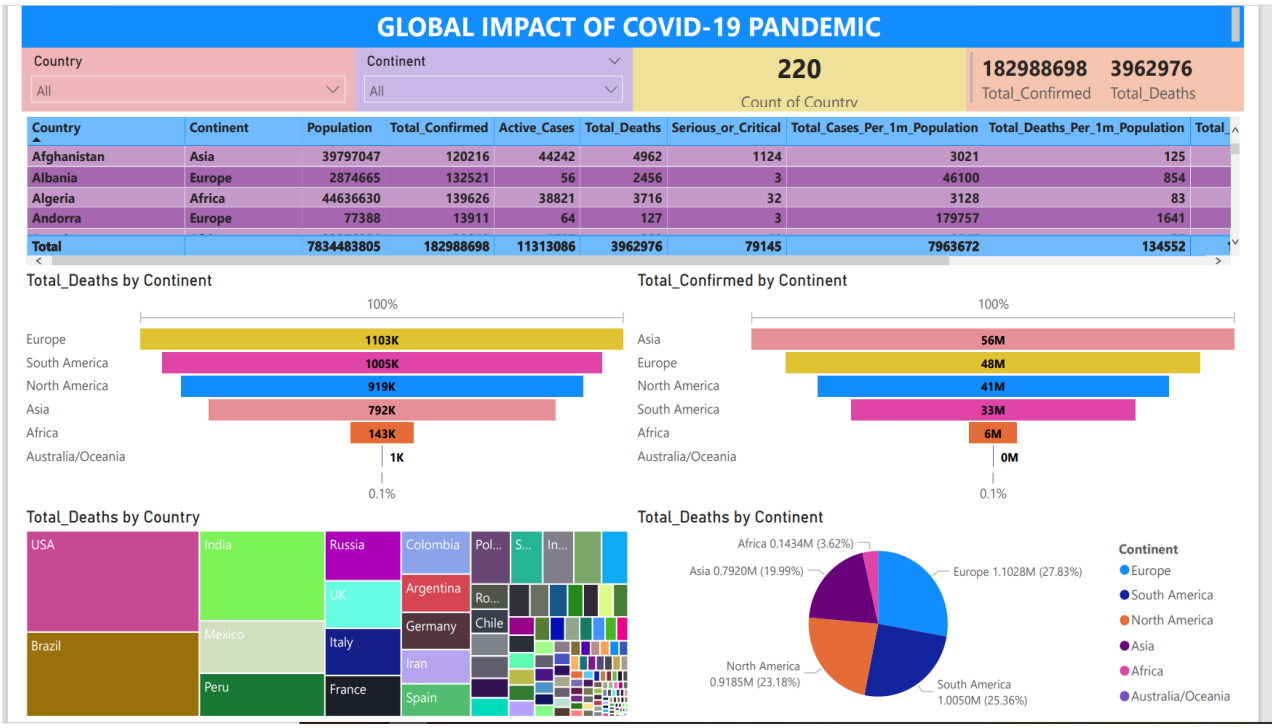


Fig 7: Visualisation Generated Using Power BI

METHODS

Pre-Processing Techniques Used

The following are some of the pre-processing techniques we carried out:

Loading the Dataset: after downloading the dataset, the first pre-processing technique we did was to load the dataset. This was done by the initial importing of python libraries such Pandas, Matplotlib, Seaborn, etc. The dataset was downloaded and named as a csv file and then loaded into Pandas data frame for cleaning and exploratory analysis.

Understanding the Dataset: this was done by knowing the features each column stands for to avoid mistakes in data analysis and modelling. We created a data frame with the names of the columns, data types, the first and last few rows' values, unique column values and statistical summary from the data dictionary

Dataset Cleaning: the dataset cleaning was done by writing python code that checked for any null value.

MODELLING AND RESULTS

Using the coronavirus data, we are going to predict features in the model using Machine Learning methods in Python. The feature we chose to predict was the Total number of Deaths in each country, as we felt it was the most relevant and important feature for a country to try to predict.

After understanding and cleaning the data, we decided to also normalise the values and remove features that were showing a low correlation or could contribute to over-fitting. This took us from our original list of headers to a restricted list of headers missing the following columns: Total Deaths Per 1m Population, Total Cases Per 1m Population, Total Tests Per 1m Population. Similarly, we didn't lose any information about coronavirus by omitting these columns as they contain information already reflected in a combination of other parameters.

We split the normalised data into training sets and validation sets and applied 7 linear Machine Learning Regression models to the training data. Then evaluated its success against the test data by calculating the normalised mean squared error (MSE), mean absolute error(MAE) and the R-squared score.

Findings

Results using all parameters

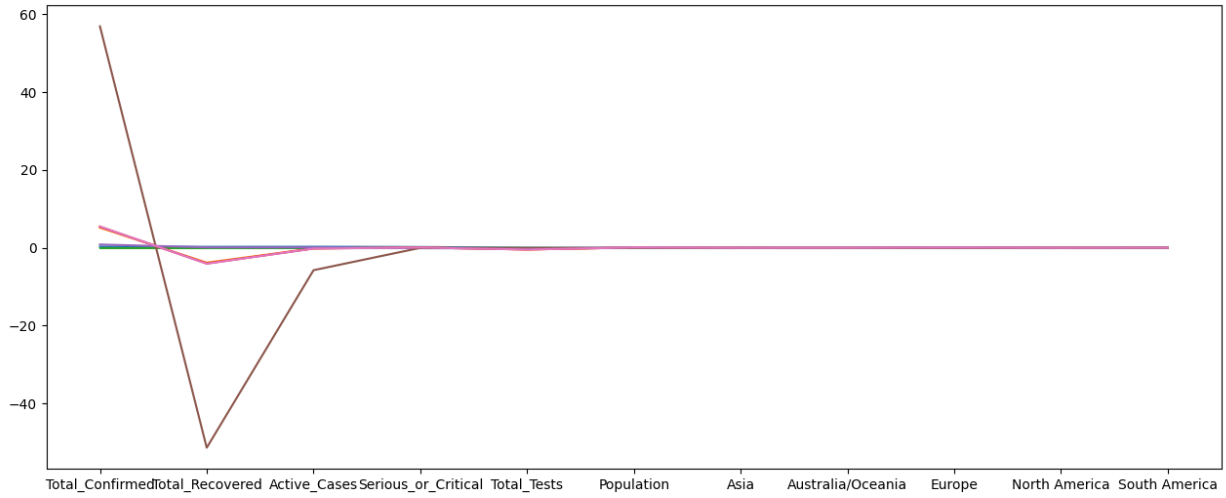
We first ran our models using all parameters except, omitting only the following columns: Total Deaths Per 1m Population, Total Cases Per 1m Population, Total Tests Per 1m Population. This gave rise to the following performance results.

Model Accuracy Results			
Model Name	MSE	MAE	R^2 Score
SGDRegressor	0.147221	0.159697	0.649311
BayesianRidge	0.203303	0.172470	0.515722
LassoLars	0.422054	0.375702	-0.005356
ARDRegression	0.199966	0.167410	0.523671
PassiveAggressive	0.135625	0.154559	0.676933
TheilSenRegressor	0.001518	0.018217	0.996385
LinearRegression	0.202269	0.171525	0.518185

Table 1: Table of scores

As you can see the Theil–Sen estimator performed the best by far, with very low errors and a very high R-squared score. This tells us it is a very good model for this data set and can accurately predict the number of deaths.

Figure1: A Graph showing the parameter weightings from the 7 Models



If we zoom into the parameter weightings given by this Theil-Sen model, (shown in brown), we can see that the magnitude of the weight of the confirmed cases and recovered cases are large compared to the other parameters and models. This is expected as the number of deaths should be the difference between the number of cases and the number of people who have recovered. However the Theil-Sen estimator appears to be

the only model that predicts this relationship. In Figure 1 we can see the other models with much smaller weights for those parameters, and we can see that they perform worse because of this fact. The exact weightings from the Thiel-Sen estimator can be seen in Table 2.

Model Parameter Weighing predicted by the Theil- Sen Model	
Parameter	Weighting
'Total Confirmed'	56.938
'Total Recovered'	-51.366
'Active Cases'	-5.7468
'Serious or Critical'	-4.778e-07
'Total Tests'	-3.918e-06
'Population'	2.229e-06
'Asia'	-0.0005
'Australia/Oceania'	-0.0207
'Europe'	-0.0013
'North America'	-0.0022
'South America'	-0.0225

Table 2: Parameter weightings for Theil-Sen Regressor

Results after removing one of the key parameters

We then repeated the process, but without one of the key features to see how well the model performs when it is missing a vital piece of information such as the Total Confirmed cases.

Model Accuracy Results			
Model Name	MSE	MAE	R^2 Score
SGDRegressor	0.140898	0.156594	0.664372
BayesianRidge	0.204898	0.188443	0.511921
LassoLars	0.422054	0.375702	-0.005356
ARDRegression	0.210587	0.185158	0.498370
PassiveAggressive	0.190714	0.179285	0.545710
TheilSenRegressor	0.219594	0.151321	0.476914
LinearRegression	0.218929	0.193680	0.478499

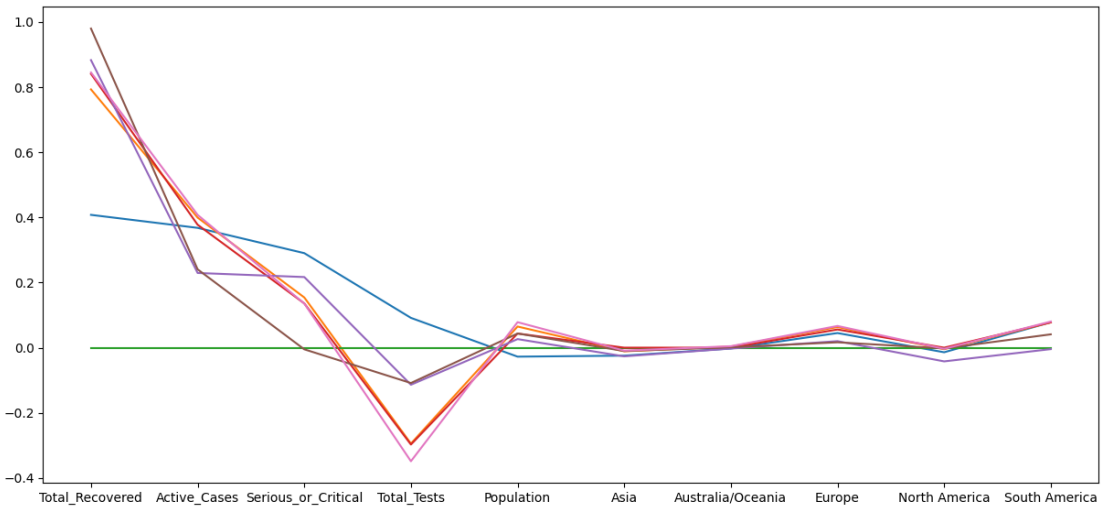
Table 3: Table of scores

As you can see in general the models performed slightly better, but the results remained around constant for all of our models. This is with the exception of the Theil-Sen model

where there is a significant decline in the performance of the Regressor. The Model with the best predicting performance is now the SGDR Regressor, which outperforms the Theil-Sen model in all categories except the mean absolute error.

The performance of the Lasso Lars Regressor remains poor, in both sets of results. However the default α is 1, optimising the penalty would produce a better Lasso estimate. for example by changing α = 10^{-6}, the Lasso results improved to MSE = 0.202308 MAE = 0.171545 and R^2 = 0.518091

Figure2: A Graph showing the parameter weightings from the 7 Models



We also plotted and compared the predicted parameter weightings of each of the models on Figure 2. From this we can see that most models weighted parameters similarly. The highest weightings were the Total Tests, Active Cases and Total Recovered, this suggests that the most important features to predicting the total coronavirus deaths using a Linear Model are these factors. We can also conclude the Population and Continent location had minimal impact on the Total Death number, compared to other parameters.

CONCLUSION

According to the box plot, the highest number of confirmed cases was in Asia. However, the number of total deaths has reduced in Asia compared to the other continents because their recovery rates are high. The challenges faced in this work is the limited coding experience in the team and the complicated data sets. The initial data was too complex for the team to use, so had to utilise another data set. A success of this work was that the team was able to use Python to apply data science techniques and visualisations

REFERENCES

1. Joseph Assaker (2021). *Covid-19 Global Dataset: Up-to-date numbers of daily Confirmed, Death and Active cases for 218 countries* (Version 51). https://www.kaggle.com/josephassaker/covid19-global-dataset?select=worldometer_coronavirus_summary_data.csv
2. Benjamin Skrainka (2015). Seven Python Tools All Data Scientists Should Know How to Use. https://blog.galvanize.com/seven-python-tools-all-data-scientists-should-know-how-to-use/