Analysis of COVID-19 vaccination progress in different countries

Ginu Varghese
Department of Computing Science and Mathematics
Dundalk Institute of Technology
Dundalk, Co.Louth, Ireland
d00251842@student.dkit.ie

***Abstract*— The COVID-19 pandemic has made a huge impact in the functioning of the global economy. The vaccination process against the deadly virus has started in most of the countries and it is the only solution to stop the spread of the coronavirus. When a herd immunity is obtained, it can also save the life of others who are not immune to this virus. In this paper, the progress of vaccination in different countries are analyzed using statistical analysis methods, graphs and tables using the data available from the Kaggle website. Hypothesis testing, ANOVA and Linear regression methods are used for analysis of the available data. In addition to this, the description of the dataset, the technologies used, interpretation of the visualizations are included in this report. The results from the analyses are included in the conclusion section.**

Keywords—COVID-19, vaccination, Hypothesis testing, ANOVA, Linear regression, interpretation.

I. INTRODUCTION

The COVID-19 has become a major concern to the global community since January 2021. COVID-19 is a contagious disease caused by the SARS-Cov-2 virus. It was first discovered in Wuhan in China [2]. There are four major variants for COVID-19 and they are the Alpha, Beta, Gamma and Delta variants [3]. The disease is causing a serious hazard to human lives in 2020. Many people have lost their lives due to the deadly virus and many are still fighting for their lives. From the WHO's report**,** almost 251 million confirmed COVID-19 cases, including 5 million deaths, are there in the world [1].

When the virus started to spread enormously, vaccine research programs started in various countries. As a result, vaccines to fight the Coronavirus has developed by the middle of 2020 and vaccination is the best effective method to stop the spread of the COVID-19 virus. The Pfizer-BioNTech COVID-19 vaccine is the first approved vaccine by the WHO [4]. It was developed by the German biotechnology company BioNTech collaborated with American company Pfizer. The Oxford-AstraZeneca vaccine, also known under the name Covershield, is the most widely used vaccine over the globe. A study in Scotland found that the AstraZeneca vaccine is 81% effective after the second dose for Alpha variant and 61% effective for the Delta variant of COVID-19 [5]. Apart from the mentioned vaccines, there are many other vaccines developed and used by different nations. The African union is using the Janssen vaccine produced by the Johnson-Johnson and India is providing the Covexin to its citizens. The vaccination has helped a lot in controlling the situation in many countries. Among the developed vaccines, different nations prefer different vaccines according to the availability, usage and doses.

Though vaccination can be a cure to the virus, all countries have not achieved 100% vaccination. Some nations have vaccinated almost half of its population while some countries like Africa are far behind. This may be due to the supply issues or the cost issues [6]. So, the analysis of vaccination data could be beneficial to understand the progress of the countries and to predict how long it will take them to achieve the goal.

By collecting the vaccination details of different countries, it is possible to understand the trends in the vaccination schemes. Hence the individuals and the authorities have accurate information to understand the progress and the preferred vaccinations of each country. The open data hub, github.com has the data related to the COVID-19 and the same data is available to download in csv format. This site has the covid vaccination data of countries based on the daily vaccinations, total vaccinated people by each dose and the vaccines used in each country. The same data is also available in Kaggle.com.

Analyzing this data will give the understanding and options to inspect the trends in the vaccination scheme in the countries. The following problems emerged while going through the data:

- Is the number of total vaccinated people in the countries equal to the number of fully vaccinated people?
- Is the vaccination trend having any relation with the continents/countries?
- Which country has attained more progress in vaccination?
- Is there any relationship between the number of vaccinations and the year?
- Which are the top 5 countries having highest and lowest vaccination rates?
- Which are factors affecting the progress of the vaccination?

In the current scenario where covid cases are again spiking in many regions an analysis and prediction on the vaccination count and progress will help the authorities and individuals to understand that herd immunity is the only solution to COVID-19 and it can be only achieved by vaccination [7]. This project attempts to analyze the above-mentioned problems to understand the progress of vaccination schemes in different countries, the vaccines used by the countries and the trend of vaccination progress in the countries in the 2020-2021.

The table below mentions the technology used for the project.

Technology
Programming language: Python (Jupyter Notebook, Spyder IDE)
Python libraries: • Pandas • Numpy • pycountry • Plotly • Matplotlib • Bokeh • Seaborn • scipy.stats • statsmodels • Flask

TABLE I. Technology used for the analysis

This paper is organized as follows: section II describes the related works. Data analytics life cycle is described in section III. The data description is given in section IV. Section V handles the visualizations and descriptive statistics is given in section VI. Hypothesis test is described in section VII, ANOVA test in section VIII and linear regression models in section IX. Analysis using bokeh is described in section X. The project diary is detailed in the section XI. Section XII gives the conclusion. Ethical Considerations is pointed out in section XIII and Acknowledgements in section XIV. Finally, References are mentioned in section XV and Appendix in section XVI.

II. RELATED WORKS

Tamilselvan S [8] wrote a paper on analyzing and predicting the COVID-19 vaccination using Machine learning. In this paper his team analyzed the vaccination status and suggested various methods to vaccine distribution using the Machine learning algorithms. They used prophet algorithms for the analysis of the dataset and predicted that around 27% of the people worldwide will be vaccinated by the month of August 2021.The complete population could be vaccinated by the end of 2022 considering the awareness among the people and their literacy.

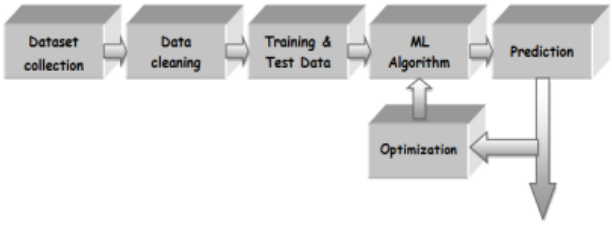


Figure.1. Proposed Machine Learning Workflow by [8]

Another paper by Narayana Darapaneni [9] on forecasting the vaccination drive in India by SIR prophet algorithm explained the vaccination drive in India and predicted the time to vaccinate the people to achieve herd immunity. They created SIR model simulation for the analysis and time series analysis was used to identify the time required to achieve herd immunity in India through vaccination.

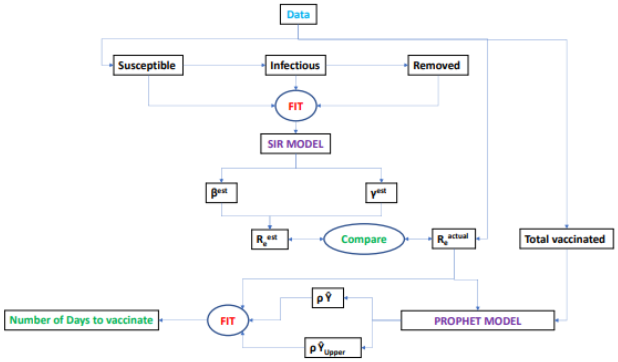


Figure.2. Proposed SIR workflow by [9]

Nasiba M. Abdulkareem [10] and team conducted a study using a machine algorithm to analyze the world COVID-19 vaccination progress. WEKA software was used to run the tests on real-world data. Random Tree, decision tree, K-nearest neighbors and Naïve bayes were used for analysis.

- *Random Tree*

  Random tree comes under the supervised machine learning algorithm. It consists of a group of decision trees to form a Random Forest. These decision trees are merged together to produce the accurate prediction [11].

- *Decision Tree*

  Decision tree algorithm comes under the supervised machine learning algorithm. Decision tree is used for solving both classification and regression problems. It is used to build a training model to predict the values of target variables from the training data [12].

- *K-Nearest Neighbors (KNN)*

  KNN is the simple algorithm for model representation. To predict a new data point using KNN, the entire training set is searched for K most neighbors and then summarizing those K most neighbors. This variable can be considered as the mean output for regression problems and mode for class value [13].

- *Naïve Bayes*

  The Naïve Bayes algorithm is used for predictive modelling. It consists of two types of probabilities i) Each class probability ii) Conditional probability of each class. These probabilities can be obtained from the training data and after calculating them the prediction can be made using the Bayes Theorem [13].

From the analysis they concluded that Decision tree is the best method based on accuracy compared to the others.

Mansi Lende [14] published a paper on COVID-19 vaccinations using fuzzy logic. In this paper, priority regions for vaccination are predicted using the fuzzy rule. Mamdani technique was used for ranking each region to decide the priority. The fuzzy logic is applied using the python fuzzy-logic-toolbox. To calculate the priority of regions each country from the dataset is considered as input variable. The score is the output variable and it has values from 9 – 49. The regions are prioritized based on this score. Python programming is used to evaluate the outputs. This model can be implemented to understand the prioritization for covid vaccine areas when there is a shortage of vaccines.

III. DATA ANALYTICS LIFE CYCLE

The different stages of this project were: the data collection, data preparation, data modelling, evaluation and API access. This is shown in Figure 3. The framework used for the project was the CRISP-DM methodology for the data collection, preparation and modelling stages. The evaluation outcomes are included in this report.

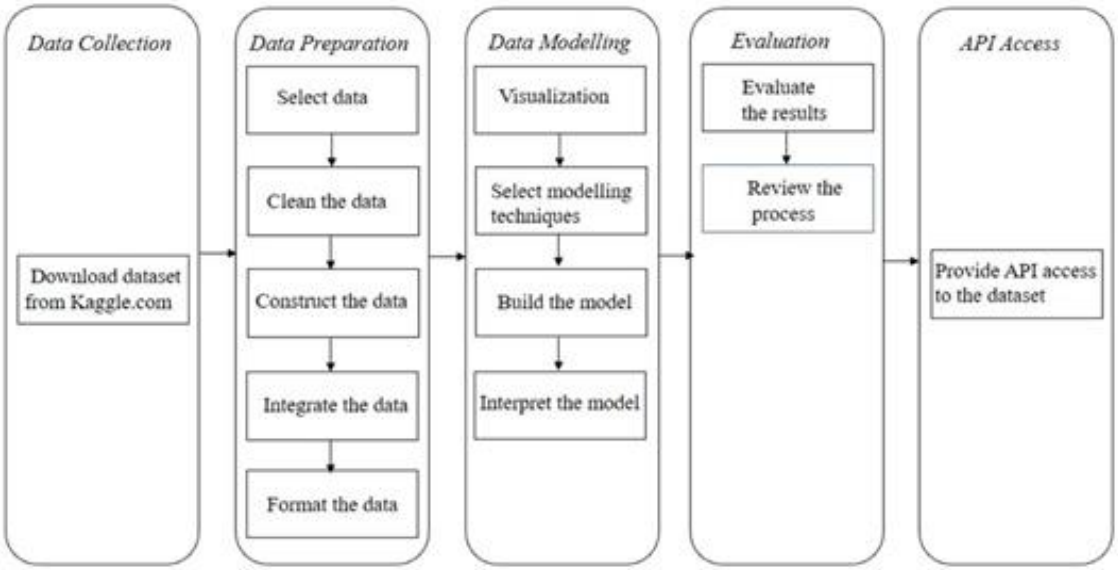


Figure 3. Methodology Implemented using CRISP-DM

A. *Data Collection*

The data collection stage included the gathering of the dataset from the Kaggle. This dataset was used in the data preparation stage. The dataset was downloaded in the CSV format.

B. *Data Preparation*

The CRISP-DM methodology was used in the data preparation stage. This stage included the selection and cleaning of the dataset, the integration of the data and formatting. Python programming libraries were used in this stage for the cleaning and integration of the dataset.

C. *Data Modelling*

In the data modelling stage, I worked with the dataset description, its visualizations and statistical analysis over a sample of the dataset with four countries including India, Ireland, Bahrain and the United Kingdom (UK). In this stage I also created more visualizations to find answers to the research questions.

D. *Evaluation*

The evaluation stage was to evaluate the results of the statistical analysis conducted. Then I came up with the results of the analysis.

E. *API Access*

In this stage, I provided API access to the cleaned dataset I used for the analysis. Flask framework was used for creating RESTful API.

IV. UNDERSTANDING THE DATA

A. *Data Description*

The dataset contains the details of the COVID-19 vaccination of 61 countries from 2020-2021.The data was collected from the Kaggle. There were 13 variables in the original dataset and 5750 observations. The data has the vaccination count and also the rate of vaccination per population of the country. As a part of cleaning and for analysis purposes I added 3 additional variables to the dataset. Finally, it had 16 variables and 5750 observations. The data is based on daily vaccinations on different dates in the time period 2020 – 2021. The details of fully vaccinated people and total vaccinations and the type vaccines used in each country are available in the dataset.

B. *Types of variables*

The TABLE II describes the type variables in the dataset. There were 16 variables in the dataset and I will be using 'daily_vaccinations_per_million', 'people_fully_vaccinated_per_hundred', 'people_vaccinated_per_hundred', 'total_vaccinations_per_hundred', 'country', 'year', 'month', 'continent' for the statistical inference.

Variable 'iso_code' is not used for analysis since it is similar to the country name as a short form for the countries. The 'date' is not used because I have created month and year variables for analysis from the date variable. Similarly, I have discarded the 'vaccines' and 'source_name'. Finally, I excluded the count variables 'daily_vaccinations', total_vaccinations', 'people_vaccinated' and 'people_fully_vaccinated' since I have the rate of vaccinations per population in the dataset, which is more appropriate for analysis. The other reason for excluding them was that if I use both of the count and rate variables, there will be higher chance for collinearity between them. The variables 'month', 'year' and 'continent' were additionally created for the analysis purpose.

Types of variables				
Variable Name	**Category**	**Type**	**Description**	**Note**
country	Nominal Categorical	String	The country in which vaccination is given	
iso_code	Nominal Categorical	String	ISO_CODE of the country	
date	Date	Datetime	The date in which the vaccinations are given.(All the dates are not included in the dataset)	
total_vaccinations	Discrete Numerical	Integer	The total number of vaccinations given in the country	

people_vaccinated	Discrete Numerical	Integer	Number of people vaccinated. It can be either single or both doses.	
people_fully_vaccinated	Discrete Numerical	Integer	Number of people who have received both doses of vaccines.	
daily_vaccinations	Discrete Numerical	Integer	It is the number of vaccinations given on that date in a country.	
total_vaccinations_per_hundred	Continuous Numerical	Float	It is the ratio between the total number of vaccinations and the total population in that country.	
people_vaccinated_per_hundred	Continuous Numerical	Float	It is the ratio between the population immunized and the total population in that country.	
people_fully_vaccinated_per_hundred	Continuous Numerical	Float	It is the ratio between the population fully immunized and the total population in that country.	
daily_vaccinations_per_million	Continuous Numerical	Integer	It is the ratio between the vaccination number and the total population in that country.	
source_name	Nominal Categorical	String	The source name where this information is available.	
year	Discrete Numerical	Integer	The year of the vaccination given	Variable was additionally created for analysis
month	Ordinal Categorical	String	Month in which the vaccination is given	Variable was additionally created for analysis
continent	Nominal Categorical	String	The continent in which the country is in.	Variable was additionally created for analysis
vaccines	Nominal Categorical	String	Name of the vaccines used in that country	

TABLE II. Type of variables in the dataset

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 5750 entries, 0 to 5219
Data columns (total 16 columns):
 #   Column                                Non-Null Count  Dtype
---  ------                                --------------  -----
 0   country                               5750 non-null   object
 1   iso_code                              5750 non-null   object
 2   date                                  5750 non-null   datetime64[ns]
 3   total_vaccinations                    5750 non-null   int64
 4   people_vaccinated                     5750 non-null   int64
 5   people_fully_vaccinated               5750 non-null   int64
 6   daily_vaccinations                    5750 non-null   int64
 7   total_vaccinations_per_hundred        5750 non-null   float64
 8   people_vaccinated_per_hundred         5750 non-null   float64
 9   people_fully_vaccinated_per_hundred   5750 non-null   float64
 10  daily_vaccinations_per_million        5750 non-null   int64
 11  vaccines                              5750 non-null   object
 12  source_name                           5750 non-null   object
 13  year                                  5750 non-null   int64
 14  month                                 5750 non-null   object
 15  continent                             5750 non-null   object
dtypes: datetime64[ns](1), float64(3), int64(6), object(6)
memory usage: 763.7+ KB
```

Figure 4. Information of the variables using the *info()* in Python

C. *Outliers and Treatments*

NOTE: From this point I will be using the sample dataset of four countries with India, Ireland, Bahrain and UK for my analysis.

I chose India because it is my home country and Ireland since I am living in Ireland now. My analysis includes both 2020 and 2021 vaccination details. But most of the countries have started their vaccination only in 2021, and not much data is available for 2020. So, I chose Bahrain as one of my 4 countries because it has started its vaccination in 2020.At this point I had 2 Asian countries and one European country. So, I decided to choose one more country in the European continent. Since the United Kingdom is in Europe and is neighboring to Ireland and also has a higher population, I chose the United Kingdom as my fourth country. The new sample dataset has 646 observations and 16 columns.

There are outliers present in four variables. They are, 'total_vaccinations', 'daily_vaccinations', 'people_vaccinated' and 'people_fully_vaccinated'. This is because the vaccination count is very much higher in some countries compared to the others. So, I decided not to remove these outliers as there is a chance it may affect other observations.

There were no missing values present in the dataset. I assured it by applying isnull().sum() functions in python on my dataframe.

```
country                               0
iso_code                              0
date                                  0
total_vaccinations                    0
people_vaccinated                     0
people_fully_vaccinated               0
daily_vaccinations                    0
total_vaccinations_per_hundred        0
people_vaccinated_per_hundred         0
people_fully_vaccinated_per_hundred   0
daily_vaccinations_per_million        0
vaccines                              0
source_name                           0
year                                  0
month                                 0
continent                             0
dtype: int64
```

Figure 5. Count of missing values in the dataframe

V. VISUALIZATIONS

A. Univariate Visualizations

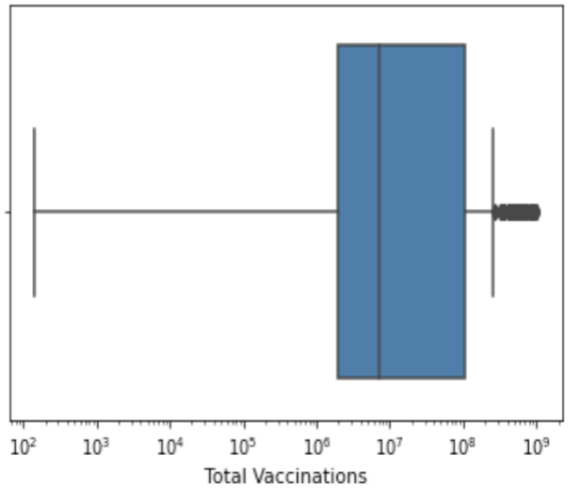
Data Visualization
total_vaccinations

Interpretation
• The data has outliers and it is skewed to the right. • The whiskers are of different length. • The mean and median of the data is different. • So, the data is not symmetrical in shape.

TABLE III. Univariate plot for total_vaccinations

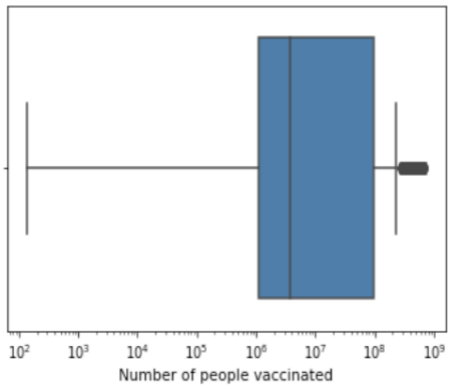
Data Visualization
people_vaccinated

Interpretation
• The data has outliers and it is skewed to the right. • The whiskers are of different length. • The mean and median of the data is different. • So, the variable 'people_vaccinated' is not symmetrical in shape.

TABLE IV. Univariate plot for people_vaccinated

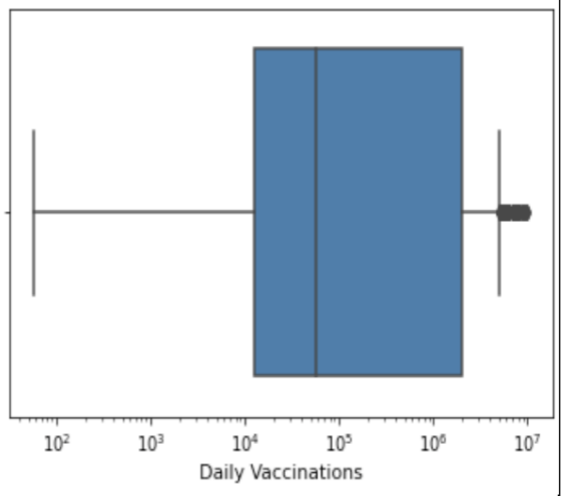
Data Visualization
daily_vaccinations

Interpretation
• The data has outliers and it is skewed to the right. • The whiskers are of different length. • The mean and median of the data is different. • So, the data is not symmetrical in shape.

TABLE V. Univariate plot for daily_vaccinations

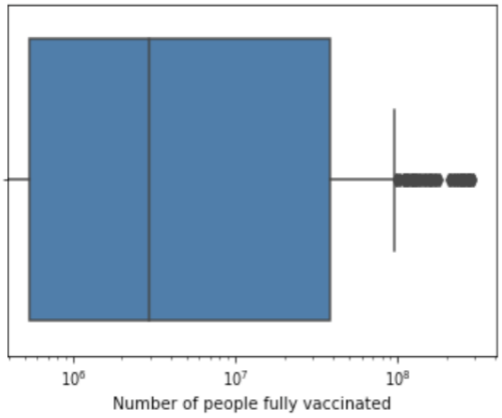
Data Visualization
people_fully_vaccinated

Interpretation
• The data has outliers and it is skewed to the right. • The whiskers are of different length. • The mean and median of the data is different. • So, the data is not symmetrical

TABLE VI. Univariate plot for people_fully_vaccinated

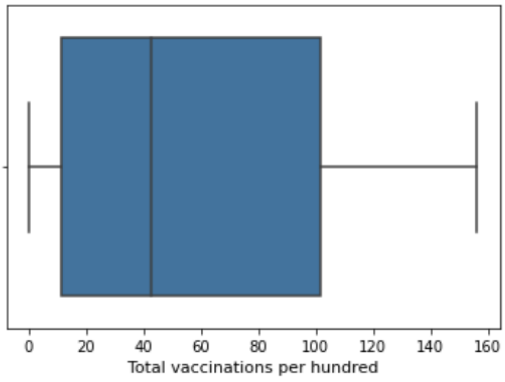
Data Visualization
total_vaccinations_per_hundred

Interpretation
• There are no outliers in the data. • The whiskers are of different length. • The mean and median of the data is different. • So, the data is not symmetrical in shape.

TABLE VII. Univariate plot for total_vaccinations_per_hundred

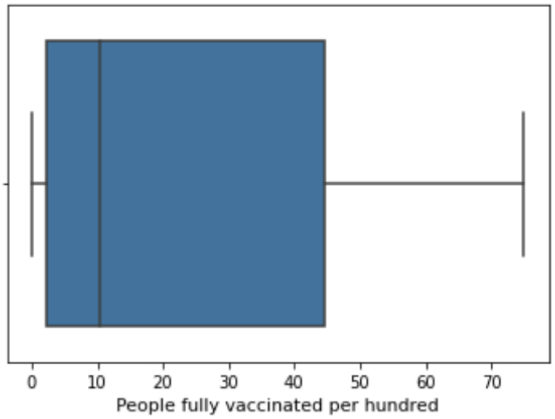
Data Visualization
people_fully_vaccinated_per_hundred

Interpretation
• There are no outliers in the data. • The whiskers are of different length. • The mean and median of the data is different. • So, the data is not symmetrical in shape.

TABLE VIII. Univariate plot for people_fully_vaccinated_per_hundred

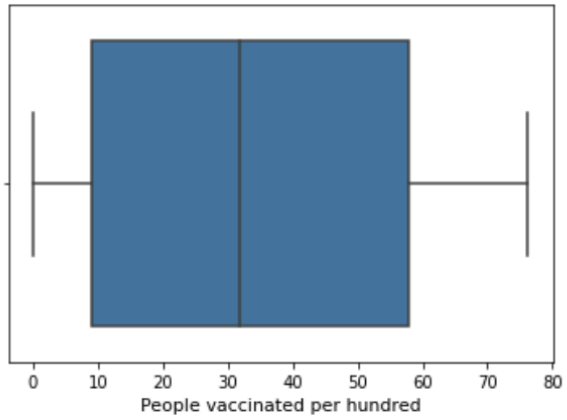
Data Visualization
people_vaccinated_per_hundred

Interpretation
• The data has no outliers. • The whiskers are of different length. • The mean and median of the data are roughly equal. • So, the data is roughly symmetrical

TABLE IX. Univariate plot for people_vaccinated_per_hundred

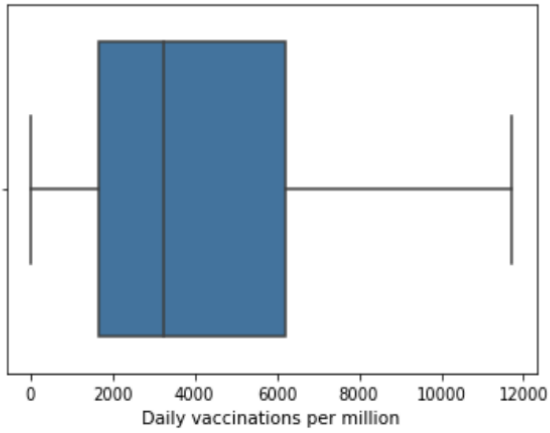
Data Visualization
daily_vaccinations_per_million

Interpretation
• There are no outliers in the data. • The whiskers are of different length. • The mean and median of the data is different. • So, the data is not symmetrical in shape.

TABLE X. Univariate plot for daily_vaccinations_per_million

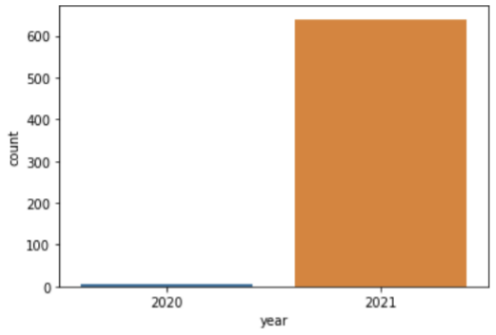
Data Visualization
year

Interpretation
• Majority of the observations in the sample dataset falls in the 2021 year. • Less than 10 % falls in the 2020 year. • More vaccinations are given in 2021 than 2020. • The data does not look balanced.

TABLE XI. Univariate plot for year

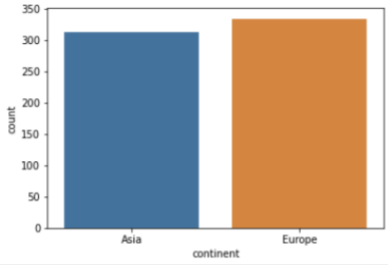
Data Visualization
continent

Interpretation
• More observations are in Europe than in Asia. • More vaccinations are given in Europe than Asia. • The data looks roughly balanced.

TABLE XII. Univariate plot for people_fully_vaccinated

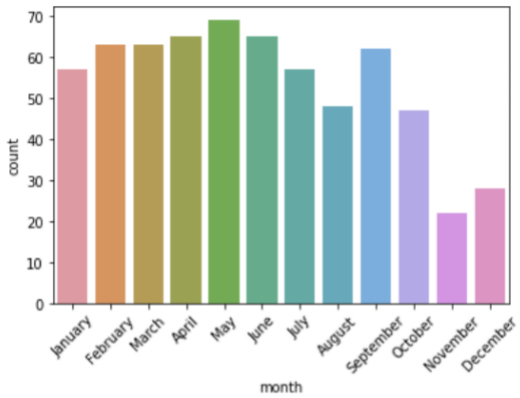
Data Visualization
month

Interpretation
• More observations are in the month of May. • That is, more vaccinations are given in May and fewer vaccinations are given in November. • The data does not look balanced.

TABLE XIII. Univariate plot for month

B. Bivariate Visualizations

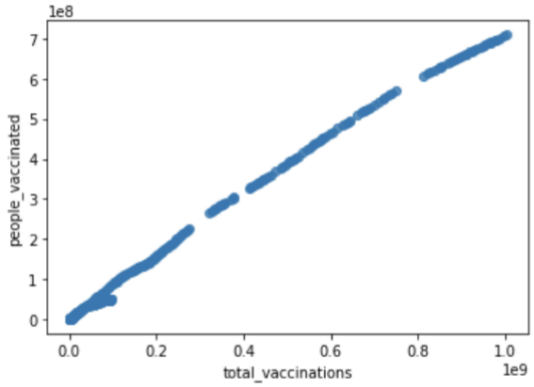
Data Visualization
total_vaccinations and people_vaccinated

Interpretation
• There is a strong positive linear relationship between the total_vaccinations and people_vaccinated.

TABLE XIV. Bivariate plot for people_vaccinated and total_vaccinations

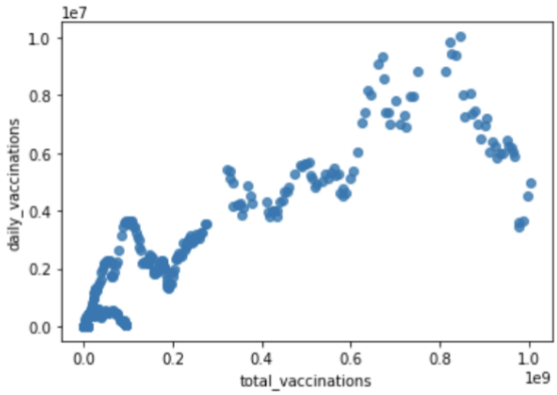
Data Visualization
total_vaccinations and daily_vaccinations

Interpretation
• There is a positive linear relationship between the total_vaccinations and daily_vaccinations.

TABLE XV. Bivariate plot for people_fully_vaccinated and daily_vaccinations

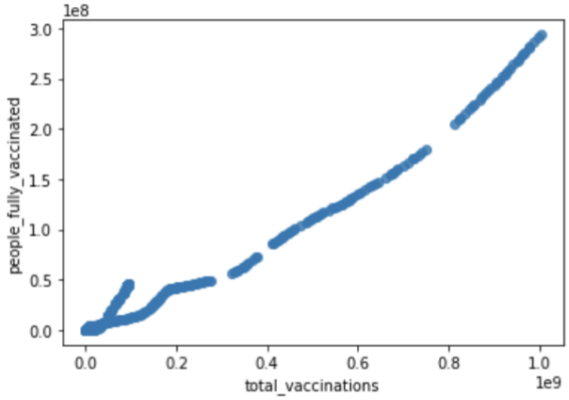
Data Visualization
total_vaccinations and people_fully_vaccinated

Interpretation
• There is a strong positive linear relationship between the total_vaccinations and people_fully_vaccinated.

TABLE XVI. Bivariate plot for people_fully_vaccinated and total_vaccinations

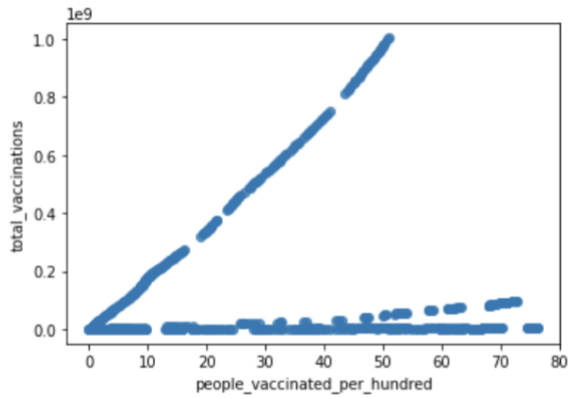
Data Visualization
total_vaccinations and people_vaccinated_per_hundred

Interpretation
• There is a positive linear relationship between the total_vaccinations and people_vaccinated_per_hundred for some data. • So the data does not look balanced.

TABLE XVII. Bivariate plot for people_vaccinated_per_hundred and total_vaccinations

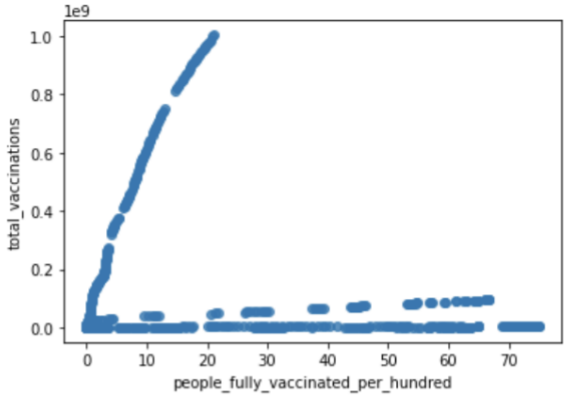
Data Visualization
total_vaccinations and people_fully_vaccinated_per_hundred

Interpretation
• There is a positive linear relationship between the total_vaccinations and people_fully _vaccinated_per_hundred. • But for some data there is not a relationship. • So the data does not look balanced.

TABLE XVIII. Bivariate plot for people_fully_vaccinated_per_hundred and total_vaccinations

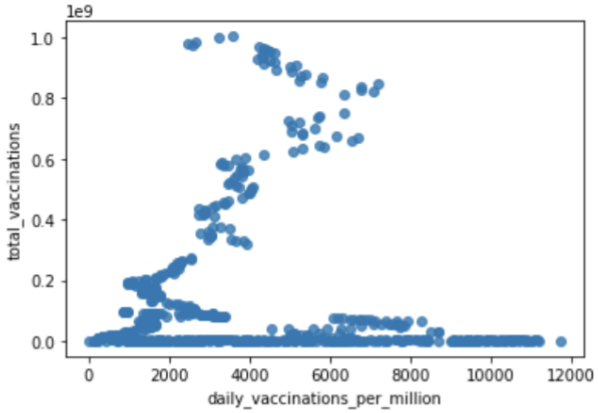
Data Visualization
total_vaccinations and daily_vaccinations_per_million

Interpretation
• There is a positive linear relationship between the data for some points. • But for some data there is not a relationship. • So, the data does not look balanced.

TABLE XIX. Bivariate plot for total_vaccinations and daily_vaccinations_per_million

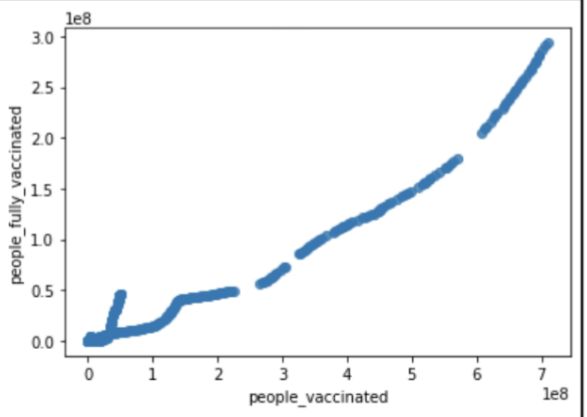
Data Visualization
people_vaccinated and people_fully_vaccinated

Interpretation
• There is a positive linear relationship between the people_vaccinated and people_fully_vaccinated

TABLE XX. Multivariate plot for people_vaccinated and people_fully_vaccinated

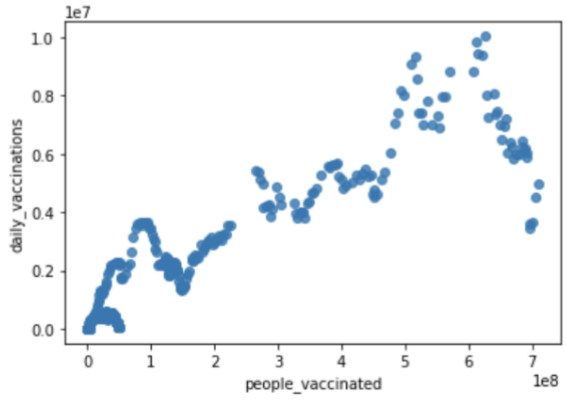
Data Visualization
people_vaccinated and daily_vaccinations

Interpretation
• There is a positive linear relationship between people_vaccinated and daily_vaccinations.

TABLE XXI. Bivariate plot for people_vaccinated and daily_vaccinations

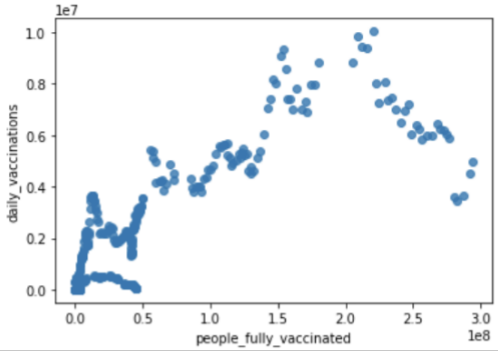
Data Visualization
people_fully_vaccinated and daily_vaccinations

Interpretation
• There is a positive linear relationship between people_fully_vaccinated and daily_vaccinations.

TABLE XXII. Bivariate plot for people_fully_vaccinated and daily_vaccinations

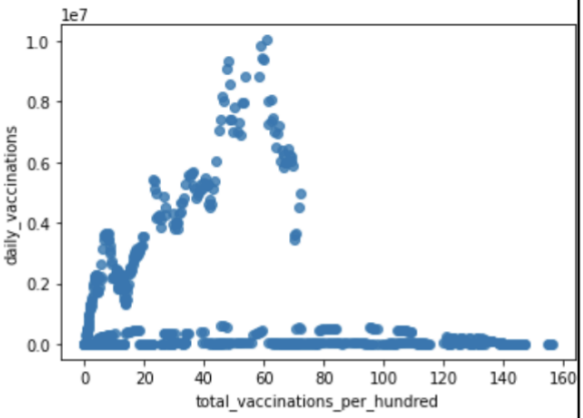
Data Visualization
total_vaccinations_per_hundred and daily_vaccinations

Interpretation
• There is a positive linear relationship between total_vaccinations_per_hundred and daily_vaccination for some datas. • But there is no relationship between the data points in some areas. • So, the data does not look balanced.

TABLE XXIII. Bivariate plot for total_vaccinations_per_hundred and daily_vaccination

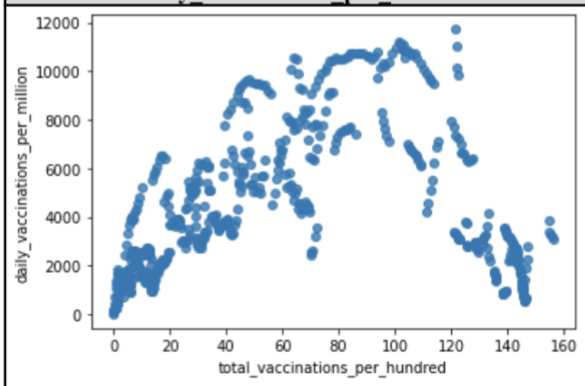
Data Visualization
total_vaccinations_per_hundred and daily_vaccinations_per_million

Interpretation
• There is a positive linear relationship between total_vaccinations_per_hundred and daily_vaccinations_per_million

TABLE XXIV. Bivariate plot for people_vaccinated and daily_vaccinations_per_million.

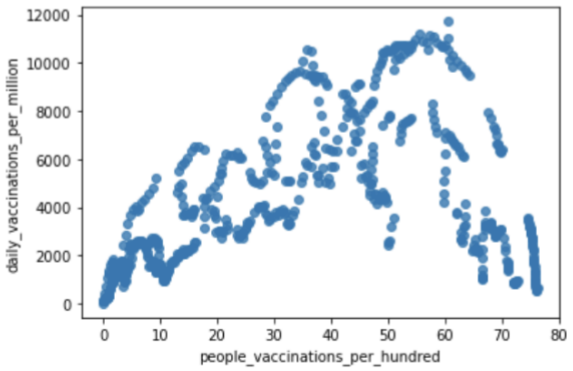
Data Visualization
people_vaccinated_per_hundred and daily_vaccinations_per_million

Interpretation
• There is a positive relationship between people_vaccinated_per_hundred and daily_vaccinations_per_million

TABLE XXV. Bivariate plot for people_vaccinated and daily_vaccinations

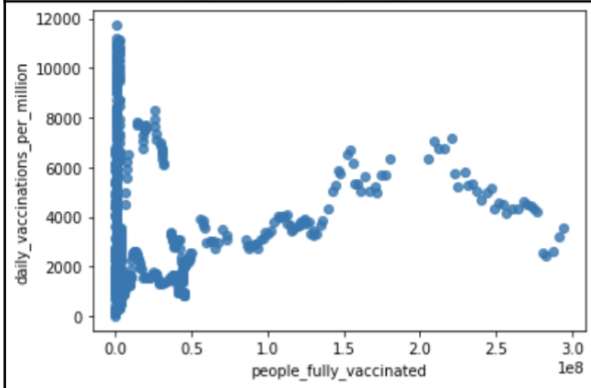
Data Visualization
people_fully_vaccinated and daily_vaccinations_per_million

Interpretation
• There is a positive linear relationship between people_vaccinated and daily_vaccinations. • But for some observations the daily vaccination has increased but people fully vaccinated has not increased. • So the data is unbalanced.

TABLE XXVI. Bivariate plot for people_vaccinated and daily_vaccinations

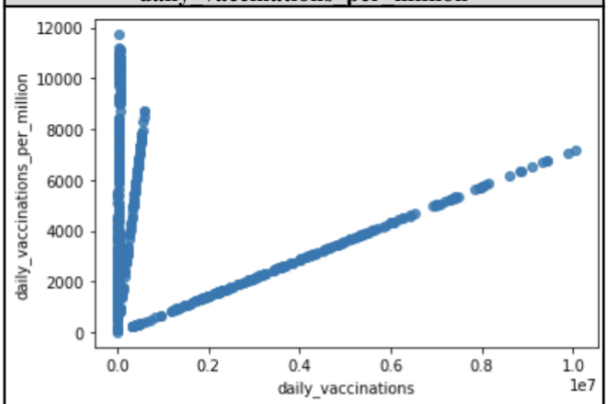
Data Visualization
daily_vaccinations and daily_vaccinations_per_million

Interpretation
• There is a positive linear relationship between the variables for some datas. • But there is no relationship between the data points in some areas. • So, the data does not look balanced.

TABLE XXVII. Bivariate plot for daily_vaccinations and daily_vaccinations_per_million

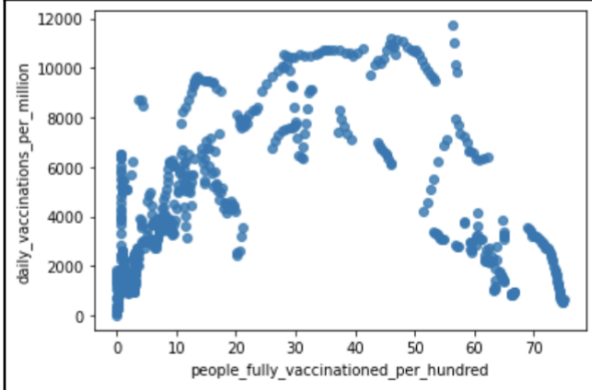
Data Visualization
people_fully_vaccinated_per_hundred and daily_vaccinations_per_million

Interpretation
• There is a positive linear relationship between people_fully_vaccinated_per_hundred and daily_vaccinations_per_million for most of the data.

TABLE XXVIII. Bivariate plot for people_vaccinated and daily_vaccinations

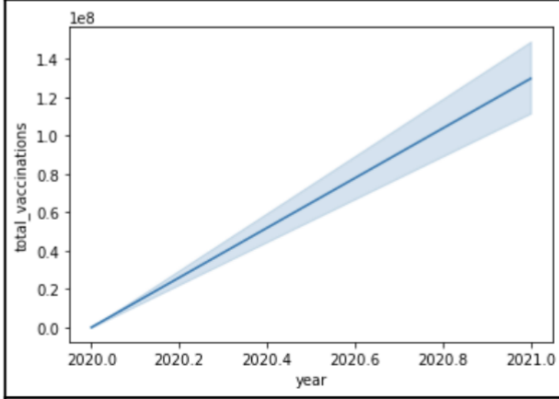
Data Visualization
Year and total_vaccinations

Interpretation
• There is a positive linear relationship between the variables..

TABLE XXIX. Bivariate plot for Year and total_vaccination

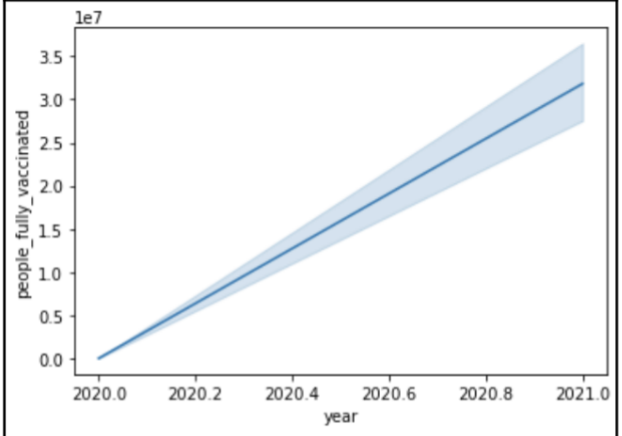
Data Visualization
Year and people_fully_vaccinated

Interpretation
• There is a positive linear relationship between the two data.

TABLE XXX. Bivariate plot for Year and people_fully_vaccinated

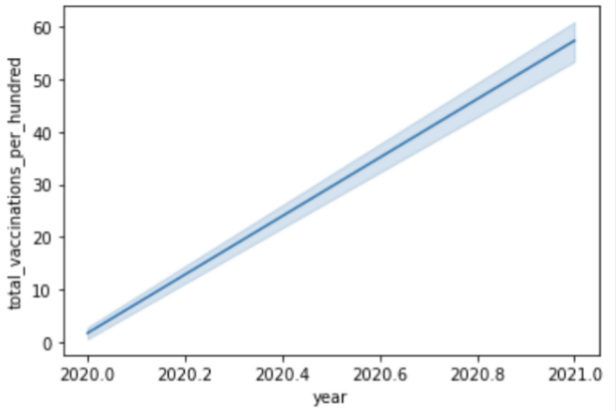
Data Visualization
year and total_vaccinations_per_hundred

Interpretation
• There is a positive linear relationship between year and total_vaccinations_per_hundred.

TABLE XXXI. Bivariate plot for year and total_vaccinations_per_hundred

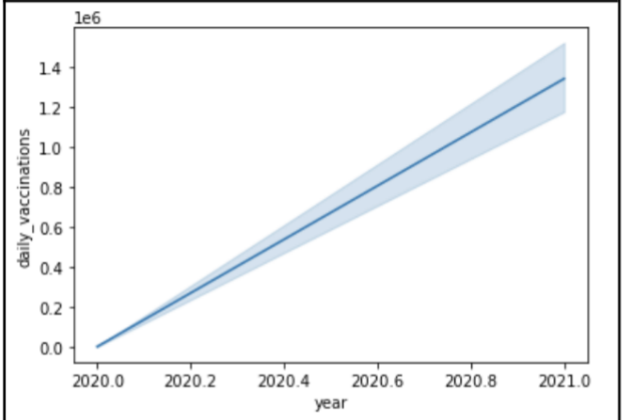
Data Visualization
Year and daily_vaccinations

Interpretation
• There is a positive linear relationship between the data.

TABLE XXXII. Bivariate plot for year and daily_vaccinations

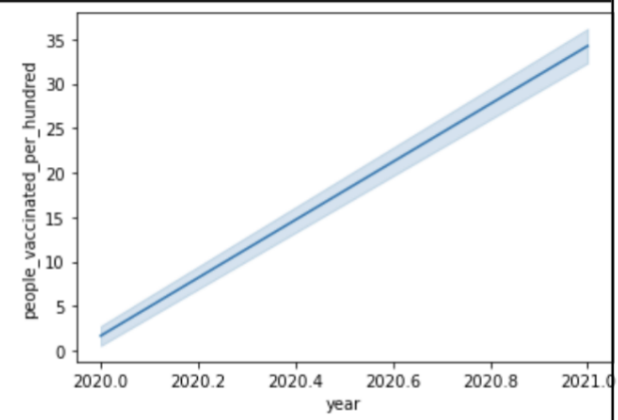
Data Visualization
Year and people_vaccinated_per_hundred

Interpretation
• There is a positive linear relationship between Year and people_vaccinated_per_hundred.

TABLE XXXIII. Bivariate plot for Year and people_vaccinated_per_hundred.

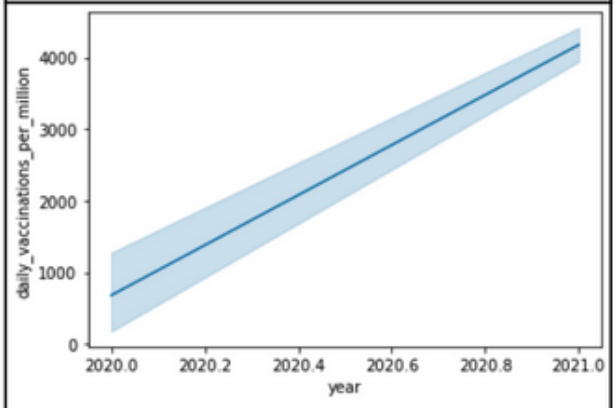
Data Visualization
year and daily_vaccinations_per_million

Interpretation
• There is a positive linear relationship between year and daily_vaccinations_per_million.

TABLE XXXIV. Bivariate plot for year and daily_vaccinations_per_million

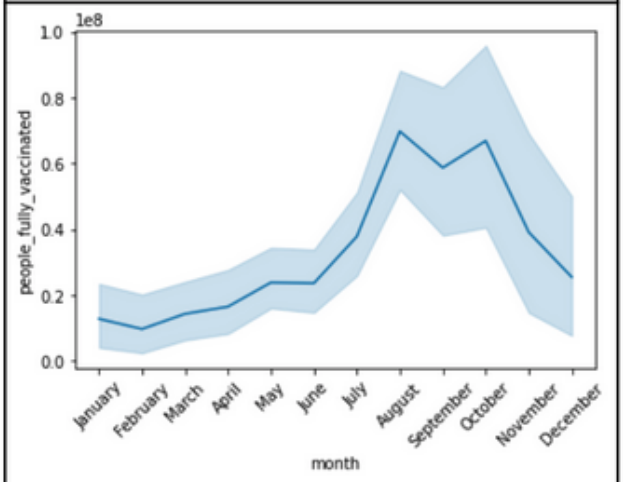
Data Visualization
month and people_fully_vaccinated

Interpretation
• There is an increasing and decreasing trend in the data between the two variables. • The vaccination has increased from January to August. Then decreased in september. • Then again it is showing an increasing trend followed by a decrease.

TABLE XXXV. Bivariate plot for month and people_fully_vaccinated

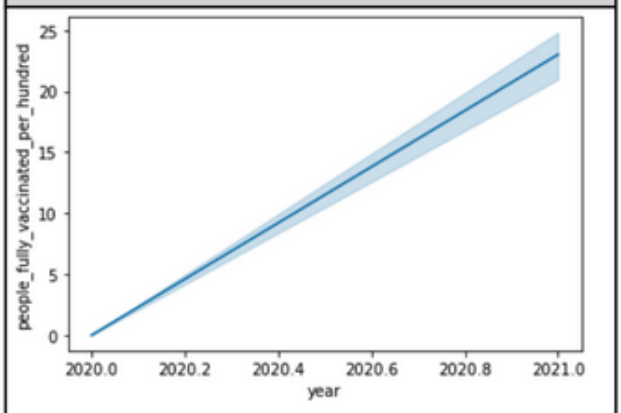
Data Visualization
Year and people_fully_vaccinated_per_hundred

Interpretation
• There is a positive linear relationship between Year and people_fully_vaccinated_per_hundred.

TABLE XXXVI. Bivariate plot for Year and people_fully_vaccinated_per_hundred

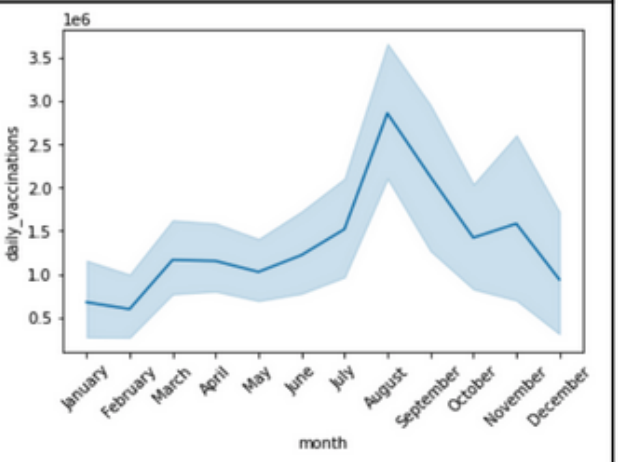
Data Visualization
month and daily_vaccinations

Interpretation
• There is an increasing and decreasing trend in the data between the two variables. • The vaccination has increased from January to August. Then decreased in september. • Then again it is showing an increasing trend followed by a decrease..

TABLE XXXVII. Bivariate plot for month and daily_vaccinations

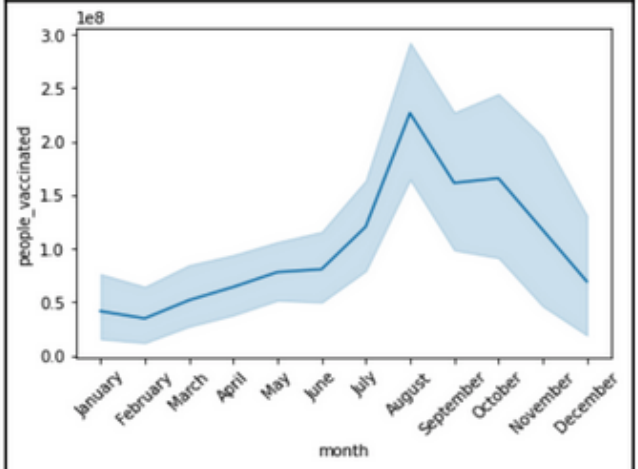
Data Visualization
month and people_vaccinated

Interpretation
• There is an increasing and decreasing trend in the data between the two variables. • The vaccination has increased from January to August. Then decreased in september. • Then again it is showing an increasing trend followed by a decrease.

TABLE XXXVIII. Bivariate plot for month and people_vaccinated

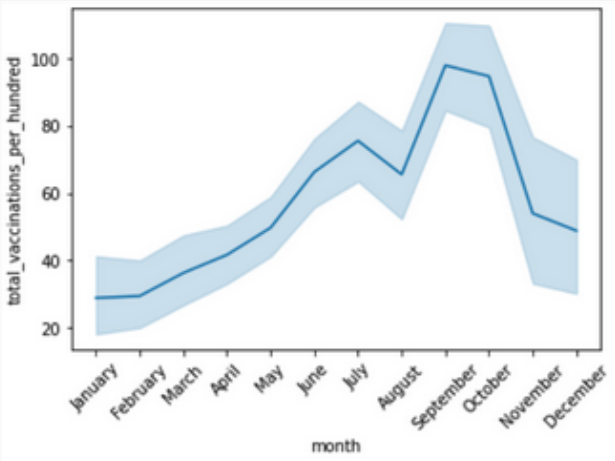
Data Visualization
month and total_vaccinations_per_hundred

Interpretation
• There is an increasing and decreasing trend in the data between the two variables. • The vaccination has increased from January to July. Then decreased in August. • Then again it is showing a increasing trend followed by a decrease

TABLE XXXIX. Bivariate plot for month and total_vaccinations_per_hundred

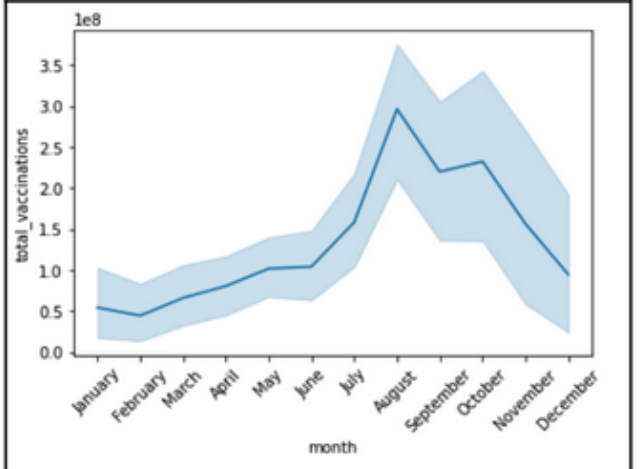
Data Visualization
month and total_vaccinations

Interpretation
• There is an increasing and decreasing trend in the data between the two variables. • The vaccination has increased from January to August. Then decreased in september. • Then again it is showing an increasing trend followed by a decrease..

TABLE XL. Bivariate plot for month and total_vaccination

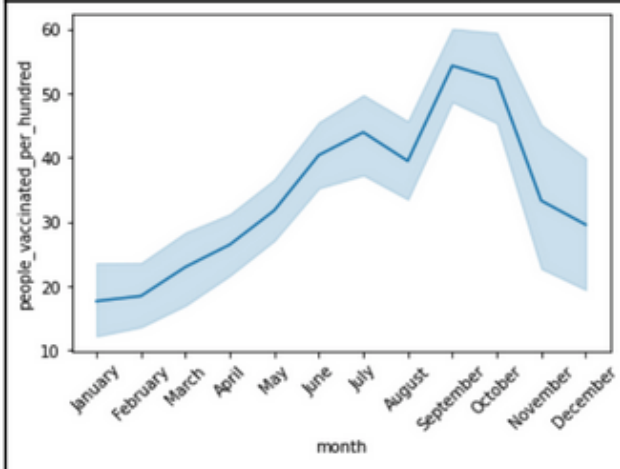
Data Visualization
month and people_vaccinated_per_hundred

Interpretation
• There is an increasing and decreasing trend in the data between the two variables. • The vaccination has increased from January to July. Then decreased in August. • Then again it is showing a increasing trend followed by a decrease

TABLE XLI. Bivariate plot for month and people_vaccinated_per_hundred

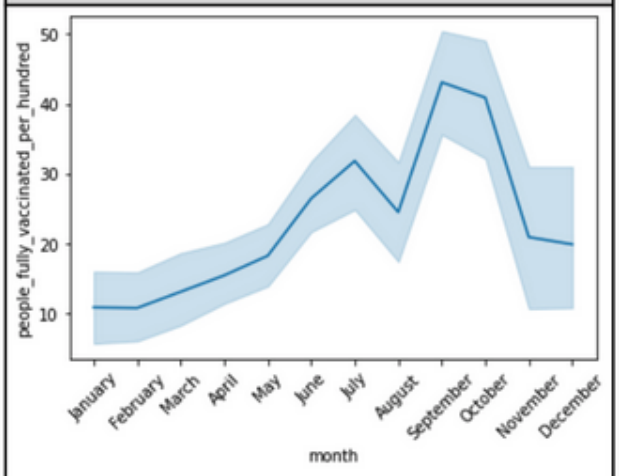
Data Visualization
month and people_fully_vaccinated_per_hundred

Interpretation
• There is an increasing and decreasing trend in the data between the two variables. • The vaccination has increased from January to July. Then decreased in August. • Then again it is showing an increasing and decreasing trend from August.

TABLE XLII. Bivariate plot for month and people_fully_vaccinated_per_hundred

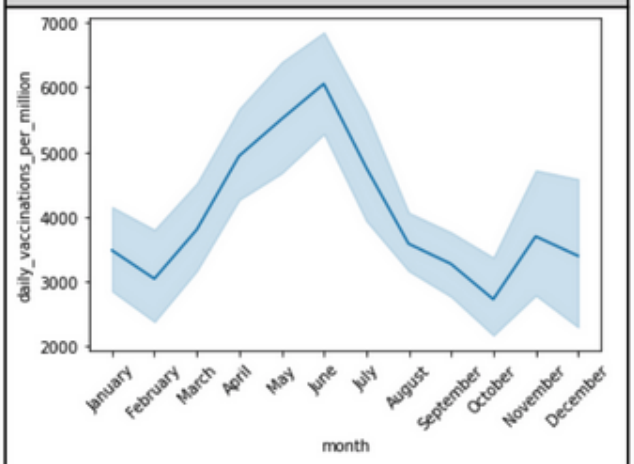
Data Visualization
month and daily_vaccinations_per_million

Interpretation
• There is an increasing and decreasing trend in the data between the two variables. • The vaccination has increased from February to June. Then decreased till October. • Then again it is showing an increasing trend from October.

TABLE XLIII. Bivariate plot for month and daily_vaccinations_per_million

Data Visualization

people_vaccinated and daily_vaccinations

vaccines	Covaxin, Oxford/AstraZeneca, Sputnik V	Johnson&Johnson, Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing, Sputnik V	All
country					
Bahrain	0	0	0	93	93
India	219	0	0	0	219
Ireland	0	230	0	0	230
United Kingdom	0	0	104	0	104
All	219	230	104	93	646

Interpretation

- The data is unbalanced.

TABLE XLIV. Bivariate plot for people_vaccinated and daily_vaccinations

Data Visualization

people_vaccinated and daily_vaccinations

vaccines	Covaxin, Oxford/AstraZeneca, Sputnik V	Johnson&Johnson, Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing, Sputnik V	All
continent					
Asia	219	0	0	93	312
Europe	0	230	104	0	334
All	219	230	104	93	646

Interpretation

- The data is unbalanced.

TABLE XLV. Bivariate plot for people_vaccinated and daily_vaccinations

C. Multivariate Visualizations

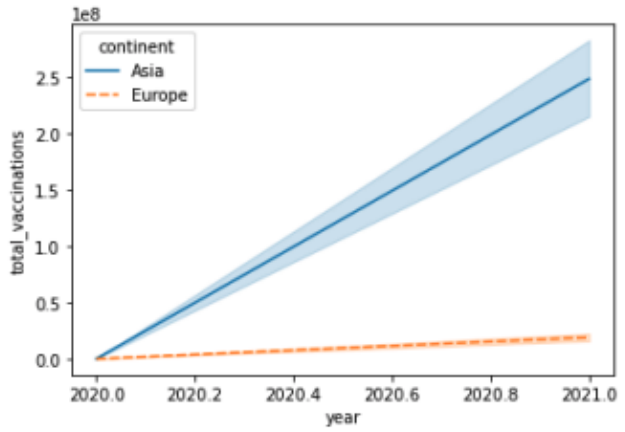
Data Visualization
Year, total_vaccinations and continent

Interpretation
• There is a rapid increase in the total vaccination from 2020 to 2021 in Asia and a gradual increase can be seen in Europe..

TABLE XLIV. Multivariate plot for Year, total_vaccinations and continent

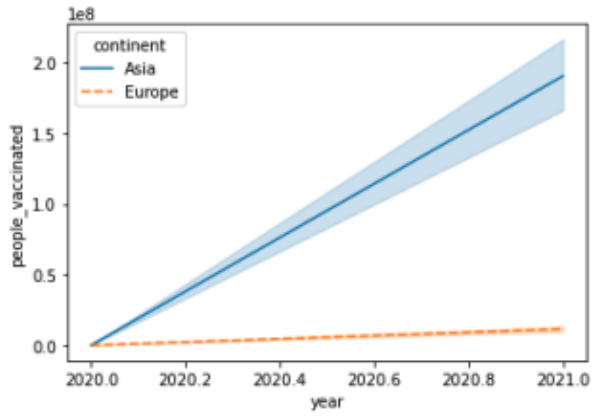
Data Visualization
Year, people_vaccinated and continent

Interpretation
• There is a rapid increase in the number of vaccinated people from 2020 to 2021 in Asia and a gradual increase can be seen in Europe.

TABLE XLVII. Multivariate plot for Year, people_vaccinated and continent

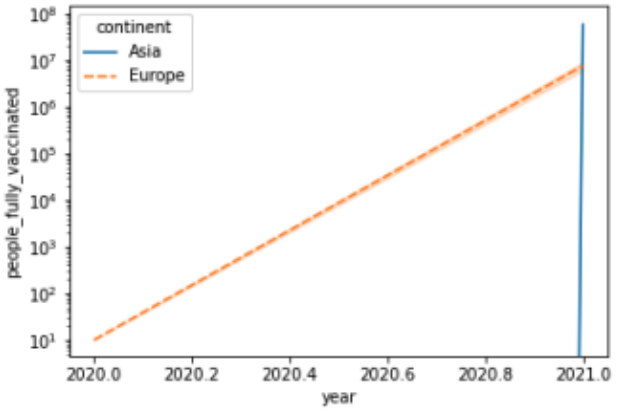
Data Visualization
Year, people_fully_vaccinated and continent

Interpretation
• There is a rapid increase in the number of fully vaccinated people from 2020 to 2021 in Europe. In Asia, all the fully vaccinations are in 2021.

TABLE XLVIII. Multivariate plot for Year, people_fully_vaccinated and continent

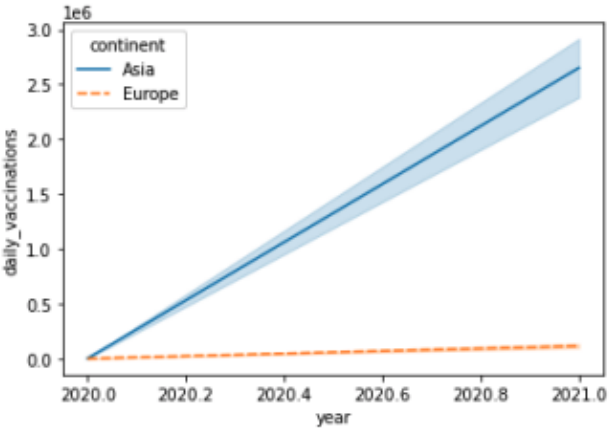
Data Visualization
year, daily_vaccinations and continent

Interpretation
• There is a positive linear relationship between people_vaccinated and daily_vaccinations.

TABLE XLIX. Multivariate plot for people_vaccinated and daily_vaccinations

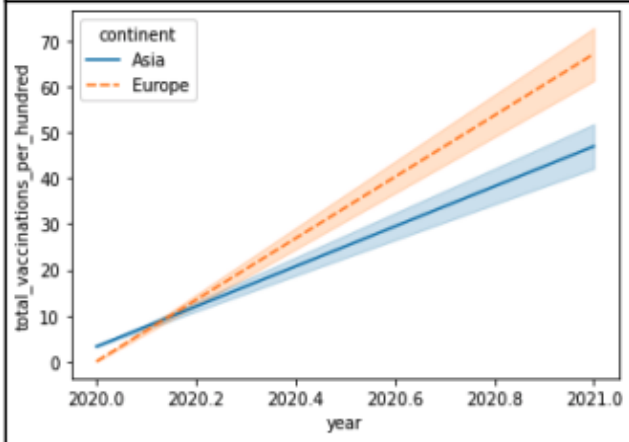
Data Visualization
year, total_vaccinations_per_hundred and continent

Interpretation
• There is a rapid increase in the number of total vaccinations per population from 2020 to 2021 in Asia and Europe. • The data is balanced.

TABLE L. Multivariate plot for year, total_vaccinations_per_hundred and continent

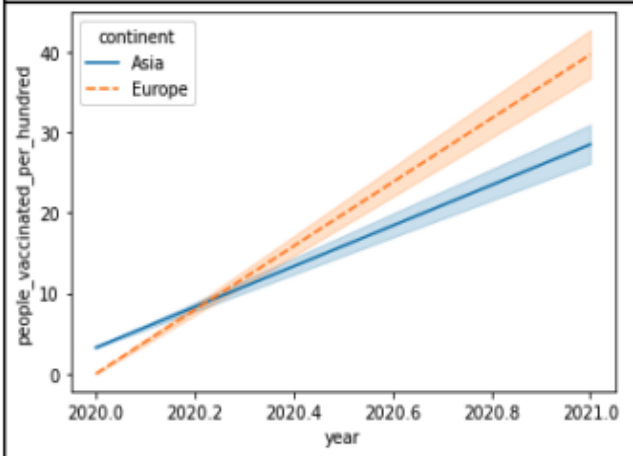
Data Visualization
year, people_vaccinated_per_hundred and continent

Interpretation
• There is an increase in the number of people vaccinated per hundred population from 2020 to 2021 in Asia and Europe. • The data is balanced.

TABLE LI. Multivariate plot for year, people_vaccinated_per_hundred and continent

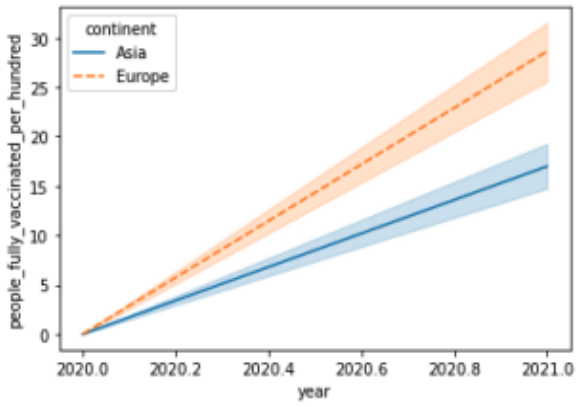
Data Visualization
year, people_fully_vaccinated_per_hundred and continent

Interpretation
• There is a rapid increase in the number of fully vaccinated people per population from 2020 to 2021 in Asia and Europe. • The data is balanced.

TABLE LII. Multivariate plot for year, people_fully_vaccinated_per_hundred and continent

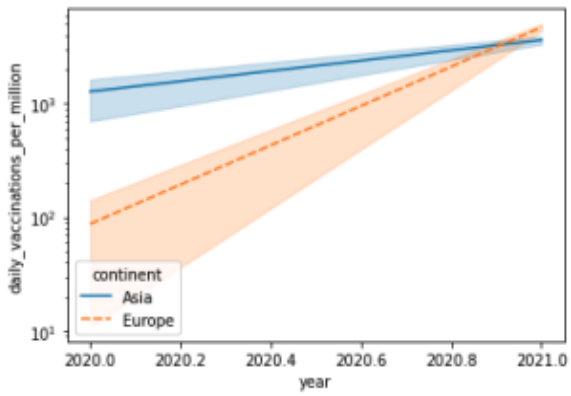
Data Visualization
year, daily_vaccinations_per_million and continent

Interpretation
• There is an increase in the number of daily vaccinations per million population from 2020 to 2021 in Asia and Europe. • The data is balanced.

TABLE LIII. Multivariate plot for year, daily_vaccinations_per_million and continent

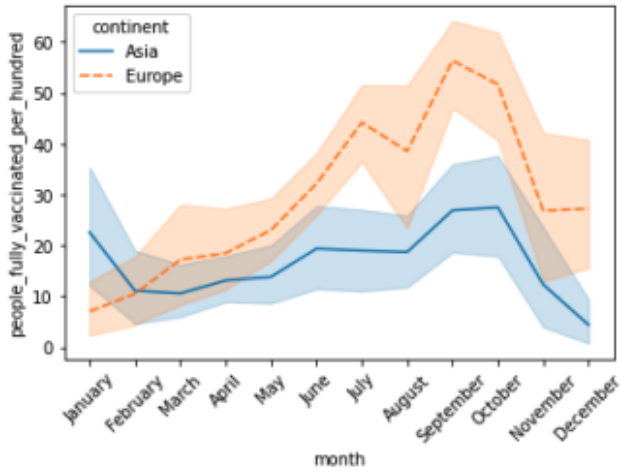
Data Visualization
Month, people_fully_vaccinated_per_hundred and continent

Interpretation
• Both Europe and Asia are showing an upward and downward trend in the months in the number of fully vaccinated people.

TABLE LIV. Multivariate plot for Month, people_fully_vaccinated_per_hundred and continent

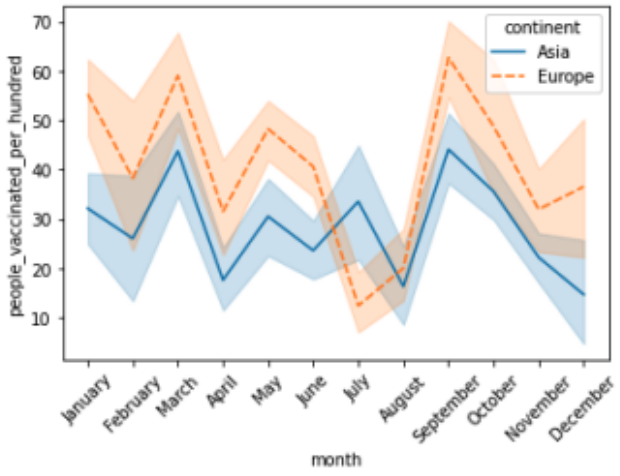
Data Visualization
Month, people_vaccinated_per_hundred and continent

Interpretation
• Both Europe and Asia are showing an upward and downward trend in the number of people vaccinated in the population from January to December.

TABLE LV. Multivariate plot for Month, people_vaccinated_per_hundred and continent

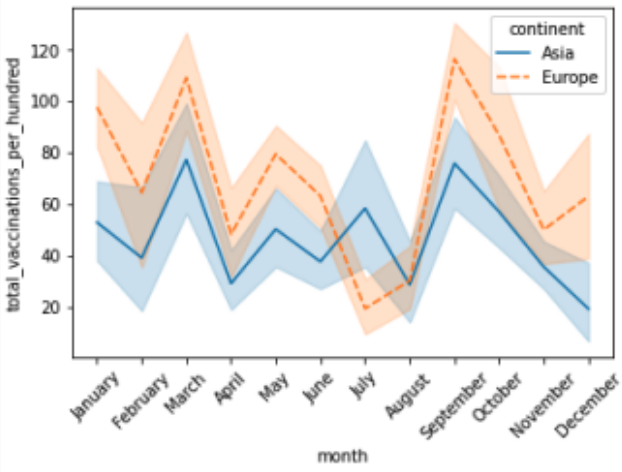
Data Visualization
Month, total_vaccinations_per_hundred and continent

Interpretation
• Both Europe and Asia are showing an upward and downward trend in the months in the number of total vaccinations in population.

TABLE LVI. Multivariate plot for Month, total_vaccinations_per_hundred and continent

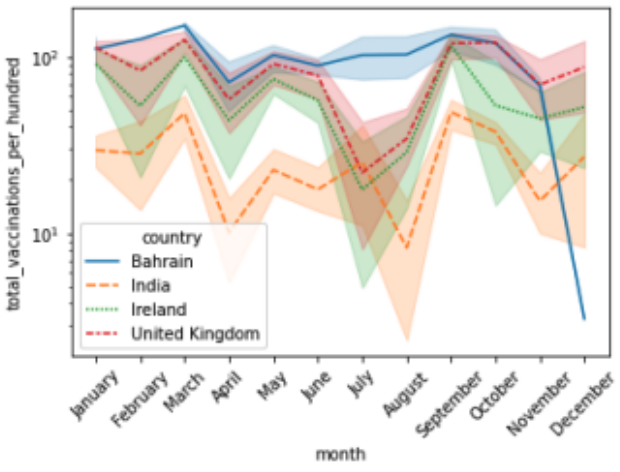
Data Visualization
Month, total_vaccinations_per_hundred and country

Interpretation
• Both Europe and Asia are showing an upward and downward trend in the number of total vaccinations in the population from January to December.

TABLE LVII. Multivariate plot for people_vaccinated and daily_vaccinations

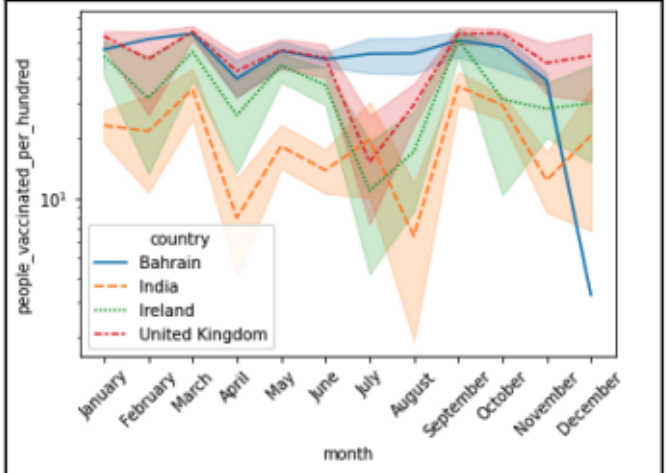
Data Visualization
Month, people_vaccinated_per_hundred and country

Interpretation
• All the four countries are showing an upward and downward trend in the number of people vaccinated in the population from January to December.

TABLE LVIII. Multivariate plot for Month, people_vaccinated_per_hundred and country

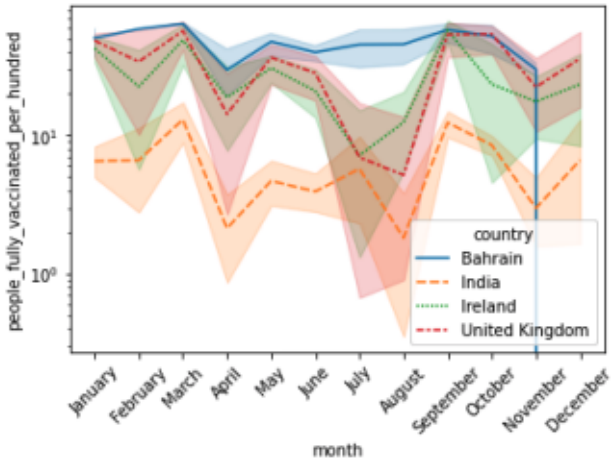
Data Visualization
Month, people_fully_vaccinated_per_hundred and country

Interpretation
• All the four countries are showing an upward and downward trend in the number of people fully vaccinated in the population from January to December.

TABLE LIX. Multivariate plot for Month, people_fully_vaccinated_per_hundred and country

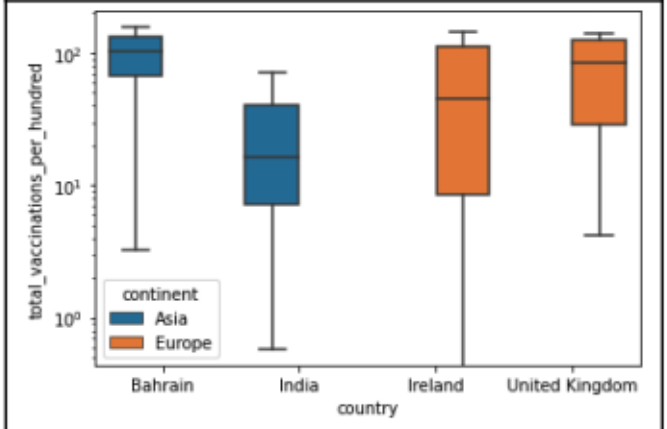
Data Visualization
country, total_vaccinations_per_hundred and continent

Interpretation
• There are no outliers present in the data. • Ireland has more spread than other countries. • The IQRs of India , Ireland and the United Kingdom slightly overlap each other. • Bahrain has the highest median among 4 countries.

TABLE LX. Multivariate plot for country, total_vaccinations_per_hundred and continent

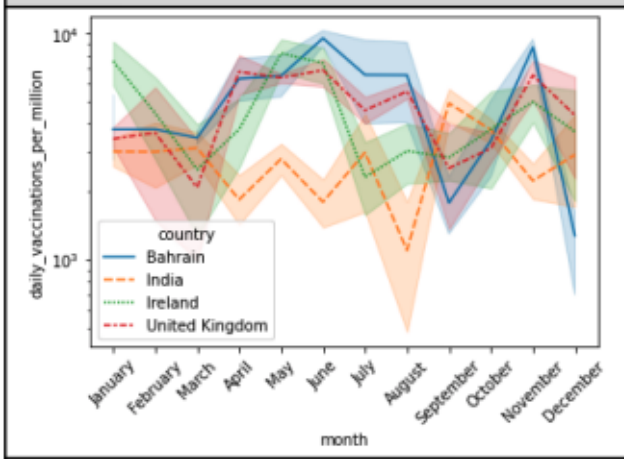
Data Visualization
Month, daily_vaccinations_per_million and country

Interpretation
• All the four countries are showing an upward and downward trend in the number of daily vaccinations in the population from January to December.

TABLE LXI. Multivariate plot for Month, daily_vaccinations_per_million and country

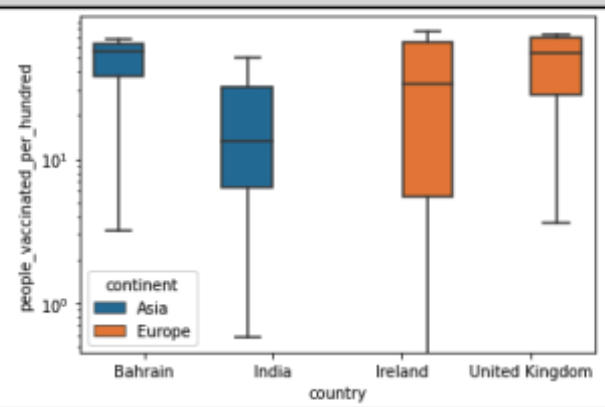
Data Visualization
country, people_vaccinated_per_hundred and continent

Interpretation
• There are no outliers present in the data. • Ireland has more spread than other countries. • The IQRs of India , Ireland and the United Kingdom slightly overlap each other. • Bahrain has the highest median among 4 countries.

TABLE LXII. Multivariate plot for country, people_vaccinated_per_hundred and continent

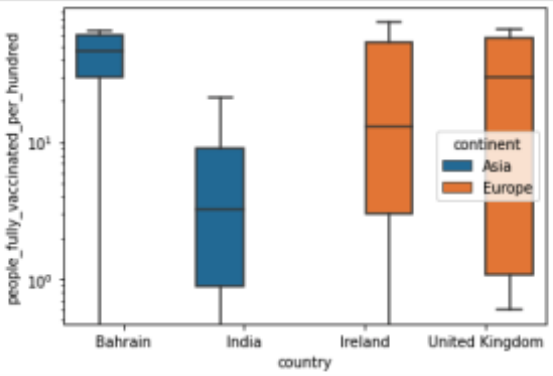
Data Visualization
country, people_fully_vaccinated_per_hundred and continent

Interpretation
• There are no outliers present in the data. • Bahrain has more spread than other countries. • The IQRs of Ireland and the United Kingdom slightly overlap each other. • Bahrain has the highest median among 4 countries.

TABLE LXIII. Multivariate plot for country, people_fully_vaccinated_per_hundred and continent

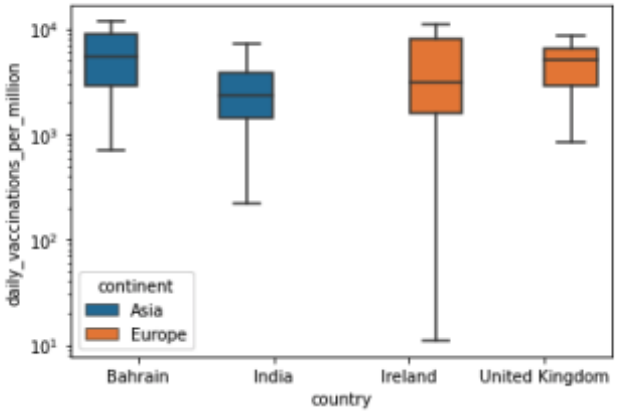
Data Visualization
country, daily_vaccinations_per_million and continent

Interpretation
• There are no outliers present in the data. • Ireland has more spread than other countries. • The IQRs of Ireland and the United Kingdom slightly overlap each other. • Bahrain has the highest median among 4 countries

TABLE LXIV. Multivariate plot for country, daily_vaccinations_per_million and continent

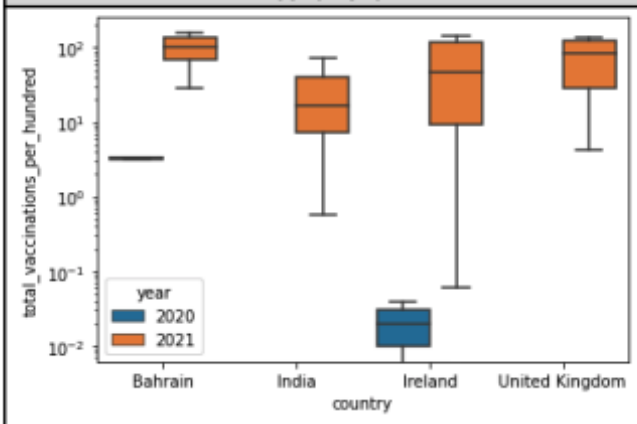
Data Visualization
country, total_vaccinations_per_hundred and continent

Interpretation
• Only Ireland and Bahrain have started vaccination in 2020. • Ireland has the highest spread. • Bahrain has the highest median in the 2021 year.

TABLE LXV. Multivariate plot for country, daily_vaccinations_per_million and continent

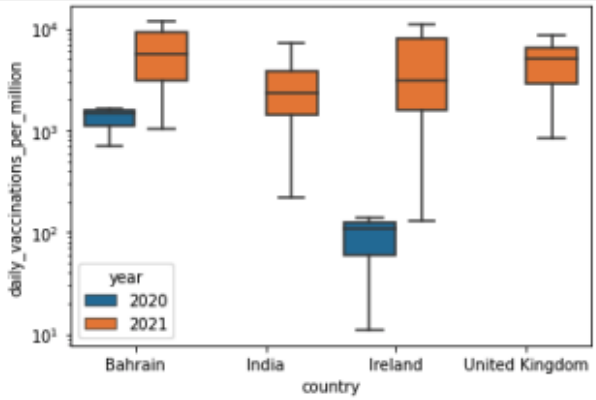
Data Visualization
country, daily_vaccinations_per_million and year

Interpretation
• Only Ireland and Bahrain have started daily vaccination in 2020. • Ireland has the highest spread. • Bahrain has the highest median in the 2021 year. • The IQRs of Ireland and the United Kingdom overlap in 2021.

TABLE LXVI. Multivariate plot for country, daily_vaccinations_per_million and year

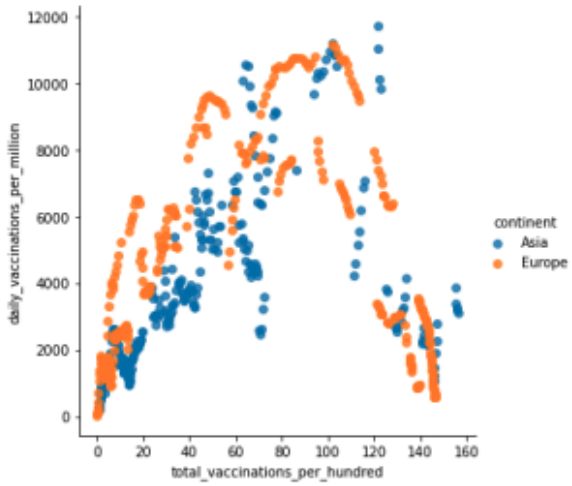
Data Visualization
total_vaccinations_per_hundred, daily_vaccinations_per_million and continent

Interpretation
• There is a linear relationship between the data for Asia and Europe. • A strong linear relationship exists between the variables for continent Asia. • There is only a small linear relationship between daily vaccination in population and total vaccination in population for Europe.

TABLE LXVII. Multivariate plot for total_vaccinations_per_hundred, daily_vaccinations_per_million and continent

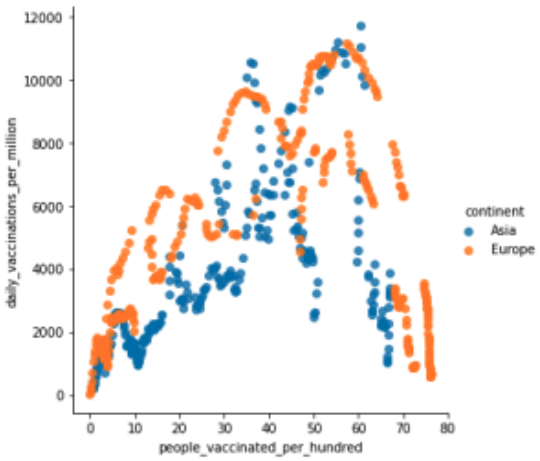
Data Visualization
people_vaccinated_per_hundred, daily_vaccinations_per_million and continent

Interpretation
• There is a linear relationship between the data for Asia and Europe. • A strong linear relationship exists between the variables for continent Asia. • There is only a small linear relationship between daily vaccination in population and people vaccinated in population for Europe

TABLE LXVIII. Multivariate plot people_vaccinated_per_hundred, daily_vaccinations_per_million and continent

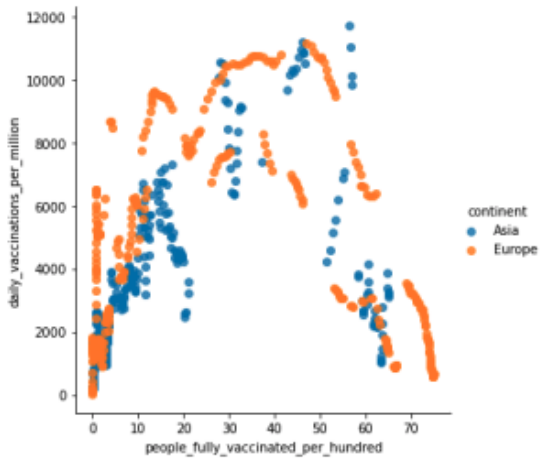
Data Visualization
people_fully_vaccinated_per_hundred, daily_vaccinations_per_million and continent

Interpretation
• There is a linear relationship between the data for Asia and Europe. • A strong linear relationship exists between the variables for continent Asia. • There is a small negative linear relationship between daily vaccination in population and people fully vaccinated in population for Europe

TABLE LXIX. Multivariate plot people_fully_vaccinated_per_hundred, daily_vaccinations_per_million and continent

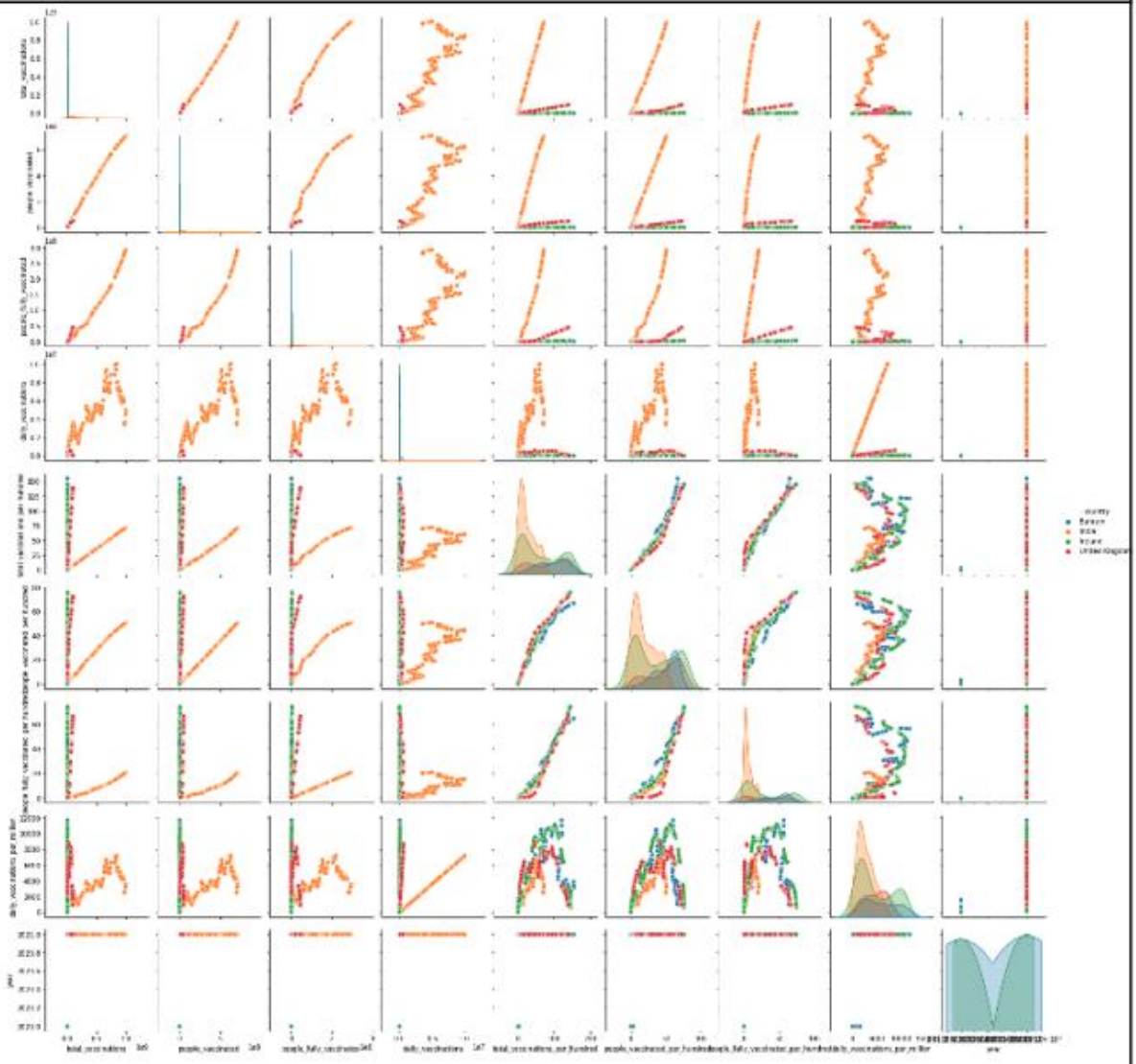
Data Visualization
Pairplot

Interpretation
• There is a positive linear relationship between the count variables..

TABLE LXX. Pair plot

VI. DESCRIPTIVE STATISTICS

This section includes the summary statistics of all variables in the dataset.Measures of central tendency, spread and correlation are also included in this section.

A. Country

There are a total of 646 observations within the *country* and it has 4 unique values. They are the United Kingdom, Ireland, India and Bahrain. The mode of the *country* variable is Ireland with a frequency of 230.

```
sel_data['country'].describe()

count         646
unique          4
top       Ireland
freq          230
Name: country, dtype: object
```

Figure 6. Summary statistics of variable 'country'

B. ISO_Code

There are a total of 646 observations within the *iso_code* and it has 4 unique values. The mode of the *iso_code* variable is IRL with a frequency of 230.

```
sel_data['iso_code'].describe()

count     646
unique      4
top       IRL
freq      230
Name: iso_code, dtype: object
```

Figure 7. Summary statistics of variable 'iso_code

C. Date

There are a total of 646 observations within the *date* and it has 295 unique values. The mode of the *date* variable is 2021-04-28 with a frequency of 4.The first date is 2020-12-29 and the last date is 2021-12-10.

```
count                    646
unique                   295
top      2021-04-28 00:00:00
freq                       4
first    2020-12-29 00:00:00
last     2021-12-10 00:00:00
Name: date, dtype: object
```

Figure 8. Summary statistics of variable 'date'

D. Total Vaccinations

The mean of the *total_vaccinations* is 1.284127e8 that is approximately 3815.62 vaccinations.The median is at 2920.82 vaccinations.There is a minimum of 10 number of total vaccinations and a maximum of 8127 total number of vaccinations.The IQR lies between 1.982450e+06 and 1.075320e+08.

```
sel_data['total_vaccinations'].describe()

count    6.460000e+02
mean     1.284127e+08
std      2.377074e+08
min      1.410000e+02
25%      1.982450e+06
50%      7.243817e+06
75%      1.075320e+08
max      1.003204e+09
Name: total_vaccinations, dtype: float64
```

Figure 9. Summary statistics of variable 'total_vaccinations'

E. People vaccinated

The mean of the *people_vaccinated* is 10626.37. That is 10626 people has vaccinated .The median is at 1529 number of people.There is a minimum of 10 number of people vaccinated and a maximum of 21135 number of people vaccinated.The IQR lies between 1.078896e+06 and 9.417913e+07.

```
sel_data['people_vaccinated'].describe()

count    6.460000e+02
mean     9.690832e+07
std      1.780241e+08
min      1.360000e+02
25%      1.078896e+06
50%      3.791142e+06
75%      9.417913e+07
max      7.095520e+08
Name: people_vaccinated, dtype: float64
```

Figure 10. Summary statistics of variable 'people_vaccinated'

F. People fully vaccinated

The mean of the *people_fully_vaccinated* is 3454. That is average of 3454 people has vaccinated with both doses .The median is at 1174 number of people.There is a minimum of 0 number of people fully vaccinated and a maximum of 8734 number of people vaccinated.The IQR lies between 0.000000e+00 and 3.859196e+07.

```
sel_data['people_fully_vaccinated'].describe()

count    6.460000e+02
mean     3.152000e+07
std      6.076850e+07
min      0.000000e+00
25%      5.360668e+05
50%      2.917430e+06
75%      3.859196e+07
max      2.936523e+08
Name: people_fully_vaccinated, dtype: float64
```

Figure 11.. Summary statistics of variable 'people_fully_vaccinated'

G. Daily vaccinations

The mean of the *daily_vaccinations* is approximately 532.52 vaccinations.The median is at 310 vaccinations.There is a minimum of 15 number of daily vaccinations given and a maximum of 1100 daily vaccinations.The IQR lies between 1.277475e+04 and 2.011813e+06.

```
sel_data['daily_vaccinations'].describe()

count    6.460000e+02
mean     1.327012e+06
std      2.200616e+06
min      5.700000e+01
25%      1.277475e+04
50%      5.686950e+04
75%      2.011813e+06
max      1.003800e+07
Name: daily_vaccinations, dtype: float64
```

Figure 12. Summary statistics of variable 'daily_vaccinations'

H. Total vaccinations per hundred

The mean of the *total_vaccinations_per_hundred* is approximately 56.89 vaccinations per hundred.The median is at 42.44 vaccinations per hundred of population.There is a minimum of 0 number of vaccinations given and a maximum of 156 vaccinations per hundred of population.The IQR lies between 11.26 and 101.76 vaccinations.

```
sel_data['total_vaccinations_per_hundred'].describe()

count    646.000000
mean      56.885201
std       50.081623
min        0.000000
25%       11.267500
50%       42.440000
75%      101.717500
max      156.140000
Name: total_vaccinations_per_hundred, dtype: float64
```

Figure 13. Summary statistics of variable 'total_vaccinations_per_hundred'

I. People vaccinated per hundred

The mean of the *people_vaccinated_per_hundred* is approximately 33.94 number of people per hundred of population.The median is at 31.8 number of people per hundred of population.There is a minimum of 0 number of people vaccinated in entire population and a maximum number of 76 people vaccinated per hundred of population.The IQR lies between 9.07 and 57.84 people vaccinated.

```
sel_data['people_vaccinated_per_hundred'].describe()

count    646.000000
mean      33.944381
std       25.453184
min        0.000000
25%        9.075000
50%       31.790000
75%       57.845000
max       76.220000
Name: people_vaccinated_per_hundred, dtype: float64
```

Figure 14. Summary statistics of variable 'people_vaccinated_per_hundred'

J. People fully vaccinated per hundred

The mean of the *people_fully_vaccinated_per_hundred* is approximately 22.77 number of people fully vaccinated per hundred of population.The median is at 10.43 that is 10 number of people fully vaccinated per hundred of population.There is a minimum of 0 number of people vaccinated in entire population and a maximum number of approximately 75 people fully vaccinated per hundred of population.The IQR lies between 2.27 and 44.72 number of people vaccinated.

```
sel_data['people_fully_vaccinated_per_hundred'].describe()

count    646.000000
mean      22.779396
std       25.124573
min        0.000000
25%        2.272500
50%       10.430000
75%       44.720000
max       74.970000
Name: people_fully_vaccinated_per_hundred, dtype: float64
```

Figure 15. Summary statistics of variable 'people_fully_vaccinated_per_hundred'

K. Daily vaccinations per million

The mean of the *daily_vaccinations_per_million*is approximately 4142.4 vaccinations per million of population.The median is at 3227.5 vaccinations per million of population.There is a minimum of 11 number of vaccinations given and a maximum of 11723 vaccinations per million of population.The IQR lies between 1652.5 and 6215.5 vaccinations.

```
sel_data['daily_vaccinations_per_million'].describe()

count      646.000000
mean      4142.394737
std       3009.103827
min         11.000000
25%       1652.500000
50%       3227.500000
75%       6215.500000
max      11723.000000
Name: daily_vaccinations_per_million, dtype: float64
```

Figure 16. Summary statistics of variable 'daily_vaccinations_per_million'

L. Vaccines

There are a total of 646 observations within the *vaccines* and it has 4 unique values.The mode of the *vaccines* variable is 'Johnson&Johnson, Moderna, Oxford/Astrazeneca, pFizer with a frequency of 230.

```
sel_data['vaccines'].describe()

count                                                  646
unique                                                   4
top       Johnson&Johnson, Moderna, Oxford/AstraZeneca, ...
freq                                                   230
Name: vaccines, dtype: object
```

Figure 17. Summary statistics of variable 'vaccines'

```
sel_data['continent'].describe()

count        646
unique         2
top       Europe
freq         334
Name: continent, dtype: object
```

M. Source name

There are a total of 646 observations for the variable *source_name* and it has 4 unique values.The mode of the *source_name* variable is 'Health Service Executive' with a frequency of 230.

```
sel_data['source_name'].describe()

count                           646
unique                            4
top       Heath Service Executive
freq                            230
Name: source_name, dtype: object
```

Figure 18. Summary statistics of variable 'source_name'

N. Year

There are a total of 646 observations for the variable *year.* The mean of year is 2020 and median is 2021.Minimum observations are in 2020 and maximum observations in 2021.The IQR lies within the year 2021.

```
sel_data['year'].describe()

count     646.000000
mean     2020.990712
std         0.096000
min      2020.000000
25%      2021.000000
50%      2021.000000
75%      2021.000000
max      2021.000000
Name: year, dtype: float64
```

Figure 19. Summary statistics of variable 'year'

O. Month

There are a total of 646 observations for the variable *month.*. There are 12 unique months present from January to December. The most vaccinations are given in the month of May with a frequency of 96 days.

```
sel_data['month'].describe()

count     646
unique     12
top       May
freq       69
Name: month, dtype: object
```

Figure 20. Summary statistics of variable 'month'

P. Continent

There are a total of 646 observations for the variable *continent.*. There are 2 unique values present in the sample dataset. They are Asia and Europe.. The most observations are in Europe with a frequency of 334.

```
sel_data['continent'].describe()
```

```
count        646
unique         2
top       Europe
freq         334
Name: continent, dtype: object
```

Figure 21. Summary statistics of variable 'continent'

Q. Correlation and spread

From the visualizations in section V the variables *'daily_vaccinations'* is showing linear relationship with variables *'people_vaccinated', 'people_fully_vaccinated' and 'total_vaccinations'*.Similarly, the variable *'daily_vaccinations_per_million'* is showing linear relationship with variables *'people_vaccinated_per_hundred', 'people_fully_vaccinated_per_hundred' and 'total_vaccinations_per_hundred*. Also variable *'year'* is showing a linear relationship between *'daily_vaccinations_per_million', 'people_vaccinated_per_hundred', 'people_fully_vaccinated_per_hundred' and 'total_vaccinations_per_hundred.*

Since the data is not normally distributed I will be using Spearman coefficient and for the data with outliers I will be using Kendall Tau, because it is the best to deal with the outliers.So for the correlation for *'daily_vaccinations' 'people_vaccinated', 'people_fully_vaccinated'* and *'total_vaccinations'* I decided to use Kendall Tau since it has outliers in the data. For the others I will be using Spearman coefficients because the data is not normally distributed. So I can not use Pearson coefficient.

- *daily_vaccinations and total_vaccinations*

```
corr_d1= kendalltau(sel_data["daily_vaccinations"], sel_data["total_vaccinations"])
corr_d1
```

```
KendalltauResult(correlation=0.7100637444815846, pvalue=1.5813370089538293e-160)
```

Figure 22. Correlation between 'daily_vaccinations and total_vaccinations'

The correlation coefficient is 0.71. So it shows a strong positive linear relationship.

- *daily_vaccinations and people_vaccinated*

```
corr_d2= kendalltau(sel_data["daily_vaccinations"], sel_data["people_vaccinated"])
corr_d2
```

```
KendalltauResult(correlation=0.731222279898405, pvalue=4.106103591132174e-170)
```

Figure 23. Correlation between 'daily_vaccinations and people_vaccinated'

The correlation coefficient is 0.73. So it shows a strong positive linear relationship between the variables *daily_vaccinations and people_vaccinated.*

- *daily_vaccinations and people_fully_vaccinated*

```
corr_d3= kendalltau(sel_data["daily_vaccinations"], sel_data["people_fully_vaccinated"])
corr_d3
```

```
KendalltauResult(correlation=0.6051656984778233, pvalue=3.9256708757503784e-117)
```

Figure 24. Correlation between 'daily_vaccinations and people_fully_vaccinated'

The correlation coefficient is 0.60. So it shows a strong positive linear relationship between the variables *daily_vaccinations and people_fully_vaccinated.*

- *daily_vaccinations_per_million and total_vaccinations_per_hundred*

```
corr_d4 = spearmanr(sel_data["total_vaccinations_per_hundred"], sel_data["daily_vaccinations_per_million"])
corr_d4
```

```
SpearmanrResult(correlation=0.41707910285019856, pvalue=1.4098452514826318e-28)
```

Figure 25. Correlation between 'daily_vaccinations and total_vaccinations_per_hundred'

The correlation coefficient is 0.41. So it shows a medium positive linear relationship between the variables *ddaily_vaccinations_per_million and total_vaccinations_per_hundred*

- *daily_vaccinations_per_million and people_vaccinated_per_hundred*

```
corr_d5= spearmanr(sel_data["daily_vaccinations_per_million"], sel_data["people_vaccinated_per_hundred"])
corr_d5
```

```
SpearmanrResult(correlation=0.4153347144694074, pvalue=2.49209972646122e-28)
```

Figure 26. Correlation between 'daily_vaccinations and people_vaccinated_per_hundred'

The correlation coefficient is 0.41. So it shows a medium positive linear relationship between the variables *ddaily_vaccinations_per_million and people_vaccinated_per_hundred*

- *daily_vaccinations_per_million and people_fully_vaccinated_per_hundred*

```
corr_d6= spearmanr(sel_data["people_fully_vaccinated_per_hundred"], sel_data["daily_vaccinations_per_million"])
corr_d6
```

```
SpearmanrResult(correlation=0.39037342213798276, pvalue=6.040720386196199e-25)
```

Figure 27. Correlation between 'daily_vaccinations_per_million and people_fully_vaccinated_per_hundred'

The correlation coefficient is 0.39. So it shows a weak positive linear relationship between the variables *daily_vaccinations_per_million and people_fully_vaccinated_per_hundred*

- *year and daily_vaccinations_per_million*

```
corr_d7= spearmanr(sel_data["total_vaccinations_per_hundred"], sel_data["year"])
corr_d7
```

```
SpearmanrResult(correlation=0.150095400856671, pvalue=0.00012854535937752598)
```

Figure 28. Correlation between 'year and daily_vaccinations_per_million'

The correlation coefficient is 0.15. So it shows a weak positive linear relationship between the variables *year and daily_vaccinations_per_million.*

- *year and total_vaccinations_per_hundred*

```
corr_d7= spearmanr(sel_data["total_vaccinations_per_hundred"], sel_data["year"])
corr_d7
```

```
SpearmanrResult(correlation=0.150095400856671, pvalue=0.00012854535937752598)
```

Figure 29. Correlation between 'year and total_vaccinations_per_hundred'

The correlation coefficient is 0.15. So it shows a very weak positive linear relationship between the variables *year and total_vaccinations_per_hundred*

- *year and people_fully_vaccinated_per_hundred*

```
corr_d9= spearmanr(sel_data["people_fully_vaccinated_per_hundred"], sel_data["year"])
corr_d9
```

```
SpearmanrResult(correlation=0.160960943767872, pvalue=3.9570424695439016e-05)
```

Figure 30. Correlation between ' year and people_fully_vaccinated_per_hundred'

The correlation coefficient is 0.16. So it shows a very weak positive linear relationship between the variables *year and people_fully_vaccinated_per_hundred*

- *year and people_vaccinated_per_hundred*

```
corr_d10= spearmanr(sel_data["people_vaccinated_per_hundred"], sel_data["year"])
corr_d10
```

```
SpearmanrResult(correlation=0.14659073931734956, pvalue=0.00018488028448843745)
```

Figure 31. Correlation between 'year and people_vaccinated_per_hundred'

The correlation coefficient is 0.14. So it shows a very weak positive linear relationship between the variables *year and people_vaccinated_per_hundred.*

VII. HYPOTHESIS TEST

A. Two sample independent test

In this section I want to assess if there is any relationship between the continents and the progress in vaccination. So I decided to use the two sample independent tests to check if the continent and daily vaccinations are related. The variables of interest in this test are the '*continent'* and *'daily_vaccinations_per_million'*

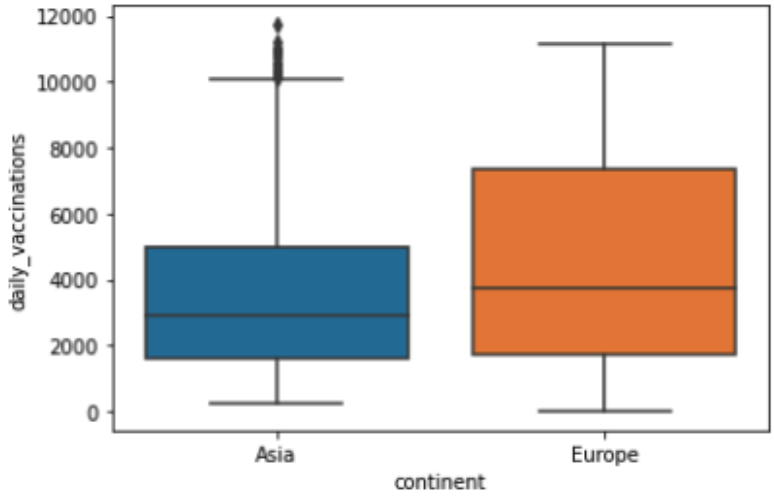


Figure 32. Bivariate plot between 'daily_vaccinations_per_million' and 'continent'

Null Hypothesis, H0: The mean daily vaccinations in Europe is equal to the mean daily vaccinations in Asia (i.e. there is no relationship between the mean daily vaccinations in Europe and Asia).

Alternate Hypothesis, H1: The mean daily vaccinations in Europe is not equal to the mean daily vaccinations in Asia (i.e. there is a relationship between the mean daily vaccinations in Europe and Asia).

Assumptions for 2 sample independent test :

- Two samples must be random and independent.
- The samples should be normally distributed.
- The samples should have the same standard deviation.

Here from the box plot it is clear that the data is not normally distributed and the Asia continent data is skewed to the right.So I won't be able to use the 2 sample independent tests here. So I decided to use the Wilcoxon- Mann-Whitney(WMW) sample test to compare the medians of the data.

Null hypothesis, H0 : The median of daily vaccinations in Europe is equal to the median of daily vaccinations in Asia (i.e. there is no relationship between the median of daily vaccinations in Europe and Asia).

Alternate Hypothesis, H1: The median of daily vaccinations in Europe is not equal to the median of daily vaccinations in Asia (i.e. there is a relationship between the median of daily vaccinations in Europe and Asia).

```
one_vacc = sel_data[sel_data["continent"] == 'Asia']['daily_vaccinations_per_million']
two_vacc = sel_data[sel_data["continent"] == 'Europe']['daily_vaccinations_per_million']

ranksums(one_vacc,two_vacc)
```

```
RanksumsResult(statistic=-3.6627538521554364, pvalue=0.0002495182805865695)
```

Figure 33. Rank sum Test

Here the *p-value* is 0.0002 and the significant level is 5% .

p-value < 0.05, so we will reject the null hypothesis, H0 and hence there is a relationship between the median of daily vaccinations in Europe and Asia.

B. Chi-Squared Test

In order to answer the research question about total vaccinations and fully vaccinated people and to assess if there is any relation between the total vaccinations and the number of fully vaccinated people in countries, I am grouping the variables '*total_vaccinations'* and '*people_fully_vaccinated'* into categories. And to assess the relationship between the daily vaccinations and the year of vaccination I am grouping *daily_vaccinations into 2 groups'*.So Iam performing the Chi-Squared test with those two variables.Here I selected the count variables instead of the rate variables because I am focusing on the vaccination count only and not on the population.I created 2 categories for both the variables.I looked into the highest and smallest numbers for both variables and then created 2 categories.

NOTE: I tried One way ANOVA with daily_vaccinations and year. But I couldn't proceed with the test since the assumptions were not satisfied. So I used the Chi-Squared Test.

- '*total_vaccinations'* and '*people_fully_vaccinated'*

1. *total_vaccinations*
   - 0 - 450000000 : Group A
   - 450000000 - 900000000 : Group B
2. *people_fully_vaccinated*
   - 0 - 120000000 *:* FullGroup A
   - 120000000 - 250000000 : FullGroup B

```
totalvacc_peoplevacc = pd.crosstab(index=desc['people_fully_vaccinated_group'], columns=desc["total_vaccination_group"])
totalvacc_peoplevacc
```

total_vaccination_group	Total A	Total B
people_fully_vaccinated_group		
FullGroup A	569	14
FullGroup B	0	42

```
chi2_contingency(totalvacc_peoplevacc)
```

```
(445.6070505375738,
 6.518583859518397e-99,
 1,
 array([[530.7632,  52.2368],
        [ 38.2368,   3.7632]]))
```

Figure 34. Chi-Squared Test 1

Assumptions for Chi-Squared Test :

1. Random sample
2. Independent observations in each cell

3. Expected values should be greater than or equal to 5

All the three assumptions are satisfied.

Null Hypothesis, H0: There is no relationship between total vaccinations and people fully vaccinated (i.e. people_fully_vaccinated and total_vaccinations are independent)

Alternative hypothesis, H1: There is relationship between total vaccinations and people fully vaccinated (i.e. people_fully_vaccinated and total_vaccinations are dependent)

Here *p-value*, 0.0007 < 0.05, so we will reject null hypothesis.Hence there is a relationship between total vaccinations and people fully vaccinated

- *daily_vaccinations and year*

```
dailyvacc_year = pd.crosstab(index=desc['year'], columns=desc["daily_vaccn_group"])
dailyvacc_year
```

daily_vaccn_group	Group A	Group B
year		
2020	6	0
2021	466	174

```
chi2_contingency(dailyvacc_year)
```

```
(1.064830730435012,
 0.3021159874898026,
 1,
 array([[  4.38390093,   1.61609907],
        [467.61609907, 172.38390093]]))
```

Figure 35. Chi-Squared Test 2

Assumptions for Chi-Squared Test :

1. Random sample
2. Independent observations in each cell
3. Expected values should be greater than or equal to 5

The third assumption is not met since only one value is greater than 5.

Hence Chi-squared test cannot be performed.So, I am performing the Fisher Exact test, since the table is 2x2 and the expected values are less than 5.

```
fisher_exact(dailyvacc_year)
```

```
(inf, 0.19878131014557648)
```

Figure 36. Fisher Exact Test

Null Hypothesis, H0: There is no relationship between daily vaccinations and year (i.e. year and daily_vaccinations are independent)

Alternative hypothesis, H1: There is relationship between daily vaccinations and year (i.e. year and daily_vaccinations are dependent)

Here p-value, 0.19 > 0.05, so we failed to reject the null hypothesis.Hence there is not enough evidence to suggest there is relationship between daily vaccinations and year.

VIII. ANOVA TEST

A. *One Way ANOVA*

- *daily_vaccinations_per_million* and *month*

Assumptions for One-Way ANOVA:

1. Each of the populations is Normally distributed with the same variance (homogeneity of variance).
2. The observations are sampled independently, the groups under consideration are independent.

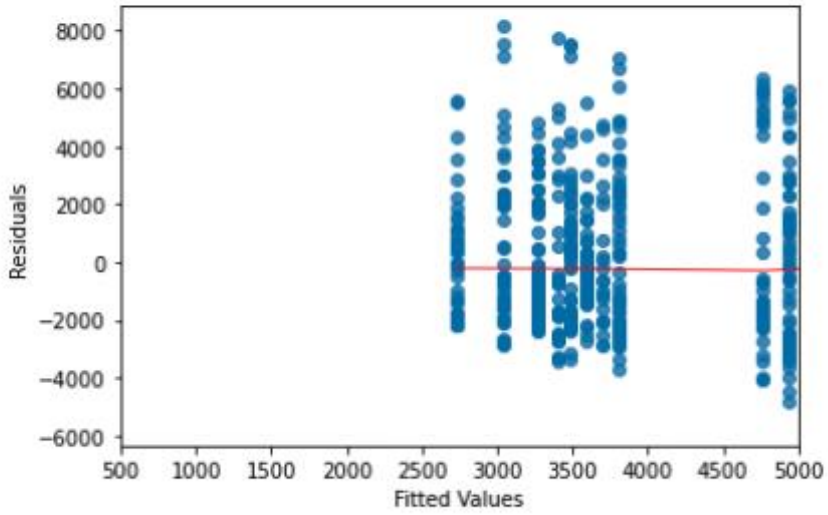


Figure 37. Fitted values vs Residuals plot for 'daily_vaccinations_per_million' and 'month'

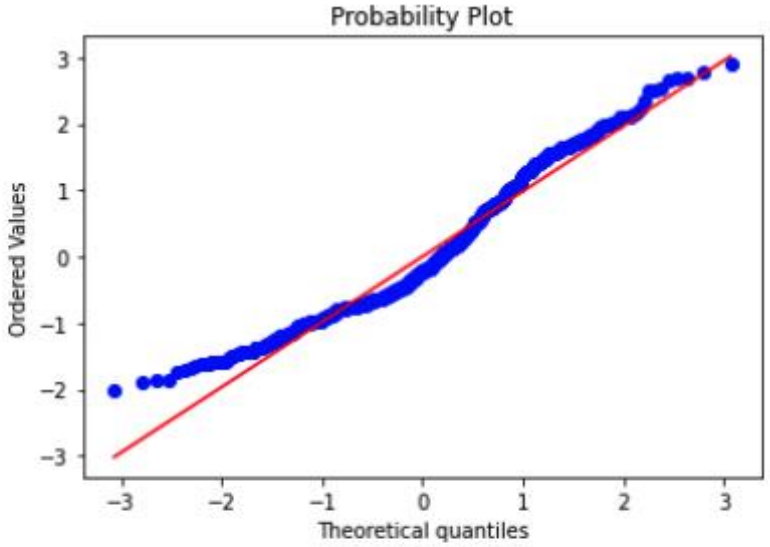


Figure 38. Probability plot of the residuals for 'daily_vaccinations_per_million' and 'month'

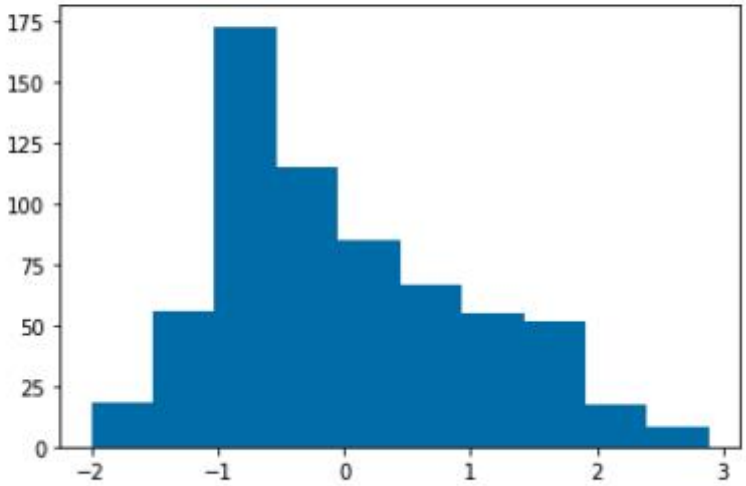


Figure 39. Histogram of the residuals for 'daily_vaccinations_per_million' and 'month'

The assumptions are roughly satisfied. Since there is roughly equal variance and equal spread and the qq plot is roughly normal.There is no strong deviation from normality.

Model Summary

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	R-squared:	0.122
Model:	OLS	Adj. R-squared:	0.107
Method:	Least Squares	F-statistic:	8.020
Date:	Sat, 11 Dec 2021	Prob (F-statistic):	3.57e-13
Time:	10:31:00	Log-Likelihood:	-6048.1
No. Observations:	646	AIC:	1.212e+04
Df Residuals:	634	BIC:	1.217e+04
Df Model:	11		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	4940.9385	352.716	14.008	0.000	4248.306	5633.571
C(month)[T.August]	-1355.3968	541.182	-2.505	0.013	-2418.123	-292.670
C(month)[T.December]	-1543.2242	642.817	-2.401	0.017	-2805.532	-280.916
C(month)[T.February]	-1896.2877	502.759	-3.772	0.000	-2883.561	-909.014
C(month)[T.January]	-1457.9385	516.021	-2.825	0.005	-2471.255	-444.621
C(month)[T.July]	-179.1139	516.021	-0.347	0.729	-1192.431	834.203
C(month)[T.June]	1113.9385	498.815	2.233	0.026	134.408	2093.469
C(month)[T.March]	-1138.6686	502.759	-2.265	0.024	-2125.942	-151.395
C(month)[T.May]	567.9021	491.533	1.155	0.248	-397.328	1533.132
C(month)[T.November]	-1241.5294	701.412	-1.770	0.077	-2618.901	135.842
C(month)[T.October]	-2213.4917	544.484	-4.065	0.000	-3282.701	-1144.282
C(month)[T.September]	-1666.4385	504.813	-3.301	0.001	-2657.747	-675.130

Omnibus:	41.339	Durbin-Watson:	0.392
Prob(Omnibus):	0.000	Jarque-Bera (JB):	41.026
Skew:	0.569	Prob(JB):	1.23e-09
Kurtosis:	2.520	Cond. No.	12.0

Figure 40. Model summary for One-way ANOVA between 'daily_vaccinations_per_million' and 'month'

```
sm.stats.anova_lm(onewaymodel1, typ=2)
```

	sum_sq	df	F	PR(>F)
C(month)	7.134142e+08	11.0	8.020213	3.571618e-13
Residual	5.126871e+09	634.0	NaN	NaN

Figure 41. One-way ANOVA between 'daily_vaccinations_per_million' and 'month'

Type =2 is used here, because the counts are not balanced.

Null Hypothesis, H0: true mean of daily vaccinations are equal across months.

H0: mu_jan = mu_feb =mu_mar = mu_apr =mu_may = mu_jun = mu_jul = mu_aug = mu_sep = mu_oct = mu_nov =mu_dec, mu_i is the true mean daily vaccinations for group i.

Alternate Hypothesis, H1: at least one mean of daily vaccinations is different across months.

p-value = 0.00000806 < 0.05, so we reject the null hypothesis and hence the true mean of daily vaccinations are equal across months.

- *daily_vaccinations_per_million* and *country*

Assumptions for One-Way ANOVA:

- Each of the populations is Normally distributed with the same variance (homogeneity of variance).
- The observations are sampled independently, the groups under consideration are independent.

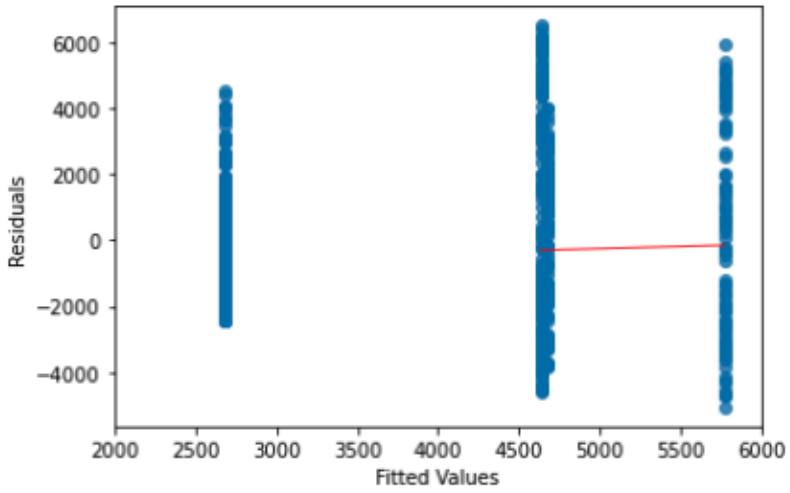


Figure 42. Fitted values vs Residuals plot for 'daily_vaccinations_per_million' and 'country'

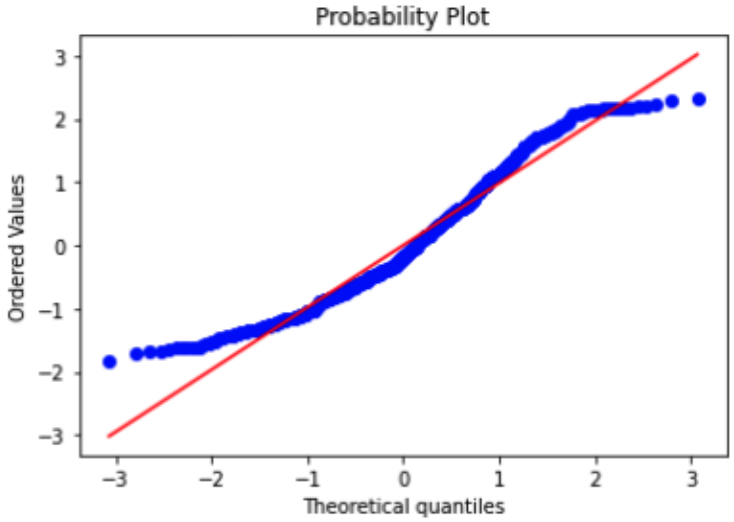


Figure 43. Probability plot of the residuals for 'daily_vaccinations_per_million' and 'country'

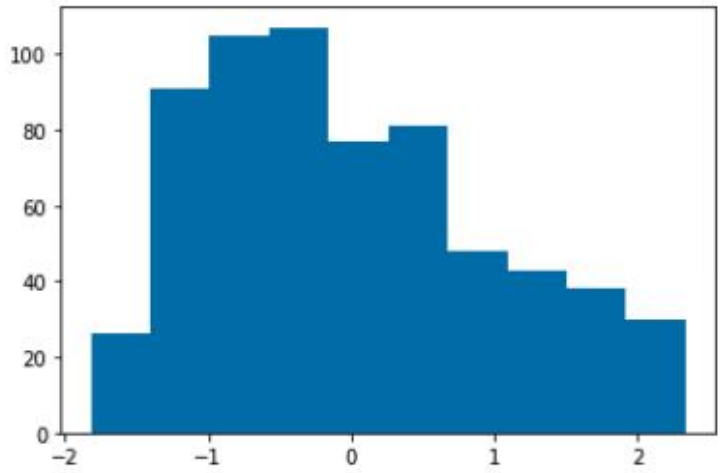


Figure 44. Histogram of the residuals for 'daily_vaccinations_per_million' and 'country'

The assumptions are roughly satisfied. There is roughly equal variance and equal spread and there is no strong deviation from normality.

Model Summary

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	**R-squared:**	0.137
Model:	OLS	**Adj. R-squared:**	0.133
Method:	Least Squares	**F-statistic:**	33.98
Date:	Sat, 11 Dec 2021	**Prob (F-statistic):**	2.15e-20
Time:	10:31:03	**Log-Likelihood:**	-6042.6
No. Observations:	646	**AIC:**	1.209e+04
Df Residuals:	642	**BIC:**	1.211e+04
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	5770.4409	290.539	19.861	0.000	5199.919	6340.962
C(country)[T.India]	-3090.4272	346.784	-8.912	0.000	-3771.396	-2409.458
C(country)[T.Ireland]	-1135.2930	344.304	-3.297	0.001	-1811.391	-459.195
C(country)[T.United Kingdom]	-1094.2005	399.872	-2.736	0.006	-1879.415	-308.986

Omnibus:	51.505	**Durbin-Watson:**	0.371
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	37.036
Skew:	0.479	**Prob(JB):**	9.07e-09
Kurtosis:	2.323	**Cond. No.**	6.30

Figure 45. Model summary for One-way ANOVA between 'daily_vaccinations_per_million' and 'country'

Type =2 is used here for the ANOVA test, because the counts are not balanced.

```
sm.stats.anova_lm(onewaymodel2, typ=2)
```

	sum_sq	df	F	PR(>F)
C(country)	8.003283e+08	3.0	33.982485	2.150283e-20
Residual	5.039957e+09	642.0	NaN	NaN

Figure 46. One-way ANOVA between 'daily_vaccinations_per_million' and 'country

Null Hypothesis, H0: true mean of daily vaccinations are equal across countries

H0: mu_bahrain = mu_india =mu_ireland = mu_uk, mu_i is the true mean daily vaccinations for group i.

Alternate Hypothesis, H1: at least one means of daily vaccinations is different across the countries.

p-value = 0.0005 < 0.05, so we reject the Null Hypothesis and hence there is a relationship between daily vaccinations across the countries.(ie..at least one mean of daily vaccinations is different)

- '*daily_vaccinations_per_million* and *continent*

Assumptions for One-Way ANOVA:

- Each of the populations is Normally distributed with the same variance (homogeneity of variance)
- The observations are sampled independently, the groups under consideration are independent.

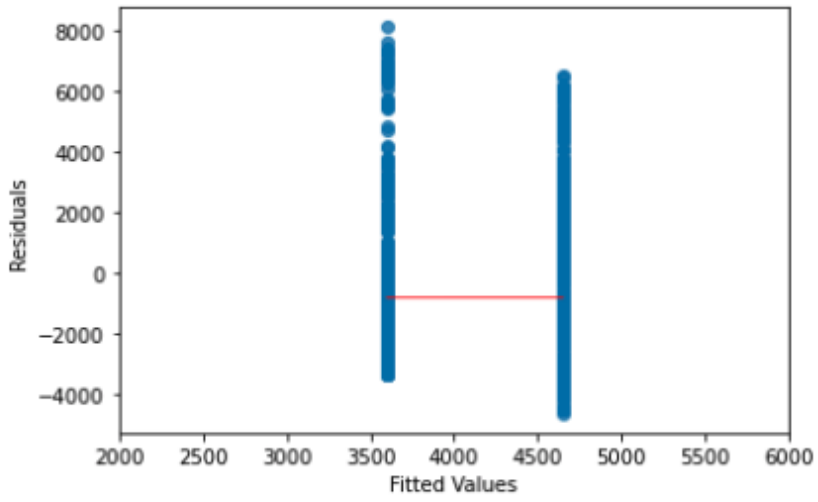


Figure 47. Fitted values vs Residuals plot for 'daily_vaccinations_per_million' and 'continent'

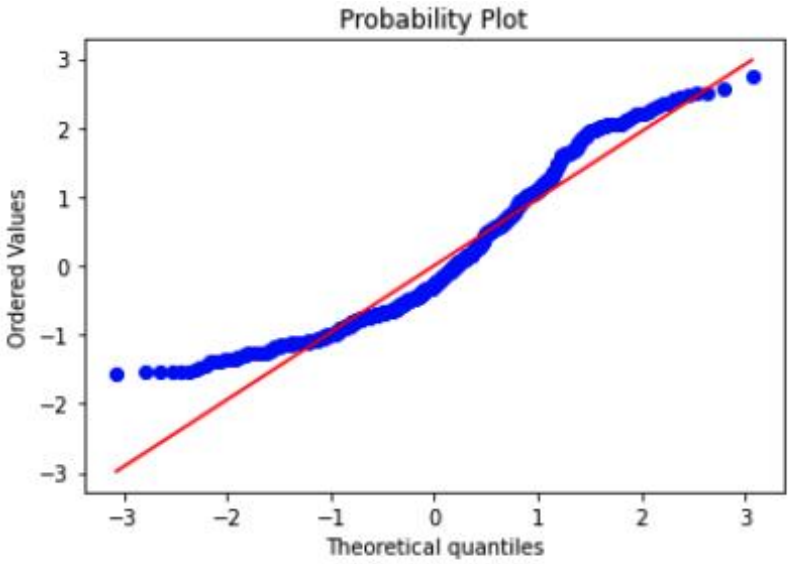


Figure 48. Probability plot for 'daily_vaccinations_per_million' and 'continent'

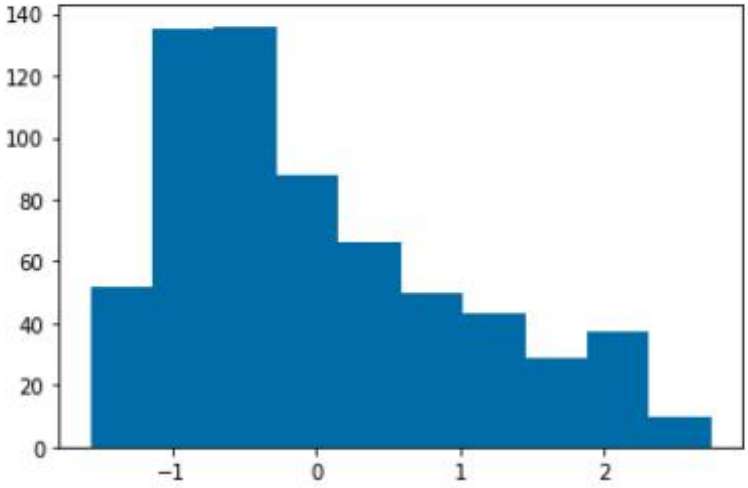


Figure 49. Histogram of residuals for 'daily_vaccinations_per_million' and 'continent'

The assumptions are roughly satisfied. There is roughly equal spread and variance and there is no strong deviation from normality.

As the counts are not balanced, I will be using Type= 2 for ANOVA test.

```
sm.stats.anova_lm(onewaymodel3, typ=2)
```

	sum_sq	df	F	PR(>F)
C(continent)	1.767461e+08	1.0	20.097766	0.000009
Residual	5.663539e+09	644.0	NaN	NaN

Figure 50. One-way ANOVA between 'daily_vaccinations_per_million' and 'continent'

Model Summary

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	**R-squared:**	0.030
Model:	OLS	**Adj. R-squared:**	0.029
Method:	Least Squares	**F-statistic:**	20.10
Date:	Sat, 11 Dec 2021	**Prob (F-statistic):**	8.71e-06
Time:	15:51:32	**Log-Likelihood:**	-6080.3
No. Observations:	646	**AIC:**	1.216e+04
Df Residuals:	644	**BIC:**	1.217e+04
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	3601.1987	167.890	21.450	0.000	3271.522	3930.876
C(continent)[T.Europe]	1046.7444	233.489	4.483	0.000	588.252	1505.236

Omnibus:	53.689	**Durbin-Watson:**	0.334
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	60.383
Skew:	0.719	**Prob(JB):**	7.73e-14
Kurtosis:	2.579	**Cond. No.**	2.66

Figure 51. Model summary for One-way ANOVA between 'daily_vaccinations_per_million' and 'continent'

Null Hypothesis, H0: true mean of daily vaccinations are equal across continents
H0: mu_europe = mu_asia , mu_i is the true mean daily vaccinations for group i.

Alternate Hypothesis, H1: at least one means of daily vaccinations is different across the continents.

p-value = 0.000009 < 0.05, so we reject the Null Hypothesis and hence there is a relationship between daily vaccinations across the continents.(ie..at least one mean of daily vaccinations is different)

B. TWO-WAY ANOVA

- *daily_vaccinations_per_million, country* and *year*

Assumptions for Two-Way ANOVA:

1. Each of the populations is Normally distributed with the same variance (homogeneity of variance)
2. The observations are sampled independently, the groups under consideration are independent.

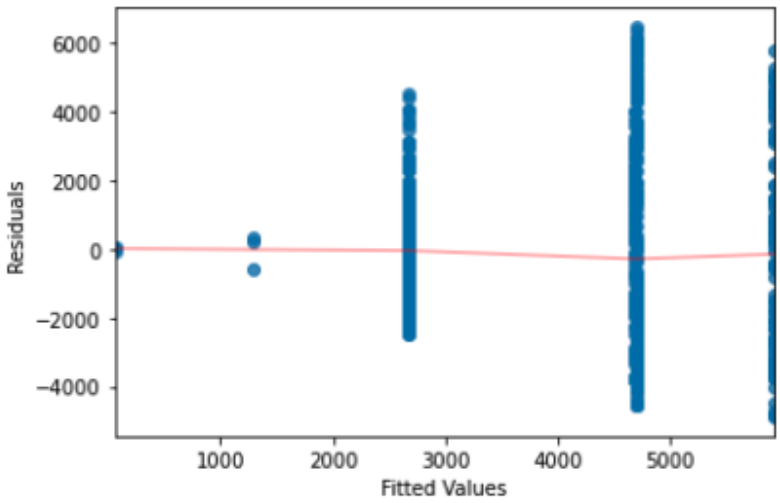


Figure 52. Fitted values vs Residuals plot for 'daily_vaccinations_per_million', 'country' and 'year'

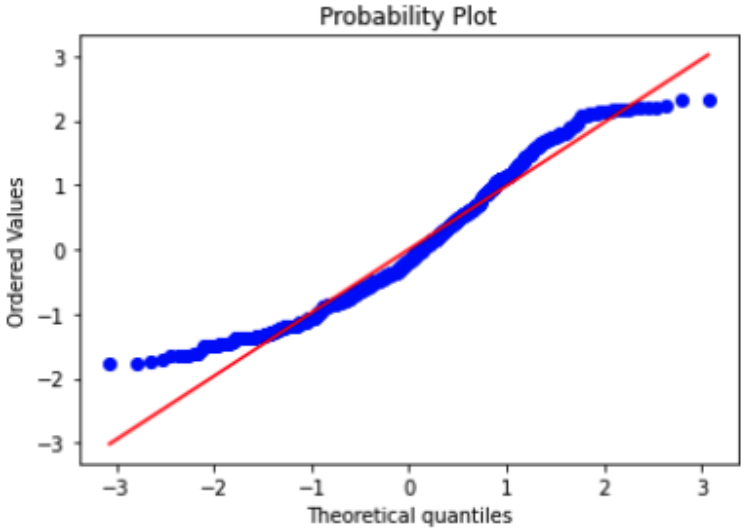


Figure 53. Probability plot for 'daily_vaccinations_per_million', 'country' and 'year'

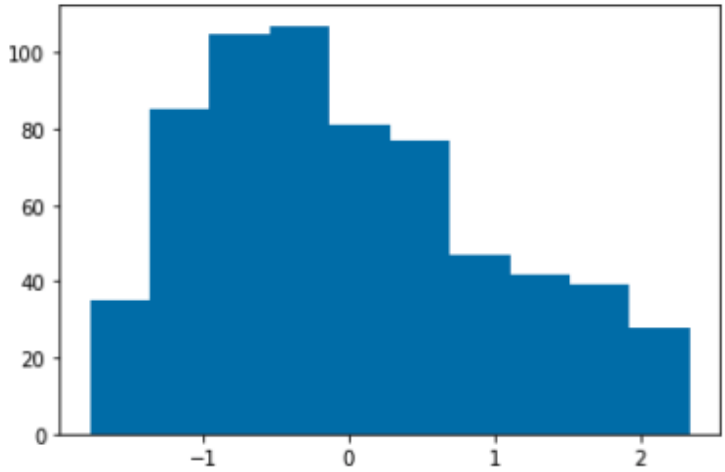


Figure 54. Histogram of residuals t for 'daily_vaccinations_per_million', 'country' and 'year'

The assumptions are roughly satisfied.There is roughly equal spread and variance and there is no strong deviation from normality.

Model Summary

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	R-squared:	0.159
Model:	OLS	Adj. R-squared:	0.153
Method:	Least Squares	F-statistic:	30.19
Date:	Sat, 11 Dec 2021	Prob (F-statistic):	4.91e-23
Time:	10:31:06	Log-Likelihood:	-6034.5
No. Observations:	646	AIC:	1.208e+04
Df Residuals:	641	BIC:	1.210e+04
Df Model:	4		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1293.9207	1143.189	1.132	0.258	-950.926	3538.768
C(country)[T.India]	-3239.6445	344.687	-9.399	0.000	-3916.496	-2562.793
C(country)[T.Ireland]	-1224.1747	340.964	-3.590	0.000	-1893.717	-554.632
C(country)[T.United Kingdom]	-1243.4178	396.888	-3.133	0.002	-2022.776	-464.059
C(year)[T.2021]	4625.7375	1143.429	4.045	0.000	2380.418	6871.057

Omnibus:	50.510	Durbin-Watson:	0.348
Prob(Omnibus):	0.000	Jarque-Bera (JB):	36.129
Skew:	0.471	Prob(JB):	1.43e-08
Kurtosis:	2.325	Cond. No.	22.1

Figure 55. Model summary for Two-way ANOVA between 'daily_vaccinations_per_million' ,'country' and 'year'

```
sm.stats.anova_lm(twoway_model1, typ=2)
```

	sum_sq	df	F	PR(>F)
C(country)	8.532785e+08	3.0	37.097954	4.079026e-22
C(year)	1.254767e+08	1.0	16.366033	5.857979e-05
Residual	4.914480e+09	641.0	NaN	NaN

Figure 56. Two-way ANOVA between 'daily_vaccinations_per_million' ,'country' and 'year'

First Hypothesis:
Null Hypothesis, H0: true daily vaccination means are equal across the countries.
Alternate Hypothesis, H1: not all the means are equal across the countries.

Second Hypothesis:
Null Hypothesis, H0: true daily vaccination means are equal across the year 2020 and 2021.
Alternate Hypothesis, H1:not all the means are equal across the year 2020 and 2021.

Both the *p-values* are less than 0.05. So we reject both the first and second null hypothesis.Hence not all the daily vaccination means are equal across countries and year.

year	country	count	mean	std	min	25%	50%	75%	max
2020	Bahrain	3.0	1276.333333	502.346826	701.0	1100.50	1500.0	1564.00	1628.0
	Ireland	3.0	87.333333	67.899435	11.0	60.50	110.0	125.50	141.0
2021	Bahrain	90.0	5920.244444	3187.917665	1035.0	3135.25	5695.0	9094.75	11723.0
	India	219.0	2680.013699	1679.367496	222.0	1426.00	2322.0	3795.50	7204.0
	Ireland	227.0	4695.251101	3552.422479	130.0	1600.50	3141.0	8139.00	11161.0
	United Kingdom	104.0	4676.240385	2295.204217	852.0	2934.00	5070.0	6513.25	8700.0

Figure 57. Mean value analysis based on Groupby function for ,'country' and 'year'

From the above table I assume that an interaction term might be needed in the model. Because the means are different across the countries and year.
So I decided to run Two-Way ANOVA with interaction.

Assumptions for Two-Way ANOVA:

1. Each of the populations is Normally distributed with the same variance (homogeneity of variance)
2. The observations are sampled independently, the groups under consideration are independent.

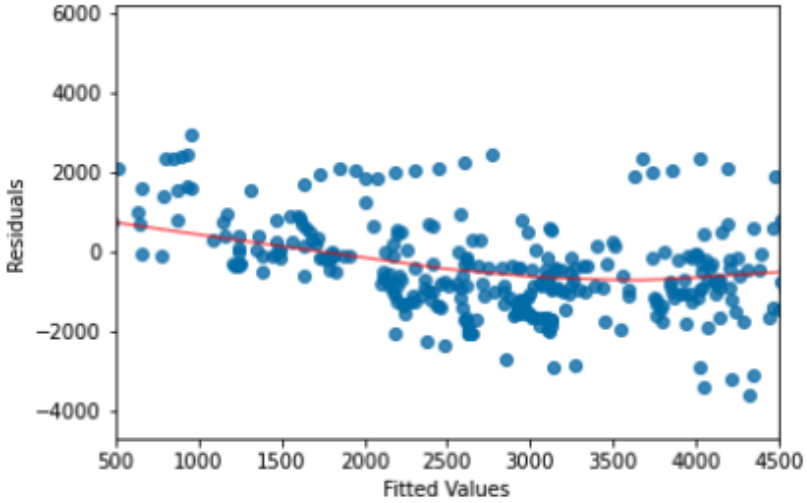


Figure 58. Fitted values vs Residuals plot for 'daily_vaccinations_per_million', 'country' and 'year' with interaction

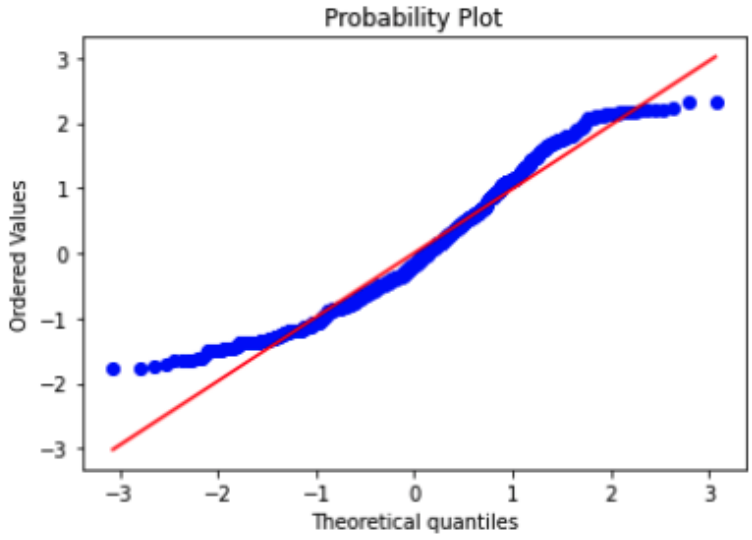


Figure 59. Probability plot for 'daily_vaccinations_per_million', 'country' and 'year' with interaction

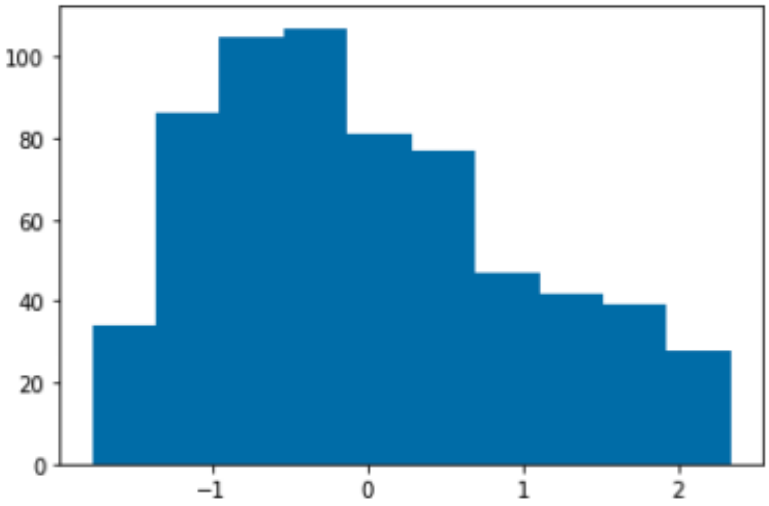


Figure 60. Histogram of residuals for 'daily_vaccinations_per_million', 'country' and 'year' with interaction

The assumptions are roughly satisfied.
There is roughly equal spread and variance and there is no strong deviation from normality.

Model Summary

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	**R-squared:**	0.159
Model:	OLS	**Adj. R-squared:**	0.152
Method:	Least Squares	**F-statistic:**	24.11
Date:	Sat, 11 Dec 2021	**Prob (F-statistic):**	2.89e-22
Time:	16:32:13	**Log-Likelihood:**	-6034.5
No. Observations:	646	**AIC:**	1.208e+04
Df Residuals:	640	**BIC:**	1.211e+04
Df Model:	5		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1276.3333	1599.883	0.798	0.425	-1865.320	4417.987
C(country)[T.India]	-1620.1154	173.482	-9.339	0.000	-1960.778	-1279.452
C(country)[T.Ireland]	-1189.0000	2262.576	-0.526	0.599	-5631.969	3253.969
C(country)[T.United Kingdom]	-622.0020	199.472	-3.118	0.002	-1013.700	-230.304
C(year)[T.2021]	4643.9111	1626.329	2.855	0.004	1450.326	7837.496
C(country)[T.India]:C(year)[T.2021]	-1620.1154	173.482	-9.339	0.000	-1960.778	-1279.452
C(country)[T.Ireland]:C(year)[T.2021]	-35.9933	2288.754	-0.016	0.987	-4530.369	4458.382
C(country)[T.United Kingdom]:C(year)[T.2021]	-622.0020	199.472	-3.118	0.002	-1013.700	-230.304

Omnibus:	50.500	**Durbin-Watson:**	0.348
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	36.128
Skew:	0.471	**Prob(JB):**	1.43e-08
Kurtosis:	2.325	**Cond. No.**	6.18e+16

Figure 61. Model summary for Two-way ANOVA between 'daily_vaccinations_per_million' and 'country' with interaction

```
sm.stats.anova_lm(twoway_model3, typ=2)
```

	sum_sq	df	F	PR(>F)
C(country)	9.128881e+07	3.0	3.962770	8.144215e-03
C(year)	7.728294e+07	1.0	10.064360	1.583995e-03
C(country):C(year)	7.618027e+08	3.0	33.069207	7.015527e-20
Residual	4.914478e+09	640.0	NaN	NaN

Figure 62. Two-way ANOVA between 'daily_vaccinations_per_million' and 'country' with interaction

First Hypothesis:
Null Hypothesis, H0: true daily vaccination means are equal across the countries.
Alternate Hypothesis, H1: not all the means are equal across the countries.

Second Hypothesis:
Null Hypothesis, H0: true daily vaccination means are equal across the year 2020 and 2021.
Alternate Hypothesis, H1:not all the means are equal across the year 2020 and 2021.

Third Hypothesis:
Null Hypothesis, H0: no interaction between country and year for daily vaccinations
Alternate Hypothesis, H1 : there is an interaction between country and year for daily vaccinations

First Hypothesis: p-value = 0.40 > 0.05, we fail to reject the null hypothesis. Hence there is not enough evidence to suggest there is a relationship between countries and daily vaccinations.

Second Hypothesis: p-value = 0.07 > 0.05, we fail to reject the null hypothesis. Hence there is not enough evidence to suggest there is a relationship between year and daily vaccinations.

Third Hypothesis: p-value= 0.0004 < 0.05 , we reject null hypothesis and hence there is an interaction between country , year and daily vaccinations.

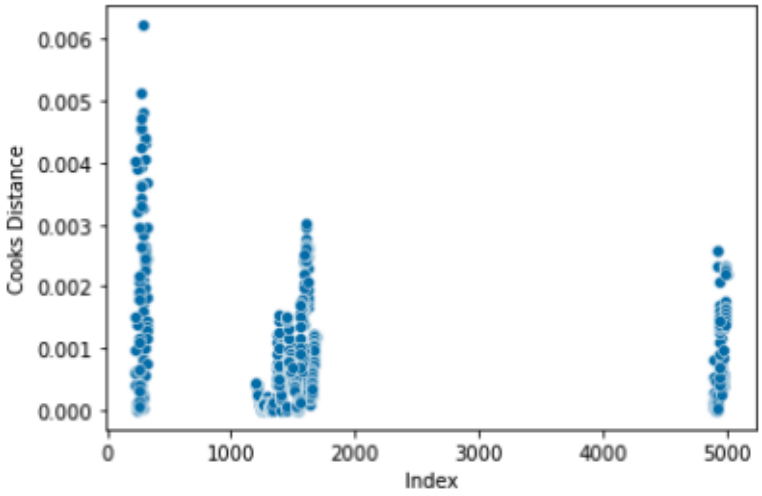


Figure 63. Cook's distance plot for 'daily_vaccinations_per_million' and 'country'

From the above figure , there is no large cook's distance. So no need to worry about the influential points.

- *daily_vaccinations_per_million, country* and *month*

Assumptions for Two-Way ANOVA:

1. Each of the populations is Normally distributed with the same variance (homogeneity of variance)
2. The observations should be sampled independently, the groups under consideration are independent.

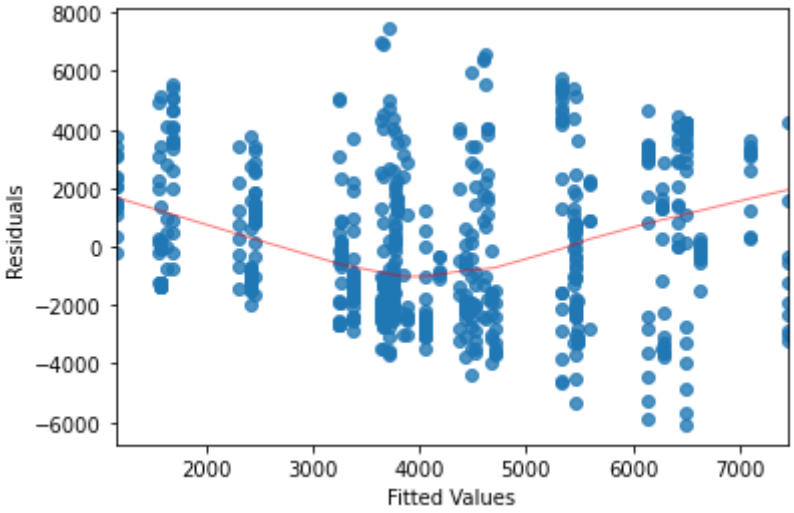


Figure 64. Fitted values vs Residuals plot for ‘daily_vaccinations_per_million’, ‘country’ and ‘month’

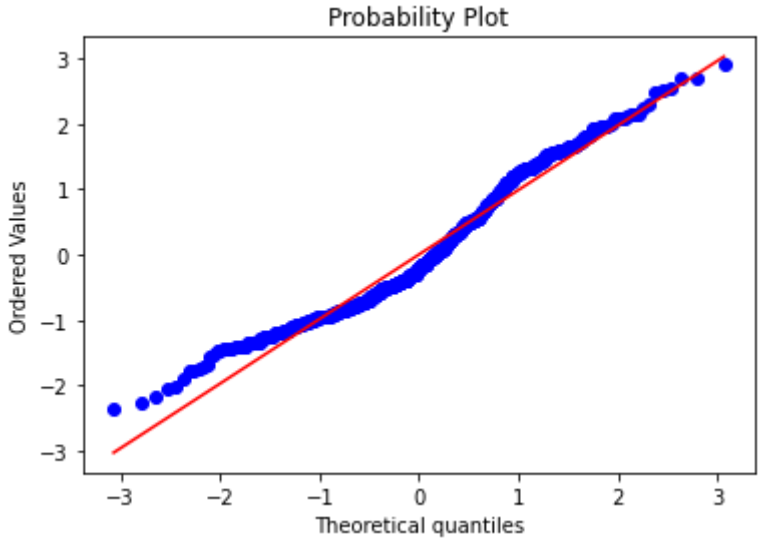


Figure 65. Probability plot for ‘daily_vaccinations_per_million’, ‘country’ and ‘month’

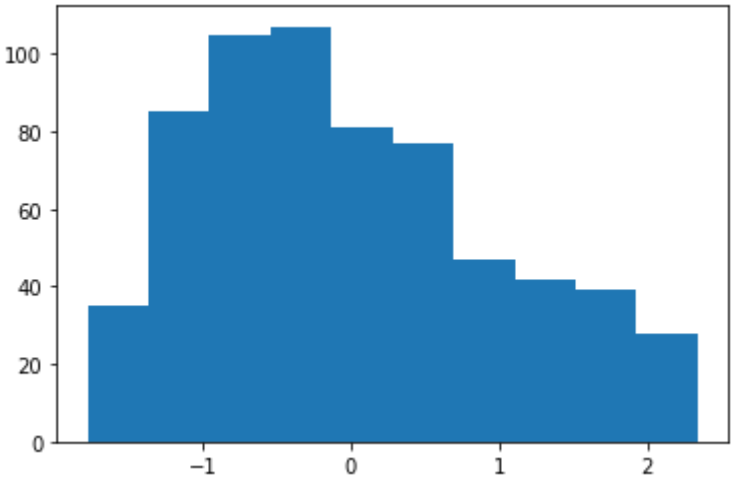


Figure 66.Histogram of residuals for ‘daily_vaccinations_per_million’, ‘country’ and ‘month’

The assumptions are roughly satisfied.There is roughly equal spread and variance and there is no strong deviation from normality.

Model Summary

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	R-squared:	0.262
Model:	OLS	Adj. R-squared:	0.245
Method:	Least Squares	F-statistic:	15.97
Date:	Sat, 11 Dec 2021	Prob (F-statistic):	1.35e-33
Time:	10:31:07	Log-Likelihood:	-5992.2
No. Observations:	646	AIC:	1.201e+04
Df Residuals:	631	BIC:	1.208e+04
Df Model:	14		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	6410.2986	398.555	16.084	0.000	5627.643	7192.954
C(country)[T.India]	-3026.3171	324.951	-9.313	0.000	-3664.434	-2388.201
C(country)[T.Ireland]	-954.0362	324.635	-2.939	0.003	-1591.532	-316.540
C(country)[T.United Kingdom]	-819.0530	377.328	-2.171	0.030	-1560.024	-78.082
C(month)[T.August]	-926.9263	500.148	-1.853	0.064	-1909.083	55.230
C(month)[T.December]	-1745.6310	593.519	-2.941	0.003	-2911.142	-580.120
C(month)[T.February]	-1798.1018	463.724	-3.878	0.000	-2708.730	-887.473
C(month)[T.January]	-1820.7308	480.079	-3.793	0.000	-2763.477	-877.984
C(month)[T.July]	-122.5347	475.128	-0.258	0.797	-1055.559	810.490
C(month)[T.June]	1036.8167	459.419	2.257	0.024	134.641	1938.992
C(month)[T.March]	-964.7341	462.745	-2.085	0.037	-1873.441	-56.028
C(month)[T.May]	675.6088	452.494	1.493	0.136	-212.967	1564.185
C(month)[T.November]	-1077.3399	647.443	-1.664	0.097	-2348.744	194.064
C(month)[T.October]	-2217.9610	502.134	-4.417	0.000	-3204.016	-1231.906
C(month)[T.September]	-1702.6149	464.872	-3.663	0.000	-2615.498	-789.731

Omnibus:	39.467	Durbin-Watson:	0.471
Prob(Omnibus):	0.000	Jarque-Bera (JB):	32.742
Skew:	0.470	Prob(JB):	7.77e-08
Kurtosis:	2.423	Cond. No.	13.5

Figure 67. Model summary for Two-way ANOVA between 'daily_vaccinations_per_million' ,'country' and 'month'

```
sm.stats.anova_lm(twoway_model2, typ=2)
```

	sum_sq	df	F	PR(>F)
C(country)	8.147105e+08	3.0	39.738960	1.564768e-23
C(month)	7.277964e+08	11.0	9.681701	2.689818e-16
Residual	4.312161e+09	631.0	NaN	NaN

Figure 68. Two-way ANOVA between 'daily_vaccinations_per_million' ,'country' and 'month'

First Hypothesis:
Null Hypothesis, H0: true daily vaccination means are equal across the countries.
Alternate Hypothesis, H1: not all the means are equal across the countries.

Second Hypothesis:
Null Hypothesis, H0: true daily vaccination means are equal across the months
Alternate Hypothesis, H1:not all the means are equal across the months.

Both the *p-values* are less than 0.05. So we reject both the first and second null hypothesis.Hence not all the daily vaccination means are equal across countries and months.

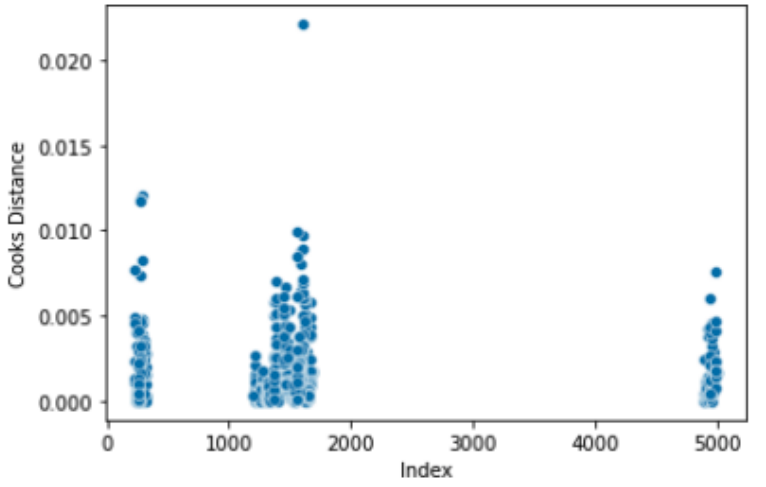


Figure 69. Cook's distance plot 1

From the above plot, there is no cook's distance greater than 0.5. So no need to worry about the influential points.

IX. LINEAR REGRESSION

In this section, a model will be constructed based on the response variable selected, and the type of response variable (i.e., categorical or numerical), and summary statistics generated in Python. The data from the sample is regarding the COVID-19 vaccination details of countries Bahrain, India, Ireland and UK.The response variable I chose is the 'daily_vaccinations_per_million' and the independent variables I am considering are 'year','country', 'month', 'continent','total_vaccinations_per_hundred','people_vaccinated_per_hundred','people_fully_vaccinated_per_hundred'.The count variables 'total_vaccinations','people_vaccinated', 'people_fully_vaccinated', and 'daily_vaccinations' are not considered because it may cause high collinearity between the variables.The other variables 'iso_code', 'vaccines', 'date', and 'source_name' are not used here. Since it does not come in the analysis scope.

Since my response variable is continuous numeric, I will be using Simple linear regression (SLR) and Multiple linear regression (MLR).For MLR, I have created dummy variables for all the categorical variables('year', 'month', 'country', 'continent'). Since total_vaccinations_per_hundred has high correlation it is used as a predictor in SLR.

A. SLR with 'daily_vaccinations_per_million' and 'total_vaccinations_per_hundred'

Assumptions for SLR:

1. A linear relationship between the independent variable and the dependent variables.
2. Residuals should be normally distributed with mean zero.
3. The residuals should have equal variance across the values of the independent variables.
4. i.i.d: Residuals are independently and identically distributed .

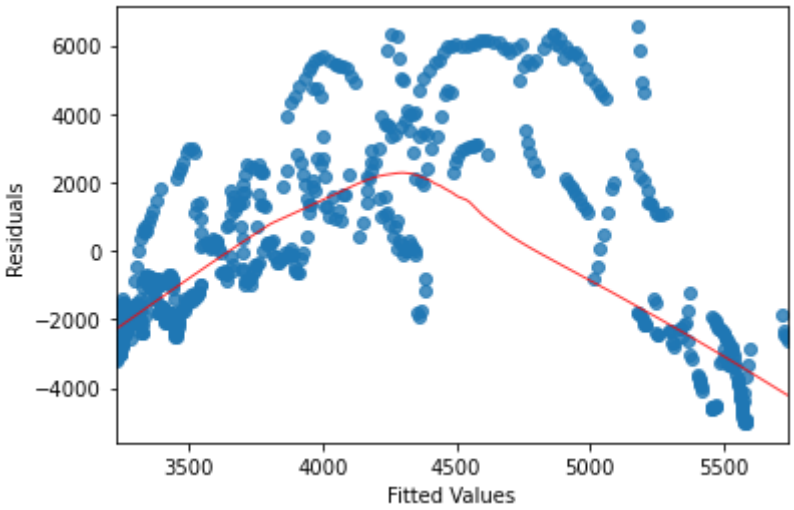


Figure 70. Fitted values vs Residuals plot for 'daily_vaccinations_per_million' and 'total_vaccinations_per_hundred'

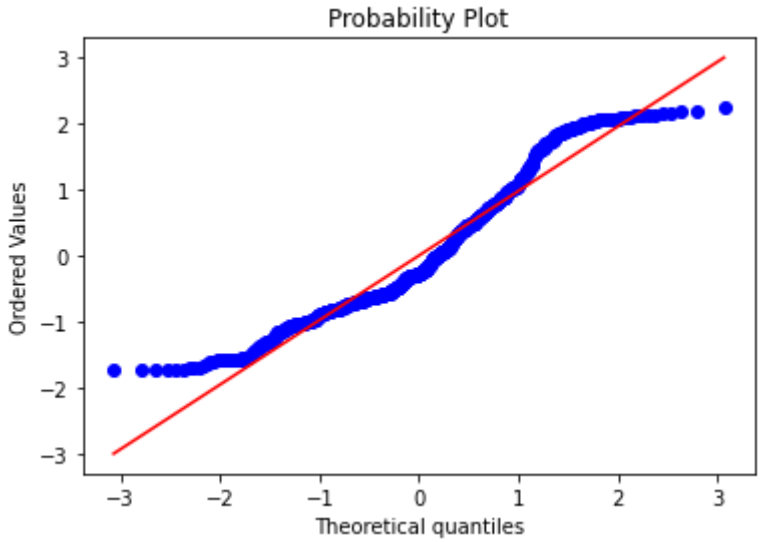


Figure 71. Probability plot for 'daily_vaccinations_per_million' and 'total_vaccinations_per_hundred'

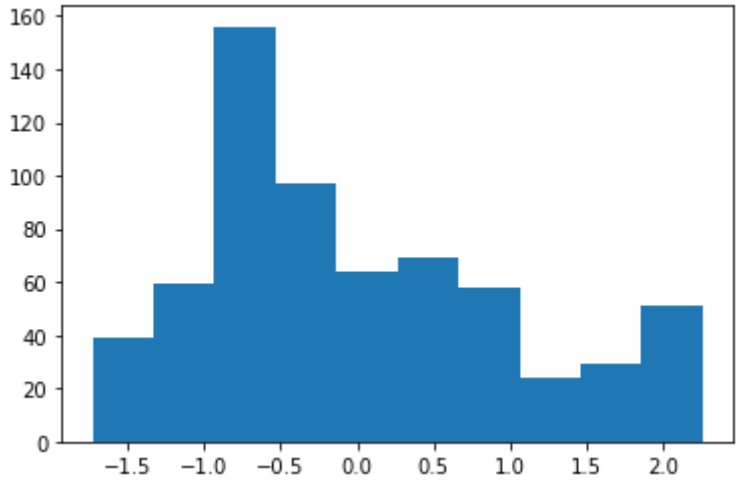


Figure 72. Histogram of residuals for 'daily_vaccinations_per_million' and 'total_vaccinations_per_hundred'

The assumptions are roughly satisfied. There is roughly equal variance and random scatter and there is no much stronger deviation from normality.

Model Summary

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	**R-squared:**	0.072
Model:	OLS	**Adj. R-squared:**	0.070
Method:	Least Squares	**F-statistic:**	49.70
Date:	Sat, 11 Dec 2021	**Prob (F-statistic):**	4.62e-12
Time:	10:31:10	**Log-Likelihood:**	-6066.2
No. Observations:	646	**AIC:**	1.214e+04
Df Residuals:	644	**BIC:**	1.215e+04
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	3227.5018	172.836	18.674	0.000	2888.111	3566.893
total_vaccinations_per_hundred	16.0831	2.281	7.050	0.000	11.604	20.563

Omnibus:	47.439	**Durbin-Watson:**	0.335
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	43.335
Skew:	0.568	**Prob(JB):**	3.89e-10
Kurtosis:	2.434	**Cond. No.**	115.

Figure 73. SLR Model summary for 'daily_vaccinations_per_million' and 'total_vaccinations_per_hundred'

```
np.sqrt(model_slr.scale)
```

```
2901.5472635993583
```

Figure 74. RSE for 'daily_vaccinations_per_million' and 'total_vaccinations_per_hundred'

The average distance of data points around the equation of line is 2901.5.

A. Full Model MLR

Assumptions for MLR:

1. A linear relationship between the dependent variable and the independent variables.
2. Residuals should be normally distributed with mean zero.
3. The variance of the residuals are similar across the values of the independent variables.
4. i.i.d: Residuals are independently and identically distributed.
5. There should be no collinearity.

I created dummy variables for the categorical data.

```
sel_data["year_21"]=pd.get_dummies(sel_data["year"])[2021]
```

Figure 75. Dummy variables created for year

```
sel_data["month_2"]=pd.get_dummies(sel_data["month"])["February"]
sel_data["month_3"]=pd.get_dummies(sel_data["month"])["March"]
sel_data["month_4"]=pd.get_dummies(sel_data["month"])["April"]
sel_data["month_5"]=pd.get_dummies(sel_data["month"])["May"]
sel_data["month_6"]=pd.get_dummies(sel_data["month"])["June"]
sel_data["month_7"]=pd.get_dummies(sel_data["month"])["July"]
sel_data["month_8"]=pd.get_dummies(sel_data["month"])["August"]
sel_data["month_9"]=pd.get_dummies(sel_data["month"])["September"]
sel_data["month_10"]=pd.get_dummies(sel_data["month"])["October"]
sel_data["month_11"]=pd.get_dummies(sel_data["month"])["November"]
sel_data["month_12"]=pd.get_dummies(sel_data["month"])["December"]
```

Figure 76. Dummy variables created for month

```
sel_data["country_ind"]=pd.get_dummies(sel_data["country"])["India"]
sel_data["country_irl"]=pd.get_dummies(sel_data["country"])["Ireland"]
sel_data["country_bah"]=pd.get_dummies(sel_data["country"])["Bahrain"]
```

Figure 77. Dummy variables created for countries

```
sel_data["conti_eur"]=pd.get_dummies(sel_data["continent"])["Europe"]
```

Figure 78. Dummy variables created for continent

- *Full model MLR*
- *'daily vaccinations per million','year','country','month',"continent','total_vaccinations_per_hundred','people_vaccinated_per_hundred','people_fully_vaccinated_per_hundred'*

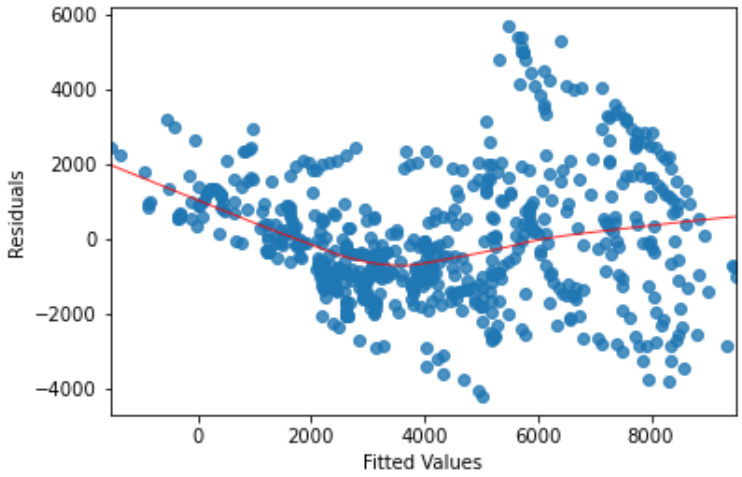


Figure 79. Residual and Fitted values plot for full model MLR

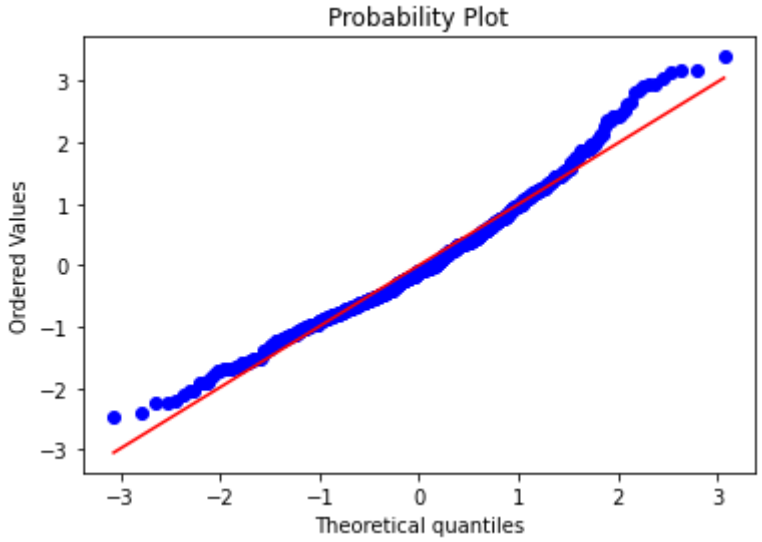


Figure 80. Probability plot for full model MLR

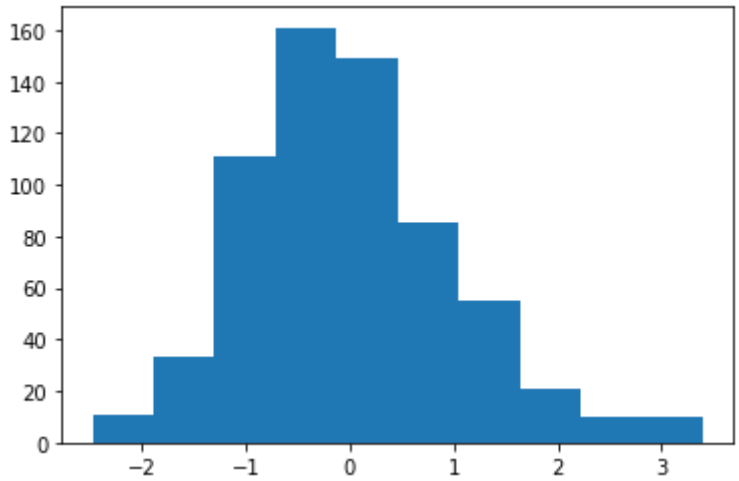


Figure 81. Histogram of residuals for ull model MLR

The assumptions are roughly satisfied. There is roughly equal variance and random scatter and there is no much stronger deviation from normality.The i.i.d assumption is also met.

Model Summary

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	R-squared:	0.684
Model:	OLS	Adj. R-squared:	0.675
Method:	Least Squares	F-statistic:	75.34
Date:	Sat, 11 Dec 2021	Prob (F-statistic):	2.08e-143
Time:	10:31:13	Log-Likelihood:	-5718.3
No. Observations:	646	AIC:	1.147e+04
Df Residuals:	627	BIC:	1.156e+04
Df Model:	18		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-968.4083	627.247	-1.544	0.123	-2200.168	263.351
people_vaccinated_per_hundred	374.6250	24.056	15.573	0.000	327.385	421.864
total_vaccinations_per_hundred	-147.5449	23.447	-6.293	0.000	-193.589	-101.500
people_fully_vaccinated_per_hundred	-68.8774	25.204	-2.733	0.006	-118.372	-19.383
year_21	3104.4078	817.007	3.800	0.000	1500.006	4708.809
month_2	-305.8513	316.524	-0.966	0.334	-927.427	315.725
month_3	-112.7019	321.460	-0.351	0.726	-743.970	518.567
month_4	359.9958	321.145	1.121	0.263	-270.654	990.646
month_5	626.7066	320.668	1.954	0.051	-3.007	1256.420
month_6	725.3279	330.047	2.198	0.028	77.197	1373.459
month_7	26.3157	343.437	0.077	0.939	-648.111	700.743
month_8	-1027.0043	362.896	-2.830	0.005	-1739.644	-314.365
month_9	-1447.2262	351.642	-4.116	0.000	-2137.764	-756.688
month_10	-1626.7927	370.579	-4.390	0.000	-2354.519	-899.067
month_11	-605.6963	439.439	-1.378	0.169	-1468.646	257.254
month_12	-232.8088	440.184	-0.529	0.597	-1097.223	631.605
country_ind	-2310.5079	269.354	-8.578	0.000	-2839.452	-1781.564
country_irl	1674.6371	222.018	7.543	0.000	1238.649	2110.625
country_bah	2678.5949	258.915	10.345	0.000	2170.150	3187.040
conti_eur	-1336.4953	260.943	-5.122	0.000	-1848.923	-824.068

Omnibus:	38.218	Durbin-Watson:	0.678
Prob(Omnibus):	0.000	Jarque-Bera (JB):	44.116
Skew:	0.585	Prob(JB):	2.63e-10
Kurtosis:	3.519	Cond. No.	1.34e+17

Figure 82. Model summary for MLR with all variables of interest.

```
np.sqrt(model_full_mlr.scale)
```

```
1716.069677663897
```

Figure 83. RSE for MLR with all variables of interest.

The equation of line, y = -968.40 + 374.62 people_vaccinated_per_hundred - 147.54 total_vaccinations_per_hundred - 68.87 people_fully_vaccinated_per_hundred + 3104.4 year_21 -305.85 month_2 -112.70 month_3 + 359.99 month_4 + 626.7 month_5 + 725.32 month_6 + 26.31 month_7 -1027.00 month_8 -1447.22 month_9 -1447.22 month_10 -605.69 month_11 -605.69 month_12 -2310.50 country_ind + 1674.63 country_irl + 2678.59 country_bah - 1336.46 conti_eur.

-968.40 is the estimated intercept, that is average y when all x are 0.

For every count increase in people_vaccinated_per_hundred, the average increase in daily_vaccinations_per_million will be 374.62 counts, keeping all other variables the same.

For every count increase in total_vaccinations_per_hundred, the average increase in daily_vaccinations_per_million will be -147.54 counts, keeping all other variables the same.

For every count increase in people_fully_vaccinated_per_hundred, the average increase in daily_vaccinations_per_million will be - 68.87 counts, keeping all other variables the same.

For every count increase in year_21, the average increase in daily_vaccinations_per_million will be 3104.4 counts, keeping all other variables the same.

For every count increase in vaccinations in month_2, the average increase in daily_vaccinations_per_million will be -305.85 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_3, the average increase in daily_vaccinations_per_million will be 359.99 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_4, the average increase in daily_vaccinations_per_million will be -305.85 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_5, the average increase in daily_vaccinations_per_million will be 626.7 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_6, the average increase in daily_vaccinations_per_million will be 725.32 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_7, the average increase in daily_vaccinations_per_million will be 26.31 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_8, the average increase in daily_vaccinations_per_million will be -1027.00 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_9, the average increase in daily_vaccinations_per_million will be -1447.22 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_10, the average increase in daily_vaccinations_per_million will be -1447.22 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_11, the average increase in daily_vaccinations_per_million will be -605.69 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_12, the average increase in daily_vaccinations_per_million will be -605.69 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_ind, the average increase in daily_vaccinations_per_million will be -305.85 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_irl, the average increase in daily_vaccinations_per_million will be 1674.63 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_bah, the average increase in daily_vaccinations_per_million will be 2678.59 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in conti_eur, the average increase in daily_vaccinations_per_million will be -1336.46 vaccination counts, keeping all other variables the same.

68% variation in daily vaccinations is being explained by this model.

1716.06 is RSE, the average spread of residuals around the equation.

There are insignificant variables present in the model. I should remove those variables. Also, I need to check the collinearity using variance inflation factor(VIF).

```
const                                0.000000
people_vaccinated_per_hundred       82.112671
total_vaccinations_per_hundred      302.013712
people_fully_vaccinated_per_hundred  87.825520
year_21                              1.347351
month_2                              1.934285
month_3                              1.995081
month_4                              2.047339
month_5                              2.151961
month_6                              2.162413
month_7                              2.081532
month_8                              1.987035
month_9                              2.353440
month_10                             2.032280
month_11                             1.393484
month_12                             1.762435
country_ind                               inf
country_irl                          2.479098
country_bah                               inf
conti_eur                                 inf
dtype: float64
```

Figure 84. VIF for MLR with all variables of interest.

There is high collinearity for the total_vaccinations_per_hundred variable. So I removed that variable along with the variables that have *p-value > 0.05*. Here 7 variables have *p-value > 0.05* they are month_2, month_3, month_4, month_5, month_7, month_11 and month_12.. So I removed them.

- *First reduced MLR model*
  Performing MLR with the significant variables from the full MLR model,

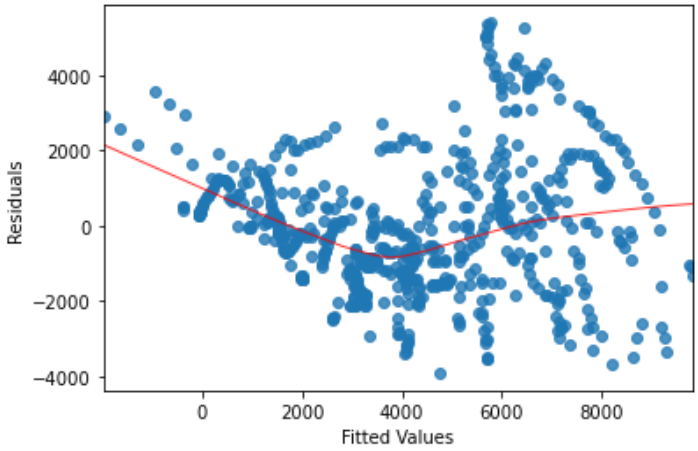


Figure 85. Residuals andFitted values plot for the first reduced MLR.

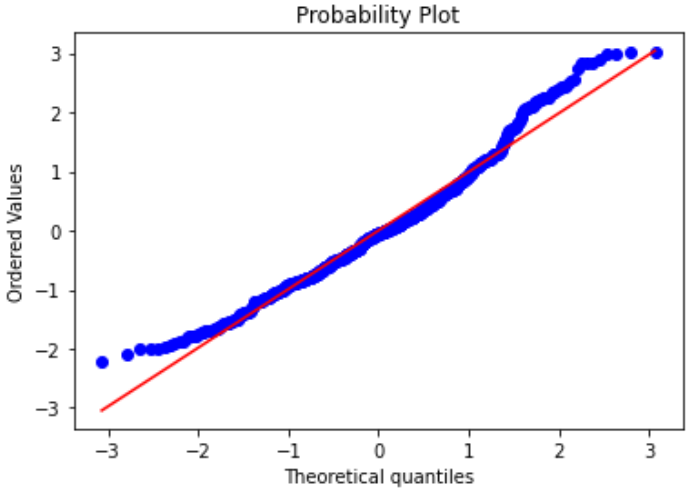


Figure 86. Probability plot for the first reduced MLR.

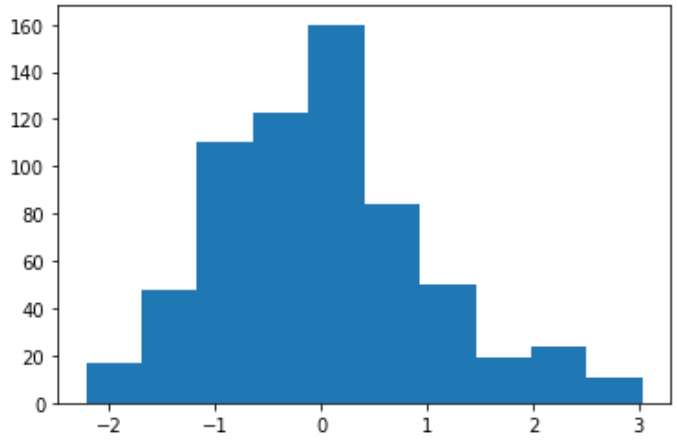


Figure 87. Histogram of residuals for the first reduced MLR.

The assumptions are roughly satisfied. There is roughly equal variance and random scatter and there is no much stronger deviation from normality.The i.i.d assumption is also met.

Model Summary

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	**R-squared:**	0.654
Model:	OLS	**Adj. R-squared:**	0.648
Method:	Least Squares	**F-statistic:**	119.8
Date:	Sat, 11 Dec 2021	**Prob (F-statistic):**	5.19e-139
Time:	10:31:15	**Log-Likelihood:**	-5747.8
No. Observations:	646	**AIC:**	1.152e+04
Df Residuals:	635	**BIC:**	1.157e+04
Df Model:	10		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-1160.6356	553.216	-2.098	0.036	-2246.989	-74.282
people_vaccinated_per_hundred	237.8681	8.455	28.134	0.000	221.265	254.471
people_fully_vaccinated_per_hundred	-222.6971	8.816	-25.261	0.000	-240.009	-205.386
year_21	2988.3757	749.171	3.989	0.000	1517.223	4459.529
month_6	669.9570	242.607	2.761	0.006	193.547	1146.367
month_8	-1352.1133	280.043	-4.828	0.000	-1902.036	-802.191
month_9	-1729.8410	261.092	-6.625	0.000	-2242.549	-1217.133
month_10	-2039.0891	288.155	-7.076	0.000	-2604.941	-1473.238
country_ind	-2035.7225	236.613	-8.604	0.000	-2500.360	-1571.085
country_irl	2024.8861	224.116	9.035	0.000	1584.787	2464.985
country_bah	2130.4219	231.057	9.220	0.000	1676.694	2584.150
conti_eur	-1255.3350	243.565	-5.154	0.000	-1733.625	-777.045

Omnibus:	35.237	**Durbin-Watson:**	0.697
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	39.557
Skew:	0.582	**Prob(JB):**	2.57e-09
Kurtosis:	3.340	**Cond. No.**	7.84e+16

Figure 88. Model summary for first reduced MLR model

```
np.sqrt(model_full_mlr1.scale)
```

```
1784.8850186108089
```

Figure 89. RSE for first reduced MLR model

The equation of line, y = -1160.63 + 237.86 people_vaccinated_per_hundred - 222.69 people_fully_vaccinated_per_hundred + 2988.37 year_21 + 669.95 month_6 -1352.11 month_8 -1729.84 month_9 -2039.08 month_10 -2035.72 country_ind + 2024.88 country_irl + 2130.42 country_bah -1255.33 conti_eur

-1160.63 is the estimated intercept, that is average y when all x are 0.

For every count increase in people_vaccinated_per_hundred, the average increase in daily_vaccinations_per_million will be 237.86 counts, keeping all other variables the same.

For every count increase in people_fully_vaccinated_per_hundred, the average increase in daily_vaccinations_per_million will be - 222.69 , keeping all other variables the same.

For every count increase in year_21, the average increase in daily_vaccinations_per_million will be 2988.37 counts, keeping all other variables the same.

For every count increase in vaccinations in month_6, the average increase in daily_vaccinations_per_million will be 669.95 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_8, the average increase in daily_vaccinations_per_million will be -1352.11 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_9, the average increase in daily_vaccinations_per_million will be -1729.84 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_10, the average increase in daily_vaccinations_per_million will be -2039.08 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_ind, the average increase in daily_vaccinations_per_million will be -2035.72 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_irl, the average increase in daily_vaccinations_per_million will be 2024.88 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_bah, the average increase in daily_vaccinations_per_million will be 2130.42 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in conti_eur, the average increase in daily_vaccinations_per_million will be -1255.33 vaccination counts, keeping all other variables the same.

65% variation in daily vaccinations is being explained by this model.

-2035.72 is RSE, the average spread of residuals around the equation.

There are insignificant variables present in the model. I should remove those. I need to check the collinearity using variance inflation factor(VIF).

```
const                                 0.000000
people_vaccinated_per_hundred         9.376282
people_fully_vaccinated_per_hundred   9.932258
year_21                               1.047226
month_6                               1.080051
month_8                               1.093803
month_9                               1.199331
month_10                              1.135856
country_ind                                inf
country_irl                           2.335154
country_bah                                inf
conti_eur                                  inf
dtype: float64
```

Figure 90. VIF for first reduced MLR model

Since 2 variables show higher multicollinearity,I removed people_fully_vaccinated_per_hundred with high collinearity.

- *Second Reduced MLR Model*

Performing MLR with significant variables from the previous model,

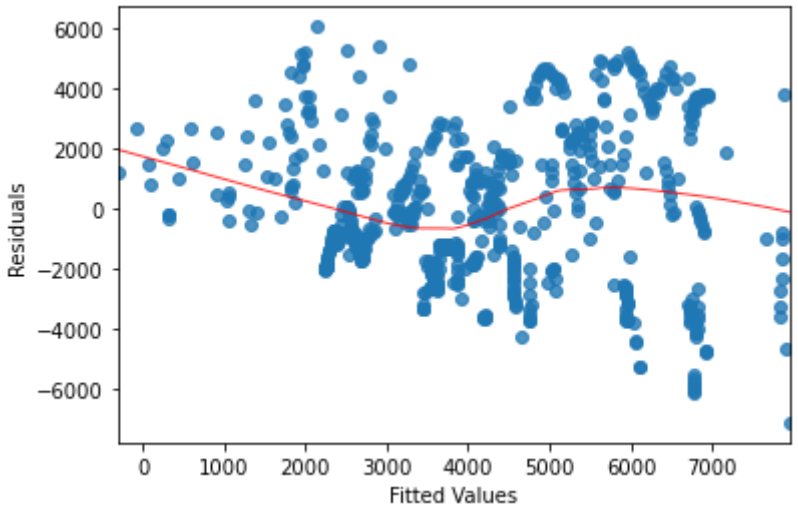


Figure 91. Residuals and Fitted values plot for second reduced MLR model

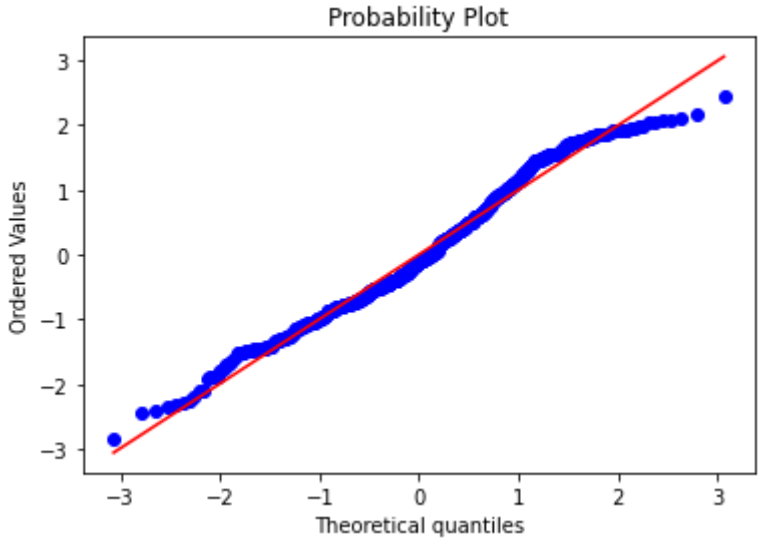


Figure 92. Probability plot for second reduced MLR model

The assumptions are roughly satisfied. There is roughly equal variance and random scatter and there is no much stronger deviation from normality.The i.i.d assumption is also met.

Model Summary

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	R-squared:	0.306
Model:	OLS	Adj. R-squared:	0.296
Method:	Least Squares	F-statistic:	31.09
Date:	Sat, 11 Dec 2021	Prob (F-statistic):	3.64e-45
Time:	10:31:17	Log-Likelihood:	-5972.4
No. Observations:	646	AIC:	1.196e+04
Df Residuals:	636	BIC:	1.201e+04
Df Model:	9		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-51.2541	780.246	-0.066	0.948	-1583.424	1480.916
people_vaccinated_per_hundred	43.7982	4.995	8.768	0.000	33.990	53.607
year_21	3135.7701	1059.932	2.958	0.003	1054.381	5217.160
month_6	1184.2395	342.042	3.462	0.001	512.572	1855.906
month_8	-948.7070	395.574	-2.398	0.017	-1725.496	-171.918
month_9	-2196.7951	368.479	-5.962	0.000	-2920.377	-1473.213
month_10	-2584.3614	406.550	-6.357	0.000	-3382.704	-1786.019
country_ind	-869.8043	328.340	-2.649	0.008	-1514.567	-225.042
country_irl	503.0906	305.421	1.647	0.100	-96.665	1102.846
country_bah	956.8360	320.234	2.988	0.003	327.992	1585.680
conti_eur	-138.2858	338.881	-0.408	0.683	-803.746	527.175

Omnibus:	21.962	Durbin-Watson:	0.451
Prob(Omnibus):	0.000	Jarque-Bera (JB):	13.042
Skew:	0.189	Prob(JB):	0.00147
Kurtosis:	2.415	Cond. No.	2.23e+17

Figure 93. Model summary for second reduced MLR model

The equation of line, y = -51.25 + 43.79 people_vaccinated_per_hundred + 3135.77 year_21 + 1184.23 month_6 -948.70 month_8 -2196.79 month_9 -2584.36 month_10 -869.80 country_ind + 503.09 country_irl + 956.83 country_bah -138.28 conti_eur.

-51.25 is the estimated intercept, that is average y when all x are 0.

For every count increase in people_vaccinated_per_hundred, the average increase in daily_vaccinations_per_million will be 43.79 counts, keeping all other variables the same.

For every count increase in year_21, the average increase in daily_vaccinations_per_million will be 3135.77 counts, keeping all other variables the same.

For every count increase in vaccinations in month_6, the average increase in daily_vaccinations_per_million will be 1184.23 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_8, the average increase in daily_vaccinations_per_million will be -948.70 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_9, the average increase in daily_vaccinations_per_million will be -2196.79 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_10, the average increase in daily_vaccinations_per_million will be 2584.36 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_ind, the average increase in daily_vaccinations_per_million will be

-869.80 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_irl, the average increase in daily_vaccinations_per_million will be 503.09 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_bah, the average increase in daily_vaccinations_per_million will be 956.83 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in conti_eur, the average increase in daily_vaccinations_per_million will be -138.28 vaccination counts, keeping all other variables the same.

30% variation in daily vaccinations is being explained by this model.

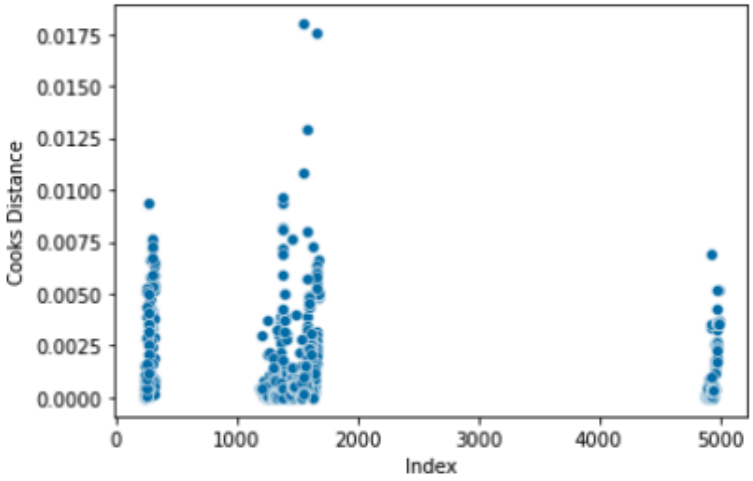


Figure 94. Cook's distance plot for second reduced MLR model.

There is no cook's distance greater than 0.5.So no need to worry about influential data points.

There are two variables with *p-value > 0.5*. They are country_irl and conti_eur. So I removed it. Performing the MLR with the significant variables.

- *Third Reduced MLR Model*

Performing the model with the significant variables from the previous model,

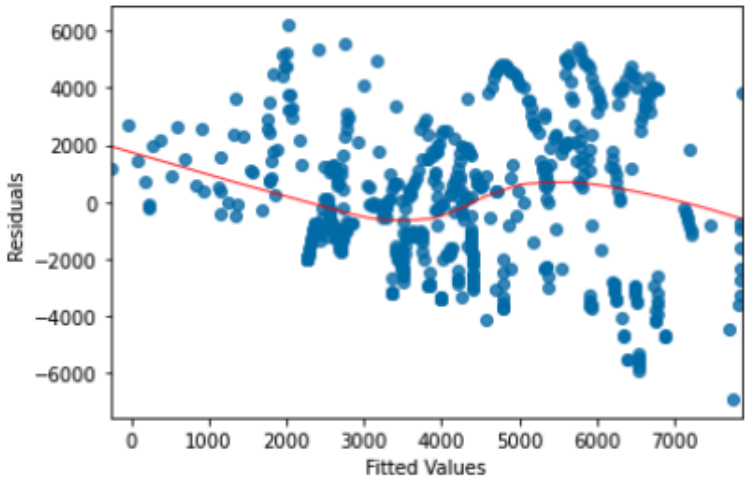


Figure 95. Residuals and fitted values plot for third reduced MLR model.

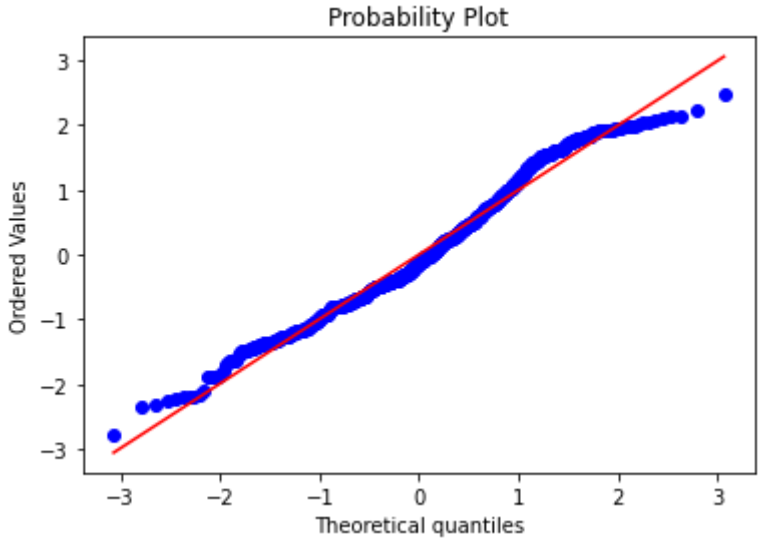


Figure 96. Probability plot for third reduced MLR model.

The assumptions are roughly satisfied. There is roughly equal variance and random scatter and there is no much stronger deviation from normality.The i.i.d assumption is also met.

Model Summary

Dep. Variable:	daily_vaccinations_per_million	R-squared:	0.303
Model:	OLS	Adj. R-squared:	0.294
Method:	Least Squares	F-statistic:	34.54
Date:	Sat, 11 Dec 2021	Prob (F-statistic):	2.24e-45
Time:	10:31:21	Log-Likelihood:	-5973.8
No. Observations:	646	AIC:	1.197e+04
Df Residuals:	637	BIC:	1.201e+04
Df Model:	8		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	229.5946	1043.210	0.220	0.826	-1818.952	2278.142
people_vaccinated_per_hundred	42.1816	4.904	8.601	0.000	32.551	51.812
year_21	3115.4994	1061.285	2.936	0.003	1031.459	5199.540
month_6	1212.7513	342.063	3.545	0.000	541.044	1884.458
month_8	-951.3802	396.102	-2.402	0.017	-1729.204	-173.556
month_9	-2133.5381	366.965	-5.814	0.000	-2854.145	-1412.931
month_10	-2585.1734	406.929	-6.304	0.000	-3384.258	-1786.089
country_ind	-1108.2326	244.216	-4.538	0.000	-1587.799	-628.667
country_bah	765.2781	302.640	2.529	0.012	170.986	1359.570

Omnibus:	21.412	Durbin-Watson:	0.447
Prob(Omnibus):	0.000	Jarque-Bera (JB):	14.393
Skew:	0.240	Prob(JB):	0.000749
Kurtosis:	2.448	Cond. No.	631.

Figure 97. Model summary for third reduced MLR model.

```
np.sqrt(model_full_mlr3.scale)
```

```
2528.7360968806156
```

Figure 98. RSE for the third reduced MLR model.

The equation of line, y = 229.59 + 42.18 people_vaccinated_per_hundred + 3115.49 year_21 + 1212.75 month_6 -951.38 month_8 -2133.53 month_9 -2565.17 month_10 -1108.23 country_ind + 765.27 country_bah

229.59 is the estimated intercept, that is average y when all x are 0.

For every count increase in people_vaccinated_per_hundred, the average increase in daily_vaccinations_per_million will be 42.18counts, keeping all other variables the same.

For every count increase in year_21, the average increase in daily_vaccinations_per_million will be 115.49 counts, keeping all other variables the same.

For every count increase in vaccinations in month_6, the average increase in daily_vaccinations_per_million will be 1212.75 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_8, the average increase in daily_vaccinations_per_million will be -951.38 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_9, the average increase in daily_vaccinations_per_million will be -2133.53 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_10, the average increase in daily_vaccinations_per_million will be -2565.17 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_ind, the average increase in daily_vaccinations_per_million will be -1108.23 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_irl, the average increase in daily_vaccinations_per_million will be 503.09 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_bah, the average increase in daily_vaccinations_per_million will be 956.83 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in conti_eur, the average increase in daily_vaccinations_per_million will be -138.28 vaccination counts, keeping all other variables the same.

30% variation in daily vaccinations is being explained by this model. And, RSE is 2528.73,the average spread of residuals around the equation.

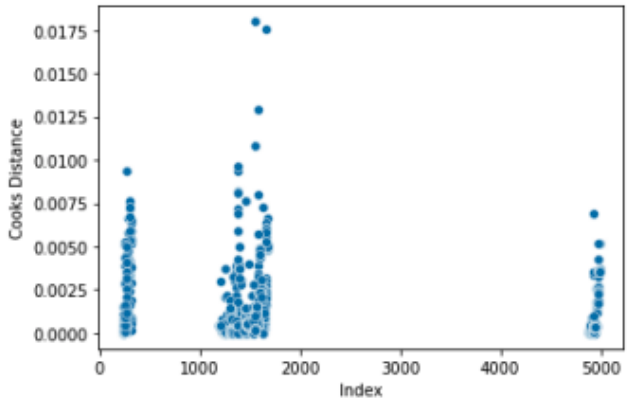


Figure 99. Cook's distance for the third reduced MLR model.

There are no points greater than 0.5.So no need to worry about the influential data points.
Checking collinearity with variance inflation factor,

```
const                            109.943415
people_vaccinated_per_hundred      1.571727
year_21                            1.047022
month_6                            1.069699
month_8                            1.090228
month_9                            1.180358
month_10                           1.128555
country_ind                        1.350147
country_bah                        1.140304
dtype: float64
```

Figure 100. VIF for the third reduced MLR model.

Variance inflation factor is less than 5 for all the variables. And from the summary function, there are no variables with *p-value > 0.5.* So all the variables are significant for the model.

To identify which among the performed MLR is preferred, I did ANOVA using the MLR models.

Null Hypothesis, H0 : Reduced MLR model is preferred.

Alternate Hypothesis, H1 : Reduced MLR model is not preferred.

```
anovaResults = anova_lm(model_full_mlr3, model_full_mlr)
anovaResults
```

	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)
0	637.0	4.073300e+09	0.0	NaN	NaN	NaN
1	627.0	1.846449e+09	10.0	2.226851e+09	75.617335	7.122837e-101

Figure 101. ANOVA between reduced MLR model and full MLR model.

p-value < 0.5(significance level, 5%).
So we will reject the null hypothesis. Hence the full model MLR is preferred.So it seems that all the variables are depending on the response variable 'daily_vaccinations_per_million'.

X. ANALYSIS USING BOKEH

In this section I will be analyzing the vaccination scheme using bokeh to analyze the dataset.I identified the top 5 countries in the vaccination scheme.

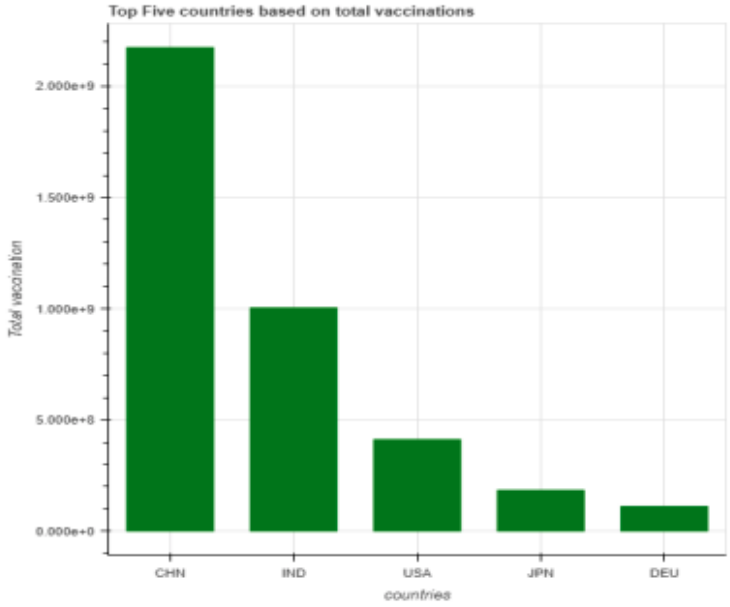


Figure 102. Bokeh visualization of top vaccinated countries.

From the visualization I identified the top 5 countries in total vaccination.They are China, India, USA, Japan and Denmark.

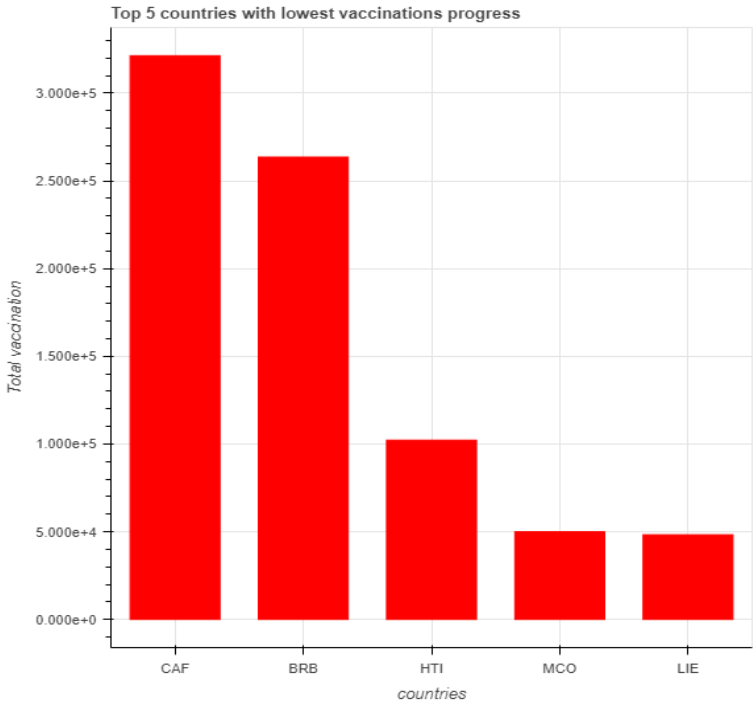


Figure 103. Bokeh visualization of least vaccinated countries.

The countries with the least number of total vaccinations are the Central African Republic, Barbados, Haiti, Monaco and Liechtenstein.The number of vaccinations and vaccinated people in each continent is calculated using the bokeh stacked bar charts as below.

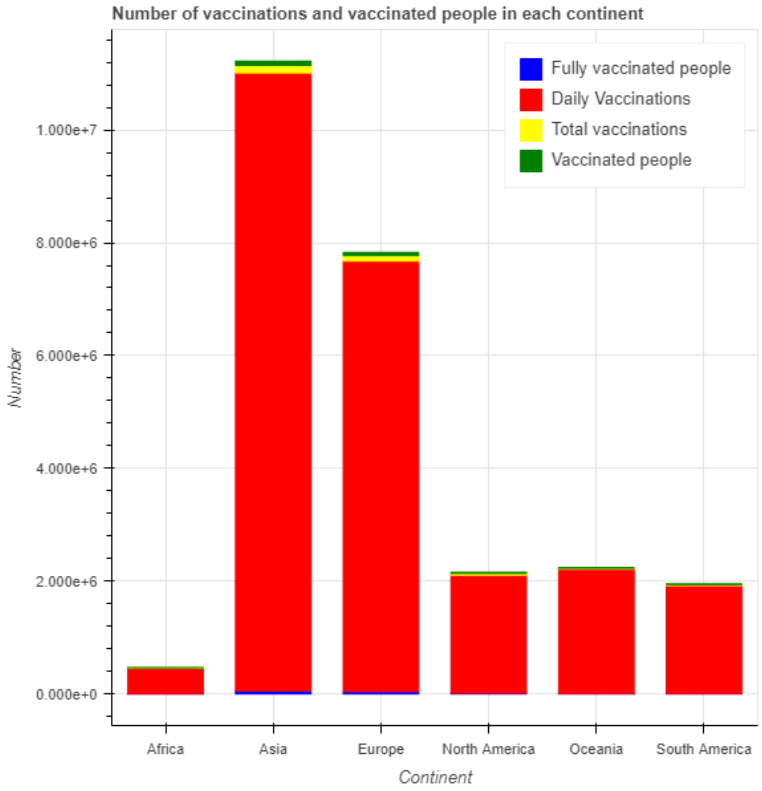


Figure 104. Bokeh visualization of vaccinations in each continent.

The difference in the number of daily vaccinations in each continent is identified from the bokeh visualizations.Asia has administered more number of daily vaccinations than other continents.Africa has the least number of daily vaccinations.

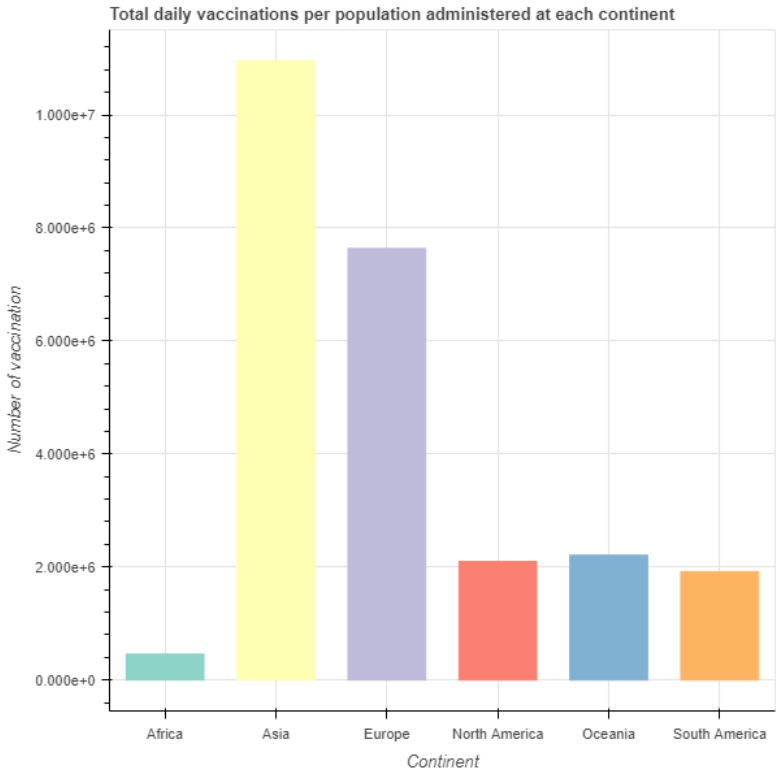


Figure 105. Bokeh visualization of daily vaccinations in each continent

XI. PROJECT DIARY

a. Data Collection

One of the major problems we faced was to gather the data needed for the analysis. An appropriate dataset was available in the Github.com and Kaggle.com. The data available in both of the sites was raw data which was better for the analysis. The datasets were available to download as csv files and the datasets contain all the relevant information required for the analysis. Since the Kaggle dataset contains the vaccine information along with the vaccination details, I decided to proceed with the Kaggle dataset.

I took 4 days to collect the dataset.

b. Quantity of the data

To solve the problems with the quantity of the dataset, I compared the datasets available in Kaggle and Github.com. It has the required number of both numerical and categorical variables. The available dataset is a population dataset so I considered the sample sets from them for better visualization.I also added additional variables for the analysis purpose.

No additional effort was used in this section. I was able to complete it along with data collection.

c. Data Management

Data cleaning and arranging was one of the main tasks during the analysis.For proper data management I used different Python libraries available.

Total of 4 days was utilized for this task.

d. Analysis , visualization and API access

Finally, when the dataset was ready, I needed a model which can be applied to get the appropriate results for the analysis. For better visualization, I used the Matplotlib, Seaborn and Bokeh libraries. In order to analyze the data and to predict the results, various statistical methods like Chi-squared test, Hypothesis test, ANOVA and Linear regression were used.

Total of 6 days was utilized for this task.

XII. CONCLUSION

The main objective of this project was to analyze the trends in the vaccination process from 2020- 2021 and to analyze which are the factors that influence the vaccinations from the available dataset.The dataset was not balanced and the information of all consecutive dates was not available in the dataset.Majority of the countries as per the dataset has not started vaccination in 2020 and it has made a unbalance in the total dataset. I chose the vaccination parameters over population for my analysis rather than the count parameters for better results.

I used ANOVA, hypothesis testing and linear regression models for the analysis.From the analysis done using Hypothesis testing, it seems that there is a relationship between the vaccinations given in the continents.

ANOVA tests conclude that there exists a relationship between the vaccinations and the country in which vaccination is given.That is vaccination progress depends on the country. Similarly, from the ANOVA it seems there is not enough evidence to prove the dependency of daily vaccination progress on the year in which the vaccinations are given or the year in which the vaccinations have started. This may be because of the unbalance in the dataset.

For the Linear regression I was able to remove the insignificant variables and to find the fitted model.

XIII. ETHICAL CONSIDERATIONS

The data required for the analysis was collected from Kaggle.com and it was accessible to the users for download. The journals and research papers used for the project are mentioned in the reference section.

XIV. ACKNOWLEDGEMENT

I would like to extend my sincere thanks to Prof. Rajesh Jaiswal, Prof. Siobhan Connolly Kernan and Prof. Jack Mc Donnel of Dundalk Institute of Technology for helping me in this project.

XV. REFERENCES

[1] World Health Organization, "WHO Coronavirus (COVID-19) Dashboard," 14 November 2021. [Online]. Available: https://covid19.who.int/. [Accessed 14 November 2021].

[2] M. Jadidi, P. Moslemi and S. Jamshi, "Targeted Vaccination for COVID-19 Using Mobile Communication Networks," in *2020 11th International Conference on Information and Knowledge Technology*, Iran, 2020.

[3] D. Diana, "Alpha, Beta, Delta, Gamma: What's important to know about SARS-CoV-2 variants of concern?," *Canadian Medical Association Journal,* vol. 193, no. 27, p. 2, 18 June 2020.

[4] The Visual and Data Journalism Team, BBC News, "Covid vaccines: How fast is progress around the world?," 8 November 2021. [Online]. Available: https://www.bbc.com/news/world-56237778. [Accessed 14 November 2021].

[5] A. Sheikh, J. McMenamin, B. Taylor and C. Robe, "SARS-CoV-2 Delta VOC in Scotland: demographics, risk of hospital admission, and vaccine effectiveness," 14 June 2021. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8201647/. [Accessed 14 November 2021].

[6] A. CARA, "Africa's COVID-19 envoy blasts EU, COVAX over vaccine crisis," 1 July 2021. [Online]. Available: https://apnews.com/article/africa-europe-coronavirus-pandemic-health-business-926c80134a8543efef16e2c7ca4d43c2. [Accessed 14 November 2021].

[7] A. M. Ahamed, S. Velan S and D. Imtiy, "Analysis of Herd Immunity Using Vaccination and Recovery Data Sets," in *2021 12th International Conference on Computing Communication and Networking Technologies*, Kharagpur, India, 2021.

[8] S. K. R, T. S, P. J and R. R, "Covid Vaccination Analysis and Prediction using Machine Learning," *IEEE,* p. 5, 10 October 2021.

[9] N. Darapaneni , S. Dhua, N. Khare, K. Ayush, K. N and S. Ghodke, "Forecasting Vaccination Drive In India for Herd Immunity using SIR and Prophet Model," *IEEE,* pp. 21-25, 0 0 2021.

[10] N. M. Abdulkareem, A. M. Abdulazeez and D. Q. Zeebaree, "COVID-19 World vaccination progress using Machine Learning classification algorithms," *Qubahan Academic Journal,* p. 6, 2021.

[11] [Online]. Available: https://builtin.com/data-science/random-forest-algorithm. [Accessed 15 November 2021].

[12] R. Saxena, "HOW DECISION TREE ALGORITHM WORKS," DataAspirant, 30 January 2017. [Online]. Available: https://dataaspirant.com/how-decision-tree-algorithm-works/. [Accessed 15 November 2021].

[13] J. Le, "The Top 10 Machine Learning Algorithms Every Beginner Should Know," Builtin Beta, 2 August 2021. [Online]. Available: https://builtin.com/data-science/tour-top-10-algorithms-machine-learning-newbies. [Accessed 15 November 2021].

[14] M. Lende, U. Thaware, R. Sharma and K. , "Prioritization of Regions for COVID Vaccinations using FUZZY Logic," in *2021 12th International Conference on Computing Communication and Networking Technologie*, Nagpur, 2021.

[15] C. MAJASKI, "Hypothesis Testing," investopedia, 2021 October 2021. [Online]. Available: https://www.investopedia.com/terms/h/hypothesistesting.asp. [Accessed 22 November 2021].

[16] B.BEERS,"Regression Definition," corporate finance institute, 30 October 2021. [Online]. Available: https://www.investopedia.com/terms/r/regression.asp. [Accessed 22 November 2021].

[17] S. S, "Difference Between ANOVA and ANCOVA," keydifferences, 11 January 2017. [Online]. Available: https://keydifferences.com/difference-between-anova-and-ancova.html. [Accessed 22 November 2021].

XVI. APPENDIX

In [273…

```
import pandas as pd
import math
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
import statsmodels.api as sm
import scipy.stats as stats
from scipy.stats import ranksums
from scipy.stats import ttest_ind
import statsmodels.formula.api as smf
from scipy.stats import chi2_contingency
from scipy.stats import fisher_exact
from scipy.stats import kendalltau
from scipy.stats import spearmanr
from scipy.stats import pearsonr
import pycountry_convert as pc
from statsmodels.formula.api import ols
from bokeh.plotting import figure, output_file, show
import bokeh.io
from bokeh.models import ColumnDataSource
from bokeh.models.tools import HoverTool
from bokeh.palettes import Spectral6
from bokeh.transform import factor_cmap
from bokeh.palettes import Set3_6
from statsmodels.graphics.factorplots import interaction_plot
from statsmodels.graphics.regressionplots import plot_partregress_grid
from statsmodels.stats.anova import anova_lm

import random
import os
os.chdir("G:\\Ginu_StudyMaterials\\Sem1\\CrossModuleProject")
```

In [2]:

```
#Reading the dataset

country_vaccinations = pd.read_csv("country_vaccinations_v1.csv", na_values =("N/A",
country_vaccinations
```

Out[2]:

	country	iso_code	date	total vaccinations	people vaccinated	people fully vaccinated	daily vaccinations	vaccinat per_hun
0	Afghanistan	AFG	11/05/2021	504502	448878	55624	13921	
1	Afghanistan	AFG	20/05/2021	547901	470341	77560	4822	
2	Afghanistan	AFG	24/05/2021	573277	476367	96910	5692	
3	Afghanistan	AFG	26/05/2021	590454	479372	111082	6768	
4	Afghanistan	AFG	27/05/2021	593313	479574	113739	6487	
...	...	...	...	...	...	...	...	

	country	iso_code	date	total vaccinations	people vaccinated	people fully vaccinated	daily vaccinations	vaccinat per_hun
5745	Barbados	BRB	13/10/2021	258231	143661	114570	1184	8
5746	Barbados	BRB	15/10/2021	260653	144569	116084	1162	9
5747	Barbados	BRB	16/10/2021	261495	144981	116514	1134	9
5748	Barbados	BRB	17/10/2021	262359	145339	117020	1145	9
5749	Barbados	BRB	21/10/2021	263548	145648	117900	1129	9

5750 rows × 13 columns

cleaning the data

In [3]:

```
# renaming the column names for better analysis
country_vaccinations.rename(columns = {"total vaccinations":"total_vaccinations","pe
print(country_vaccinations)
```

```
         country iso_code        date  total_vaccinations  people_vaccinated  \
0     Afghanistan      AFG  11/05/2021              504502              448878
1     Afghanistan      AFG  20/05/2021              547901              470341
2     Afghanistan      AFG  24/05/2021              573277              476367
3     Afghanistan      AFG  26/05/2021              590454              479372
4     Afghanistan      AFG  27/05/2021              593313              479574
...           ...      ...         ...                 ...                 ...
5745     Barbados      BRB  13/10/2021              258231              143661
5746     Barbados      BRB  15/10/2021              260653              144569
5747     Barbados      BRB  16/10/2021              261495              144981
5748     Barbados      BRB  17/10/2021              262359              145339
5749     Barbados      BRB  21/10/2021              263548              145648

      people_fully_vaccinated  daily_vaccinations  \
0                       55624               13921
1                       77560                4822
2                       96910                5692
3                      111082                6768
4                      113739                6487
...                       ...                 ...
5745                   114570                1184
5746                   116084                1162
5747                   116514                1134
5748                   117020                1145
5749                   117900                1129

      total_vaccinations_per_hundred  people_vaccinated_per_hundred  \
0                               1.27                           1.13
1                               1.38                           1.18
2                               1.44                           1.20
3                               1.48                           1.20
4                               1.49                           1.20
```

```
...                                 ...                                ...
5745                               89.75                              49.93
5746                               90.60                              50.25
5747                               90.89                              50.39
5748                               91.19                              50.52
5749                               91.60                              50.62

      people_fully_vaccinated_per_hundred  daily_vaccinations_per_million  \
0                                    0.14                             349
1                                    0.19                             121
2                                    0.24                             143
3                                    0.28                             170
4                                    0.29                             163
...                                   ...                             ...
5745                                39.82                            4115
5746                                40.35                            4039
5747                                40.50                            3941
5748                                40.67                            3980
5749                                40.98                            3924

                                                 vaccines  \
0     Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
1     Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
2     Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
3     Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
4     Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
...                                                 ...
5745  Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm...
5746  Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm...
5747  Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm...
5748  Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm...
5749  Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm...

                    source_name
0     World Health Organization
1     World Health Organization
2     World Health Organization
3     World Health Organization
4     World Health Organization
...                         ...
5745         Ministry of Health
5746         Ministry of Health
5747         Ministry of Health
5748         Ministry of Health
5749         Ministry of Health

[5750 rows x 13 columns]
```

In [4]:

```
# checking if any missing values are there in the dataset
country_vaccinations.isnull().sum()
```

Out[4]:

```
country                              0
iso_code                             0
date                                 0
total_vaccinations                   0
people_vaccinated                    0
people_fully_vaccinated              0
daily_vaccinations                   0
total_vaccinations_per_hundred       0
people_vaccinated_per_hundred        0
people_fully_vaccinated_per_hundred  0
daily_vaccinations_per_million       0
vaccines                             0
source_name                          0
dtype: int64
```

There are no missing values present in the dataset.

In [5]:
```
# checking if any duplicate values are present in the dataset
country_vaccinations.duplicated().sum()
```

Out[5]:
```
0
```

There are no duplicate values present in the dataset.

In [6]:
```
country_vaccinations.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5750 entries, 0 to 5749
Data columns (total 13 columns):
 #   Column                               Non-Null Count  Dtype  
---  ------                               --------------  -----  
 0   country                              5750 non-null   object 
 1   iso_code                             5750 non-null   object 
 2   date                                 5750 non-null   object 
 3   total_vaccinations                   5750 non-null   int64  
 4   people_vaccinated                    5750 non-null   int64  
 5   people_fully_vaccinated              5750 non-null   int64  
 6   daily_vaccinations                   5750 non-null   int64  
 7   total_vaccinations_per_hundred       5750 non-null   float64
 8   people_vaccinated_per_hundred        5750 non-null   float64
 9   people_fully_vaccinated_per_hundred  5750 non-null   float64
 10  daily_vaccinations_per_million       5750 non-null   int64  
 11  vaccines                             5750 non-null   object 
 12  source_name                          5750 non-null   object 
dtypes: float64(3), int64(5), object(5)
memory usage: 584.1+ KB
```

In [7]:
```
# The country names are in shuffled order in the last section , so sorting it in asc

country_vaccinations.sort_values(['country'], ascending =True, inplace =True)
```

In [8]:
```
country_vaccinations
```

Out[8]:

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinate
0	Afghanistan	AFG	11/05/2021	504502	448878	5562
16	Afghanistan	AFG	21/10/2021	1201286	770542	43074
15	Afghanistan	AFG	11/07/2021	962093	742934	21915
14	Afghanistan	AFG	07/07/2021	934463	735213	19925
13	Afghanistan	AFG	05/07/2021	915671	726349	18932
...	...	...	...	...	...	
5252	Zimbabwe	ZWE	23/08/2021	3829011	2345043	148396

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinate
5251	Zimbabwe	ZWE	22/08/2021	3773199	2312117	146108
5250	Zimbabwe	ZWE	21/08/2021	3753242	2301341	145190
5199	Zimbabwe	ZWE	14/05/2021	775219	571458	20376
5219	Zimbabwe	ZWE	26/06/2021	1249486	747328	50215

5750 rows × 13 columns

In [9]:
```
# changing the date to pandas datetime format

cd_date = pd.to_datetime(country_vaccinations['date'].astype(str), format ='%d/%m/%Y
```

In [10]:
```
# creating separate columns for year and month for analysis.
country_vaccinations['year'] = cd_date.dt.year
country_vaccinations['month'] = cd_date.dt.month
```

In [11]:
```
# changing month values from Numbers to text

country_vaccinations['date'] = pd.to_datetime(country_vaccinations['date'])
country_vaccinations['month'] = country_vaccinations['date'].dt.month_name()
print(country_vaccinations)
```

```
          country iso_code       date  total_vaccinations  people_vaccinated  \
0     Afghanistan      AFG 2021-11-05              504502             448878
16    Afghanistan      AFG 2021-10-21             1201286             770542
15    Afghanistan      AFG 2021-11-07              962093             742934
14    Afghanistan      AFG 2021-07-07              934463             735213
13    Afghanistan      AFG 2021-05-07              915671             726349
...           ...      ...        ...                 ...                ...
5252     Zimbabwe      ZWE 2021-08-23             3829011            2345043
5251     Zimbabwe      ZWE 2021-08-22             3773199            2312117
5250     Zimbabwe      ZWE 2021-08-21             3753242            2301341
5199     Zimbabwe      ZWE 2021-05-14              775219             571458
5219     Zimbabwe      ZWE 2021-06-26             1249486             747328

      people_fully_vaccinated  daily_vaccinations  \
0                       55624               13921
16                     430744                1119
15                     219159                7455
14                     199250                6801
13                     189322                8989
...                       ...                 ...
5252                  1483968               69004
5251                  1461082               69134
5250                  1451901               74048
5199                   203761               19211
5219                   502158               16169

      total_vaccinations_per_hundred  people_vaccinated_per_hundred  \
```

```
0                                1.27                                1.13
16                               3.02                                1.93
15                               2.42                                1.87
14                               2.35                                1.85
13                               2.30                                1.82
...                               ...                                 ...
5252                            25.37                               15.54
5251                            25.00                               15.32
5250                            24.87                               15.25
5199                             5.14                                3.79
5219                             8.28                                4.95

      people_fully_vaccinated_per_hundred  daily_vaccinations_per_million  \
0                                    0.14                             349
16                                   1.08                              28
15                                   0.55                             187
14                                   0.50                             171
13                                   0.48                             226
...                                   ...                             ...
5252                                 9.83                            4572
5251                                 9.68                            4581
5250                                 9.62                            4906
5199                                 1.35                            1273
5219                                 3.33                            1071

                                              vaccines  \
0     Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
16    Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
15    Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
14    Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
13    Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
...                                                 ...
5252  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac...
5251  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac...
5250  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac...
5199  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac...
5219  Oxford/AstraZeneca, Sinopharm/Beijing, Sinovac...

                     source_name  year     month
0     World Health Organization  2021  November
16    World Health Organization  2021   October
15    World Health Organization  2021  November
14    World Health Organization  2021      July
13    World Health Organization  2021       May
...                         ...   ...       ...
5252         Ministry of Health  2021    August
5251         Ministry of Health  2021    August
5250         Ministry of Health  2021    August
5199         Ministry of Health  2021       May
5219         Ministry of Health  2021      June

[5750 rows x 15 columns]
```

In [12]:

```
# printing the number of countries in the datset
d = list(country_vaccinations['country'].unique())
d = len(d)
print(d)
```

```
61
```

In [13]:

```
# adding continent column for visualizations and analysis

def convert(row):
    cn_code = pc.country_name_to_country_alpha2(row.country, cn_name_format = "defau
    conti_code = pc.country_alpha2_to_continent_code(cn_code)
    return conti_code
```

In [14]:
```
country_vaccinations["continent"] = country_vaccinations.apply(convert, axis =1 )
country_vaccinations
```

Out[14]:

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	da
0	Afghanistan	AFG	2021-11-05	504502	448878	55624	
16	Afghanistan	AFG	2021-10-21	1201286	770542	430744	
15	Afghanistan	AFG	2021-11-07	962093	742934	219159	
14	Afghanistan	AFG	2021-07-07	934463	735213	199250	
13	Afghanistan	AFG	2021-05-07	915671	726349	189322	
...	...	...	...	...	...	...	
5252	Zimbabwe	ZWE	2021-08-23	3829011	2345043	1483968	
5251	Zimbabwe	ZWE	2021-08-22	3773199	2312117	1461082	
5250	Zimbabwe	ZWE	2021-08-21	3753242	2301341	1451901	
5199	Zimbabwe	ZWE	2021-05-14	775219	571458	203761	
5219	Zimbabwe	ZWE	2021-06-26	1249486	747328	502158	

5750 rows × 16 columns

In [15]:
```
# converting continent codes to Names
conti_names = {"AS": "Asia", "SA":"South America", "AN":"Antartica", "OC":"Oceania",
country_vaccinations["continent"] = country_vaccinations["continent"].map(conti_name
country_vaccinations
```

Out[15]:

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	da
0	Afghanistan	AFG	2021-11-05	504502	448878	55624	

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	da
16	Afghanistan	AFG	2021-10-21	1201286	770542	430744	
15	Afghanistan	AFG	2021-11-07	962093	742934	219159	
14	Afghanistan	AFG	2021-07-07	934463	735213	199250	
13	Afghanistan	AFG	2021-05-07	915671	726349	189322	
...	...	...	...	...	...	...	
5252	Zimbabwe	ZWE	2021-08-23	3829011	2345043	1483968	
5251	Zimbabwe	ZWE	2021-08-22	3773199	2312117	1461082	
5250	Zimbabwe	ZWE	2021-08-21	3753242	2301341	1451901	
5199	Zimbabwe	ZWE	2021-05-14	775219	571458	203761	
5219	Zimbabwe	ZWE	2021-06-26	1249486	747328	502158	

5750 rows × 16 columns

In [16]:
```
country_vaccinations.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 5750 entries, 0 to 5219
Data columns (total 16 columns):
 #   Column                             Non-Null Count  Dtype         
---  ------                             --------------  -----         
 0   country                            5750 non-null   object        
 1   iso_code                           5750 non-null   object        
 2   date                               5750 non-null   datetime64[ns]
 3   total_vaccinations                 5750 non-null   int64         
 4   people_vaccinated                  5750 non-null   int64         
 5   people_fully_vaccinated            5750 non-null   int64         
 6   daily_vaccinations                 5750 non-null   int64         
 7   total_vaccinations_per_hundred     5750 non-null   float64       
 8   people_vaccinated_per_hundred      5750 non-null   float64       
 9   people_fully_vaccinated_per_hundred  5750 non-null   float64       
 10  daily_vaccinations_per_million     5750 non-null   int64         
 11  vaccines                           5750 non-null   object        
 12  source_name                        5750 non-null   object        
 13  year                               5750 non-null   int64         
 14  month                              5750 non-null   object        
 15  continent                          5750 non-null   object        
```

```
dtypes: datetime64[ns](1), float64(3), int64(6), object(6)
memory usage: 763.7+ KB
```

In [17]:
```
# checking if any missing values are there in the dataset
country_vaccinations.isnull().sum()
```

Out[17]:
```
country                              0
iso_code                             0
date                                 0
total_vaccinations                   0
people_vaccinated                    0
people_fully_vaccinated              0
daily_vaccinations                   0
total_vaccinations_per_hundred       0
people_vaccinated_per_hundred        0
people_fully_vaccinated_per_hundred  0
daily_vaccinations_per_million       0
vaccines                             0
source_name                          0
year                                 0
month                                0
continent                            0
dtype: int64
```

In [18]:
```
# saving cleaned dataset to csv
country_vaccinations.to_csv("country_vaccinations_v1.1.csv", index=False)
```

Description of the data. This should include, brief discussion of the data and variables of interest, identification of outliers/missing data, and treatment of any outliers/missing data. (10 marks)

The dataset contains the COVID-19 vaccination details of different countries. It includes the daily vaccinations given in each countries, the total vaccinated and fully vacinated peoples numbers for each day for 61 countries.The data is for the time period 2020 and 2021.It was collected from the Kaggle.com. The columns year, month and continent was additionally created for the analysis purpose.

There are 5750 observations and 16 variables for the dataset.

In [19]:
```
# selecting a sample dataset of countries of Asia and Europe for the analysis and pl
# I am selecting India and Ireland because India is my home country and Ireland is t
# I selected Baharain since it is the only Asian country which has started vaccinati
# Then I selected United Kingdom which is the neighbouring country and it has a larg

values=["India","Ireland", "Bahrain", "United Kingdom"]
sel_data = country_vaccinations[country_vaccinations["country"].isin(values)]
sel_data
#sel_data = country_vaccinations.loc[(country_vaccinations['country'] == 'Ireland')
#sel_data
```

Out[19]:

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily
295	Bahrain	BHR	2021-07-30	2297602	1107217	1052384	
296	Bahrain	BHR	2021-07-31	2305354	1108252	1055185	

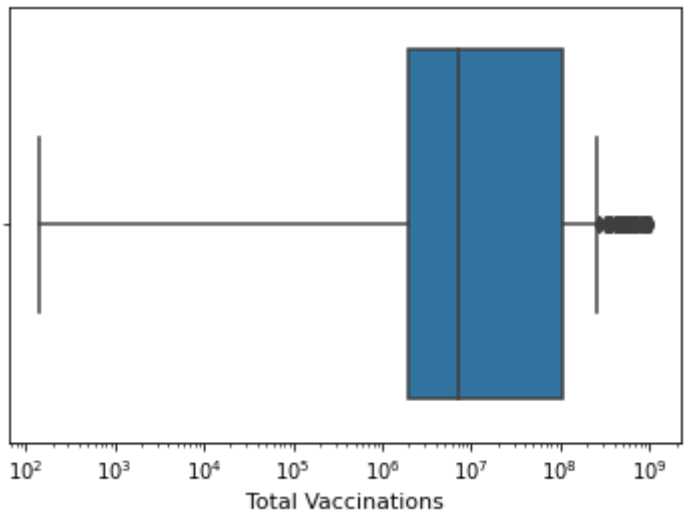
	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily
294	Bahrain	BHR	2021-07-24	2266289	1101005	1041059	
293	Bahrain	BHR	2021-07-23	2262698	1100267	1039944	
292	Bahrain	BHR	2021-07-22	2257391	1099361	1038168	
...	...	...	...	...	...	...	
4975	United Kingdom	GBR	2021-11-09	92419795	48424317	43995478	
4974	United Kingdom	GBR	2021-10-09	92293823	48396404	43897419	
4987	United Kingdom	GBR	2021-10-18	94863415	49462425	45400990	
4989	United Kingdom	GBR	2021-10-20	95014529	49554407	45460122	
4939	United Kingdom	GBR	2021-05-16	56992075	36704672	20287403	

646 rows × 16 columns

Univariate plots

In [20]:

```
# total_vaccinations

boxplot = sns.boxplot(x="total_vaccinations", data =sel_data )
boxplot.set_xlabel("Total Vaccinations", fontsize=11)
boxplot.set_xscale('log')
```



In [21]:
```
sel_data['total_vaccinations'].describe()
```

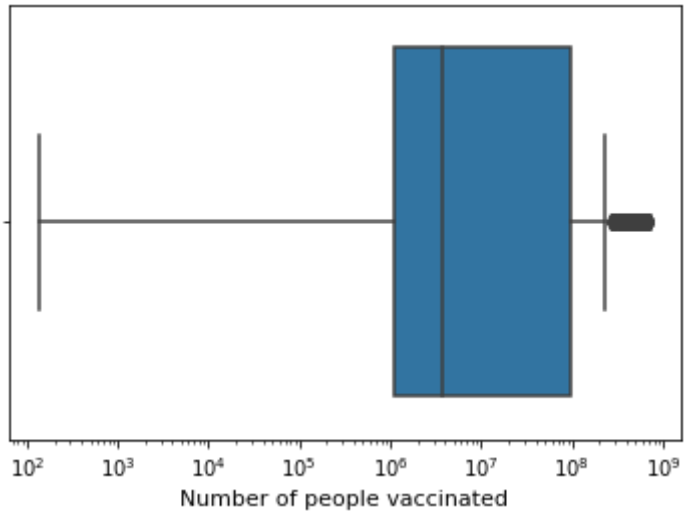
Out[21]:
```
count    6.460000e+02
mean     1.284127e+08
std      2.377074e+08
min      1.410000e+02
25%      1.982450e+06
50%      7.243817e+06
75%      1.075320e+08
max      1.003204e+09
Name: total_vaccinations, dtype: float64
```

The data has outliers and it is skewed to the right.

The mean and median are different and the whiskers are of different length.

Therefore the variable 'total_vaccinations' is not symmetrical in shape.

In [22]:
```
# people_vaccinated

boxplot = sns.boxplot(x="people_vaccinated", data =sel_data )
boxplot.set_xlabel("Number of people vaccinated", fontsize=11)
boxplot.set_xscale('log')
```

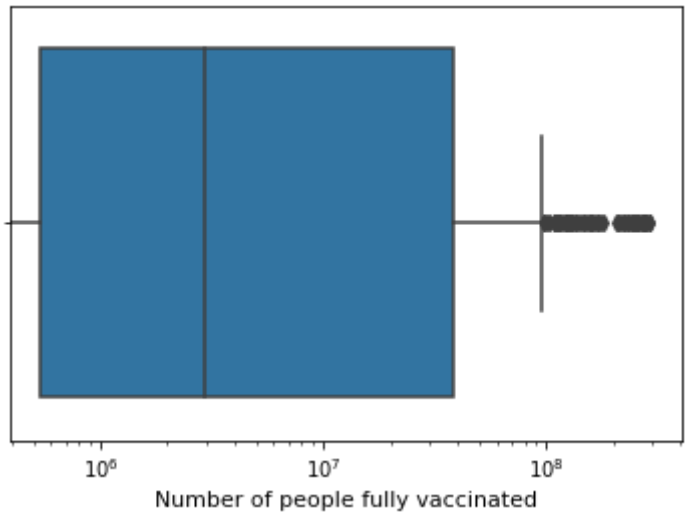


The data has outliers and it is skewed to the right.

The mean and median are roughly equal but, the whiskers are of different length.

Therefore the variable 'people_vaccinated' is not symmetrical in shape.

In [23]:
```
# people_fully_vaccinated

boxplot = sns.boxplot(x="people_fully_vaccinated", data =sel_data )
boxplot.set_xlabel("Number of people fully vaccinated", fontsize=11)
boxplot.set_xscale('log')
```



In [24]:
```
sel_data['people_fully_vaccinated'].describe()
```

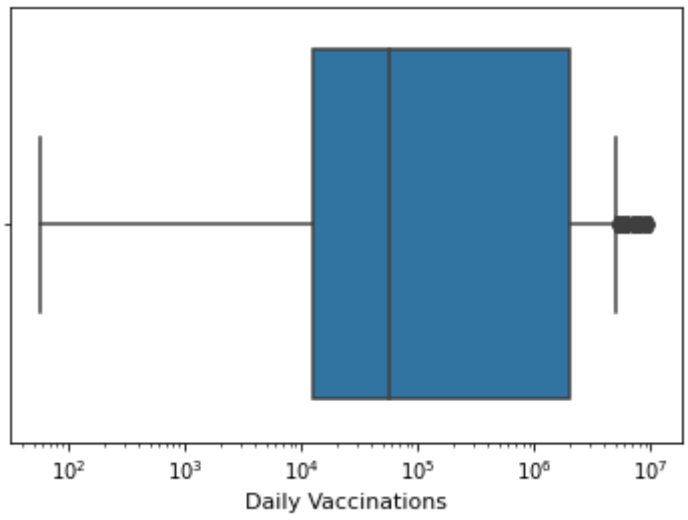
Out[24]:
```
count    6.460000e+02
mean     3.152000e+07
std      6.076850e+07
min      0.000000e+00
25%      5.360668e+05
50%      2.917430e+06
75%      3.859196e+07
max      2.936523e+08
Name: people_fully_vaccinated, dtype: float64
```

The data has outliers and it is skewed to the right.

The mean and median are different and the whiskers are of different length.

Therefore the variable 'people_vaccinated' is not symmetrical in shape.

In [25]:
```
# daily_vaccinations

boxplot = sns.boxplot(x="daily_vaccinations", data =sel_data )
boxplot.set_xlabel("Daily Vaccinations", fontsize=11)
boxplot.set_xscale('log')
```



In [26]:
```
sel_data['daily_vaccinations'].describe()
```

```
count    6.460000e+02
```

Out[26]:
```
mean      1.327012e+06
std       2.200616e+06
min       5.700000e+01
25%       1.277475e+04
50%       5.686950e+04
75%       2.011813e+06
max       1.003800e+07
Name: daily_vaccinations, dtype: float64
```

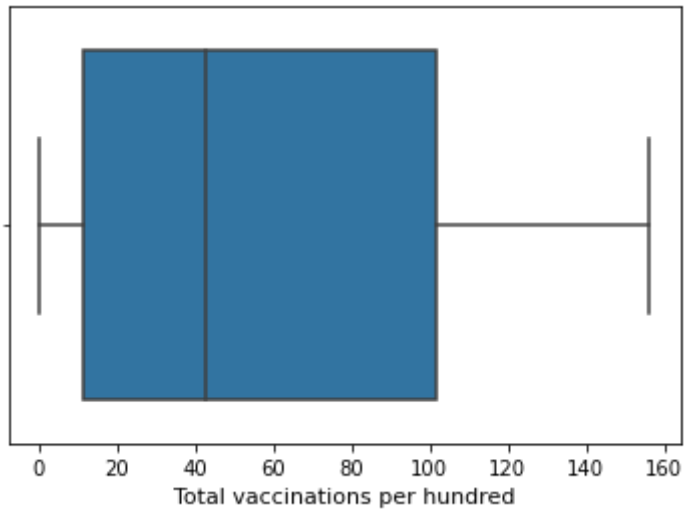
The data has outliers and it is skewed to the right.

The mean and median are roughly equal but, the whiskers are of different length.

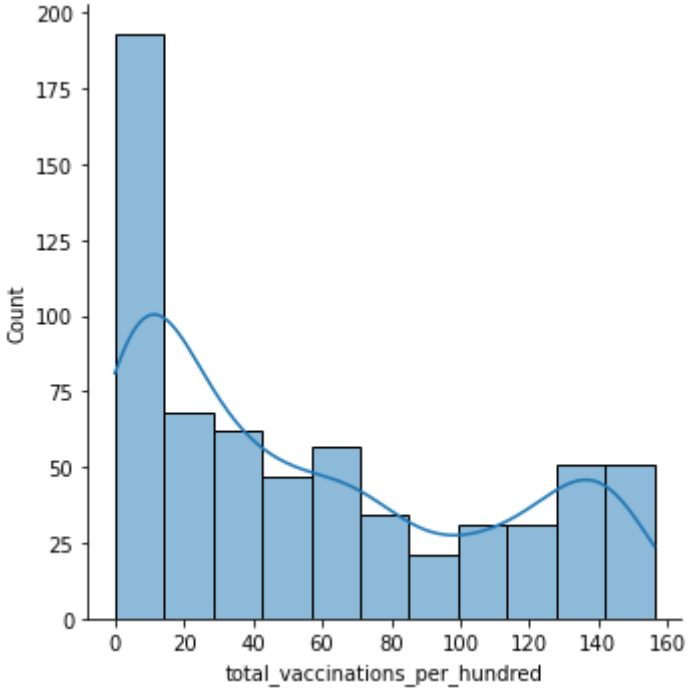
Therefore the variable 'daily_vaccinations' is not symmetrical in shape.

In [27]:
```
# total_vaccinations_per_hundred

boxplot = sns.boxplot(x="total_vaccinations_per_hundred", data =sel_data )
boxplot.set_xlabel("Total vaccinations per hundred", fontsize=11)
```

Out[27]:
```
Text(0.5, 0, 'Total vaccinations per hundred')
```



In [28]:
```
sns.displot(x='total_vaccinations_per_hundred', data=sel_data, kde=True)

plt.show()
```



The data has no outliers present.

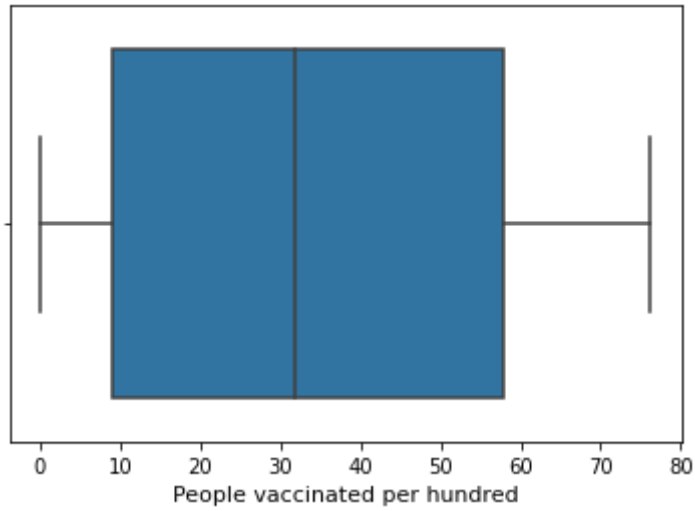
The mean and median are different and the whiskers are of different length.

The does not looks symmetrical.

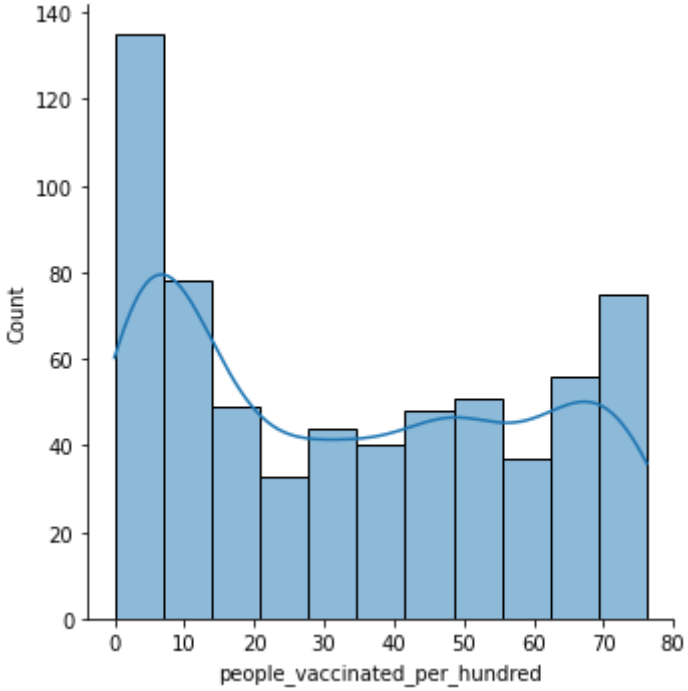
Therefore the variable 'daily_vaccinations' is not symmetrical in shape.

In [29]:
```
# people_vaccinated_per_hundred

boxplot = sns.boxplot(x="people_vaccinated_per_hundred", data =sel_data )
boxplot.set_xlabel("People vaccinated per hundred", fontsize=11)
```

Out[29]:
```
Text(0.5, 0, 'People vaccinated per hundred')
```



In [30]:
```
sns.displot(x='people_vaccinated_per_hundred', data=sel_data, kde=True)
plt.show()
```



The data has no outliers present.

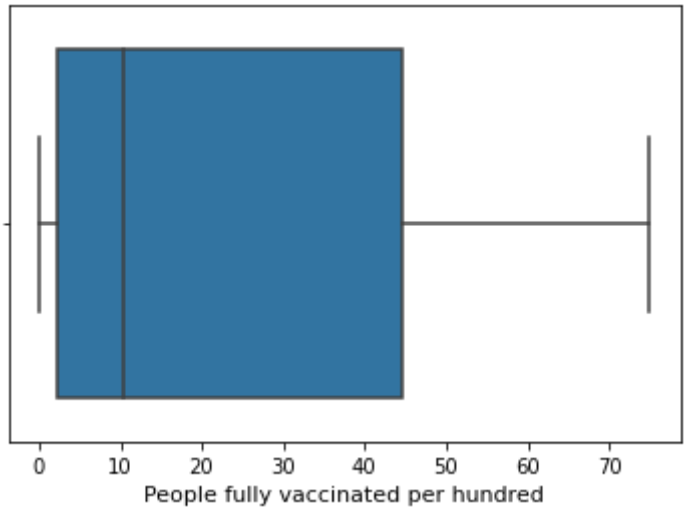
The mean and median are different and the whiskers are of different length.

The data loks roughly symmetrical symmetrical.

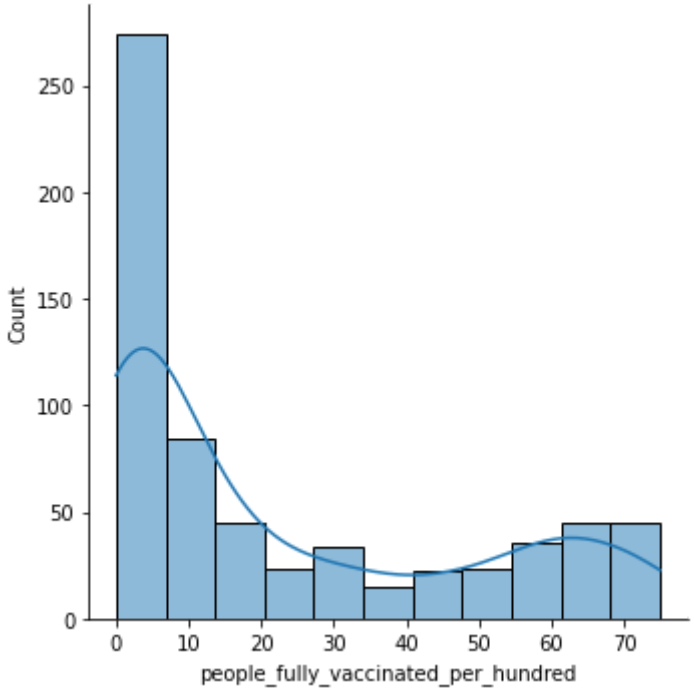
Therefore the variable 'people_vaccinated_per_hundred' is roughly symmetrical in shape.

In [31]:
```
# people_fully_vaccinated_per_hundred

boxplot = sns.boxplot(x="people_fully_vaccinated_per_hundred", data =sel_data )
boxplot.set_xlabel("People fully vaccinated per hundred", fontsize=11)
```

Out[31]:
```
Text(0.5, 0, 'People fully vaccinated per hundred')
```



In [32]:
```
sns.displot(x='people_fully_vaccinated_per_hundred', data=sel_data, kde=True)
plt.show()
```



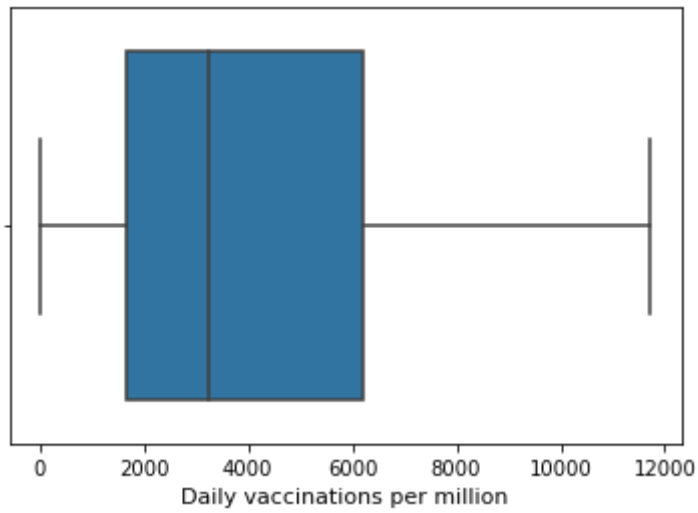
The data has no outliers.

The mean and median are different and the whiskers are of different length.

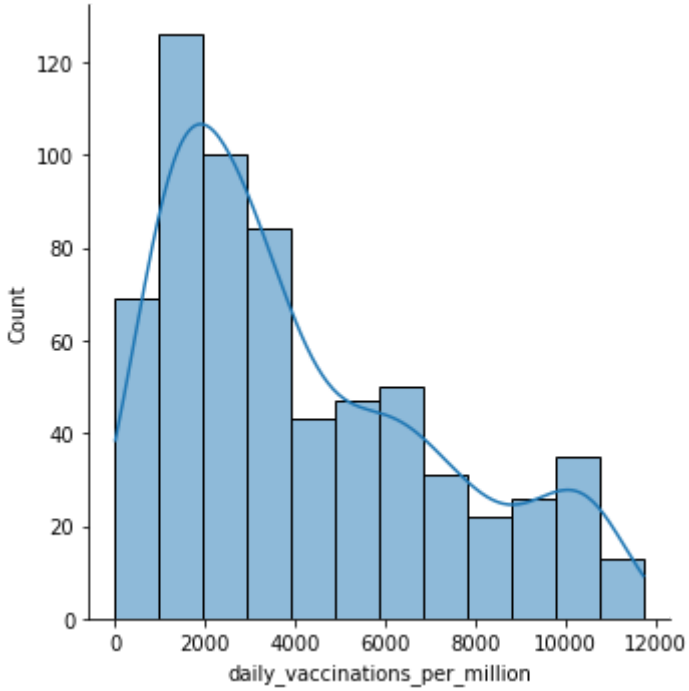
Therefore the variable 'people_fully_vaccinated_per_hundred' is roughly symmetrical in shape

In [33]:
```
# daily_vaccinations_per_million

boxplot = sns.boxplot(x="daily_vaccinations_per_million", data =sel_data )
boxplot.set_xlabel("Daily vaccinations per million", fontsize=11)
```

Out[33]:
```
Text(0.5, 0, 'Daily vaccinations per million')
```



In [34]:
```
sns.displot(x='daily_vaccinations_per_million', data=sel_data, kde=True)
plt.show()
```

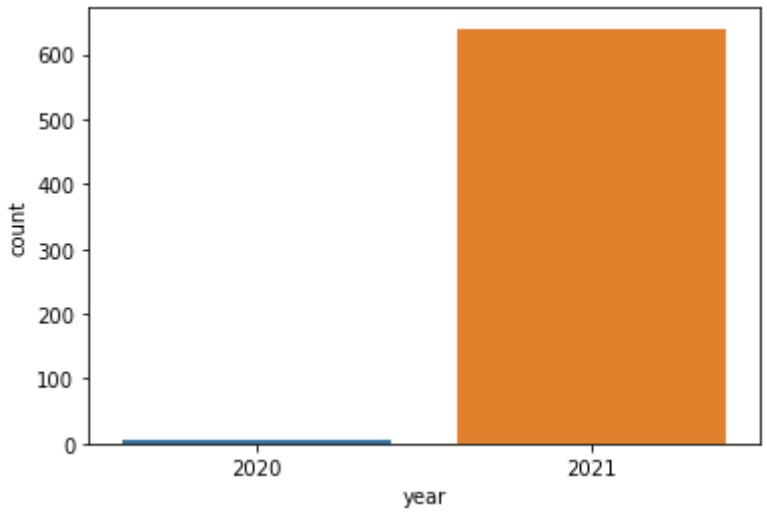


The data has outliers and it is skewed to the right.

The mean and median are roughly equal and the whiskers are of different length.

Therefore the variable 'daily_vaccinations_per_million' is not symmetrical in shape

In [35]:
```
# year

sns.countplot(x='year', data=sel_data)
plt.show()
```

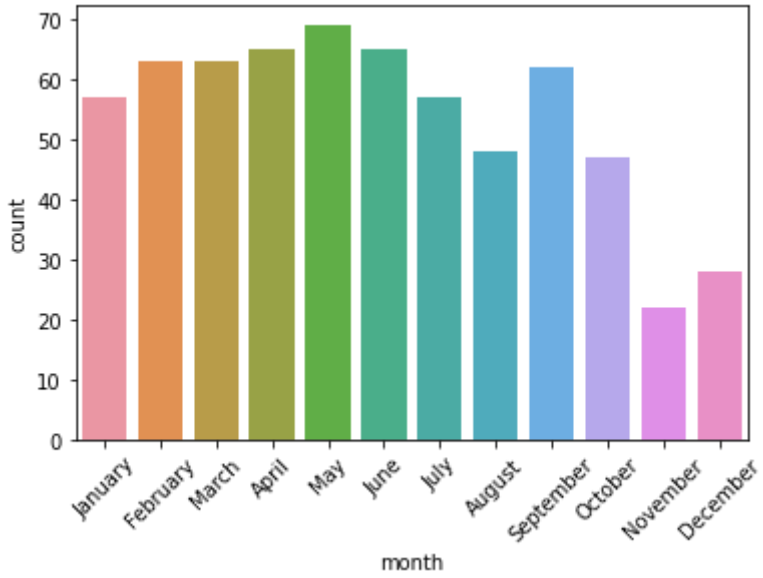


Majority of the observations in the sample falls in the 2021 year.

less than 10 % falls in the 2020 year.

In [36]:
```
# month
mon_data = sel_data.copy()
mon_data['month'] = pd.Categorical(mon_data['month'],
                                   categories=['January', 'February', 'March', 'Apri
                                   ordered=True)
chart = sns.countplot(x='month', data=mon_data)
```

```
chart.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

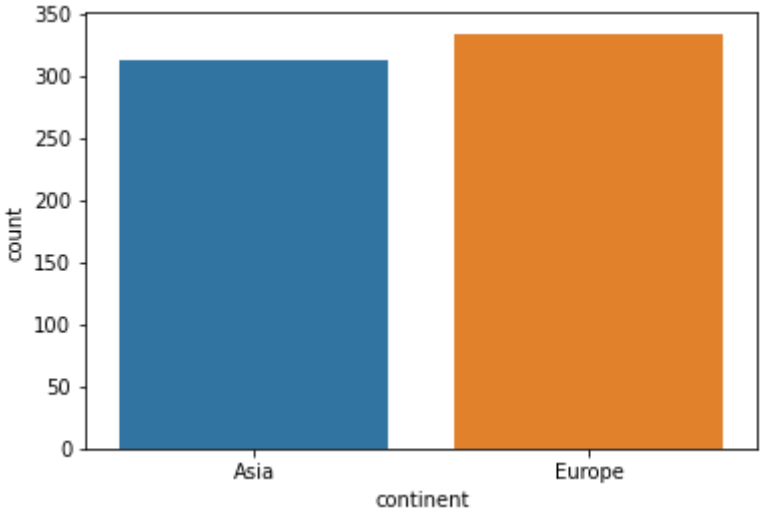


In [37]:
```
sel_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 646 entries, 295 to 4939
Data columns (total 16 columns):
 #   Column                               Non-Null Count  Dtype
---  ------                               --------------  -----
 0   country                              646 non-null    object
 1   iso_code                             646 non-null    object
 2   date                                 646 non-null    datetime64[ns]
 3   total_vaccinations                   646 non-null    int64
 4   people_vaccinated                    646 non-null    int64
 5   people_fully_vaccinated              646 non-null    int64
 6   daily_vaccinations                   646 non-null    int64
 7   total_vaccinations_per_hundred       646 non-null    float64
 8   people_vaccinated_per_hundred        646 non-null    float64
 9   people_fully_vaccinated_per_hundred  646 non-null    float64
 10  daily_vaccinations_per_million       646 non-null    int64
 11  vaccines                             646 non-null    object
 12  source_name                          646 non-null    object
 13  year                                 646 non-null    int64
 14  month                                646 non-null    object
 15  continent                            646 non-null    object
dtypes: datetime64[ns](1), float64(3), int64(6), object(6)
memory usage: 102.0+ KB
```

More number of observations are in the 7 th month ie. July. It shows more vaccinations are taken in the month of july.Least number is in the 12 th month ie. december.

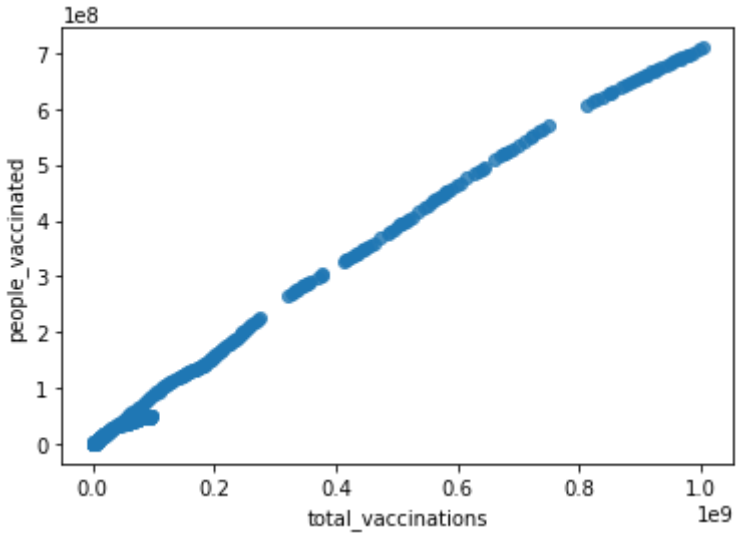
In [38]:
```
sns.countplot(x='continent', data=sel_data)
plt.show()
```



More number of obervationss lies in the Europe continent in the selected sample.

Bivariate Plots

In [39]:
```
# total_vaccinations and people_vaccinated
sns.regplot(x='total_vaccinations', y='people_vaccinated', data=sel_data,fit_reg=Fal
plt.ylabel("people_vaccinated")
plt.xlabel("total_vaccinations")
plt.show()
```

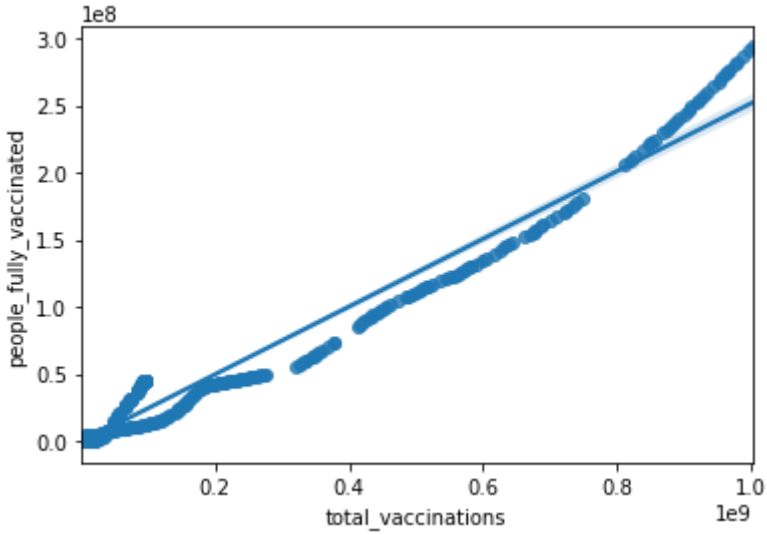


There is a relationship between the number of total vaccinations and the people vaccinated.

3 outliers are there in the data.

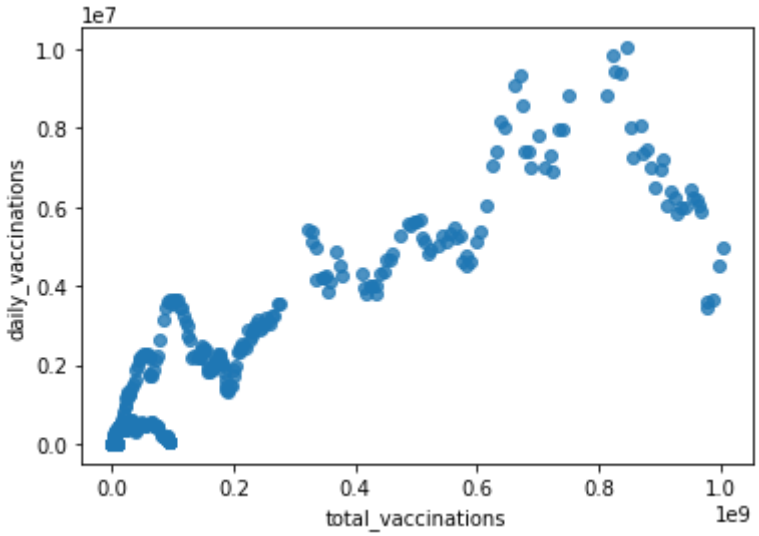
It is showing a strong positive linear relationship.

In [40]:
```
p1 = sns.regplot(x='total_vaccinations', y='people_fully_vaccinated', data=sel_data,
plt.ylabel("people_fully_vaccinated")
plt.xlabel("total_vaccinations")
plt.show()
```



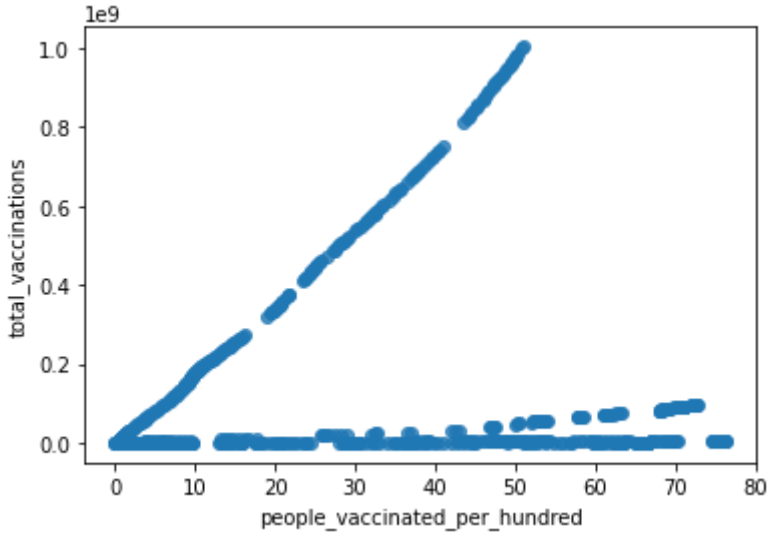
In [41]:

```
p1 = sns.regplot(x='total_vaccinations', y='daily_vaccinations', data=sel_data, fit_
plt.ylabel("daily_vaccinations")
plt.xlabel("total_vaccinations")
plt.show()
```



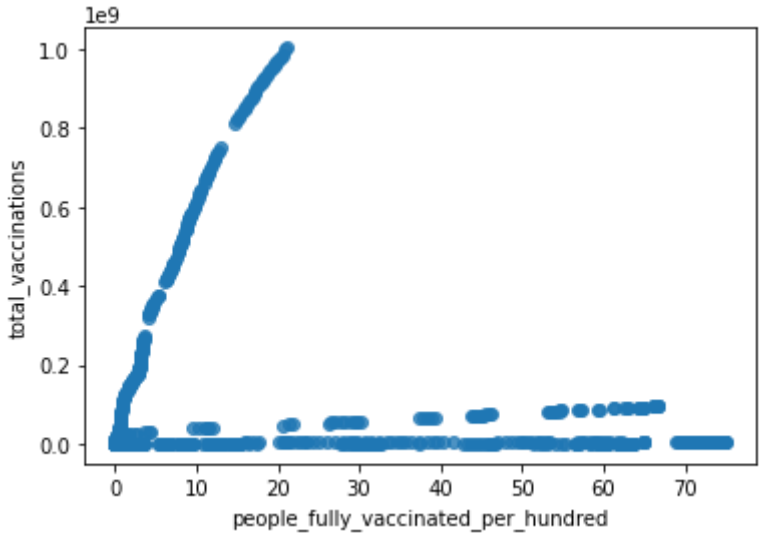
In [42]:

```
p1 = sns.regplot(y='total_vaccinations', x='people_vaccinated_per_hundred', data=sel
plt.xlabel("people_vaccinated_per_hundred")
plt.ylabel("total_vaccinations")
plt.show()
```



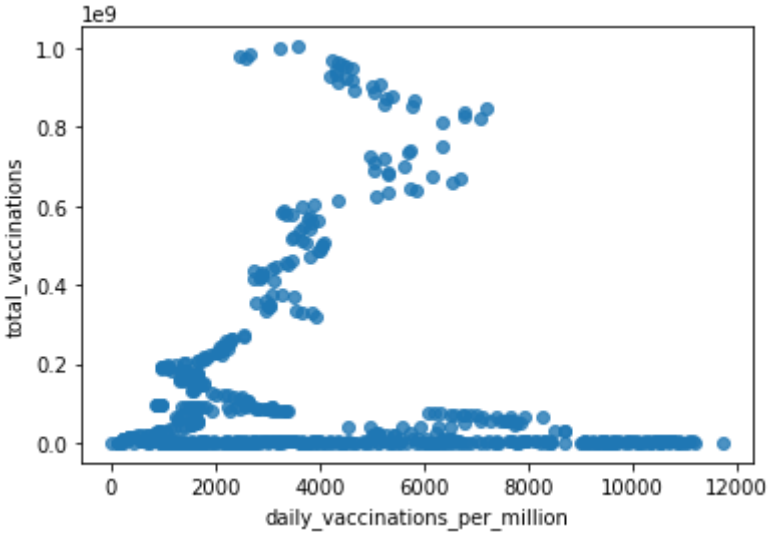
In [43]:

```
p1 = sns.regplot(y='total_vaccinations', x='people_fully_vaccinated_per_hundred', da
plt.xlabel("people_fully_vaccinated_per_hundred")
plt.ylabel("total_vaccinations")
plt.show()
```



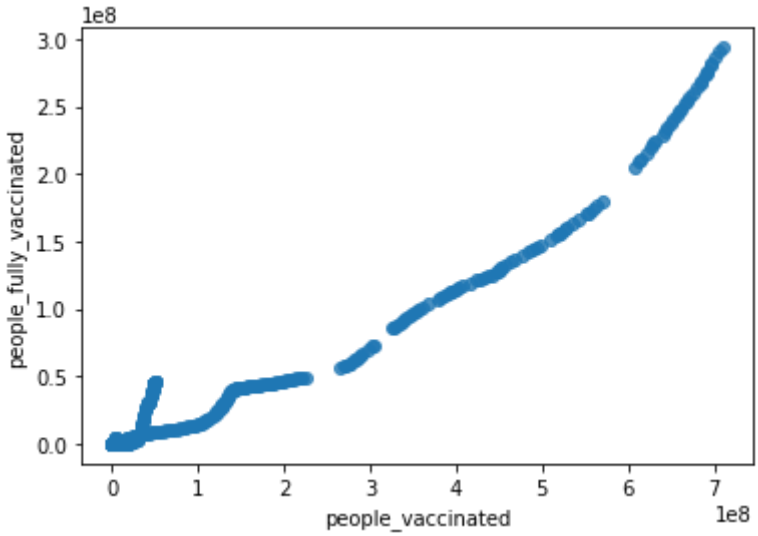
In [44]:

```
p1 = sns.regplot(y='total_vaccinations', x='daily_vaccinations_per_million', data=se
plt.xlabel("daily_vaccinations_per_million")
plt.ylabel("total_vaccinations")
plt.show()
```



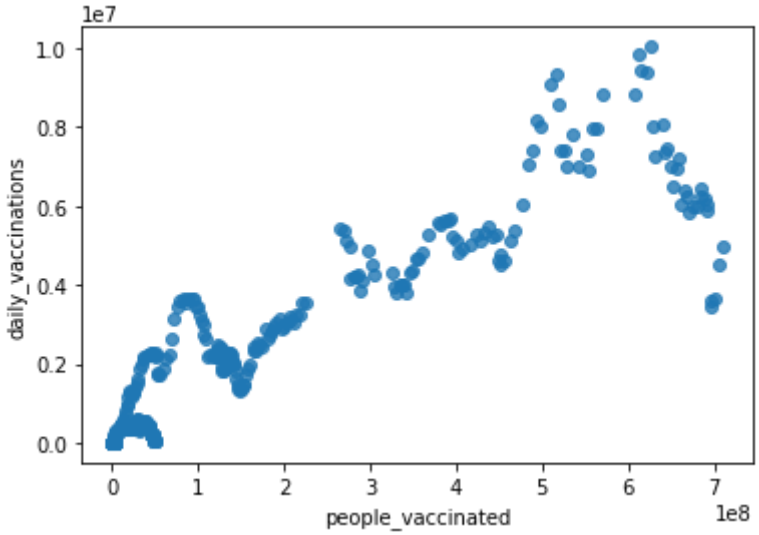
In [45]:

```
p1 = sns.regplot(x='people_vaccinated', y='people_fully_vaccinated', data=sel_data,
plt.xlabel("people_vaccinated")
plt.ylabel("people_fully_vaccinated")
plt.show()
```



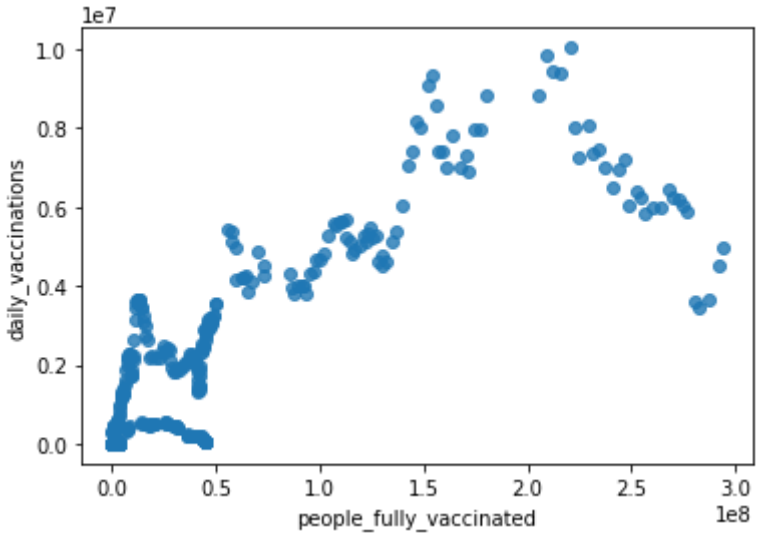
In [46]:

```
p1 = sns.regplot(x='people_vaccinated', y='daily_vaccinations', data=sel_data, fit_r
plt.xlabel("people_vaccinated")
plt.ylabel("daily_vaccinations")
plt.show()
```



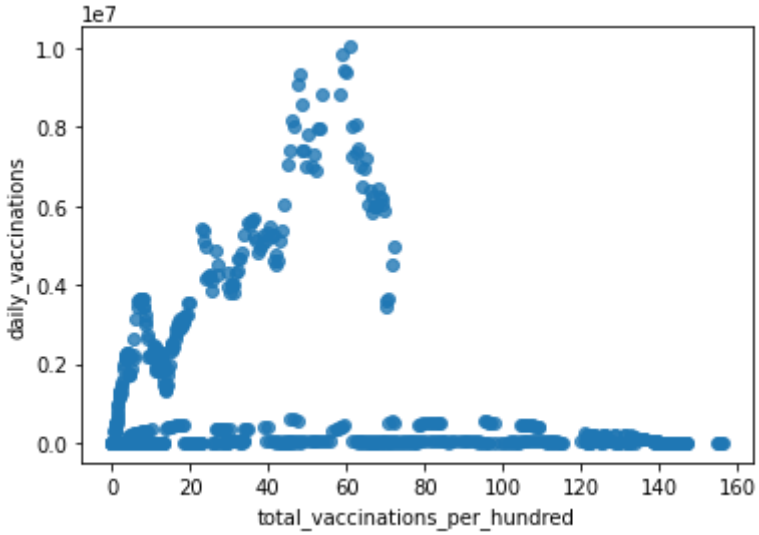
In [47]:

```
p1 = sns.regplot(x='people_fully_vaccinated', y='daily_vaccinations', data=sel_data,
plt.xlabel("people_fully_vaccinated")
plt.ylabel("daily_vaccinations")
plt.show()
```



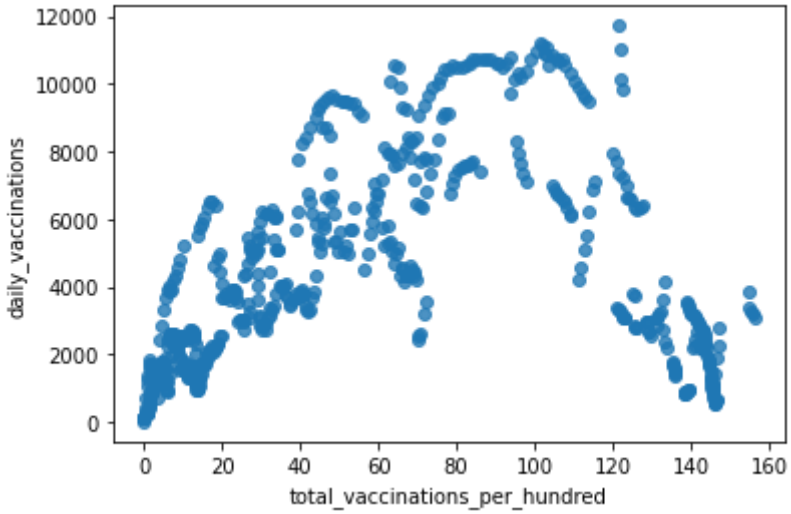
In [48]:

```
p1 = sns.regplot(x='total_vaccinations_per_hundred', y='daily_vaccinations', data=se
plt.xlabel("total_vaccinations_per_hundred")
plt.ylabel("daily_vaccinations")
plt.show()
```



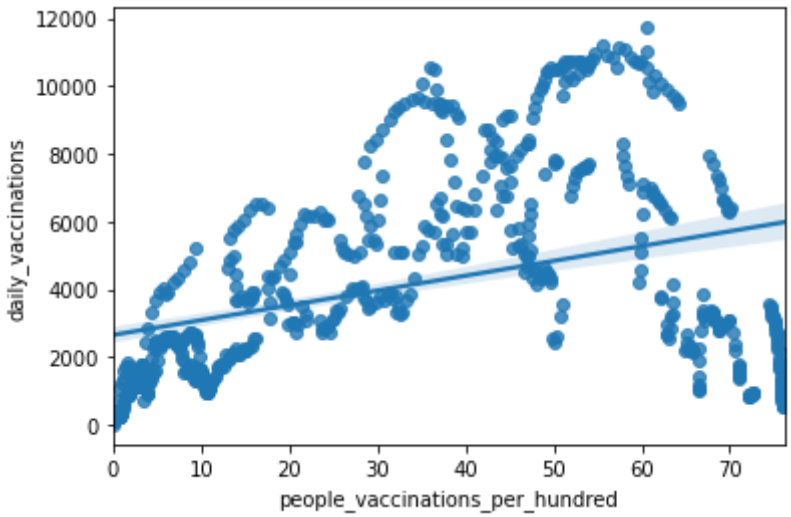
In [49]:

```
p1 = sns.regplot(x='total_vaccinations_per_hundred', y='daily_vaccinations_per_milli
plt.xlabel("total_vaccinations_per_hundred")
plt.ylabel("daily_vaccinations")
plt.show()
```



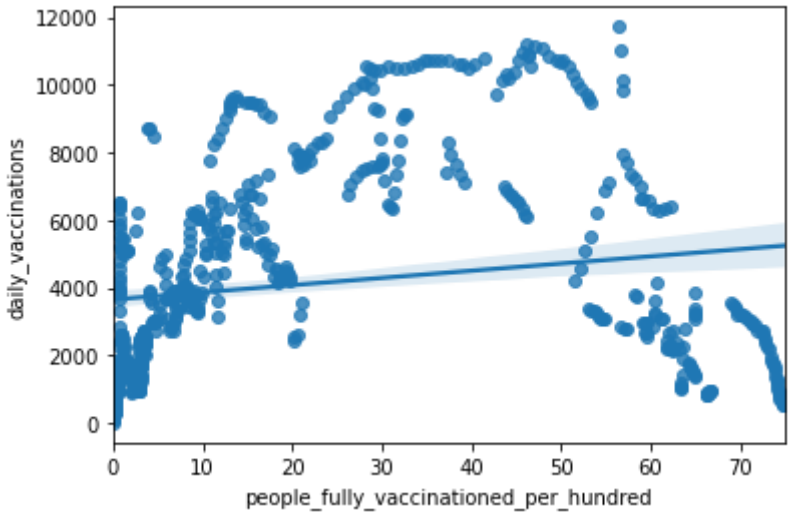
In [258…

```
p1 = sns.regplot(x='people_vaccinated_per_hundred', y='daily_vaccinations_per_millio
plt.xlabel("people_vaccinations_per_hundred")
plt.ylabel("daily_vaccinations")
plt.show()
```



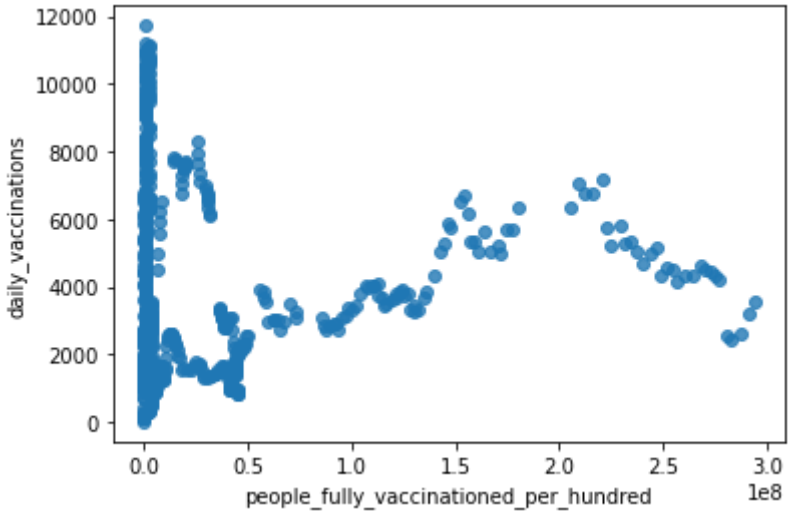
In [255…

```
p1 = sns.regplot(x='people_fully_vaccinated_per_hundred', y='daily_vaccinations_per_
plt.xlabel("people_fully_vaccinationed_per_hundred")
plt.ylabel("daily_vaccinations")
plt.show()
```



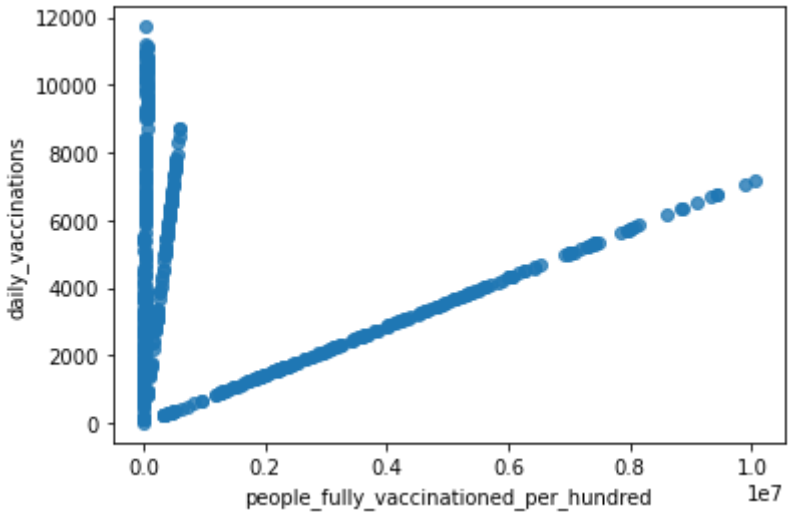
In [260…

```
p1 = sns.regplot(x='people_fully_vaccinated', y='daily_vaccinations_per_million', da
plt.xlabel("people_fully_vaccinationed_per_hundred")
plt.ylabel("daily_vaccinations")
plt.show()
```

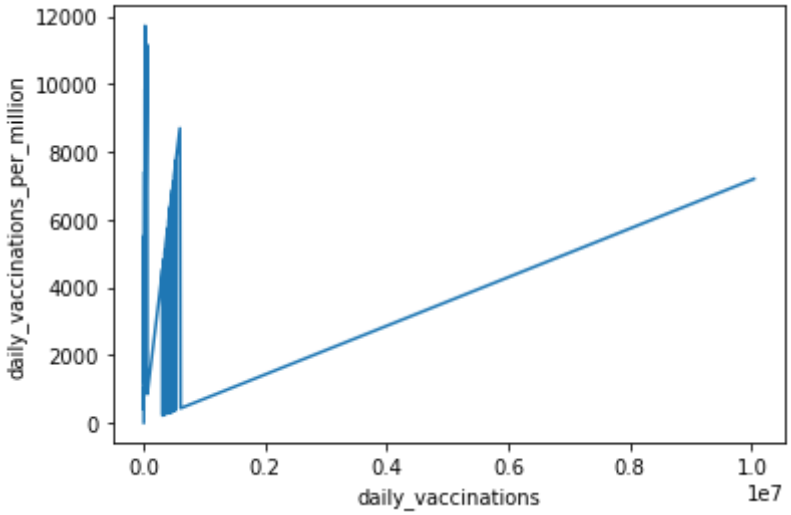


In [53]:

```
p1 = sns.regplot(x='daily_vaccinations', y='daily_vaccinations_per_million', data=se
plt.xlabel("people_fully_vaccinationed_per_hundred")
plt.ylabel("daily_vaccinations")
plt.show()

sns.lineplot(data=sel_data, x="daily_vaccinations", y="daily_vaccinations_per_millio
```

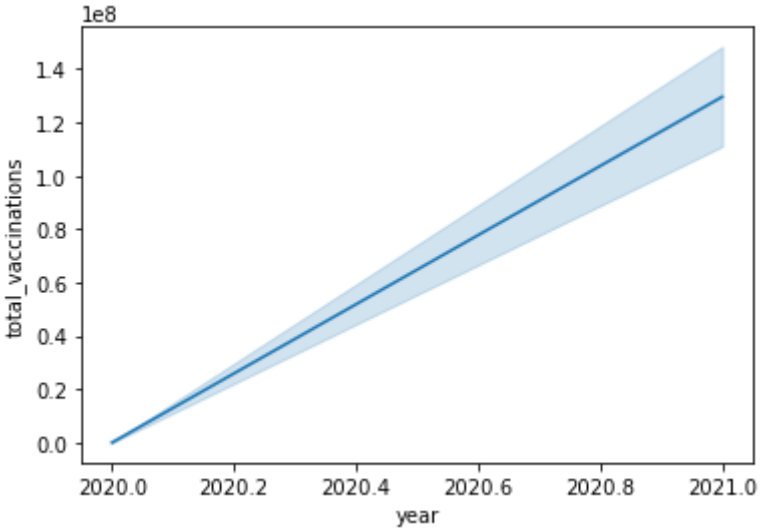


Out[53]: <AxesSubplot:xlabel='daily_vaccinations', ylabel='daily_vaccinations_per_million'>



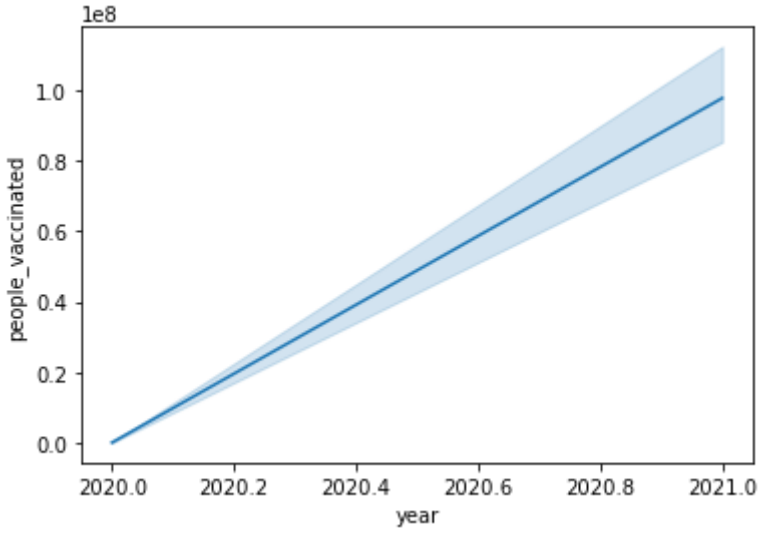
In [54]:
```
sns.lineplot(data=sel_data, x="year", y="total_vaccinations")
```

Out[54]: <AxesSubplot:xlabel='year', ylabel='total_vaccinations'>



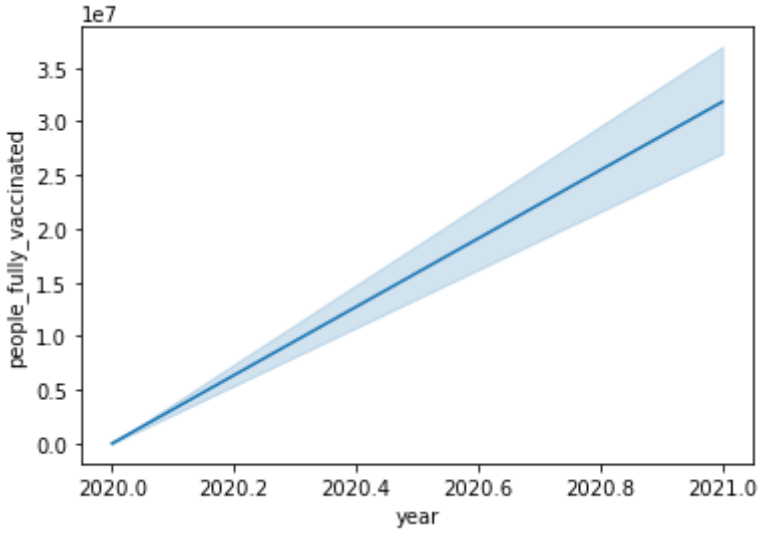
In [55]:
```
sns.lineplot(data=sel_data, x="year", y="people_vaccinated")
```

Out[55]: <AxesSubplot:xlabel='year', ylabel='people_vaccinated'>



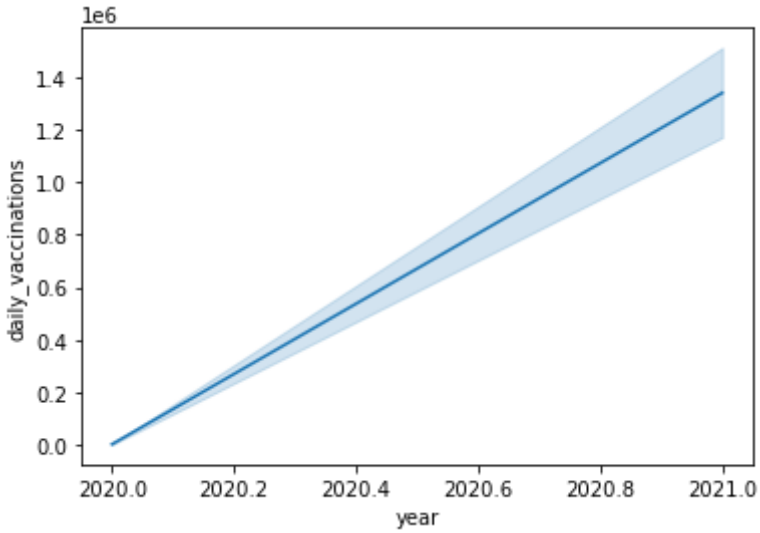
In [56]:
```
sns.lineplot(data=sel_data, x="year", y="people_fully_vaccinated")
```

Out[56]: <AxesSubplot:xlabel='year', ylabel='people_fully_vaccinated'>



In [57]:
```
sns.lineplot(data=sel_data, x="year", y="daily_vaccinations")
```

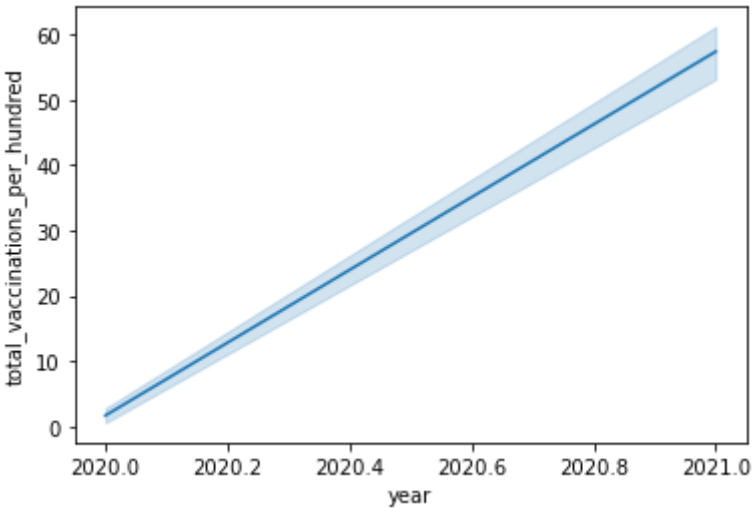
Out[57]: <AxesSubplot:xlabel='year', ylabel='daily_vaccinations'>



In [58]:

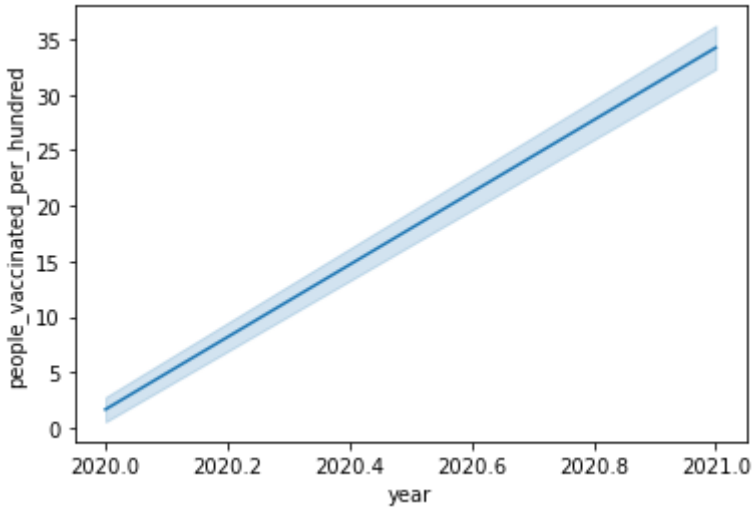
```
sns.lineplot(data=sel_data, x="year", y="total_vaccinations_per_hundred")
```

Out[58]: <AxesSubplot:xlabel='year', ylabel='total_vaccinations_per_hundred'>



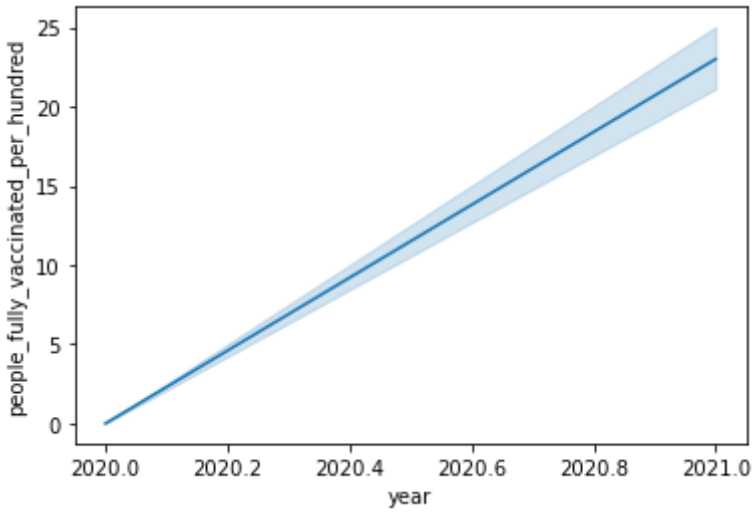
In [59]:
```
sns.lineplot(data=sel_data, x="year", y="people_vaccinated_per_hundred")
```

Out[59]: <AxesSubplot:xlabel='year', ylabel='people_vaccinated_per_hundred'>



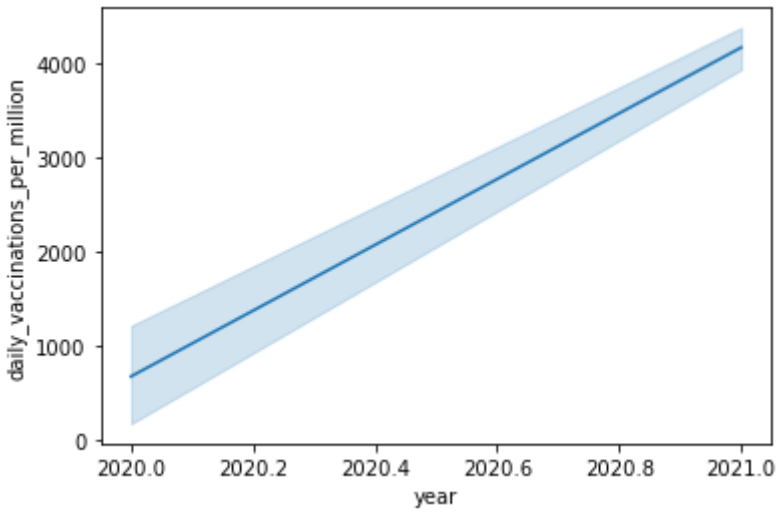
In [60]:
```
sns.lineplot(data=sel_data, x="year", y="people_fully_vaccinated_per_hundred")
```

Out[60]: <AxesSubplot:xlabel='year', ylabel='people_fully_vaccinated_per_hundred'>



In [61]:
```
sns.lineplot(data=sel_data, x="year", y="daily_vaccinations_per_million")
```

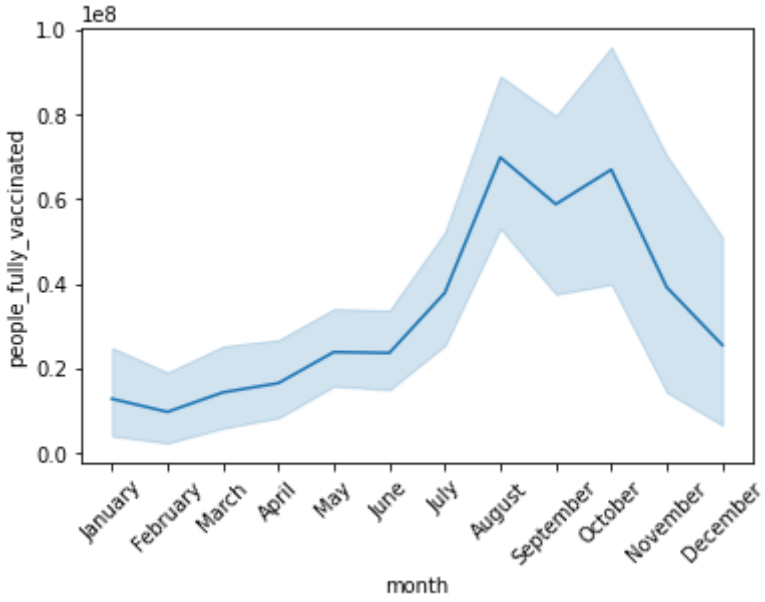
Out[61]:
```
<AxesSubplot:xlabel='year', ylabel='daily_vaccinations_per_million'>
```



In [62]:
```
#sel_data.set_index(['year'],inplace=True)
#sel_data.sort_index()
```

In [63]:
```
#sel_data['month'] = pd.Categorical(sel_data['month'],
                                    #categories=['January', 'February', 'March', 'Apr
                                    #ordered=True)

q2 = sns.lineplot(data=mon_data, x="month", y="people_fully_vaccinated")
q2.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

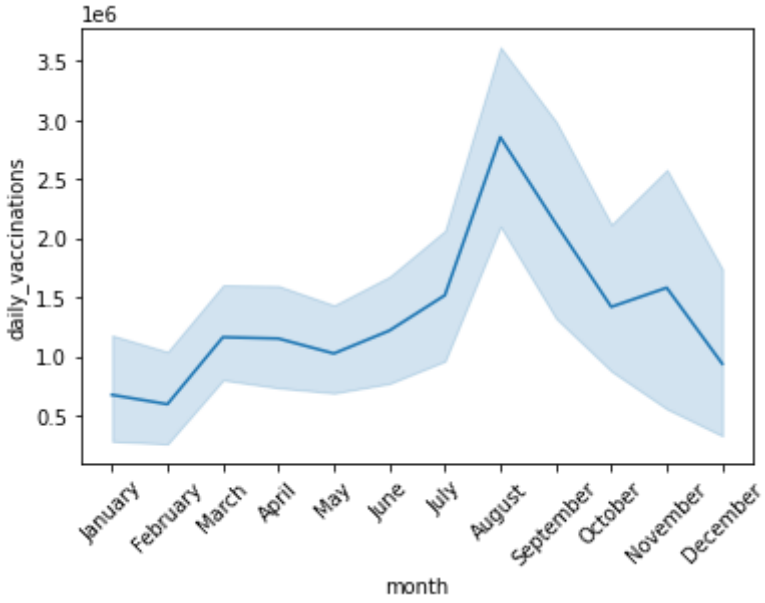
```
<ipython-input-63-b1b37ce66a44>:6: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  q2.set_xticklabels(chart.get_xticklabels(), rotation=45)
```



In [64]:
```
#sel_data['month'] = pd.Categorical(sel_data['month'],
                                    #categories=['January', 'February', 'March', 'Apr
                                    #ordered=True)

q3 = sns.lineplot(data=mon_data, x="month", y="daily_vaccinations")
```

```
q3.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

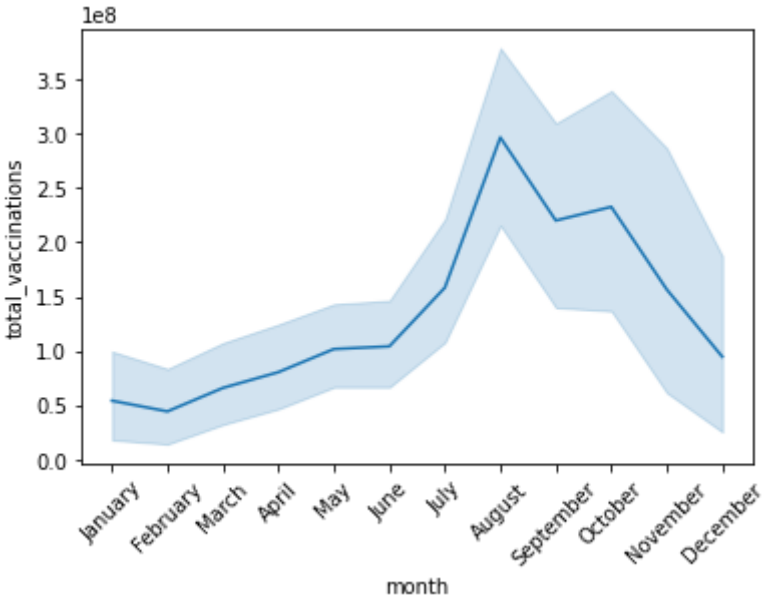
```
<ipython-input-64-da2632f6f991>:6: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  q3.set_xticklabels(chart.get_xticklabels(), rotation=45)
```



In [65]:

```
#sel_data['month'] = pd.Categorical(sel_data['month'],
                                    #categories=['January', 'February', 'March', 'Apr
                                    #ordered=True)

q2 = sns.lineplot(data=mon_data, x="month", y="total_vaccinations")
q2.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

```
<ipython-input-65-aa9eaeb95665>:6: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  q2.set_xticklabels(chart.get_xticklabels(), rotation=45)
```

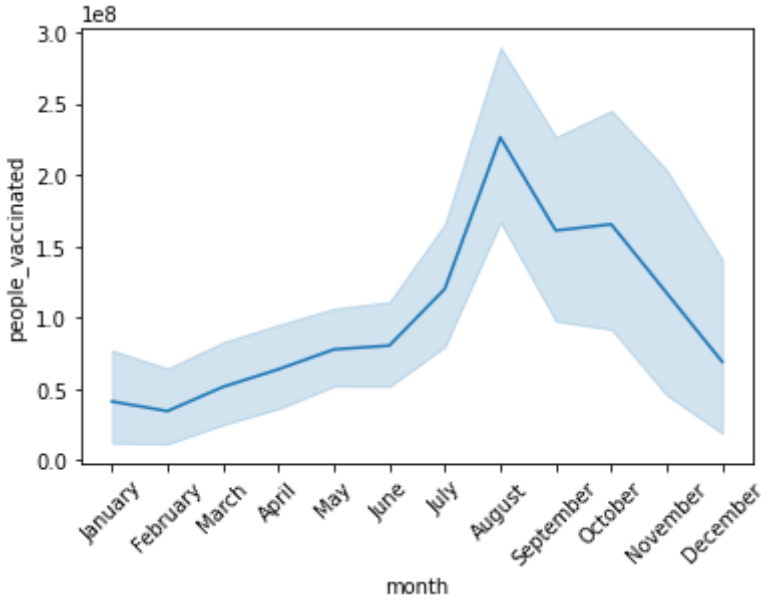


In [66]:

```
#sel_data['month'] = pd.Categorical(sel_data['month'],
                                    #categories=['January', 'February', 'March', 'Apr
                                    #ordered=True)

q1 = sns.lineplot(data=mon_data, x="month", y="people_vaccinated")
```

```
q1.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

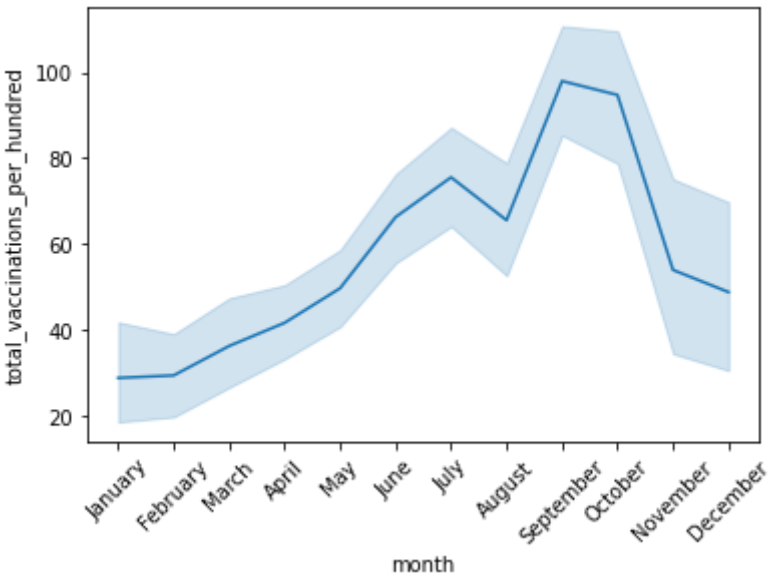
```
<ipython-input-66-d6e9fc05d1f5>:6: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  q1.set_xticklabels(chart.get_xticklabels(), rotation=45)
```



In [67]:

```
#sel_data['month'] = pd.Categorical(sel_data['month'],
                                    #categories=['January', 'February', 'March', 'Apr
                                    #ordered=True)

q4 = sns.lineplot(data=mon_data, x="month", y="total_vaccinations_per_hundred")
q4.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

```
<ipython-input-67-4377a4d79bfc>:6: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  q4.set_xticklabels(chart.get_xticklabels(), rotation=45)
```

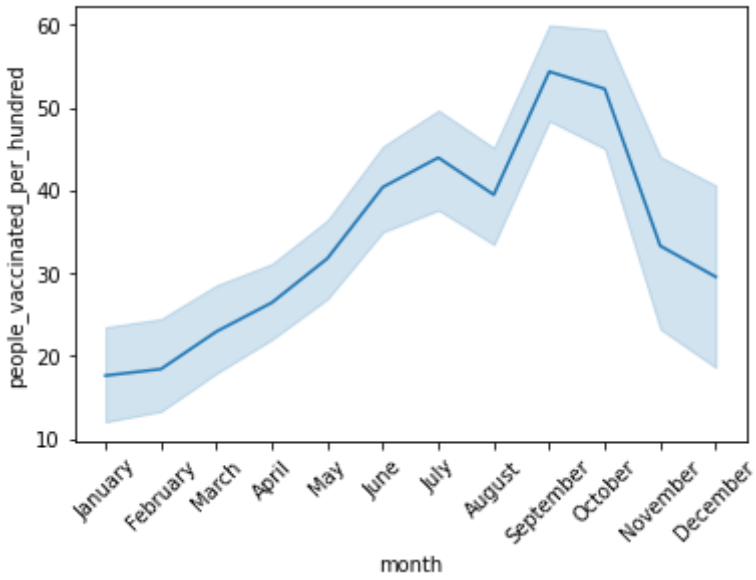


In [68]:

```
#sel_data['month'] = pd.Categorical(sel_data['month'],
                                    #categories=['January', 'February', 'March', 'Apr
                                    #ordered=True)

q5 = sns.lineplot(data=mon_data, x="month", y="people_vaccinated_per_hundred")
```

```
q5.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

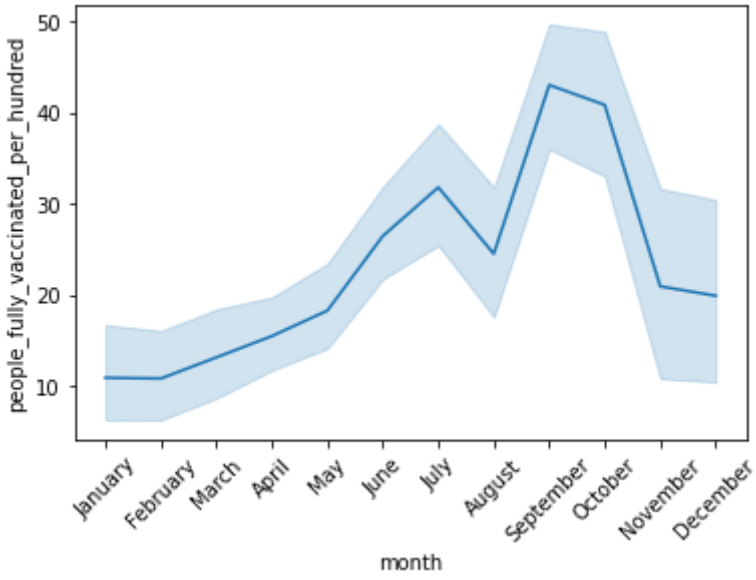
```
<ipython-input-68-6dea5ca31254>:6: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  q5.set_xticklabels(chart.get_xticklabels(), rotation=45)
```



In [69]:

```
#sel_data['month'] = pd.Categorical(sel_data['month'],
                                    #categories=['January', 'February', 'March', 'Apr
                                    #ordered=True)

q6 = sns.lineplot(data=mon_data, x="month", y="people_fully_vaccinated_per_hundred")
q6.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

```
<ipython-input-69-db4f4a123952>:6: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  q6.set_xticklabels(chart.get_xticklabels(), rotation=45)
```

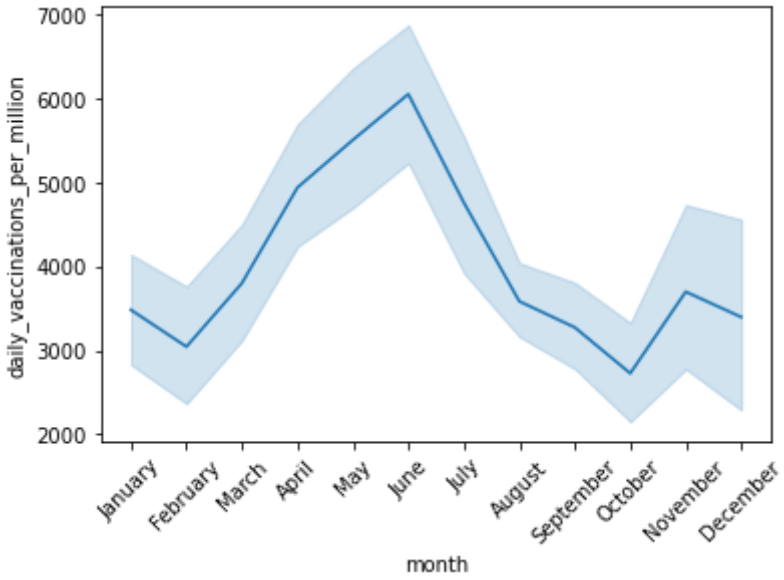


In [70]:

```
#sel_data['month'] = pd.Categorical(sel_data['month'],
                                    #categories=['January', 'February', 'March', 'Apr
                                    #ordered=True)

q7 = sns.lineplot(data=mon_data, x="month", y="daily_vaccinations_per_million")
```

```
q7.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

```
<ipython-input-70-7adc3d86f966>:6: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  q7.set_xticklabels(chart.get_xticklabels(), rotation=45)
```



In [71]:

```
sel_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 646 entries, 295 to 4939
Data columns (total 16 columns):
 #   Column                                Non-Null Count  Dtype
---  ------                                --------------  -----
 0   country                               646 non-null    object
 1   iso_code                              646 non-null    object
 2   date                                  646 non-null    datetime64[ns]
 3   total_vaccinations                    646 non-null    int64
 4   people_vaccinated                     646 non-null    int64
 5   people_fully_vaccinated               646 non-null    int64
 6   daily_vaccinations                    646 non-null    int64
 7   total_vaccinations_per_hundred        646 non-null    float64
 8   people_vaccinated_per_hundred         646 non-null    float64
 9   people_fully_vaccinated_per_hundred   646 non-null    float64
 10  daily_vaccinations_per_million        646 non-null    int64
 11  vaccines                              646 non-null    object
 12  source_name                           646 non-null    object
 13  year                                  646 non-null    int64
 14  month                                 646 non-null    object
 15  continent                             646 non-null    object
dtypes: datetime64[ns](1), float64(3), int64(6), object(6)
memory usage: 102.0+ KB
```

In [72]:

```
# country and vaccines

T1 = pd.crosstab(index=sel_data['country'],columns=sel_data['vaccines'],margins=True
T1
```

Out[72]:

vaccines	Covaxin, Oxford/AstraZeneca, Sputnik V	Johnson&Johnson, Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing, Sputnik V	All
country					

vaccines	Covaxin, Oxford/AstraZeneca, Sputnik V	Johnson&Johnson, Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing, Sputnik V	All
country					
Bahrain	0	0	0	93	93
India	219	0	0	0	219
Ireland	0	230	0	0	230
United Kingdom	0	0	104	0	104
All	219	230	104	93	646

In [73]:
```
T2 = pd.crosstab(index=sel_data['continent'],columns=sel_data['vaccines'],margins=Tr
T2
```

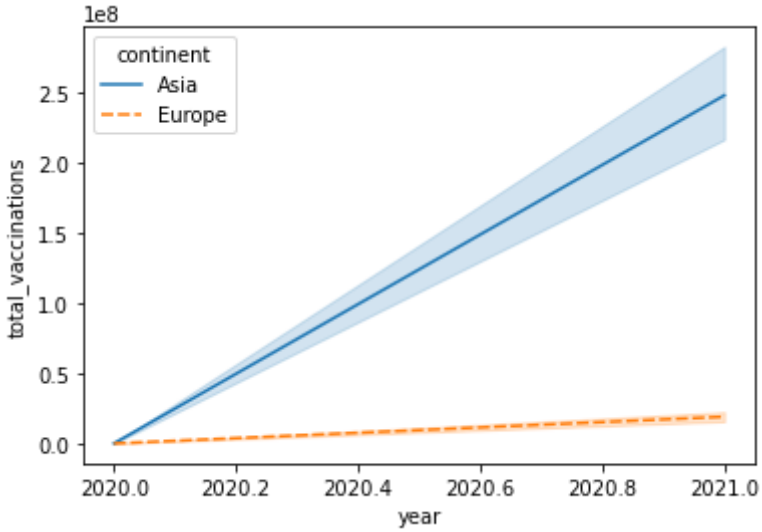
Out[73]:

vaccines	Covaxin, Oxford/AstraZeneca, Sputnik V	Johnson&Johnson, Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	Moderna, Oxford/AstraZeneca, Pfizer/BioNTech	Oxford/AstraZeneca, Pfizer/BioNTech, Sinopharm/Beijing, Sputnik V	All
continent					
Asia	219	0	0	93	312
Europe	0	230	104	0	334
All	219	230	104	93	646

Multivariate plots

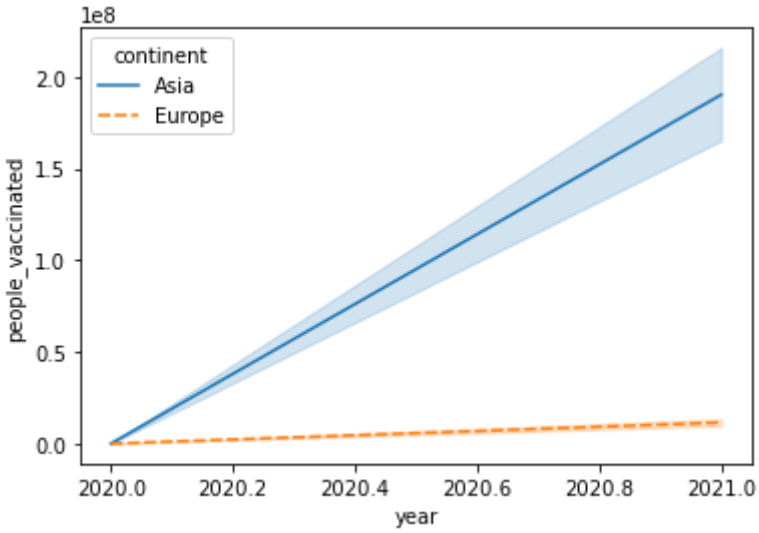
In [74]:
```
sns.lineplot(data=sel_data, x="year", y="total_vaccinations",hue='continent',style='
```

Out[74]: <AxesSubplot:xlabel='year', ylabel='total_vaccinations'>

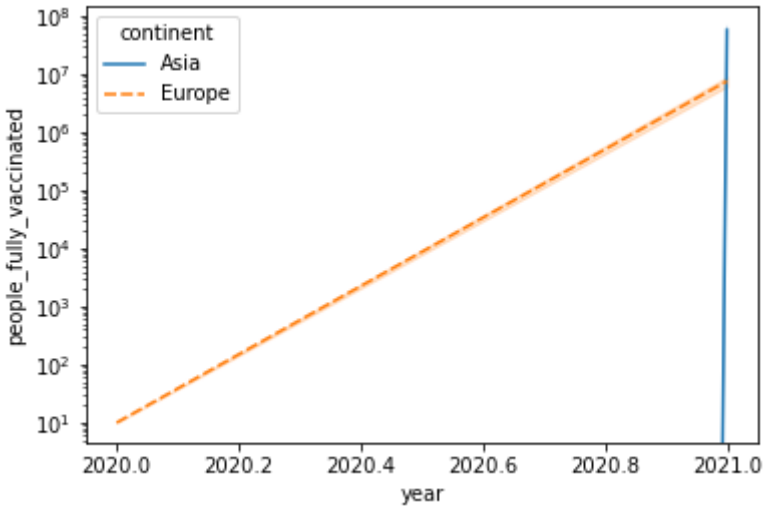


In [75]:
```
sns.lineplot(data=sel_data, x="year", y="people_vaccinated",hue='continent',style='c
```

Out[75]: <AxesSubplot:xlabel='year', ylabel='people_vaccinated'>

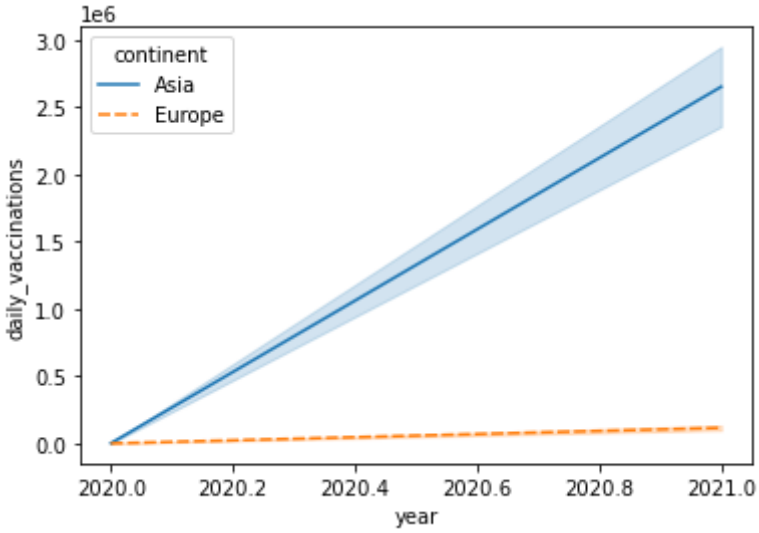


In [76]:
```
p1 = sns.lineplot(data=sel_data, x="year", y="people_fully_vaccinated",hue='continen
p1.set_yscale('log')
```



In [77]:
```
sns.lineplot(data=sel_data, x="year", y="daily_vaccinations",hue='continent',style='
```

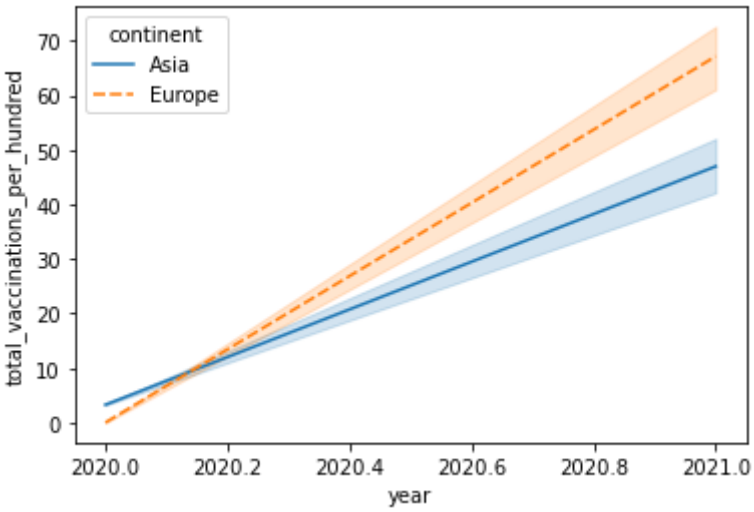
Out[77]: <AxesSubplot:xlabel='year', ylabel='daily_vaccinations'>



In [78]:

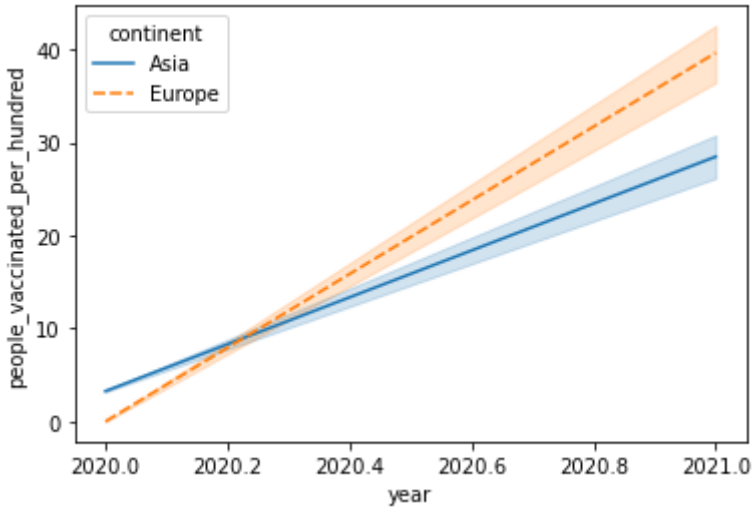
```
sns.lineplot(data=sel_data, x="year", y="total_vaccinations_per_hundred",hue='contin
```

Out[78]: <AxesSubplot:xlabel='year', ylabel='total_vaccinations_per_hundred'>



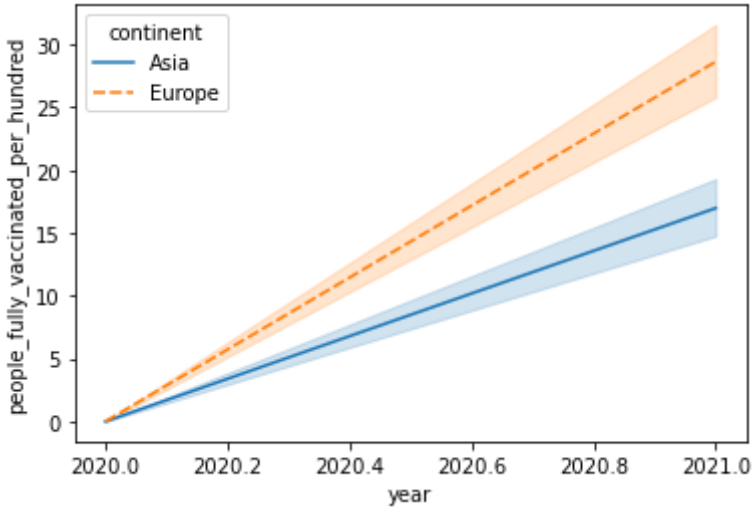
In [79]:
```
sns.lineplot(data=sel_data, x="year", y="people_vaccinated_per_hundred",hue='contine
```

Out[79]: <AxesSubplot:xlabel='year', ylabel='people_vaccinated_per_hundred'>

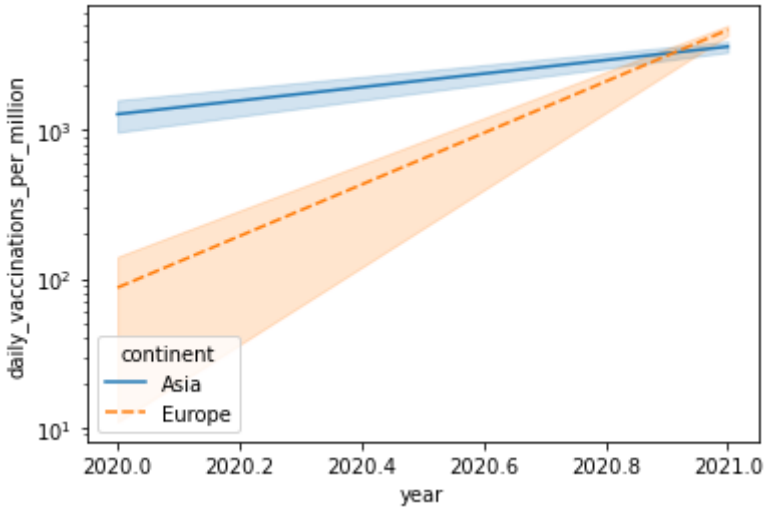


In [80]:
```
sns.lineplot(data=sel_data, x="year", y="people_fully_vaccinated_per_hundred",hue='c
```

Out[80]: <AxesSubplot:xlabel='year', ylabel='people_fully_vaccinated_per_hundred'>

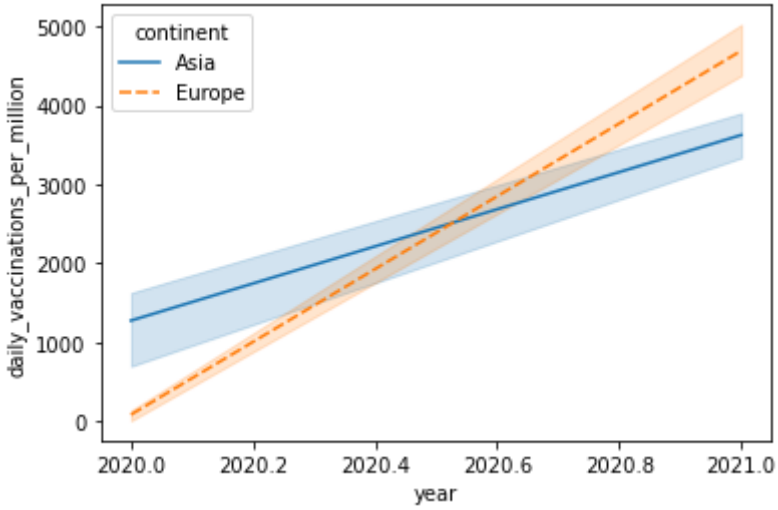


In [81]:
```
p4= sns.lineplot(data=sel_data, x="year", y="daily_vaccinations_per_million",hue='co
p4.set_yscale('log')
```



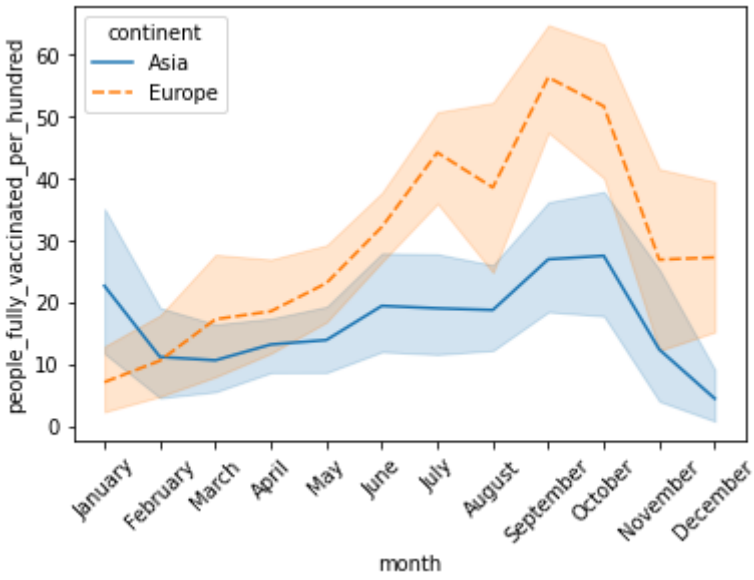
In [82]:
```
sns.lineplot(data=sel_data, x="year", y="daily_vaccinations_per_million",hue='contin
```

Out[82]: <AxesSubplot:xlabel='year', ylabel='daily_vaccinations_per_million'>



In [83]:
```
#sel_data['month'] = pd.Categorical(sel_data['month'],
                                    #categories=['January', 'February', 'March', 'Apr
                                    #ordered=True)

q2 = sns.lineplot(data=mon_data, x="month", y="people_fully_vaccinated_per_hundred",
q2.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

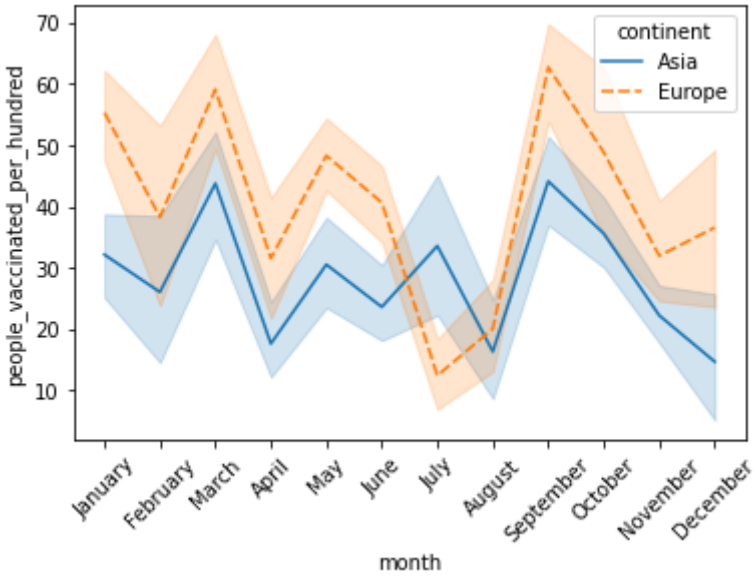
```
<ipython-input-83-261dc664a721>:6: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  q2.set_xticklabels(chart.get_xticklabels(), rotation=45)
```



In [84]:

```
#sel_data['month'] = pd.Categorical(sel_data['month'],
                                    #categories=['January', 'February', 'March', 'Apr
                                    #ordered=True)

q3 = sns.lineplot(data=sel_data, x="month", y="people_vaccinated_per_hundred",hue='c
q3.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

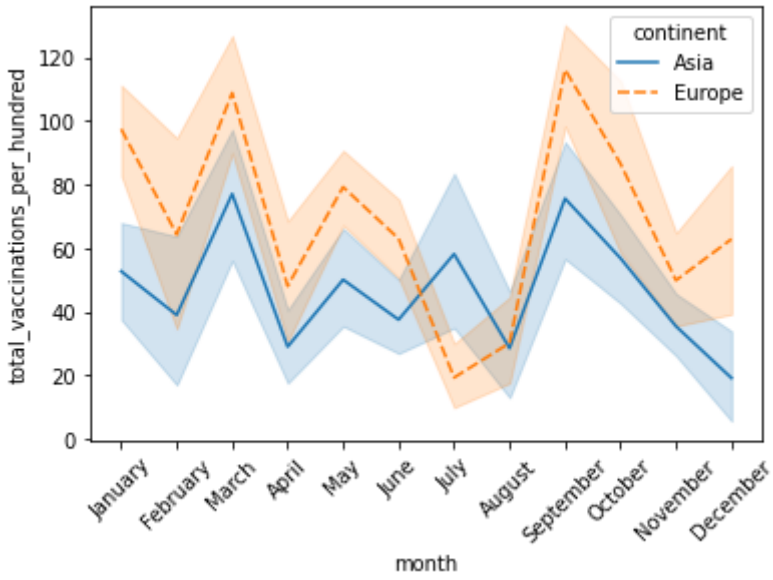
```
<ipython-input-84-3091c0934605>:6: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  q3.set_xticklabels(chart.get_xticklabels(), rotation=45)
```



In [85]:

```
#sel_data['month'] = pd.Categorical(sel_data['month'],
                                    #categories=['January', 'February', 'March', 'Apr
                                    #ordered=True)

q2 = sns.lineplot(data=sel_data, x="month", y="total_vaccinations_per_hundred",hue='
q2.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

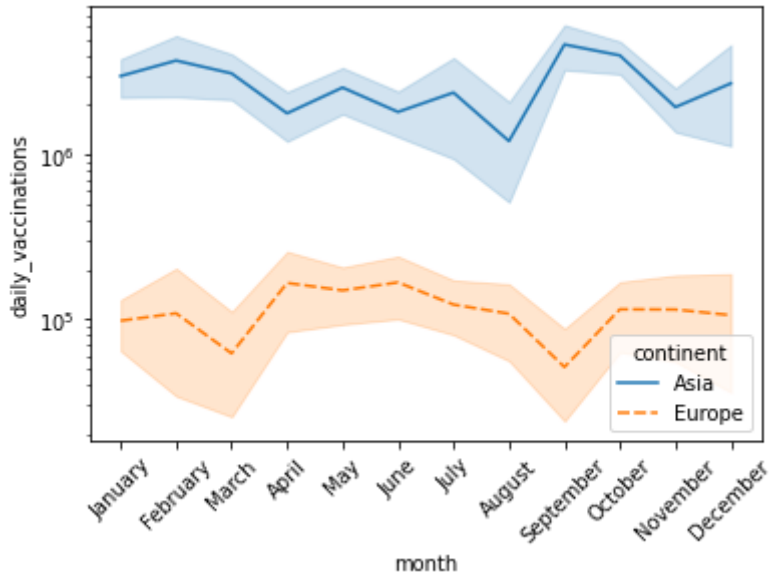
```
<ipython-input-85-5e55e4843816>:6: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  q2.set_xticklabels(chart.get_xticklabels(), rotation=45)
```



In [86]:

```
#sel_data['month'] = pd.Categorical(sel_data['month'],
                                    #categories=['January', 'February', 'March', 'Apr
                                    #ordered=True)

p3 = sns.lineplot(data=sel_data, x="month", y="daily_vaccinations",hue='continent',s
p3.set_yscale('log')
p3.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

```
<ipython-input-86-8c5e7dc1397e>:7: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  p3.set_xticklabels(chart.get_xticklabels(), rotation=45)
```

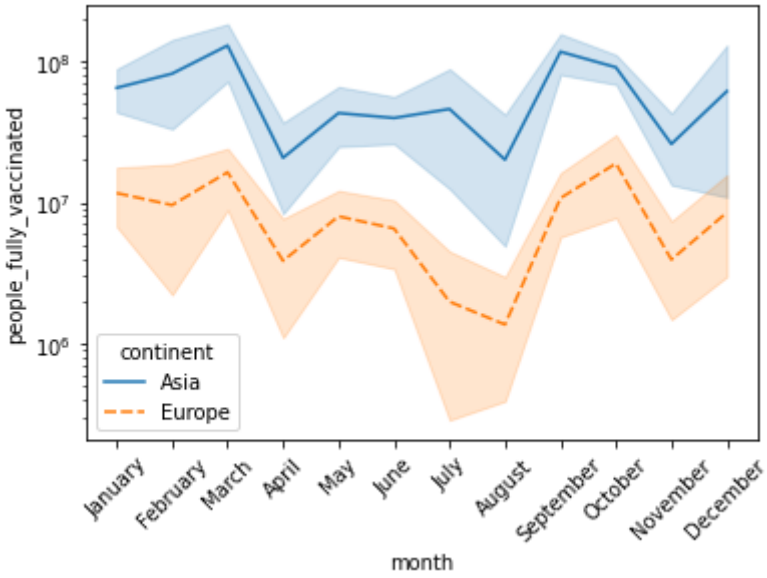


In [87]:

```
#sel_data['month'] = pd.Categorical(sel_data['month'],
                                    #categories=['January', 'February', 'March', 'Apr
                                    #ordered=True)

p2 = sns.lineplot(data=sel_data, x="month", y="people_fully_vaccinated",hue='contine
p2.set_yscale('log')
p2.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

```
<ipython-input-87-0c0b530d418d>:7: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
```

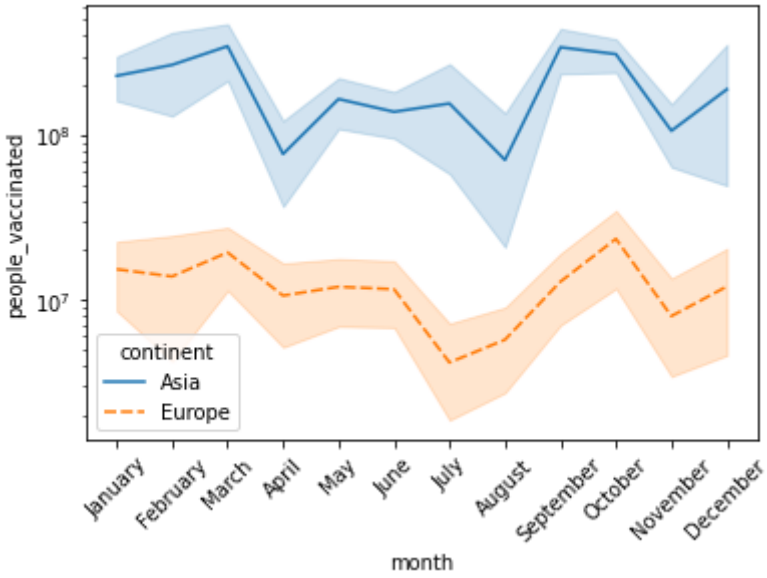
```
  p2.set_xticklabels(chart.get_xticklabels(), rotation=45)
```



In [88]:

```
#sel_data['month'] = pd.Categorical(sel_data['month'],
                                    #categories=['January', 'February', 'March', 'Apr
                                    #ordered=True)

L1 = sns.lineplot(data=sel_data, x="month", y="people_vaccinated",hue='continent',st
L1.set_yscale('log')
L1.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

```
<ipython-input-88-b271076d9a9e>:7: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  L1.set_xticklabels(chart.get_xticklabels(), rotation=45)
```

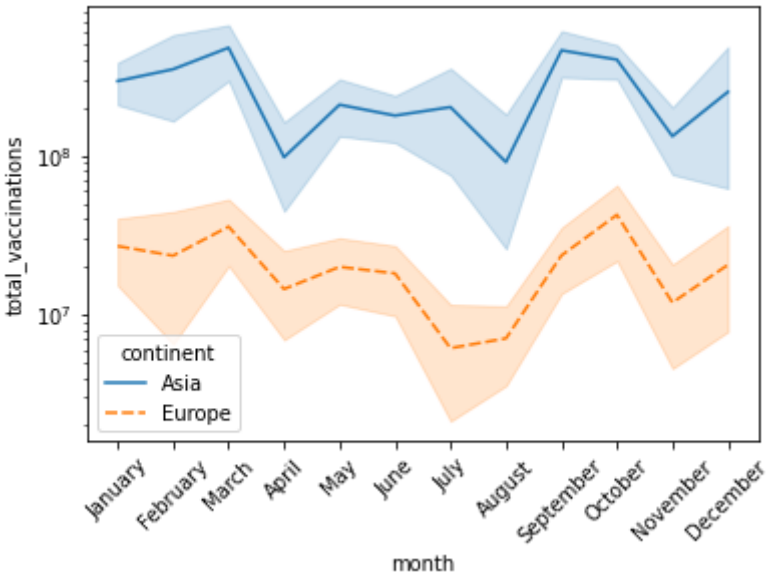


In [89]:

```
#sel_data['month'] = pd.Categorical(sel_data['month'],
                                    #categories=['January', 'February', 'March', 'Apr
                                    #ordered=True)

p =sns.lineplot(data=sel_data, x="month", y="total_vaccinations",hue='continent',sty
p.set_yscale('log')
p.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

```
<ipython-input-89-74ce950b1c75>:7: UserWarning: FixedFormatter should only be used t
```

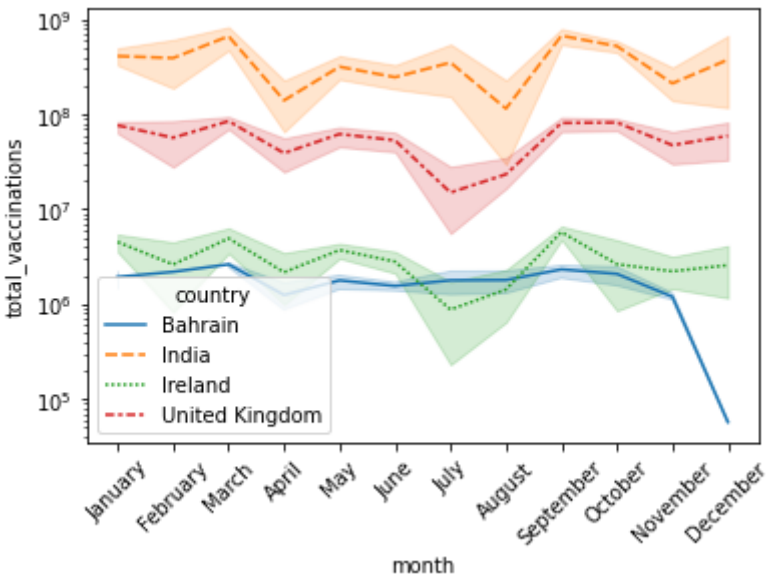
```
ogether with FixedLocator
  p.set_xticklabels(chart.get_xticklabels(), rotation=45)
```



In [90]:

```
#sel_data['month'] = pd.Categorical(sel_data['month'],
                                    #categories=['January', 'February', 'March', 'Apr
                                    #ordered=True)

p5 = sns.lineplot(data=sel_data, x="month", y="total_vaccinations",hue='country',sty
p5.set_yscale('log')
p5.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

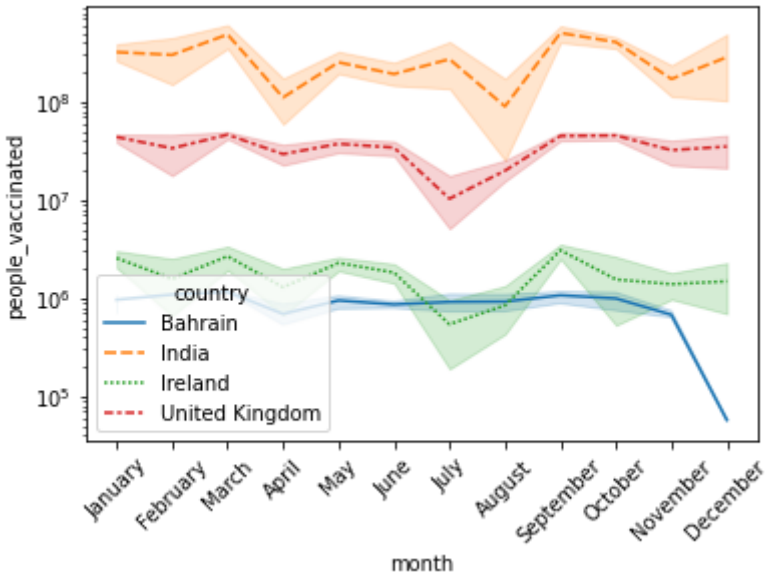
```
<ipython-input-90-b5e364933d36>:7: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  p5.set_xticklabels(chart.get_xticklabels(), rotation=45)
```



In [91]:

```
L3 =sns.lineplot(data=sel_data, x="month", y="people_vaccinated",hue='country',style
L3.set_yscale('log')
L3.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

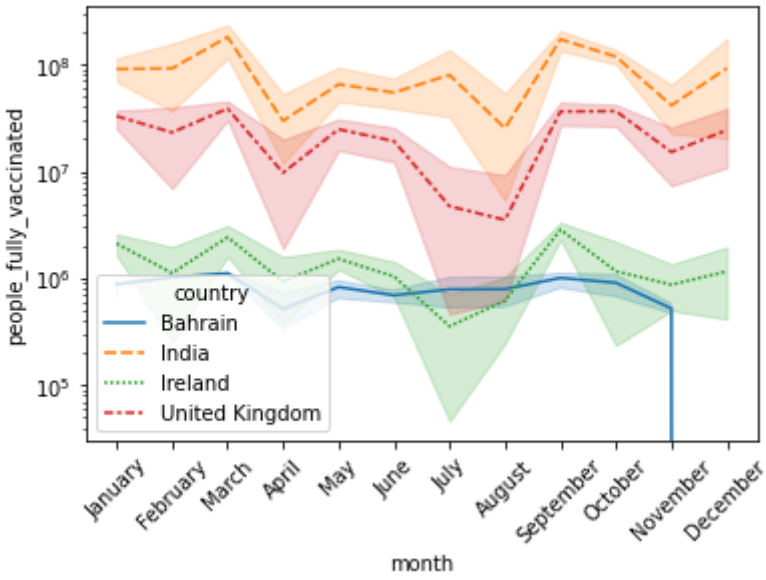
```
<ipython-input-91-f4644f1f4113>:3: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  L3.set_xticklabels(chart.get_xticklabels(), rotation=45)
```



In [92]:

```
L4 = sns.lineplot(data=sel_data, x="month", y="people_fully_vaccinated",hue='country
L4.set_yscale('log')
L4.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

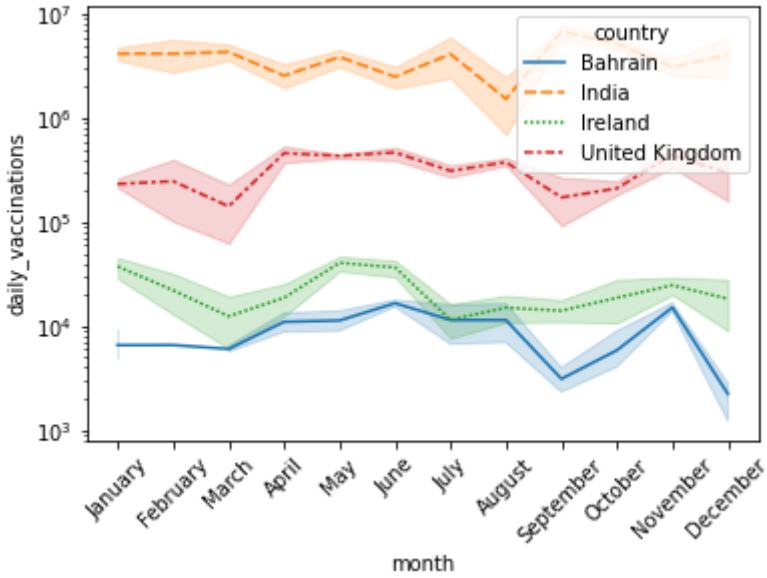
```
<ipython-input-92-04d9098185c2>:3: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  L4.set_xticklabels(chart.get_xticklabels(), rotation=45)
```



In [93]:

```
L5 = sns.lineplot(data=sel_data, x="month", y="daily_vaccinations",hue='country',sty
L5.set_yscale('log')
L5.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

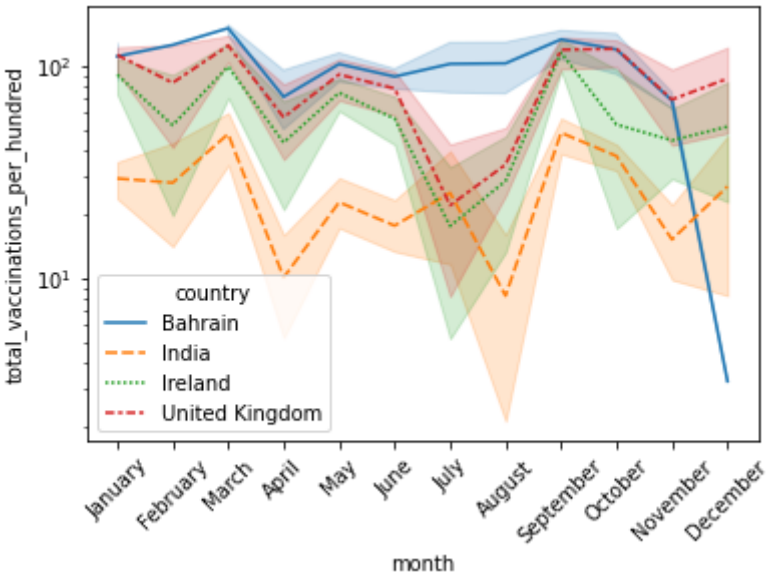
```
<ipython-input-93-6594d267d5be>:3: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  L5.set_xticklabels(chart.get_xticklabels(), rotation=45)
```



In [94]:

```
L6 = sns.lineplot(data=sel_data, x="month", y="total_vaccinations_per_hundred",hue='
L6.set_yscale('log')
L6.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

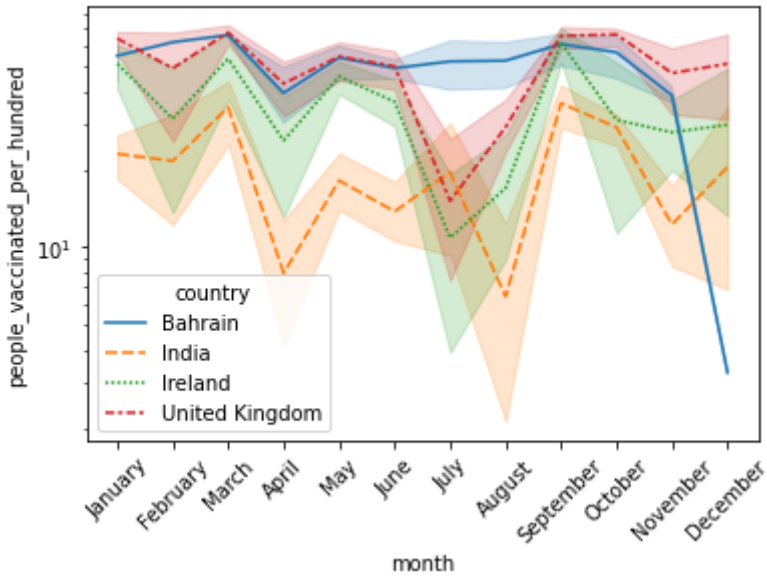
```
<ipython-input-94-cd27e668ad99>:3: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  L6.set_xticklabels(chart.get_xticklabels(), rotation=45)
```



In [95]:

```
L7 = sns.lineplot(data=sel_data, x="month", y="people_vaccinated_per_hundred",hue='c
L7.set_yscale('log')
L7.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

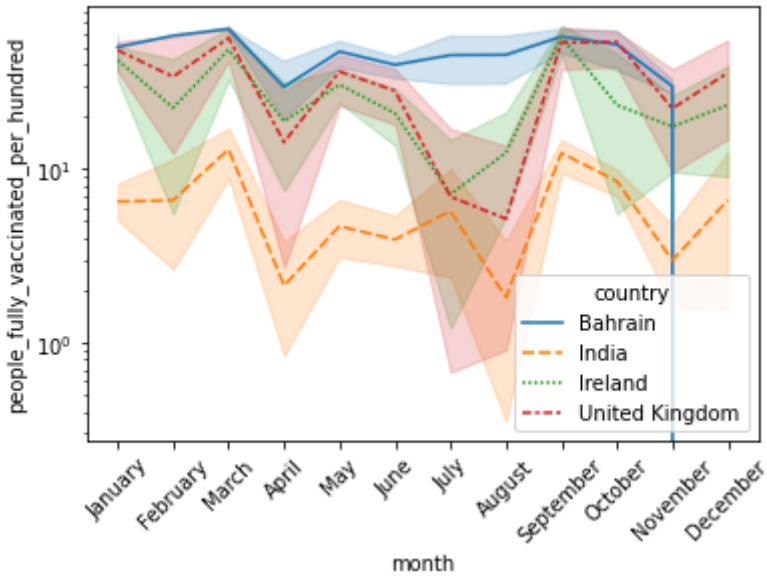
```
<ipython-input-95-69c2591d8b2d>:3: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  L7.set_xticklabels(chart.get_xticklabels(), rotation=45)
```



In [96]:

```
L8 = sns.lineplot(data=sel_data, x="month", y="people_fully_vaccinated_per_hundred",
L8.set_yscale('log')
L8.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

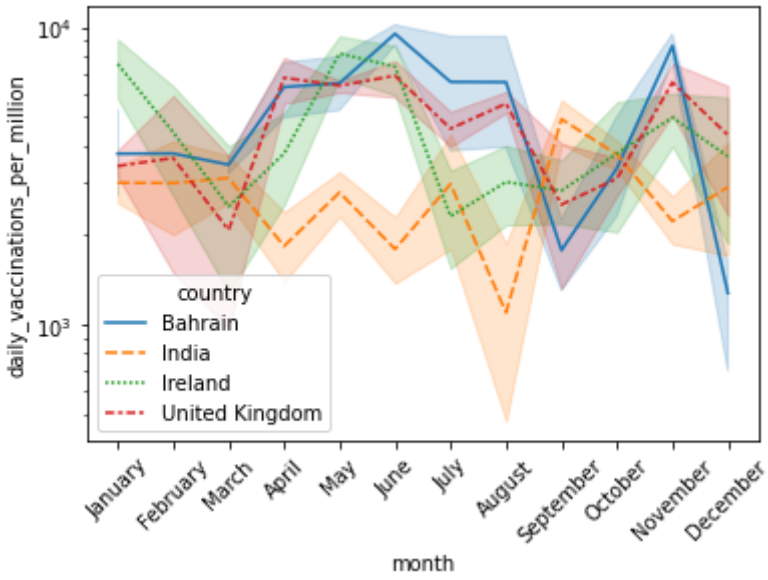
```
<ipython-input-96-c3f3acfe059d>:3: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  L8.set_xticklabels(chart.get_xticklabels(), rotation=45)
```



In [97]:

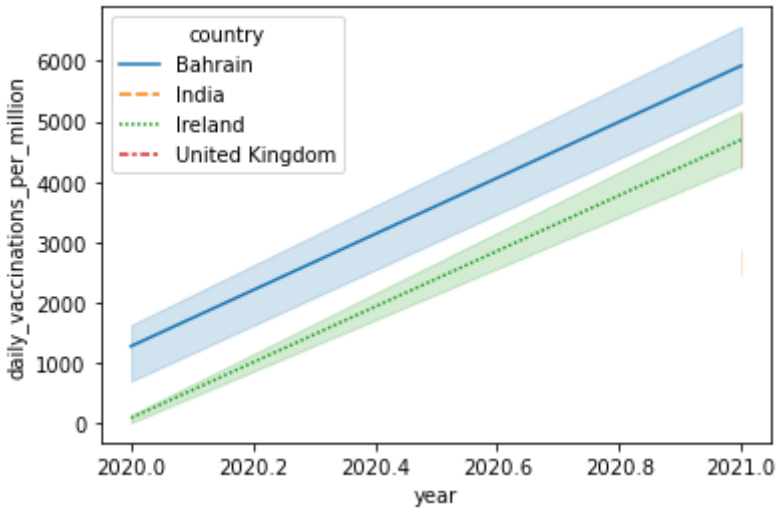
```
L9 = sns.lineplot(data=sel_data, x="month", y="daily_vaccinations_per_million",hue='
L9.set_yscale('log')
L9.set_xticklabels(chart.get_xticklabels(), rotation=45)
plt.show()
```

```
<ipython-input-97-c1abd03f2c4f>:3: UserWarning: FixedFormatter should only be used t
ogether with FixedLocator
  L9.set_xticklabels(chart.get_xticklabels(), rotation=45)
```



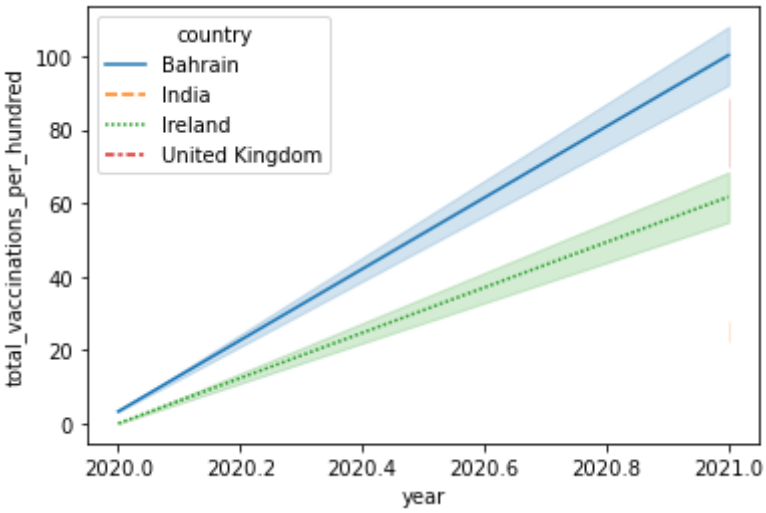
In [98]:
```
sns.lineplot(data=sel_data, x="year", y="daily_vaccinations_per_million",hue='countr
```

Out[98]: <AxesSubplot:xlabel='year', ylabel='daily_vaccinations_per_million'>



In [99]:
```
sns.lineplot(data=sel_data, x="year", y="total_vaccinations_per_hundred",hue='countr
```

Out[99]: <AxesSubplot:xlabel='year', ylabel='total_vaccinations_per_hundred'>

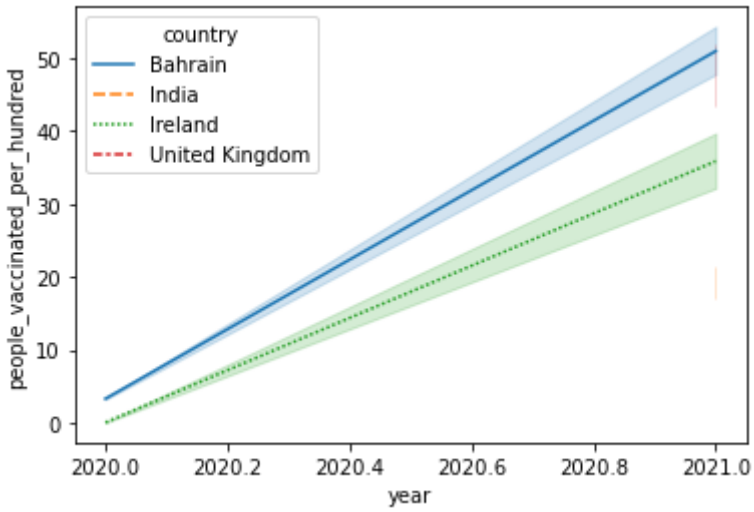


In [100…

```
sns.lineplot(data=sel_data, x="year", y="people_vaccinated_per_hundred",hue='country
```

Out[100…

```
<AxesSubplot:xlabel='year', ylabel='people_vaccinated_per_hundred'>
```

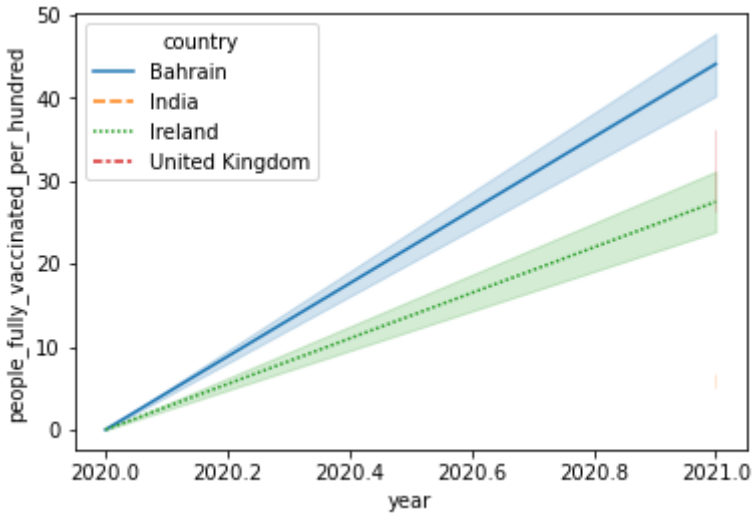


In [101…

```
sns.lineplot(data=sel_data, x="year", y="people_fully_vaccinated_per_hundred",hue='c
```

Out[101…

```
<AxesSubplot:xlabel='year', ylabel='people_fully_vaccinated_per_hundred'>
```



In [102…

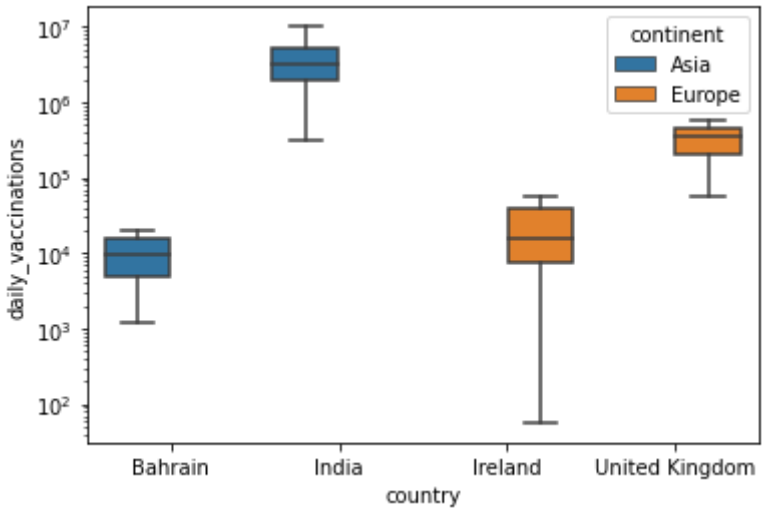
```
#total_vaccinations_per_hundred people_vaccinated_per_hundred   people_fully_vaccina
```

In [103…

```
#sns.histplot(sel_data['daily_vaccinations'] , color="yellow" ,label="male")
#sns.histplot(sel_data['people_fully_vaccinated'] , color="red",label="female")
#plt.legend()
#plt.xlabel("Time (days)")
#plt.xscale('log')
#plt.yscale('log')
#plt.show()
```

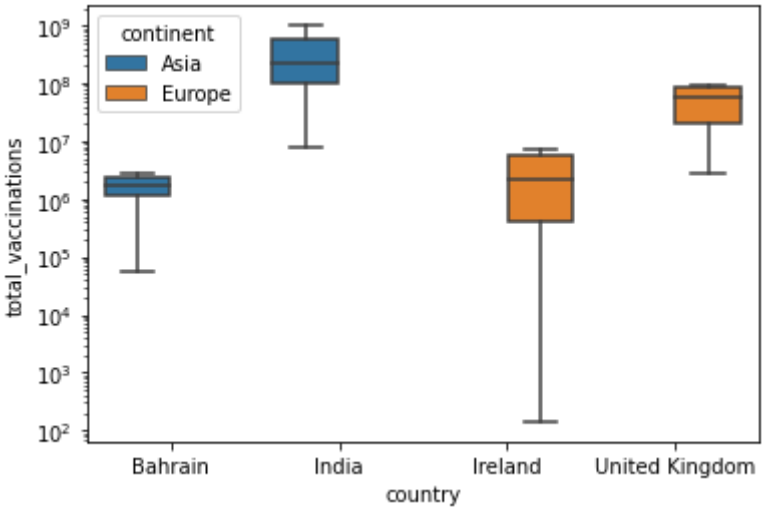
In [104…

```
sns.boxplot(x="country", y="daily_vaccinations", hue='continent', data=sel_data)
plt.ylabel("daily_vaccinations")
plt.yscale('log')
plt.show()
```



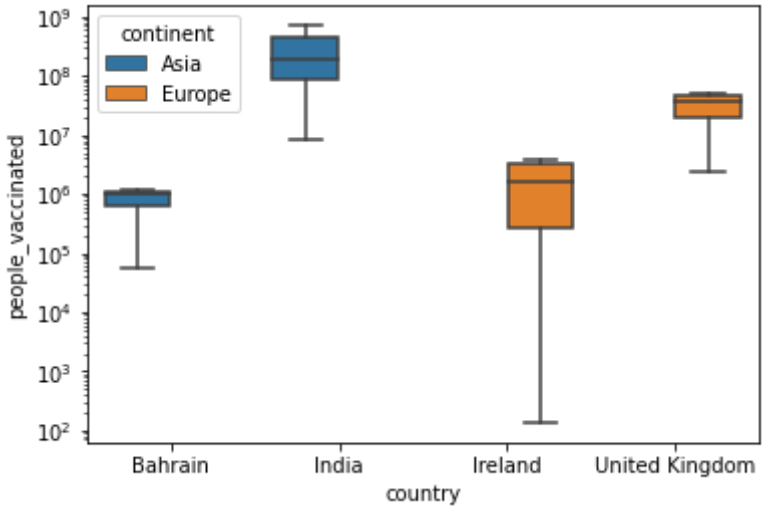
In [105…

```
sns.boxplot(x="country", y="total_vaccinations", hue='continent', data=sel_data)
plt.ylabel("total_vaccinations")
plt.yscale('log')
plt.show()
```



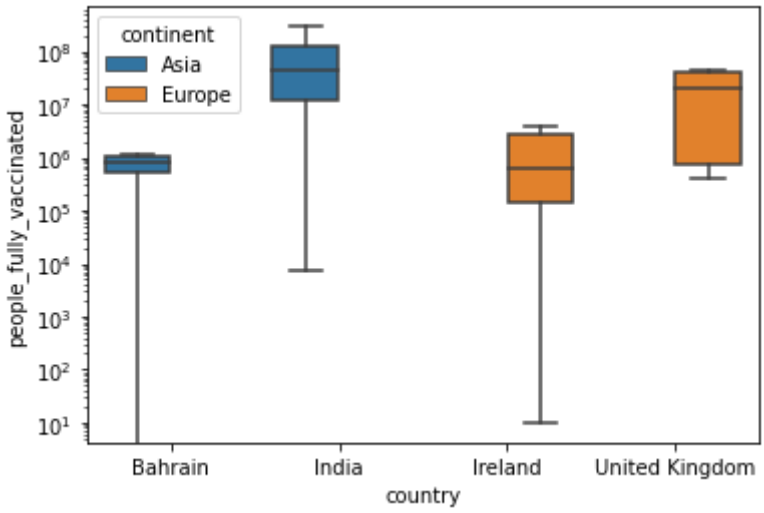
In [106…

```
sns.boxplot(x="country", y="people_vaccinated", hue='continent', data=sel_data)
plt.ylabel("people_vaccinated")
plt.yscale('log')
plt.show()
```



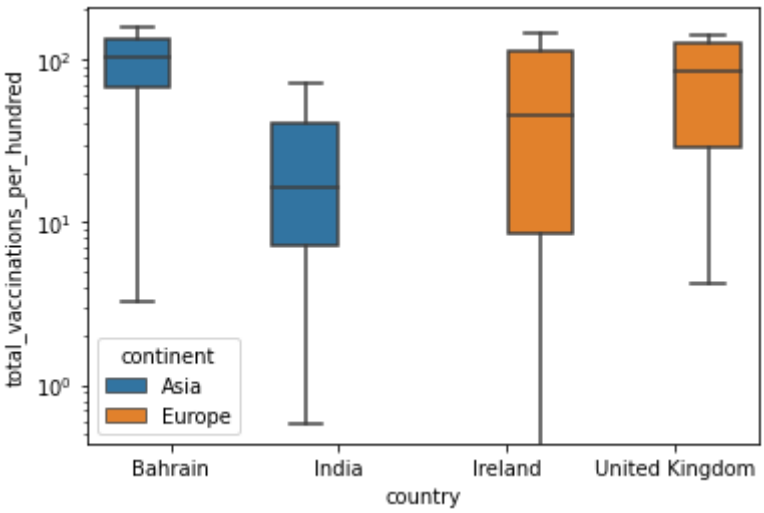
In [107…

```
sns.boxplot(x="country", y="people_fully_vaccinated", hue='continent', data=sel_data
plt.ylabel("people_fully_vaccinated")
plt.yscale('log')
plt.show()
```



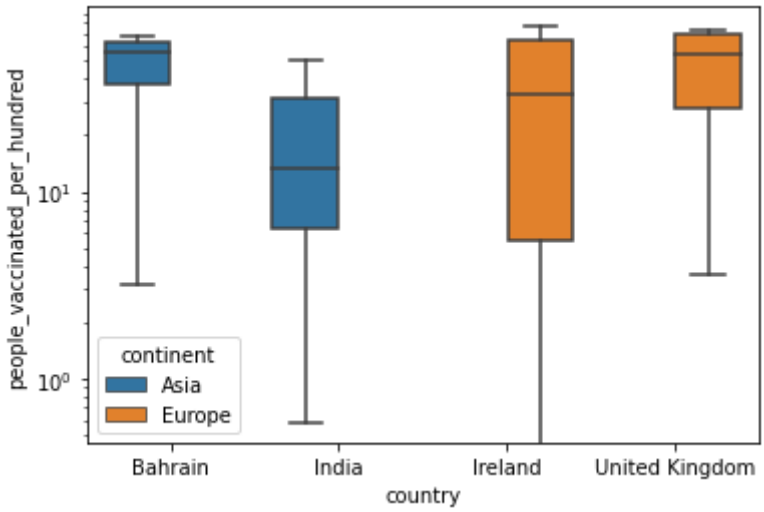
In [108…

```
sns.boxplot(x="country", y="total_vaccinations_per_hundred", hue='continent', data=s
plt.ylabel("total_vaccinations_per_hundred")
plt.yscale('log')
plt.show()
```



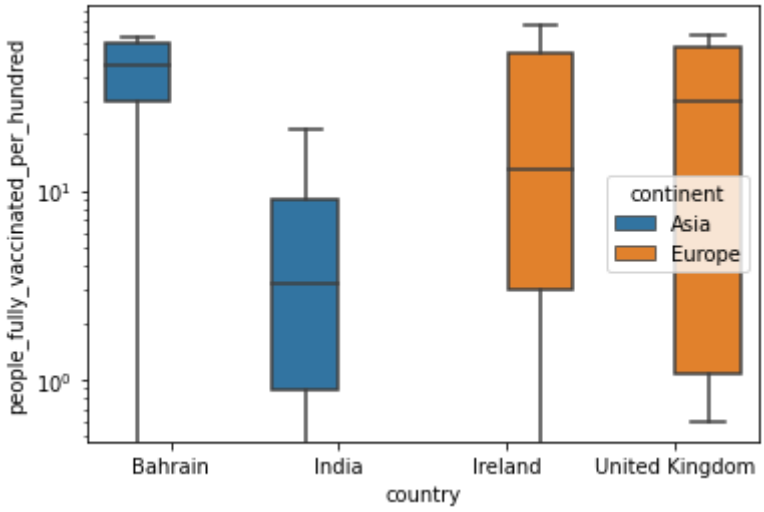
In [109…

```
sns.boxplot(x="country", y="people_vaccinated_per_hundred", hue='continent', data=se
plt.ylabel("people_vaccinated_per_hundred")
plt.yscale('log')
plt.show()
```



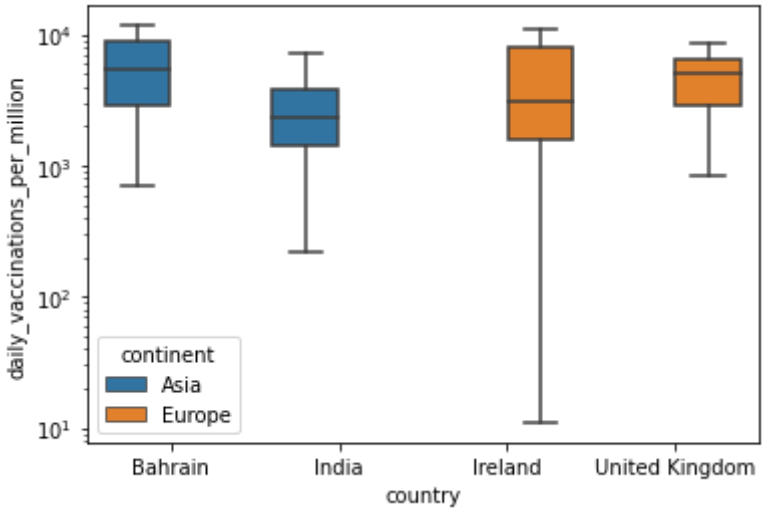
In [110…

```
sns.boxplot(x="country", y="people_fully_vaccinated_per_hundred", hue='continent', d
plt.ylabel("people_fully_vaccinated_per_hundred")
plt.yscale('log')
plt.show()
```



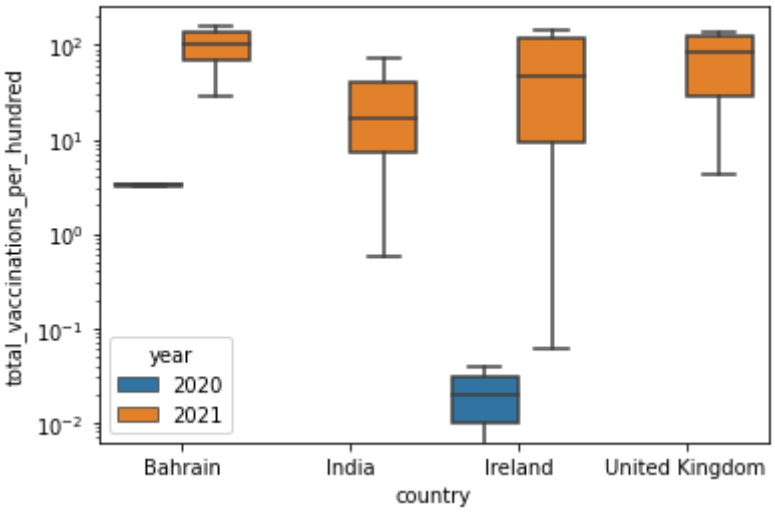
In [111…

```
sns.boxplot(x="country", y="daily_vaccinations_per_million", hue='continent', data=s
plt.ylabel("daily_vaccinations_per_million")
plt.yscale('log')
plt.show()
```



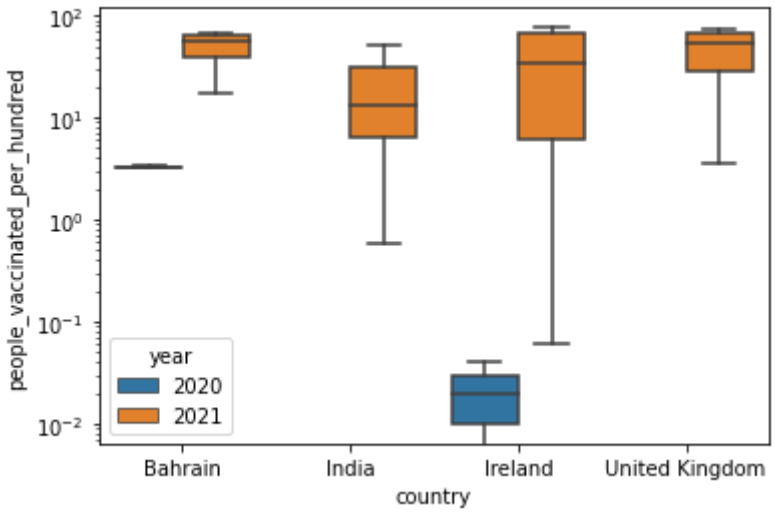
In [112…

```
sns.boxplot(x="country", y="total_vaccinations_per_hundred", hue='year', data=sel_da
plt.ylabel("total_vaccinations_per_hundred")
plt.yscale('log')
plt.show()
```



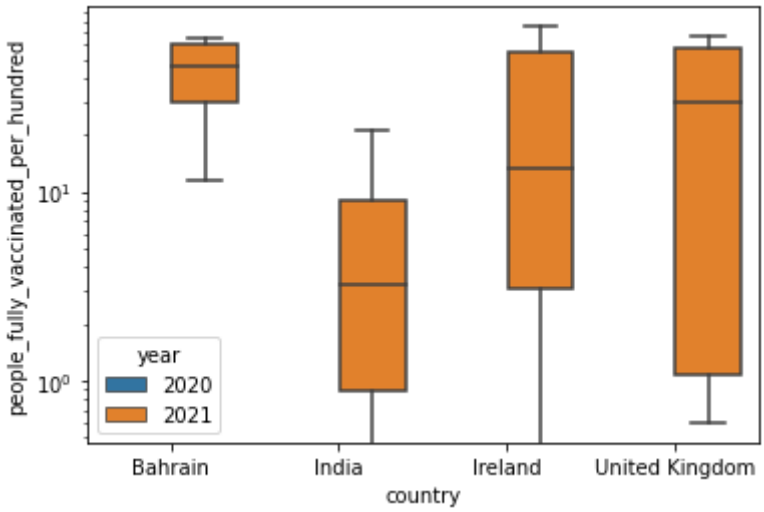
In [113…

```
sns.boxplot(x="country", y="people_vaccinated_per_hundred", hue='year', data=sel_dat
plt.ylabel("people_vaccinated_per_hundred")
plt.yscale('log')
plt.show()
```



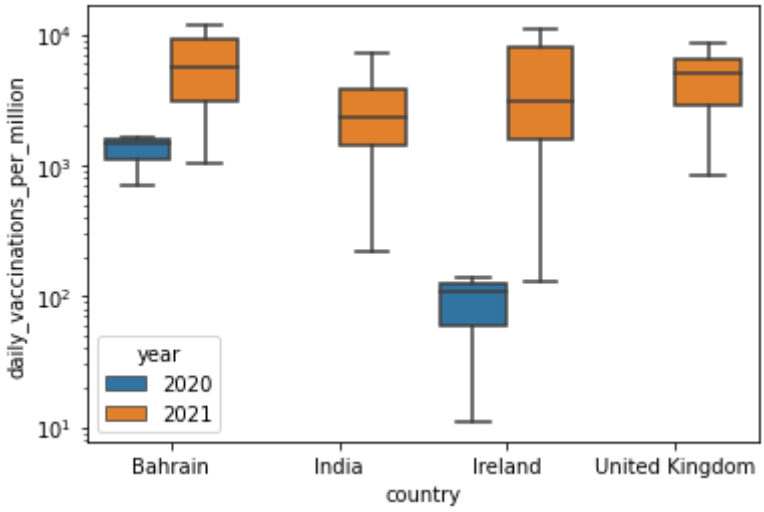
In [114…

```
sns.boxplot(x="country", y="people_fully_vaccinated_per_hundred", hue='year', data=s
plt.ylabel("people_fully_vaccinated_per_hundred")
plt.yscale('log')
plt.show()
```



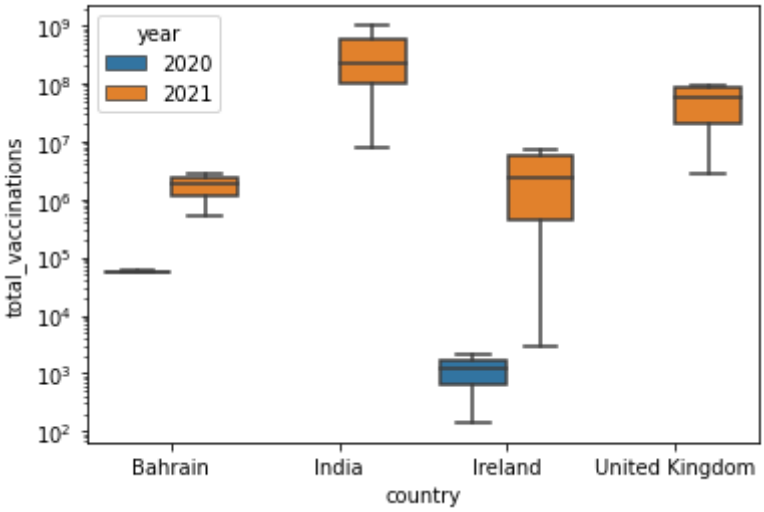
In [115…

```
sns.boxplot(x="country", y="daily_vaccinations_per_million", hue='year', data=sel_da
plt.ylabel("daily_vaccinations_per_million")
plt.yscale('log')
plt.show()
```



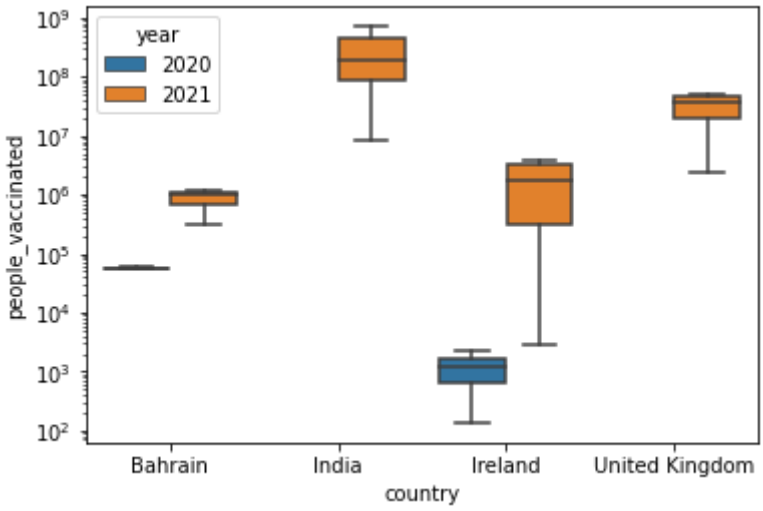
In [116…

```
sns.boxplot(x="country", y="total_vaccinations", hue='year', data=sel_data)
plt.ylabel("total_vaccinations")
plt.yscale('log')
plt.show()
```



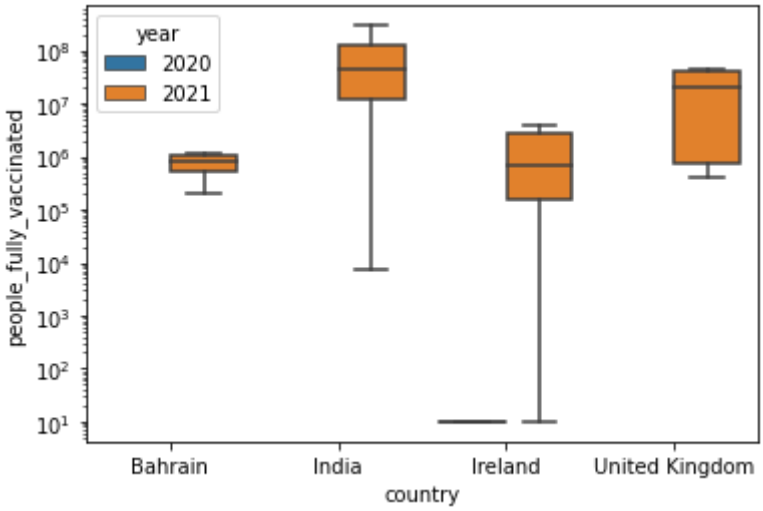
In [117…

```
sns.boxplot(x="country", y="people_vaccinated", hue='year', data=sel_data)
plt.ylabel("people_vaccinated")
plt.yscale('log')
plt.show()
```



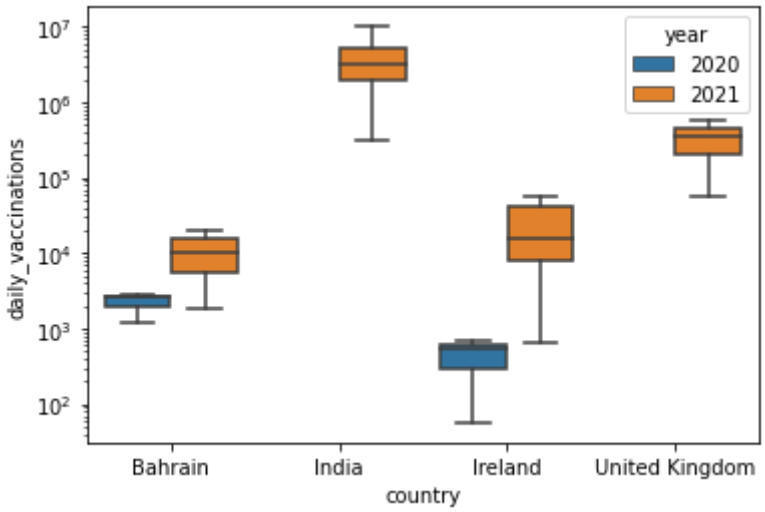
In [118…

```
sns.boxplot(x="country", y="people_fully_vaccinated", hue='year', data=sel_data)
plt.ylabel("people_fully_vaccinated")
plt.yscale('log')
plt.show()
```



In [119…

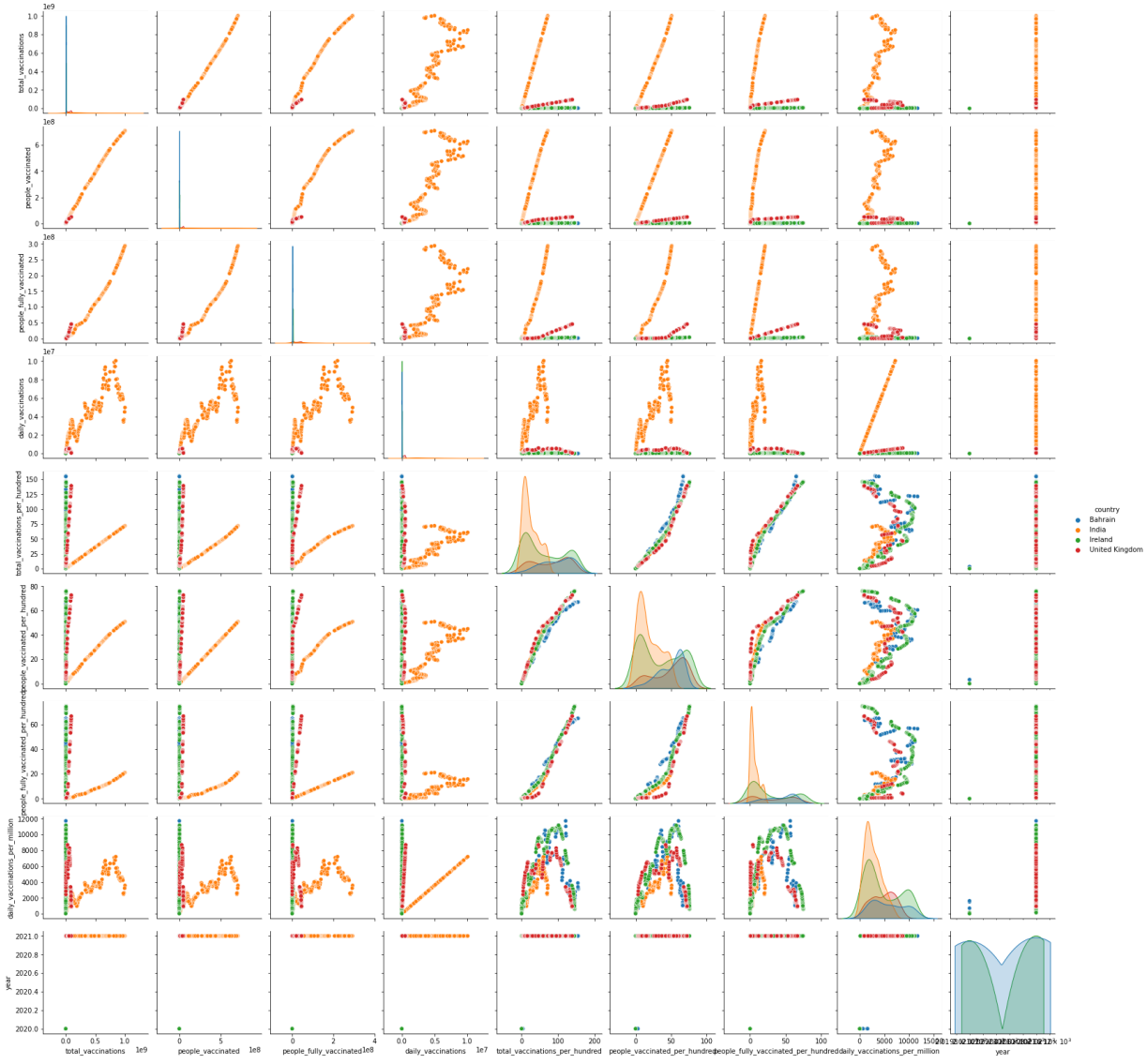
```
sns.boxplot(x="country", y="daily_vaccinations", hue='year', data=sel_data)
plt.ylabel("daily_vaccinations")
plt.yscale('log')
plt.show()
```



In [120…

```
sns.pairplot(sel_data,hue="country")
plt.yscale('log')
plt.xscale('log')
plt.show()
```

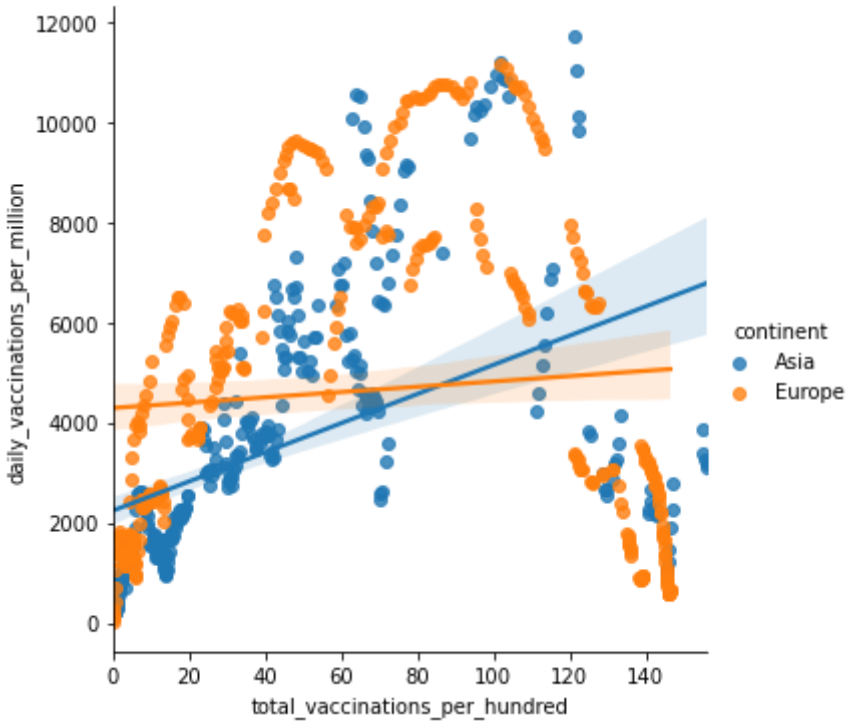
```
C:\Users\Chikku\anaconda3\lib\site-packages\seaborn\distributions.py:306: UserWarnin
g: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
C:\Users\Chikku\anaconda3\lib\site-packages\seaborn\distributions.py:306: UserWarnin
g: Dataset has 0 variance; skipping density estimate.
  warnings.warn(msg, UserWarning)
```



In [257…

```
sns.lmplot(x='total_vaccinations_per_hundred', y='daily_vaccinations_per_million', h
plt.xlabel("total_vaccinations_per_hundred")
plt.ylabel("daily_vaccinations_per_million")
```

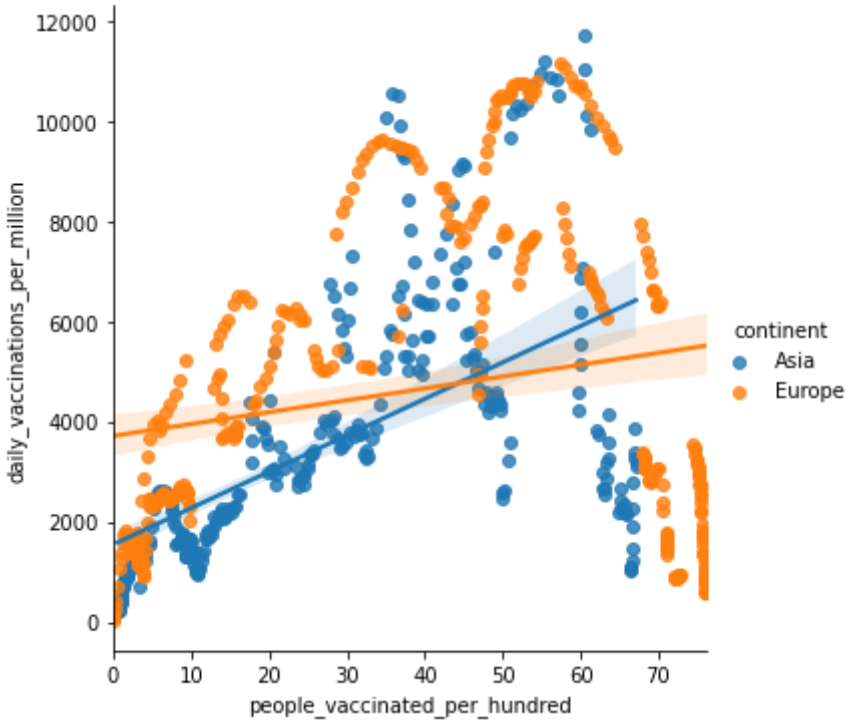
Out[257… Text(16.579765625, 0.5, 'daily_vaccinations_per_million')



In [256…

```
sns.lmplot(x='people_vaccinated_per_hundred', y='daily_vaccinations_per_million', hu
plt.xlabel("people_vaccinated_per_hundred")
plt.ylabel("daily_vaccinations_per_million")
```

Out[256… Text(16.579765625, 0.5, 'daily_vaccinations_per_million')

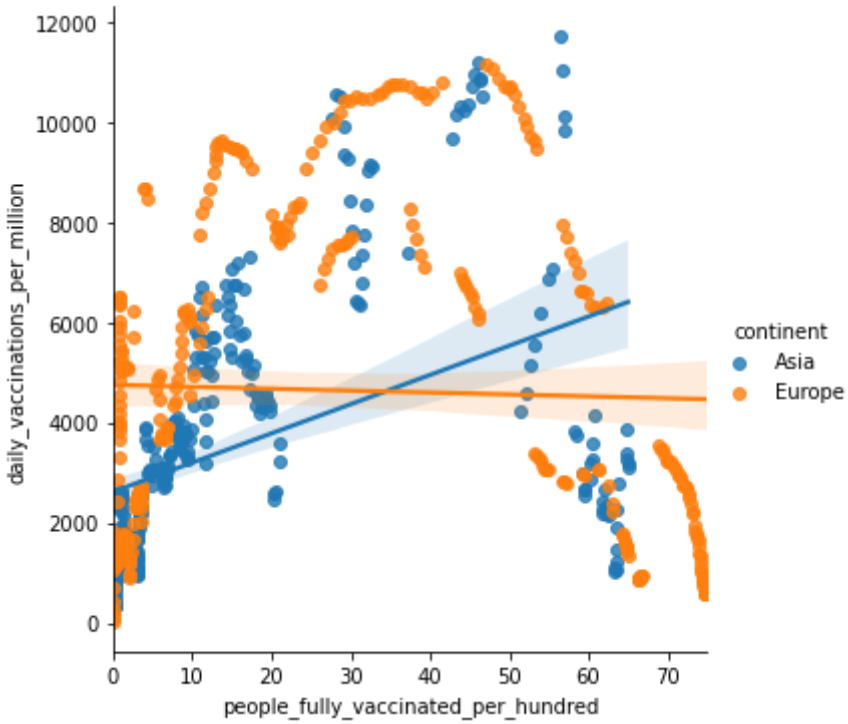


In [254…

```
sns.lmplot(x='people_fully_vaccinated_per_hundred', y='daily_vaccinations_per_millio
plt.xlabel("people_fully_vaccinated_per_hundred")
```

```
plt.ylabel("daily_vaccinations_per_million")
```

Out[254… Text(16.579765625, 0.5, 'daily_vaccinations_per_million')



3. Descriptive Statistics - generate and analyse basic statistics for all variables - including measures of central tendency and spread and correlation. (10 marks)

In [124…
```
sel_data['country'].describe()
```

Out[124…
```
count         646
unique          4
top       Ireland
freq          230
Name: country, dtype: object
```

In [125…
```
sel_data['iso_code'].describe()
```

Out[125…
```
count     646
unique      4
top       IRL
freq      230
Name: iso_code, dtype: object
```

In [126…
```
sel_data['date'].describe()
```

```
<ipython-input-126-045948d072e9>:1: FutureWarning: Treating datetime data as categor
ical rather than numeric in `.describe` is deprecated and will be removed in a futur
e version of pandas. Specify `datetime_is_numeric=True` to silence this warning and
adopt the future behavior now.
  sel_data['date'].describe()
```

Out[126…
```
count                    646
unique                   295
top       2021-04-28 00:00:00
freq                       4
first     2020-12-29 00:00:00
last      2021-12-10 00:00:00
Name: date, dtype: object
```

In [127…
```
sel_data['total_vaccinations'].describe()
```

Out[127…
```
count    6.460000e+02
mean     1.284127e+08
std      2.377074e+08
min      1.410000e+02
25%      1.982450e+06
50%      7.243817e+06
75%      1.075320e+08
max      1.003204e+09
Name: total_vaccinations, dtype: float64
```

In [128…
```
sel_data['people_vaccinated'].describe()
```

Out[128…
```
count    6.460000e+02
mean     9.690832e+07
std      1.780241e+08
min      1.360000e+02
25%      1.078896e+06
50%      3.791142e+06
75%      9.417913e+07
max      7.095520e+08
Name: people_vaccinated, dtype: float64
```

In [129…
```
sel_data['people_fully_vaccinated'].describe()
```

Out[129…
```
count    6.460000e+02
mean     3.152000e+07
std      6.076850e+07
min      0.000000e+00
25%      5.360668e+05
50%      2.917430e+06
75%      3.859196e+07
max      2.936523e+08
Name: people_fully_vaccinated, dtype: float64
```

In [130…
```
sel_data['daily_vaccinations'].describe()
```

Out[130…
```
count    6.460000e+02
mean     1.327012e+06
std      2.200616e+06
min      5.700000e+01
25%      1.277475e+04
50%      5.686950e+04
75%      2.011813e+06
max      1.003800e+07
Name: daily_vaccinations, dtype: float64
```

In [131…
```
sel_data['total_vaccinations_per_hundred'].describe()
```

Out[131…
```
count    646.000000
mean      56.885201
std       50.081623
min        0.000000
25%       11.267500
50%       42.440000
75%      101.717500
max      156.140000
Name: total_vaccinations_per_hundred, dtype: float64
```

In [132…
```
sel_data['people_vaccinated_per_hundred'].describe()
```

Out[132…
```
count    646.000000
mean      33.944381
std       25.453184
min        0.000000
25%        9.075000
50%       31.790000
75%       57.845000
max       76.220000
Name: people_vaccinated_per_hundred, dtype: float64
```

In [133…
```
sel_data['people_fully_vaccinated_per_hundred'].describe()
```

Out[133…
```
count    646.000000
mean      22.779396
std       25.124573
min        0.000000
25%        2.272500
50%       10.430000
75%       44.720000
max       74.970000
Name: people_fully_vaccinated_per_hundred, dtype: float64
```

In [134…
```
sel_data['daily_vaccinations_per_million'].describe()
```

Out[134…
```
count      646.000000
mean      4142.394737
std       3009.103827
min         11.000000
25%       1652.500000
50%       3227.500000
75%       6215.500000
max      11723.000000
Name: daily_vaccinations_per_million, dtype: float64
```

In [135…
```
sel_data['vaccines'].describe()
```

Out[135…
```
count                                                   646
unique                                                    4
top       Johnson&Johnson, Moderna, Oxford/AstraZeneca, ...
freq                                                    230
Name: vaccines, dtype: object
```

In [136…
```
sel_data['source_name'].describe()
```

Out[136…
```
count                          646
unique                           4
top       Heath Service Executive
freq                           230
Name: source_name, dtype: object
```

In [137…
```
sel_data['year'].describe()
```

Out[137…
```
count     646.000000
mean     2020.990712
std         0.096000
min      2020.000000
25%      2021.000000
50%      2021.000000
75%      2021.000000
max      2021.000000
Name: year, dtype: float64
```

In [138…

```
sel_data['month'].describe()
```

Out[138…
```
count     646
unique     12
top       May
freq       69
Name: month, dtype: object
```

In [139…
```
sel_data['continent'].describe()
```

Out[139…
```
count        646
unique         2
top       Europe
freq         334
Name: continent, dtype: object
```

The top is the most common value. The freq is the most common value's frequency

Correletion and Spread

From Data Visualization, the variables that show a linear relationship were daily_vaccinations and total_vaccinations, daily_vaccinations and people_vaccinated, and daily_vaccinations and people_fully_vaccinated.

So, the coefficient to be used to measure the strength of the correlation between variables that shows linear relationship is Pearson coefficient.

But for those with outliers Kendalltau will be more appropriate.

In [140…
```
corr_d1= kendalltau(sel_data["daily_vaccinations"], sel_data["total_vaccinations"])
corr_d1
```

Out[140…
```
KendalltauResult(correlation=0.7100637444815846, pvalue=1.5813370089538293e-160)
```

In [141…
```
corr_d2= kendalltau(sel_data["daily_vaccinations"], sel_data["people_vaccinated"])
corr_d2
```

Out[141…
```
KendalltauResult(correlation=0.731222279898405, pvalue=4.106103591132174e-170)
```

In [142…
```
corr_d3= kendalltau(sel_data["daily_vaccinations"], sel_data["people_fully_vaccinate
corr_d3
```

Out[142…
```
KendalltauResult(correlation=0.6051656984778233, pvalue=3.9256708757503784e-117)
```

In [143…
```
corr_d4 = spearmanr(sel_data["total_vaccinations_per_hundred"], sel_data["daily_vacc
corr_d4
```

Out[143…
```
SpearmanrResult(correlation=0.41707910285019856, pvalue=1.4098452514826318e-28)
```

In [261…
```
corr_d5= spearmanr(sel_data["daily_vaccinations_per_million"], sel_data["people_vacc
corr_d5
```

Out[261…
```
SpearmanrResult(correlation=0.4153347144694074, pvalue=2.49209972646122e-28)
```

In [145…

```
corr_d6= spearmanr(sel_data["people_fully_vaccinated_per_hundred"], sel_data["daily_
corr_d6
```

Out[145... SpearmanrResult(correlation=0.39037342213798276, pvalue=6.040720386196199e-25)

In [146...
```
corr_d7= spearmanr(sel_data["total_vaccinations_per_hundred"], sel_data["year"])
corr_d7
```

Out[146... SpearmanrResult(correlation=0.150095400856671, pvalue=0.00012854535937752598)

In [147...
```
corr_d8= spearmanr(sel_data["year"], sel_data["daily_vaccinations_per_million"])
corr_d8
```

Out[147... SpearmanrResult(correlation=0.138629511884896, pvalue=0.0004097339718941793)

In [148...
```
corr_d9= spearmanr(sel_data["people_fully_vaccinated_per_hundred"], sel_data["year"]
corr_d9
```

Out[148... SpearmanrResult(correlation=0.160960943767872, pvalue=3.9570424695439016e-05)

In [262...
```
corr_d10= spearmanr(sel_data["people_vaccinated_per_hundred"], sel_data["year"])
corr_d10
```

Out[262... SpearmanrResult(correlation=0.14659073931734956, pvalue=0.00018488028448843745)

The Kendalltau coefficient is greater than 0.7 for all the groups. Hence it has a strong positive relationship.

In [150...
```
#total_vaccinations     people_vaccinated       people_fully_vaccinated daily_vaccin
```

In [151...
```
#total_vaccinations_per_hundred people_vaccinated_per_hundred    people_fully_vaccina
```

Hypothesis Testing

Inorder to do the Hypothesis Testing with 'daily_vaccinations_per_million' discrete variable and others, I am converting 'daily_vaccinations' into catogories.

In [152...
```
desc = sel_data.sort_values(['daily_vaccinations_per_million'], ascending =False )
desc
```

Out[152...

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_
285	Bahrain	BHR	2021-06-30	2122296	1057888	987140	
274	Bahrain	BHR	2021-02-06	1775070	969225	805845	
1607	Ireland	IRL	2021-12-07	5071466	2858652	2343709	

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_
1608	Ireland	IRL	2021-07-13	5136618	2888501	2388286	
286	Bahrain	BHR	2021-01-07	2128340	1059186	991125	
...	...	...	...	...	...	...	
1453	Ireland	IRL	2021-04-01	5128	5122	11	
1449	Ireland	IRL	2020-12-31	2187	2182	10	
1452	Ireland	IRL	2021-03-01	3974	3969	10	
1448	Ireland	IRL	2020-12-30	1176	1171	10	
1447	Ireland	IRL	2020-12-29	141	136	10	

646 rows × 16 columns

In [153…

```
#add a new column category next to the daily_vaccination_group group.
category = pd.cut(desc.daily_vaccinations_per_million,bins=[0,6000, 12000],labels=['
desc.insert(3,'daily_vaccn_group',category)
desc
```

Out[153…

	country	iso_code	date	daily_vaccn_group	total_vaccinations	people_vaccinated	people_full
285	Bahrain	BHR	2021-06-30	Group B	2122296	1057888	
274	Bahrain	BHR	2021-02-06	Group B	1775070	969225	
1607	Ireland	IRL	2021-12-07	Group B	5071466	2858652	

	country	iso_code	date	daily_vaccn_group	total_vaccinations	people_vaccinated	people_full
1608	Ireland	IRL	2021-07-13	Group B	5136618	2888501	
286	Bahrain	BHR	2021-01-07	Group B	2128340	1059186	
...	...	...	...	...	...	...	
1453	Ireland	IRL	2021-04-01	Group A	5128	5122	
1449	Ireland	IRL	2020-12-31	Group A	2187	2182	
1452	Ireland	IRL	2021-03-01	Group A	3974	3969	
1448	Ireland	IRL	2020-12-30	Group A	1176	1171	
1447	Ireland	IRL	2020-12-29	Group A	141	136	

646 rows × 17 columns

In [154...

```
# daily_vaccinations_per_million and country

#Since India is a developing country and all othres are developed countries , I assu

dailyvacc_country = pd.crosstab(index=desc['country'], columns=desc["daily_vaccn_gro
dailyvacc_country
```

Out[154...

daily_vaccn_group	Group A	Group B
country		
Bahrain	49	44
India	210	9
Ireland	146	84
United Kingdom	67	37

In [155...

```
chi2_contingency(dailyvacc_country)
```

Out[155...

```
(92.28592455630574,
```

```
7.071404158680109e-20,
3,
array([[ 67.9504644,  25.0495356],
       [160.0123839,  58.9876161],
       [168.0495356,  61.9504644],
       [ 75.9876161,  28.0123839]]))
```

Assumptions :

1. Random sample

2. Independent observations in each cell

3. Expected values should be greater than or equal to 5

The assumptions are roughly satisfied since only 1 out of 8 value have value less than 5.

Null Hypothesis, H0: There is no relationship between daily vaccinations per population and country (i.e. country and daily_vaccinations are independent)

Alternative hypothesis, H1: There is relationship between daily vaccinations and country (i.e. country and daily_vaccinations are dependent)

Here p-value, -0.78 < 0.05, so we will reject null hypothesis.

Hence there is a relationship between daily vaccinations per population and country.

In [156…

```
# daily_vaccinations and year

dailyvacc_year = pd.crosstab(index=desc['year'], columns=desc["daily_vaccn_group"])
dailyvacc_year
```

Out[156…

daily_vaccn_group	Group A	Group B
year		
2020	6	0
2021	466	174

In [157…

```
chi2_contingency(dailyvacc_year)
```

Out[157…

```
(1.064830730435012,
0.3021159874898026,
1,
array([[  4.38390093,   1.61609907],
       [467.61609907, 172.38390093]]))
```

Assumptions :

1. Random sample

2. Independent observations in each cell

3. Expected values should be greater than or equal to 5

The third assumption is not met since only one value is greater than 5.

Hence Chi-squared test cannot be performed.

So, I am performing the Fisher Exact test, since the table is 2x2 and the expected values are lesser than 5.

In [158…
```
fisher_exact(dailyvacc_year)
```

Out[158… (inf, 0.19878131014557648)

Null Hypothesis, H0: There is no relationship between daily vaccinations and year (i.e. year and daily_vaccinations are independent)

Alternative hypothesis, H1: There is relationship between daily vaccinations and year (i.e. year and daily_vaccinations are dependent)

Here p-value, 0.19 > 0.05, so we failed reject the null hypothesis.

Hence there is not enough evidence to suggest there is relationship between daily vaccinations and year

In [159…
```
# daily_vaccinations and month

dailyvacc_month = pd.crosstab(index=desc['month'], columns=desc["daily_vaccn_group"]
dailyvacc_month
```

Out[159…

daily_vaccn_group	Group A	Group B
month		
April	38	27
August	45	3
December	21	7
February	53	10
January	49	8
July	42	15
June	31	34
March	46	17
May	34	35
November	18	4
October	43	4
September	52	10

In [160…
```
chi2_contingency(dailyvacc_month)
```

Out[160…
```
(80.02840304829743,
 1.457184452112441e-12,
 11,
 array([[47.49226006, 17.50773994],
        [35.07120743, 12.92879257],
        [20.45820433,  7.54179567],
        [46.03095975, 16.96904025],
        [41.64705882, 15.35294118],
        [41.64705882, 15.35294118],
        [47.49226006, 17.50773994],
        [46.03095975, 16.96904025],
        [50.41486068, 18.58513932],
        [16.07430341,  5.92569659],
        [34.34055728, 12.65944272],
        [45.3003096 , 16.6996904 ]]))
```

Assumptions :

1. Random sample

2. Independent observations in each cell

3. Expected values should be greater than or equal to 5

All the assumptions are satisfied.

Null Hypothesis, H0: There is no relationship between daily vaccinations and month (i.e. total_vaccinations and month are independent)

Alternative hypothesis, H1: There is relationship between daily vaccinations and month (i.e. total_vaccinations and daily_vaccinations are dependent)

Here p-value, -8.05 < 0.05, so we reject the null hypothesis.

Hence there is relationship between daily vaccinations and month.

In order to answer the research question, I need to consider total vaccinations and fully vaccinated people. So Iam performing the Chi-Squared test with those two variables.

In [161…
```
#add a new column categories next to the total_vaccinations group.

category = pd.cut(desc.total_vaccinations,bins=[0,450000000, 900000000],labels=['Tot
desc.insert(5,'total_vaccination_group',category)
```

In [162…
```
#add a new column category next to the people_fully_vaccinated group.
category = pd.cut(desc.people_fully_vaccinated,bins=[0,120000000, 250000000],labels=
desc.insert(5,'people_fully_vaccinated_group',category)
```

In [163…
```
# people_fully_vaccinated and total_vaccination

totalvacc_peoplevacc = pd.crosstab(index=desc['people_fully_vaccinated_group'], colu
totalvacc_peoplevacc
```

Out[163...

total_vaccination_group	Total A	Total B
people_fully_vaccinated_group		
FullGroup A	569	14
FullGroup B	0	42

In [164...

```
chi2_contingency(totalvacc_peoplevacc)
```

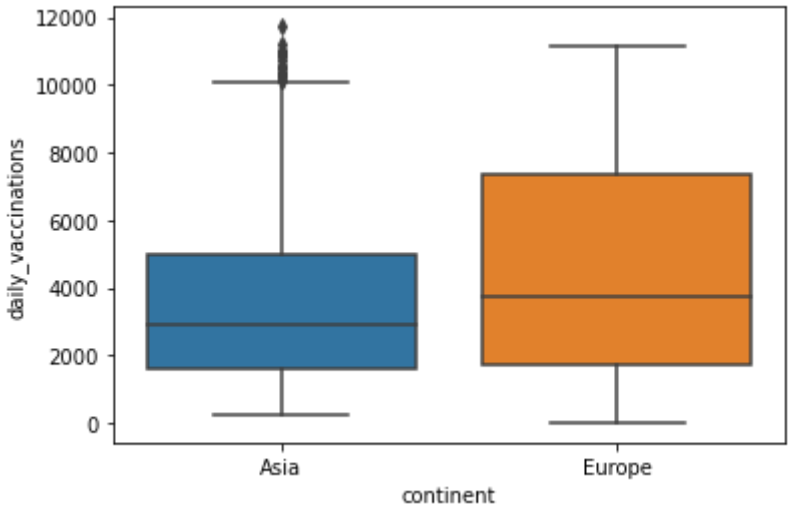
Out[164...

```
(445.6070505375738,
 6.518583859518397e-99,
 1,
 array([[530.7632,  52.2368],
        [ 38.2368,   3.7632]]))
```

```
  Assumptions :

  1. Random sample

  2. Independent observations in each cell

  3. Expected values should be greater than or equal to 5

  All the three assumptions are satisfied.

  Null Hypothesis, H0: There is no relationship between total
  vaccinations and people fully vaccinated (i.e. people_fully_vaccinated
  and  total_vaccinations are independent)

  Alternative hypothesis, H1: There is relationship between total
  vaccinations and people fully vaccinated (i.e. people_fully_vaccinated
  and  total_vaccinations are dependent)

  Here p-value, -58.78 <  0.05, so we will reject null hypothesis.

  Hence there is a relationship between total vaccinations and people
  fully vaccinated
```

In [165...

```
# 2 sample independent test for "continent" and "daily_vaccinations"

sns.boxplot(x="continent", y="daily_vaccinations_per_million", data=sel_data)
plt.ylabel("daily_vaccinations")
plt.show()
```



In [166...
```
sel_data.groupby("continent").describe()
```

Out[166...

continent	count	mean	std	min	25%	50%	75%	total_vacc
Asia	312.0	2.456784e+08	2.992326e+08	56041.0	2496287.0	123237578.0	4.291994e+08	1.003
Europe	334.0	1.887102e+07	3.003651e+07	141.0	982412.0	4702801.5	1.238887e+07	9.509

2 rows × 72 columns

So here Iam performing 2 sample independent test to check the means of the data.

Null hypothesis, H0 : The mean daily vaccinations in Europe is equal to the mean daily vaccinations in Asia (i.e. there is no relationship between the mean daily vaccinations in Europe and Asia)

Alternate Hypothesis, H1: The mean daily vaccinations in Europe is not equal to the mean daily vaccinations in Asia (i.e. there is a relationship between the mean daily vaccinations in Europe and Asia)

But since ,there is no equal spread and the data is not symmtrical and has outliers, I won't be able to continue the 2 sample independent test.

So I am performing the Wilcoxon- Mann-Whitney(WMW) sample test.

Null hypothesis, H0 : The median of daily vaccinations in Europe is equal to the median of daily vaccinations in Asia (i.e. there is no relationship between the median of daily vaccinations in Europe and Asia)

Alternate Hypothesis, H1: The median of daily vaccinations in Europe is not equal to the median of daily vaccinations in Asia (i.e. there

is a relationship between the median of daily vaccinations in Europe and Asia)

In [167…
```
one_vacc = sel_data[sel_data["continent"] == 'Asia']['daily_vaccinations_per_million
two_vacc = sel_data[sel_data["continent"] == 'Europe']['daily_vaccinations_per_milli

ranksums(one_vacc,two_vacc)
```

Out[167… RanksumsResult(statistic=-3.6627538521554364, pvalue=0.0002495182805865695)

The statistic and pvalue is < 0.5.

One way ANOVA test

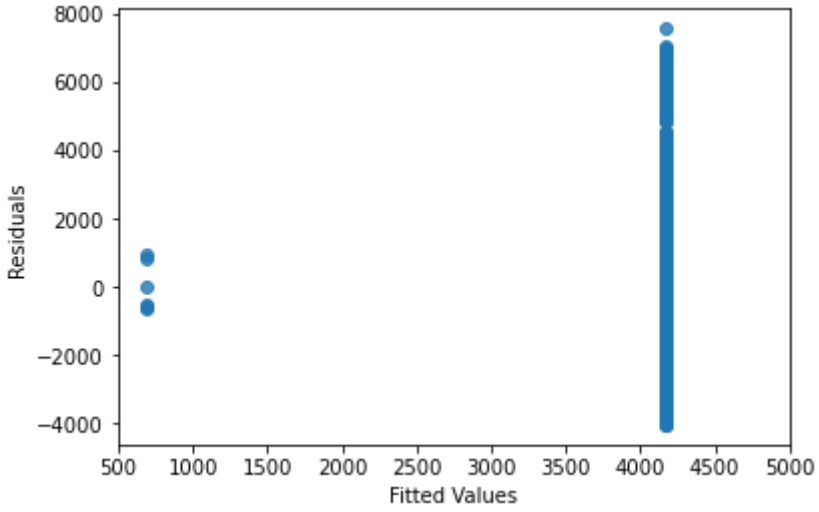
Assumptions:
1.Each of the populations is Normally distributed with the same variance (homogeneity of variance)

2.The observations are sampled independently, the groups under consideration are independent

In [168…
```
#perform one-way ANOVA
onewaymodel = ols('daily_vaccinations_per_million ~ C(year)', data=sel_data).fit()
#fitted values
model_fitted_vals = onewaymodel.fittedvalues
#model residuals
model_residuals = onewaymodel.resid
#standardised residuals
model_norm_residuals = onewaymodel.get_influence().resid_studentized_internal
```

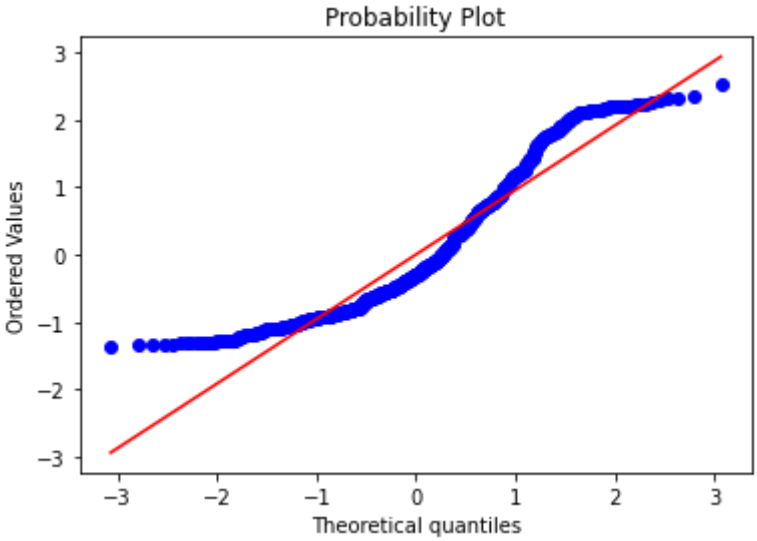
In [169…
```
sns.regplot(x=model_fitted_vals,y=model_residuals,
 ci=False,lowess=True,
 line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8})
plt.xlabel("Fitted Values")
plt.ylabel("Residuals")
#plt.xscale("log")
#plt.yscale("log")
plt.xlim(500, 5000)
plt.show()
```

```
C:\Users\Chikku\anaconda3\lib\site-packages\statsmodels\nonparametric\smoothers_lowe
ss.py:214: RuntimeWarning: invalid value encountered in true_divide
  res, _ = _lowess(y, x, x, np.ones_like(x),
```

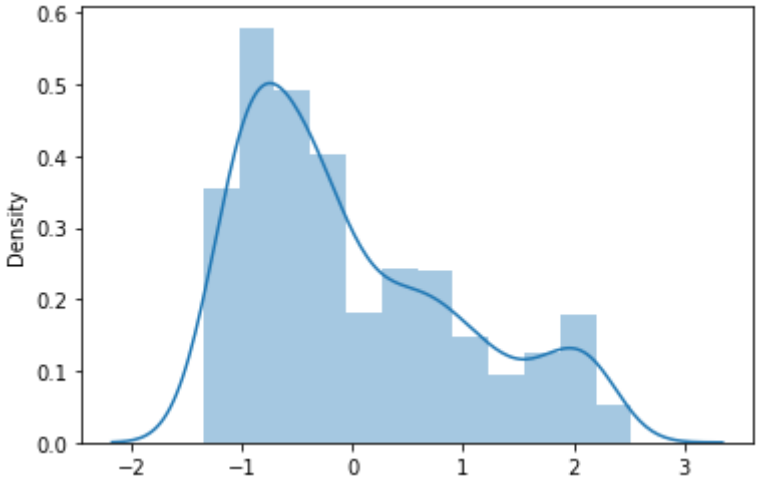


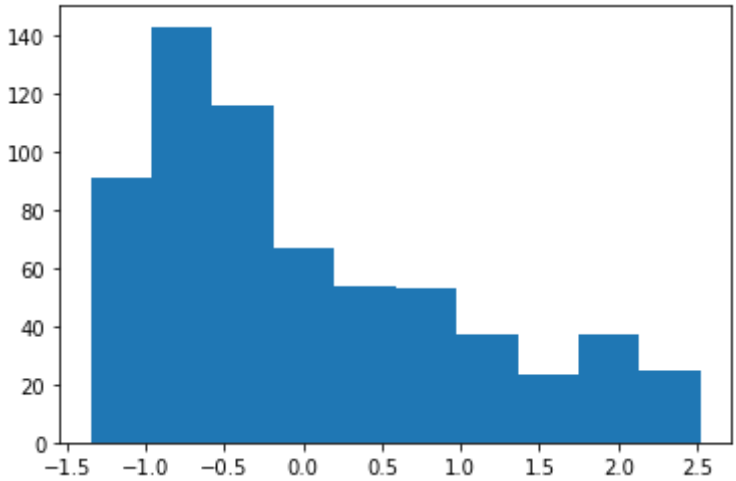
In [170…

```
stats.probplot(model_norm_residuals, plot=sns.mpl.pyplot)
plt.show()

sns.distplot(model_norm_residuals, kde=True)
plt.show()

plt.hist(model_norm_residuals)
plt.show()
```




```
C:\Users\Chikku\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWar
ning: `distplot` is a deprecated function and will be removed in a future version. P
lease adapt your code to use either `displot` (a figure-level function with similar
flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```





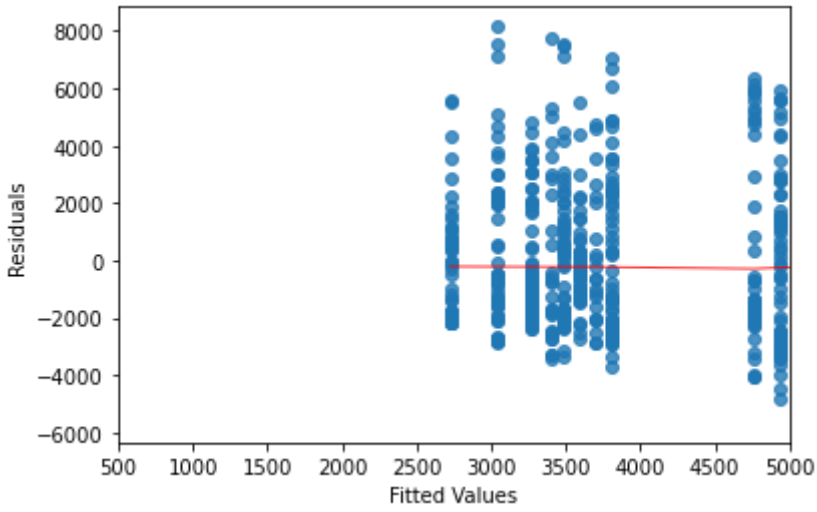
The assumptions are not since there is a strong deviation from normality. met so I won't be able to continue with One-way ANOVA.

In [171…

```
#perform one-way ANOVA  for month and daily_vaccinations
onewaymodel1 = ols('daily_vaccinations_per_million ~ C(month)', data=sel_data).fit()
#fitted values
model_fitted_vals1 = onewaymodel1.fittedvalues
#model residuals
model_residuals1 = onewaymodel1.resid
#standardised residuals
model_norm_residuals1 = onewaymodel1.get_influence().resid_studentized_internal
```

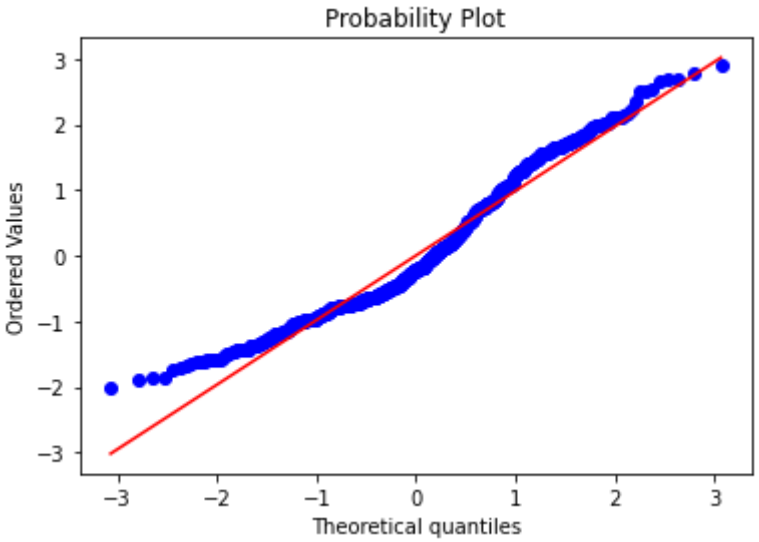
In [172…

```
sns.regplot(x=model_fitted_vals1,y=model_residuals1,
 ci=False,lowess=True,
 line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8})
plt.xlabel("Fitted Values")
plt.ylabel("Residuals")
#plt.xscale("log")
#plt.yscale("log")
plt.xlim(500, 5000)
plt.show()
```

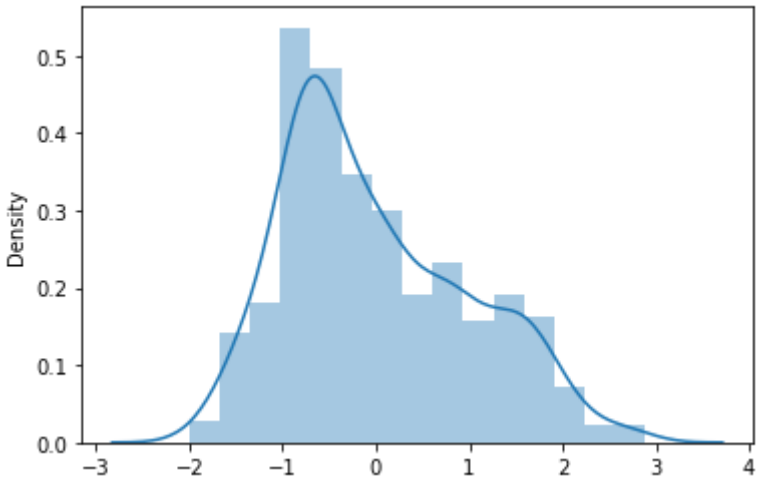


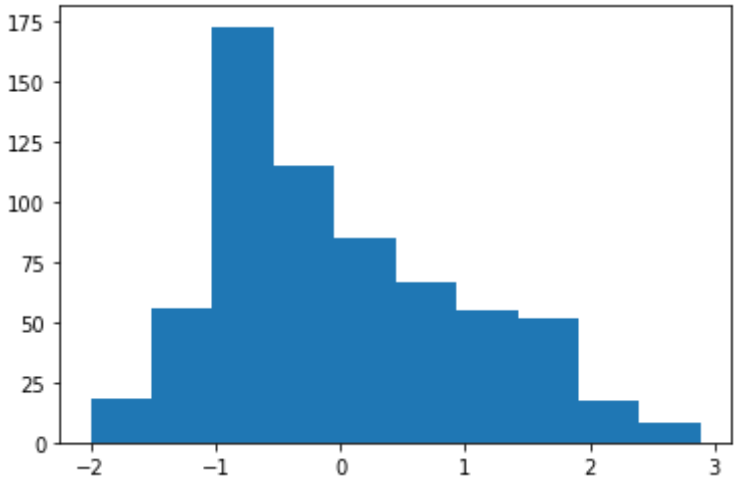
In [173…

```
stats.probplot(model_norm_residuals1, plot=sns.mpl.pyplot)
plt.show()

sns.distplot(model_norm_residuals1, kde=True)
plt.show()

plt.hist(model_norm_residuals1)
plt.show()
```




```
C:\Users\Chikku\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWar
ning: `distplot` is a deprecated function and will be removed in a future version. P
lease adapt your code to use either `displot` (a figure-level function with similar
flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```





The assumptions are roughly satisfied. Since there is roughly equal variance and equal spread and the qq plot is roughly normal.There is no strong deviation from normality.

In [174...

```
onewaymodel1.summary()
```

Out[174...

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	**R-squared:**	0.122
Model:	OLS	**Adj. R-squared:**	0.107
Method:	Least Squares	**F-statistic:**	8.020
Date:	Sat, 11 Dec 2021	**Prob (F-statistic):**	3.57e-13
Time:	10:31:00	**Log-Likelihood:**	-6048.1
No. Observations:	646	**AIC:**	1.212e+04
Df Residuals:	634	**BIC:**	1.217e+04
Df Model:	11		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	4940.9385	352.716	14.008	0.000	4248.306	5633.571
C(month)[T.August]	-1355.3968	541.182	-2.505	0.013	-2418.123	-292.670
C(month)[T.December]	-1543.2242	642.817	-2.401	0.017	-2805.532	-280.916

C(month)[T.February]	-1896.2877	502.759	-3.772	0.000	-2883.561	-909.014
C(month)[T.January]	-1457.9385	516.021	-2.825	0.005	-2471.255	-444.621
C(month)[T.July]	-179.1139	516.021	-0.347	0.729	-1192.431	834.203
C(month)[T.June]	1113.9385	498.815	2.233	0.026	134.408	2093.469
C(month)[T.March]	-1138.6686	502.759	-2.265	0.024	-2125.942	-151.395
C(month)[T.May]	567.9021	491.533	1.155	0.248	-397.328	1533.132
C(month)[T.November]	-1241.5294	701.412	-1.770	0.077	-2618.901	135.842
C(month)[T.October]	-2213.4917	544.484	-4.065	0.000	-3282.701	-1144.282
C(month)[T.September]	-1666.4385	504.813	-3.301	0.001	-2657.747	-675.130

Omnibus:	41.339	**Durbin-Watson:**	0.392
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	41.026
Skew:	0.569	**Prob(JB):**	1.23e-09
Kurtosis:	2.520	**Cond. No.**	12.0

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [175…
```
sel_data.groupby("month").describe()
```

Out[175…

total_v

	count	mean	std	min	25%	50%	75%	
month								
April	65.0	8.042143e+07	1.575861e+08	5128.0	1274232.00	4616601.0	1.226226e+08	9.1
August	48.0	2.964396e+08	2.892673e+08	39631.0	3087934.50	130914059.5	5.774673e+08	9.3
December	28.0	9.466291e+07	2.238413e+08	141.0	231798.00	5720027.5	7.662974e+07	9.6
February	63.0	4.430666e+07	1.467098e+08	3967.0	458621.50	7240455.0	1.427191e+07	9.0
January	57.0	5.415627e+07	1.631537e+08	2968.0	158251.00	2314773.0	6.822981e+06	8.9
July	57.0	1.581765e+08	2.207314e+08	26996.0	5197939.00	20989010.0	4.118464e+08	9.2
June	65.0	1.041527e+08	1.765267e+08	15322.0	3562675.00	4428233.0	1.626036e+08	9.2
March	63.0	6.591019e+07	1.554865e+08	3974.0	917284.50	27032671.0	5.432507e+07	9.0
May	69.0	1.017012e+08	1.605336e+08	8602.0	2294340.00	19497704.0	1.851916e+08	9.1
November	22.0	1.565667e+08	2.628086e+08	60714.0	2355877.75	33005850.5	1.565291e+08	9.5
October	47.0	2.325604e+08	3.680221e+08	52415.0	4842445.00	39587893.0	2.056904e+08	1.0
September	62.0	2.199804e+08	3.474209e+08	47668.0	6439109.50	7209959.5	2.197186e+08	9.4

12 rows × 72 columns

In [176…
```
# the counts are not equal so I am using type=2

sm.stats.anova_lm(onewaymodel1, typ=2)
```

Out[176…

	sum_sq	df	F	PR(>F)
C(month)	7.134142e+08	11.0	8.020213	3.571618e-13
Residual	5.126871e+09	634.0	NaN	NaN

H0: true mean of daily vaccinations are equal across months

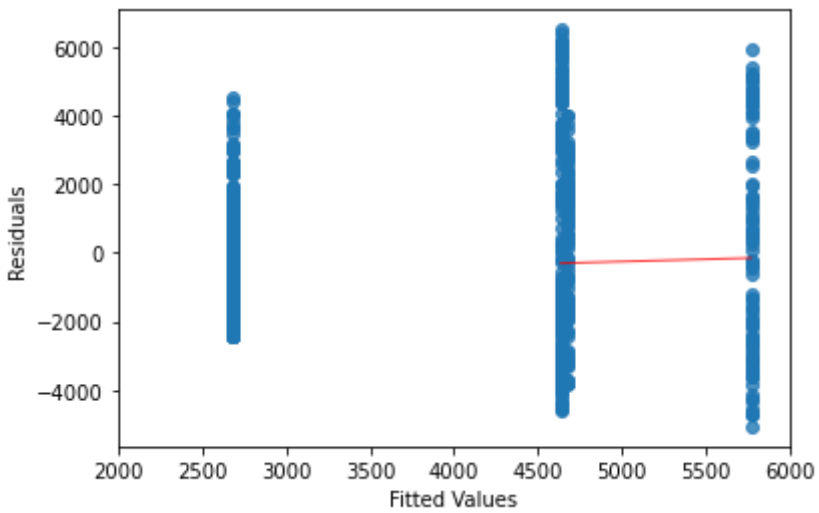
H0: mu_jan = mu_feb =mu_mar = mu_apr =mu_may = mu_jun = mu_jul = mu_aug = mu_sep = mu_oct = mu_nov =mu_dec,
mu_i is the true mean daily vaccinations for group i.

H1: at least one mean of daily vaccinations is different

p-value = 2.86 > 0.05, so we fail to reject the H0 and hence there is no enough evidence to suggest that the true mean of #daily vaccinations are equal across months.

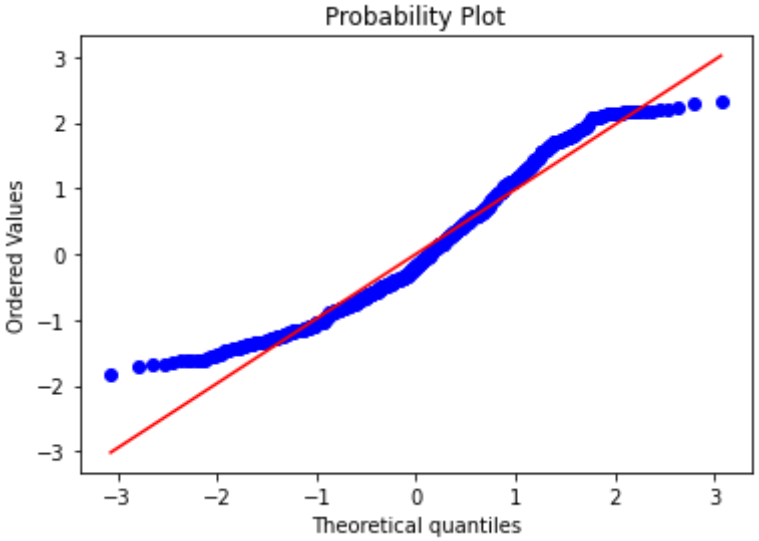
In [177…
```
#perform one-way ANOVA  for country and daily_vaccinations
onewaymodel2 = ols('daily_vaccinations_per_million ~ C(country)', data=sel_data).fit
#fitted values
model_fitted_vals2 = onewaymodel2.fittedvalues
#model residuals
model_residuals2 = onewaymodel2.resid
#standardised residuals
model_norm_residuals2 = onewaymodel2.get_influence().resid_studentized_internal
```

In [178…
```
sns.regplot(x=model_fitted_vals2,y=model_residuals2,
 ci=False,lowess=True,
 line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8})
plt.xlabel("Fitted Values")
plt.ylabel("Residuals")
#plt.xscale("log")
#plt.yscale("log")
plt.xlim(2000, 6000)
plt.show()
```

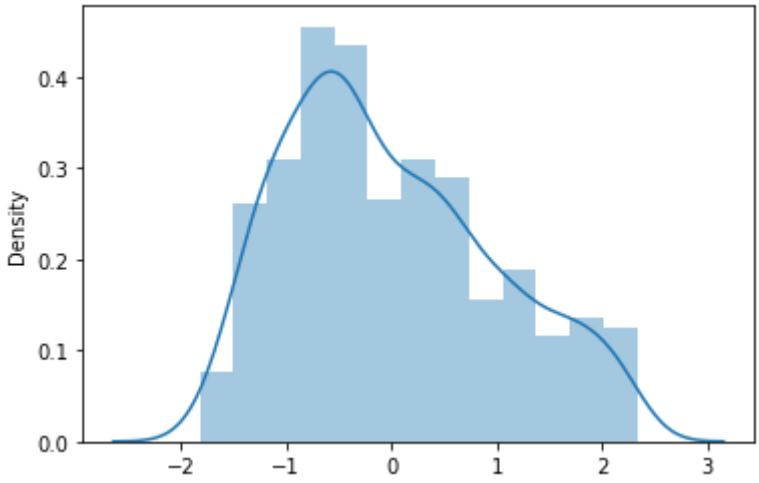


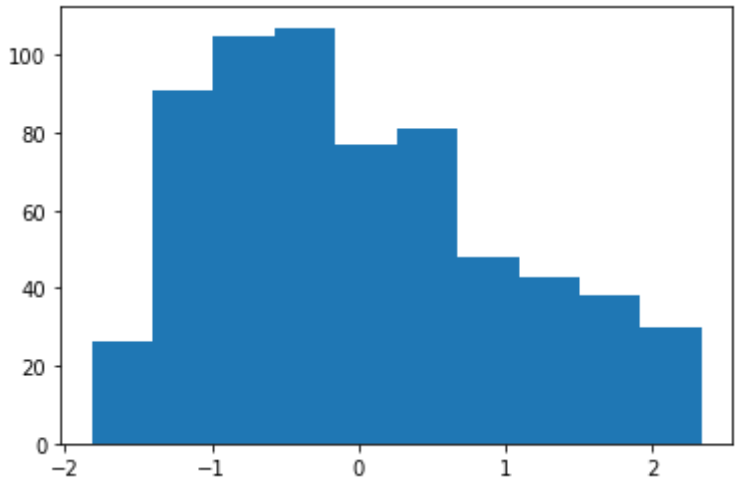
In [179…

```
stats.probplot(model_norm_residuals2, plot=sns.mpl.pyplot)
plt.show()

sns.distplot(model_norm_residuals2, kde=True)
plt.show()

plt.hist(model_norm_residuals2)
plt.show()
```




```
C:\Users\Chikku\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWar
ning: `distplot` is a deprecated function and will be removed in a future version. P
lease adapt your code to use either `displot` (a figure-level function with similar
flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```





In [180…
```
# The assumptions are satisfied. Since there is no equal variance and equal spre
```

The assumptions are roughly satisfied.
There is rougly equal spred and variance and there is no strong devaition from normality.

In [181…
```
onewaymodel2.summary()
```

Out[181…

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	**R-squared:**	0.137
Model:	OLS	**Adj. R-squared:**	0.133
Method:	Least Squares	**F-statistic:**	33.98
Date:	Sat, 11 Dec 2021	**Prob (F-statistic):**	2.15e-20
Time:	10:31:03	**Log-Likelihood:**	-6042.6
No. Observations:	646	**AIC:**	1.209e+04
Df Residuals:	642	**BIC:**	1.211e+04
Df Model:	3		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
Intercept	5770.4409	290.539	19.861	0.000	5199.919	6340.962
C(country)[T.India]	-3090.4272	346.784	-8.912	0.000	-3771.396	-2409.458
C(country)[T.Ireland]	-1135.2930	344.304	-3.297	0.001	-1811.391	-459.195
C(country)[T.United Kingdom]	-1094.2005	399.872	-2.736	0.006	-1879.415	-308.986

Omnibus:	51.505	**Durbin-Watson:**	0.371
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	37.036
Skew:	0.479	**Prob(JB):**	9.07e-09
Kurtosis:	2.323	**Cond. No.**	6.30

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [182…
```
#sel_data.groupby("country").describe()
```

In [183…
```
# the counts are not equal so I am using type=2

sm.stats.anova_lm(onewaymodel2, typ=2)
```

Out[183…

	sum_sq	**df**	**F**	**PR(>F)**
C(country)	8.003283e+08	3.0	33.982485	2.150283e-20

	sum_sq	df	F	PR(>F)
Residual	5.039957e+09	642.0	NaN	NaN

In [184…

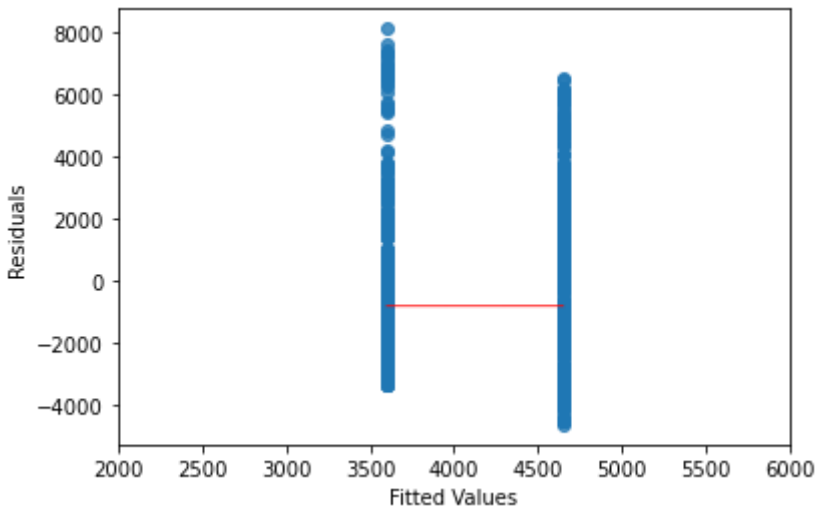
```
#H0: true mean of daily vaccinations are equal across countries

#H0: mu_afghanistan = mu_india =mu_ireland = mu_iceland,
#mu_i is the true mean daily vaccinations for group i.

#H1: at least one mean of daily vaccinations is different

#p-value = -95.57 <  0.05, so we  reject the H0 and hence there is a relationshi
```

In [185…

```
#perform one-way ANOVA  for continent and daily_vaccinations
onewaymodel3 = ols('daily_vaccinations_per_million ~ C(continent)', data=sel_data).f
#fitted values
model_fitted_vals3 = onewaymodel3.fittedvalues
#model residuals
model_residuals3 = onewaymodel3.resid
#standardised residuals
model_norm_residuals3 = onewaymodel3.get_influence().resid_studentized_internal
```

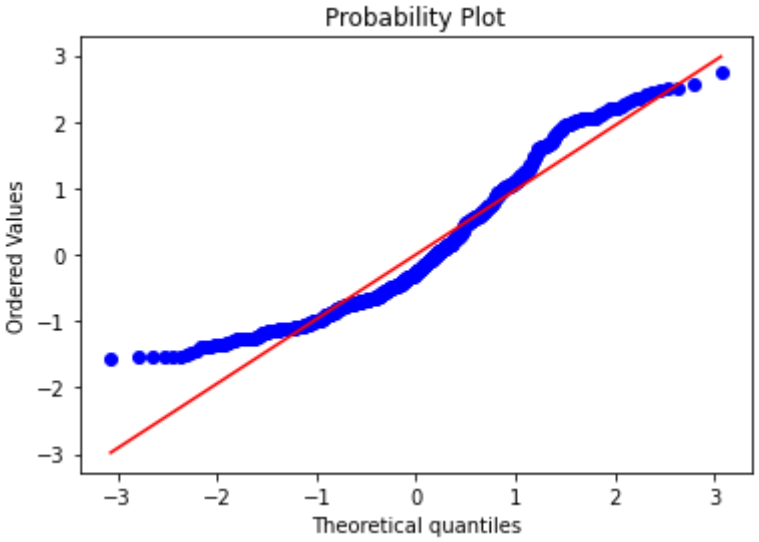
In [186…

```
sns.regplot(x=model_fitted_vals3,y=model_residuals3,
 ci=False,lowess=True,
 line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8})
plt.xlabel("Fitted Values")
plt.ylabel("Residuals")
#plt.xscale("log")
#plt.yscale("log")
plt.xlim(2000, 6000)
plt.show()
```

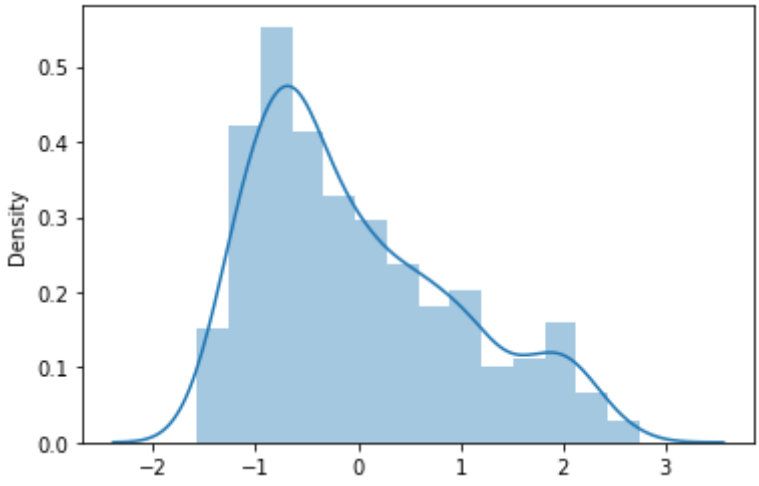


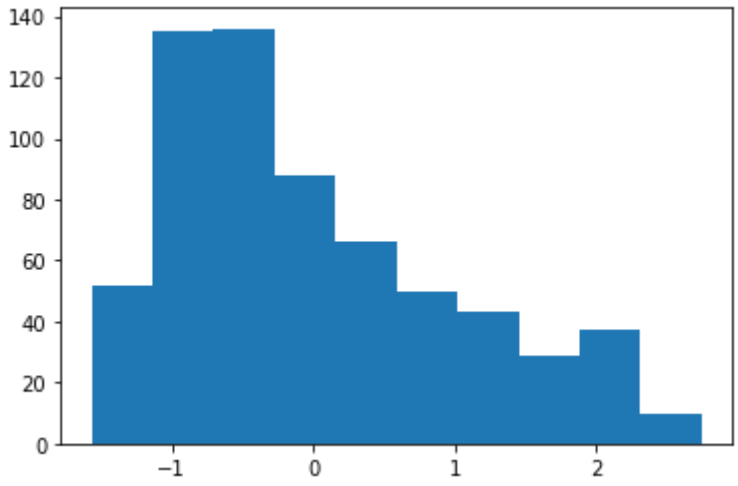
In [187…

```
stats.probplot(model_norm_residuals3, plot=sns.mpl.pyplot)
plt.show()

sns.distplot(model_norm_residuals3, kde=True)
plt.show()

plt.hist(model_norm_residuals3)
plt.show()
```




```
C:\Users\Chikku\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWar
ning: `distplot` is a deprecated function and will be removed in a future version. P
lease adapt your code to use either `displot` (a figure-level function with similar
flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```





The assumptions are roughly satisfied.
There is rougly equal spred and variance and there is no strong devaition from normality

In [263…

```
onewaymodel3.summary()
```

Out[263…

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	**R-squared:**	0.030
Model:	OLS	**Adj. R-squared:**	0.029
Method:	Least Squares	**F-statistic:**	20.10
Date:	Sat, 11 Dec 2021	**Prob (F-statistic):**	8.71e-06
Time:	15:51:32	**Log-Likelihood:**	-6080.3
No. Observations:	646	**AIC:**	1.216e+04
Df Residuals:	644	**BIC:**	1.217e+04
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	3601.1987	167.890	21.450	0.000	3271.522	3930.876
C(continent)[T.Europe]	1046.7444	233.489	4.483	0.000	588.252	1505.236

Omnibus:	53.689	**Durbin-Watson:**	0.334
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	60.383
Skew:	0.719	**Prob(JB):**	7.73e-14
Kurtosis:	2.579	**Cond. No.**	2.66

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [189…
```
# the counts are not equal so I am using type=2

sm.stats.anova_lm(onewaymodel3, typ=2)
```

Out[189…

	sum_sq	df	F	PR(>F)
C(continent)	1.767461e+08	1.0	20.097766	0.000009
Residual	5.663539e+09	644.0	NaN	NaN

TWO WAY ANOVA TESTS

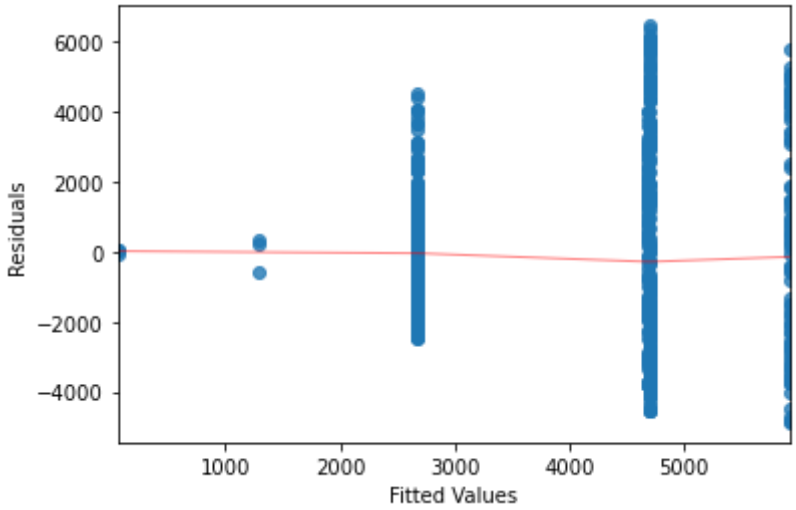
```
Assumptions:
1.Each of the populations is Normally distributed with the same
variance (homogeneity of variance)

2.The observations are sampled independently, the groups under
consideration are independent
```

In [190…
```
#perform two-way ANOVA without interaction between daily vaccinations, country and y

twoway_model1 = ols('daily_vaccinations_per_million ~ C(country) + C(year)', data=se
#fitted values
twoway_model_fitted_vals1 = twoway_model1.fittedvalues
```

```
#model residuals
twoway_model_residuals1 = twoway_model1.resid
#standardised residuals
twoway_model_norm_residuals1 = twoway_model1.get_influence().resid_studentized_inter
```

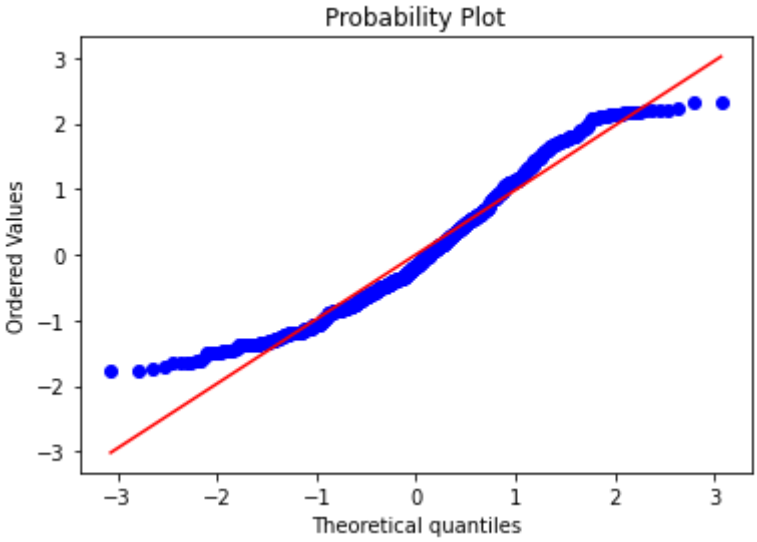
In [191…

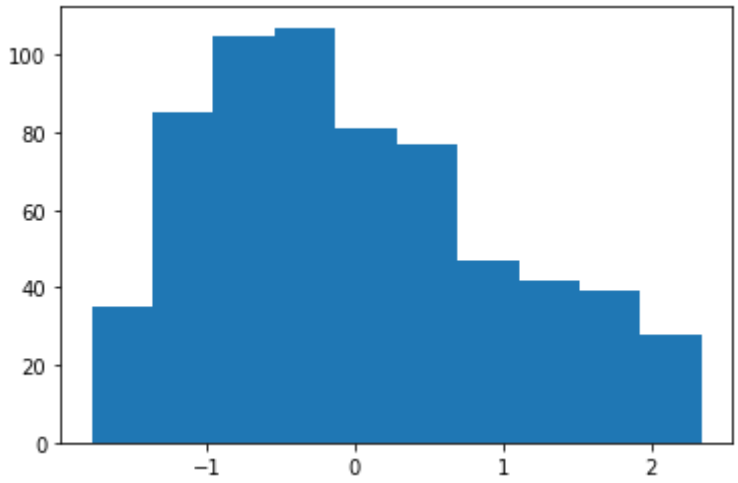
```
sns.regplot(x=twoway_model_fitted_vals1,y=twoway_model_residuals1,ci=False,lowess=Tr
plt.xlabel("Fitted Values")
plt.ylabel("Residuals")
plt.show()
```



In [192…

```
stats.probplot(twoway_model_norm_residuals1, plot=sns.mpl.pyplot)
plt.show()
plt.hist(twoway_model_norm_residuals1)
plt.show()
```





The assumptions are roughly satisfied.

There is rougly equal spred and variance and there is no strong devaition from normality.

In [193…

```
twoway_model1.summary()
```

Out[193…

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	**R-squared:**	0.159
Model:	OLS	**Adj. R-squared:**	0.153
Method:	Least Squares	**F-statistic:**	30.19
Date:	Sat, 11 Dec 2021	**Prob (F-statistic):**	4.91e-23
Time:	10:31:06	**Log-Likelihood:**	-6034.5
No. Observations:	646	**AIC:**	1.208e+04
Df Residuals:	641	**BIC:**	1.210e+04
Df Model:	4		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1293.9207	1143.189	1.132	0.258	-950.926	3538.768
C(country)[T.India]	-3239.6445	344.687	-9.399	0.000	-3916.496	-2562.793
C(country)[T.Ireland]	-1224.1747	340.964	-3.590	0.000	-1893.717	-554.632
C(country)[T.United Kingdom]	-1243.4178	396.888	-3.133	0.002	-2022.776	-464.059
C(year)[T.2021]	4625.7375	1143.429	4.045	0.000	2380.418	6871.057

Omnibus:	50.510	**Durbin-Watson:**	0.348
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	36.129
Skew:	0.471	**Prob(JB):**	1.43e-08
Kurtosis:	2.325	**Cond. No.**	22.1

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [194...

```
sm.stats.anova_lm(twoway_model1, typ=2)
```

Out[194...

	sum_sq	df	F	PR(>F)
C(country)	8.532785e+08	3.0	37.097954	4.079026e-22
C(year)	1.254767e+08	1.0	16.366033	5.857979e-05
Residual	4.914480e+09	641.0	NaN	NaN

In [264...

```
sel_data.groupby(["year", "country"]).daily_vaccinations_per_million.describe()
```

Out[264...

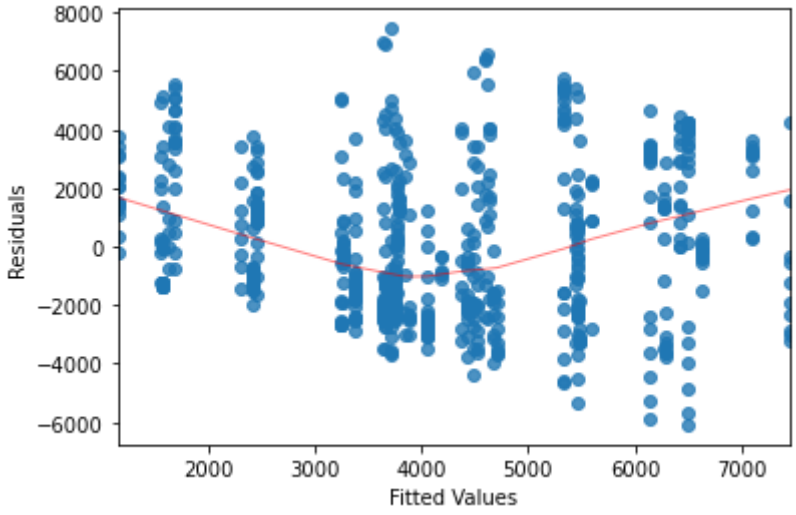
		count	mean	std	min	25%	50%	75%	max
year	**country**								
2020	**Bahrain**	3.0	1276.333333	502.346826	701.0	1100.50	1500.0	1564.00	1628.0
	Ireland	3.0	87.333333	67.899435	11.0	60.50	110.0	125.50	141.0
2021	**Bahrain**	90.0	5920.244444	3187.917665	1035.0	3135.25	5695.0	9094.75	11723.0
	India	219.0	2680.013699	1679.367496	222.0	1426.00	2322.0	3795.50	7204.0
	Ireland	227.0	4695.251101	3552.422479	130.0	1600.50	3141.0	8139.00	11161.0
	United Kingdom	104.0	4676.240385	2295.204217	852.0	2934.00	5070.0	6513.25	8700.0

In [195...

```
#perform two-way ANOVA without interaction between daily vaccinations, country and m

twoway_model2 = ols('daily_vaccinations_per_million ~ C(country) + C(month)', data=s
#fitted values
twoway_model_fitted_vals2 = twoway_model2.fittedvalues
#model residuals
twoway_model_residuals2 = twoway_model2.resid
#standardised residuals
twoway_model_norm_residuals2 = twoway_model2.get_influence().resid_studentized_inter
```

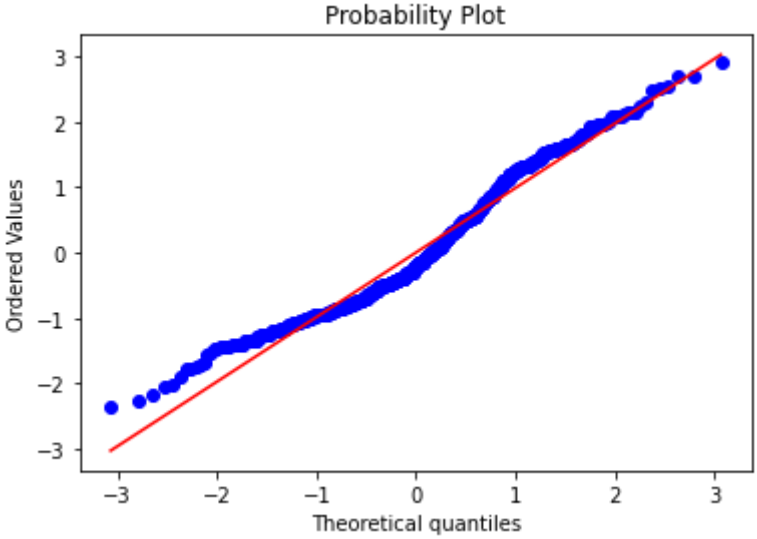
In [196...

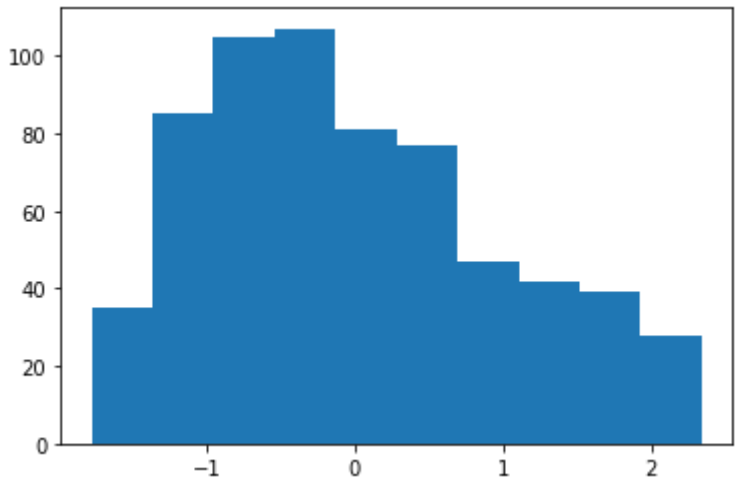
```
sns.regplot(x=twoway_model_fitted_vals2,y=twoway_model_residuals2,ci=False,lowess=Tr
plt.xlabel("Fitted Values")
plt.ylabel("Residuals")
#plt.xscale("log")
#plt.yscale("log")
#plt.xlim(1000000,1000000)
plt.show()
```



In [197…

```
stats.probplot(twoway_model_norm_residuals2, plot=sns.mpl.pyplot)
plt.show()
plt.hist(twoway_model_norm_residuals1)
plt.show()
```





The assumptions are roughly satisfied.

In [198…

```
twoway_model2.summary()
```

Out[198…

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	**R-squared:**	0.262
Model:	OLS	**Adj. R-squared:**	0.245
Method:	Least Squares	**F-statistic:**	15.97
Date:	Sat, 11 Dec 2021	**Prob (F-statistic):**	1.35e-33
Time:	10:31:07	**Log-Likelihood:**	-5992.2
No. Observations:	646	**AIC:**	1.201e+04
Df Residuals:	631	**BIC:**	1.208e+04
Df Model:	14		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	6410.2986	398.555	16.084	0.000	5627.643	7192.954
C(country)[T.India]	-3026.3171	324.951	-9.313	0.000	-3664.434	-2388.201
C(country)[T.Ireland]	-954.0362	324.635	-2.939	0.003	-1591.532	-316.540
C(country)[T.United Kingdom]	-819.0530	377.328	-2.171	0.030	-1560.024	-78.082
C(month)[T.August]	-926.9263	500.148	-1.853	0.064	-1909.083	55.230
C(month)[T.December]	-1745.6310	593.519	-2.941	0.003	-2911.142	-580.120
C(month)[T.February]	-1798.1018	463.724	-3.878	0.000	-2708.730	-887.473
C(month)[T.January]	-1820.7308	480.079	-3.793	0.000	-2763.477	-877.984
C(month)[T.July]	-122.5347	475.128	-0.258	0.797	-1055.559	810.490
C(month)[T.June]	1036.8167	459.419	2.257	0.024	134.641	1938.992
C(month)[T.March]	-964.7341	462.745	-2.085	0.037	-1873.441	-56.028
C(month)[T.May]	675.6088	452.494	1.493	0.136	-212.967	1564.185
C(month)[T.November]	-1077.3399	647.443	-1.664	0.097	-2348.744	194.064
C(month)[T.October]	-2217.9610	502.134	-4.417	0.000	-3204.016	-1231.906
C(month)[T.September]	-1702.6149	464.872	-3.663	0.000	-2615.498	-789.731

Omnibus:	39.467	**Durbin-Watson:**	0.471
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	32.742
Skew:	0.470	**Prob(JB):**	7.77e-08
Kurtosis:	2.423	**Cond. No.**	13.5

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [199…
```
sm.stats.anova_lm(twoway_model2, typ=2)
```

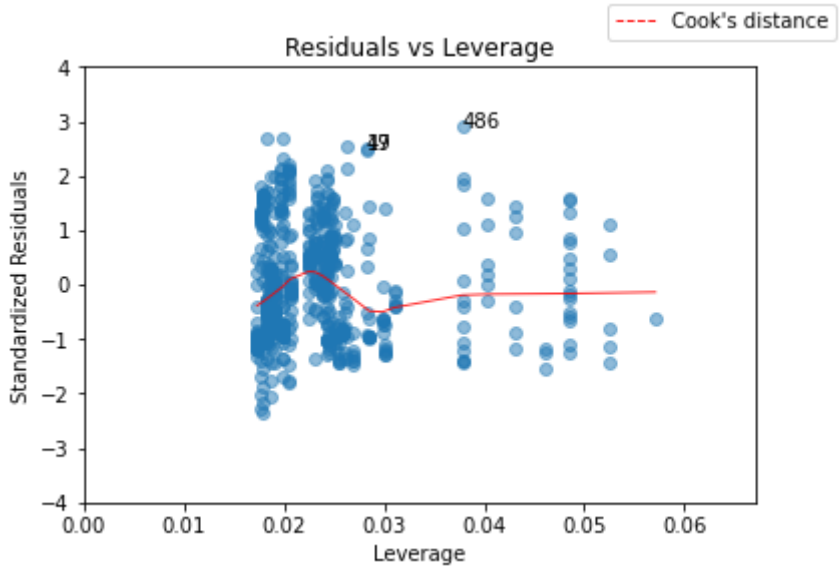
Out[199…

	sum_sq	df	F	PR(>F)

	sum_sq	df	F	PR(>F)
C(country)	8.147105e+08	3.0	39.738960	1.564768e-23
C(month)	7.277964e+08	11.0	9.681701	2.689818e-16
Residual	4.312161e+09	631.0	NaN	NaN

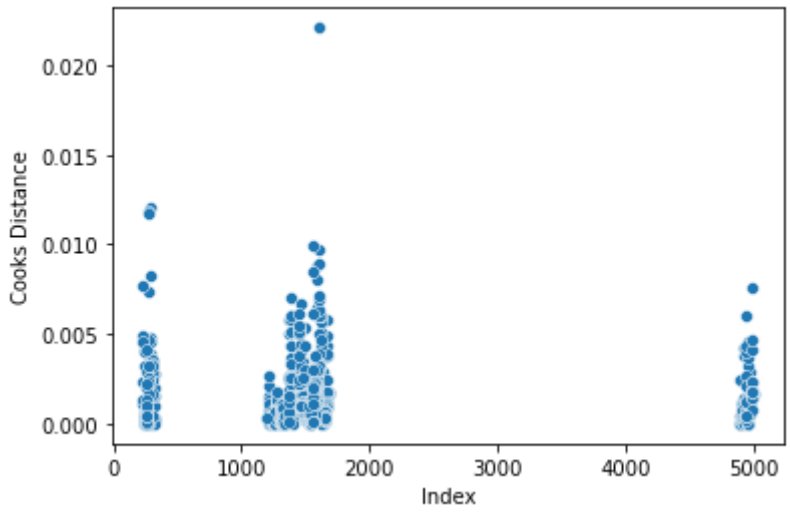
In [200…

```
def graph(formula, x_range, label=None):
 """
 Helper function for plotting cook's distance lines
 """
 x = x_range
 y = formula(x)
 plt.plot(x, y, label=label, lw=1, ls='--', color='red')
# Leverage, from statsmodels internals
model_leverage = twoway_model2.get_influence().hat_matrix_diag
# cook's distance, from statsmodels internals
model_cooks = twoway_model2.get_influence().cooks_distance[0]
plot_cooks = plt.figure();
plt.scatter(model_leverage, twoway_model_norm_residuals2, alpha=0.5);
sns.regplot(x=model_leverage, y=twoway_model_norm_residuals2,
 scatter=False,
 ci=False,
 lowess=True,
 line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8});
plot_cooks.axes[0].set_xlim(0, max(model_leverage)+0.01)
plot_cooks.axes[0].set_ylim(-4, 4)
plot_cooks.axes[0].set_title('Residuals vs Leverage')
plot_cooks.axes[0].set_xlabel('Leverage')
plot_cooks.axes[0].set_ylabel('Standardized Residuals');
# annotations
leverage_top_3 = np.flip(np.argsort(model_cooks), 0)[:3]
for i in leverage_top_3:
 plot_cooks.axes[0].annotate(i,
 xy=(model_leverage[i],
 twoway_model_norm_residuals2[i]));
p = len(twoway_model2.params) # number of model parameters
graph(lambda x: np.sqrt((0.5 * p * (1 - x)) / x),
 np.linspace(0.001, max(model_leverage), 50),
 'Cook\'s distance') # 0.5 line
graph(lambda x: np.sqrt((1 * p * (1 - x)) / x),
 np.linspace(0.001, max(model_leverage), 50)) # 1 line
plot_cooks.legend(loc='upper right');
plt.show()
```



In [201…

```
sns.scatterplot(x=sel_data.index, y=model_cooks)
plt.xlabel('Index')
plt.ylabel('Cooks Distance')
plt.show()
```



There is no cooks distance greater than 0.5. So no need to worry about influential data points.

In [265…

```
#perform two-way ANOVA with interaction
twoway_model3 = ols('daily_vaccinations_per_million ~ C(country) + C(year) + C(count
twoway_model3.summary()
```

Out[265…

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	**R-squared:**	0.159
Model:	OLS	**Adj. R-squared:**	0.152
Method:	Least Squares	**F-statistic:**	24.11
Date:	Sat, 11 Dec 2021	**Prob (F-statistic):**	2.89e-22
Time:	16:32:13	**Log-Likelihood:**	-6034.5
No. Observations:	646	**AIC:**	1.208e+04
Df Residuals:	640	**BIC:**	1.211e+04
Df Model:	5		

Covariance Type: nonrobust

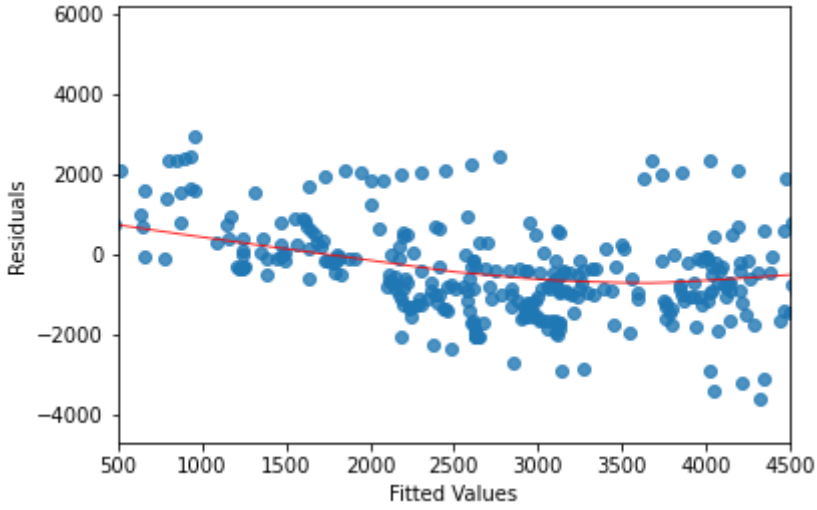
	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1276.3333	1599.883	0.798	0.425	-1865.320	4417.987
C(country)[T.India]	-1620.1154	173.482	-9.339	0.000	-1960.778	-1279.452
C(country)[T.Ireland]	-1189.0000	2262.576	-0.526	0.599	-5631.969	3253.969
C(country)[T.United Kingdom]	-622.0020	199.472	-3.118	0.002	-1013.700	-230.304
C(year)[T.2021]	4643.9111	1626.329	2.855	0.004	1450.326	7837.496
C(country)[T.India]:C(year)[T.2021]	-1620.1154	173.482	-9.339	0.000	-1960.778	-1279.452
C(country)[T.Ireland]:C(year)[T.2021]	-35.9933	2288.754	-0.016	0.987	-4530.369	4458.382
C(country)[T.United Kingdom]:C(year)[T.2021]	-622.0020	199.472	-3.118	0.002	-1013.700	-230.304

Omnibus:	50.500	**Durbin-Watson:**	0.348
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	36.128
Skew:	0.471	**Prob(JB):**	1.43e-08
Kurtosis:	2.325	**Cond. No.**	6.18e+16

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 4.31e-31. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

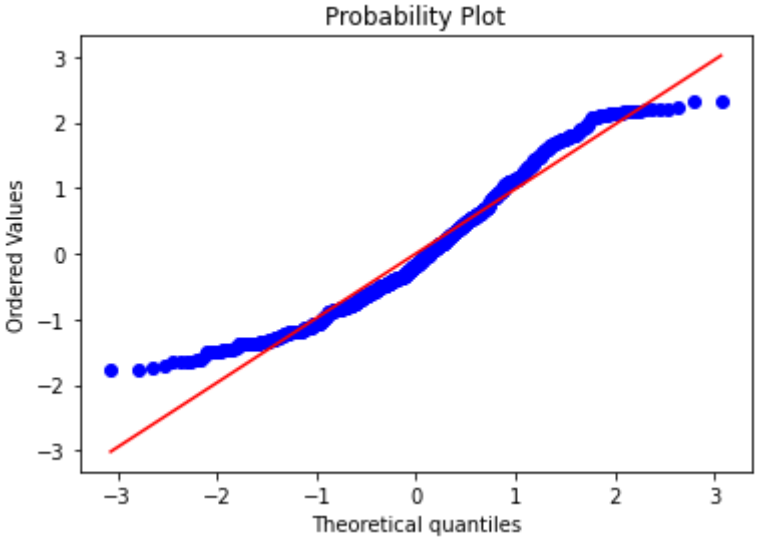
In [266…

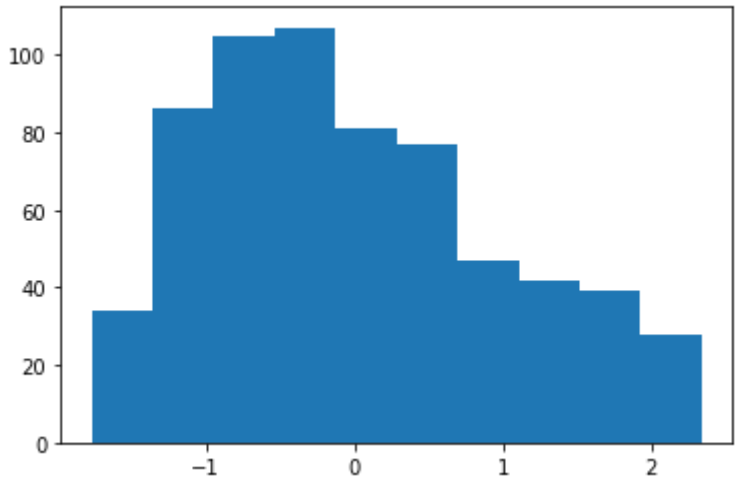
```
#fitted values
twoway_model_fitted_vals3 = twoway_model3.fittedvalues
#model residuals
twoway_model_residuals3 = twoway_model3.resid
#standardised residuals
twoway_model_norm_residuals3 = twoway_model3.get_influence().resid_studentized_inter
sns.regplot(x=model_fitted_vals,y=model_residuals,
 ci=False,lowess=True,
 line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8})
plt.xlabel("Fitted Values")
plt.ylabel("Residuals")
plt.xlim(500, 4500)
plt.show()
```



In [267…

```
stats.probplot(twoway_model_norm_residuals3, plot=sns.mpl.pyplot)
plt.show()

plt.hist(twoway_model_norm_residuals3)
plt.show()
```





In [268…

```
sm.stats.anova_lm(twoway_model3, typ=2)
```

Out[268…

	sum_sq	df	F	PR(>F)
C(country)	9.128881e+07	3.0	3.962770	8.144215e-03

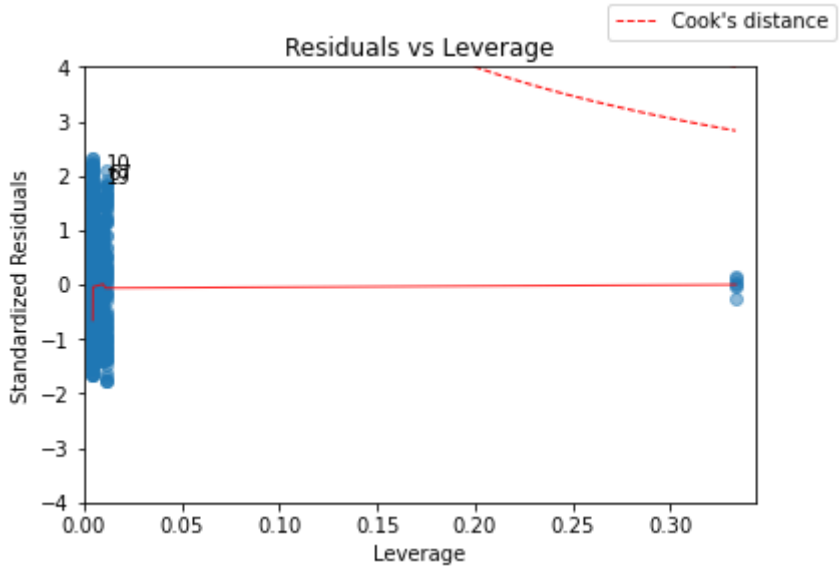
	sum_sq	df	F	PR(>F)
C(year)	7.728294e+07	1.0	10.064360	1.583995e-03
C(country):C(year)	7.618027e+08	3.0	33.069207	7.015527e-20
Residual	4.914478e+09	640.0	NaN	NaN

In [207…

```
# country , month, vaccinations
```

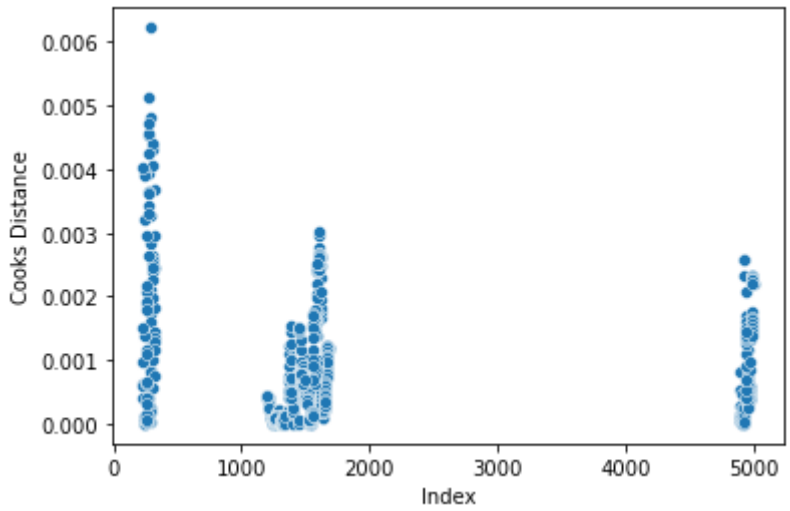
In [269…

```
def graph(formula, x_range, label=None):
 """
 Helper function for plotting cook's distance lines
 """
 x = x_range
 y = formula(x)
 plt.plot(x, y, label=label, lw=1, ls='--', color='red')
# leverage, from statsmodels internals
model_leverage = twoway_model3.get_influence().hat_matrix_diag
# cook's distance, from statsmodels internals
model_cooks = twoway_model3.get_influence().cooks_distance[0]
plot_cooks = plt.figure();
plt.scatter(model_leverage, twoway_model_norm_residuals3, alpha=0.5);
sns.regplot(x=model_leverage, y=twoway_model_norm_residuals3,
 scatter=False,
 ci=False,
 lowess=True,
 line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8});
plot_cooks.axes[0].set_xlim(0, max(model_leverage)+0.01)
plot_cooks.axes[0].set_ylim(-4, 4)
plot_cooks.axes[0].set_title('Residuals vs Leverage')
plot_cooks.axes[0].set_xlabel('Leverage')
plot_cooks.axes[0].set_ylabel('Standardized Residuals');
# annotations
leverage_top_3 = np.flip(np.argsort(model_cooks), 0)[:3]
for i in leverage_top_3:
 plot_cooks.axes[0].annotate(i,
 xy=(model_leverage[i],
 twoway_model_norm_residuals3[i]));
p = len(twoway_model3.params) # number of model parameters
graph(lambda x: np.sqrt((0.5 * p * (1 - x)) / x),
 np.linspace(0.001, max(model_leverage), 50),
 'Cook\'s distance') # 0.5 line
graph(lambda x: np.sqrt((1 * p * (1 - x)) / x),
 np.linspace(0.001, max(model_leverage), 50)) # 1 line
plot_cooks.legend(loc='upper right');
plt.show()
```



In [271…

```
sns.scatterplot(x=sel_data.index, y=model_cooks)
plt.xlabel('Index')
plt.ylabel('Cooks Distance')
plt.show()
```



Linear Regression

In this section, a model will be constructed based on the response variable selected, and the type of response variable (i.e., categorical or numerical), and summary statistics generated in Python.

The data from the sample is regarding the COVID-19 vaccination details of countries Afghanistan, India, Ireland and Iceland.
The response variable I choose is the "daily_vaccinations_per_million" and the independendt variables I am considering are "total_vaccinations","people_vaccinated", "people_fully_vaccinated", "year","country", "month", "continent", "total_vaccinations_per_hundred","people_vaccinated_per_hundred", "people_fully_vaccinated_per_hundred".

The other variables "iso_code", "vaccines", "date", and "source_name" are not using here. Since it does not come in the analysis scope.

Since my response varibale is continuous numeric, I will be using SLR

and MLR.
First I am creating dummy variables for all the categorical variables('year', 'month', 'country', 'continent).
since total_vaccinations_per_hundred has high correlation it is using as perdictor in SLR.

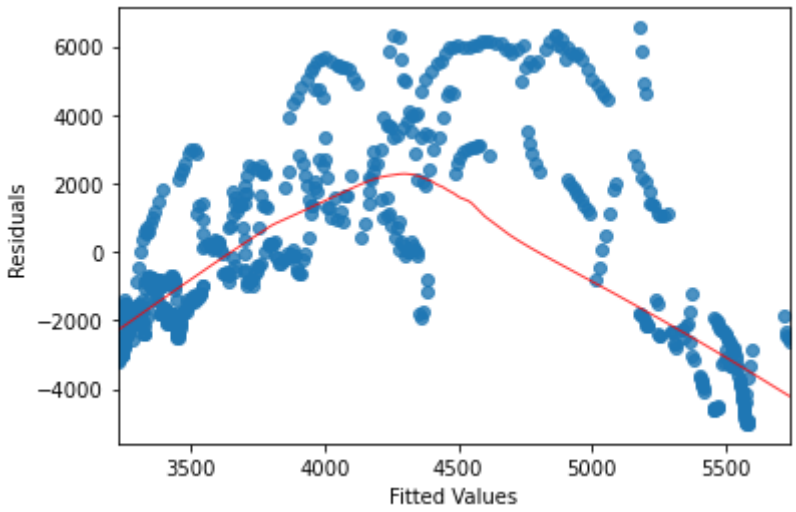
SLR with "daily_vaccinations_per_million" and "total_vaccinations_per_hundred"

In [208…

```
X = sel_data["total_vaccinations_per_hundred"]
X = sm.add_constant(X)
y = sel_data['daily_vaccinations_per_million']
model_slr = sm.OLS(y, X).fit()
#fitted values
model_fitted_vals = model_slr.fittedvalues
#model residuals
model_residuals = model_slr.resid
#standardised residuals
model_norm_residuals = model_slr.get_influence().resid_studentized_internal
```

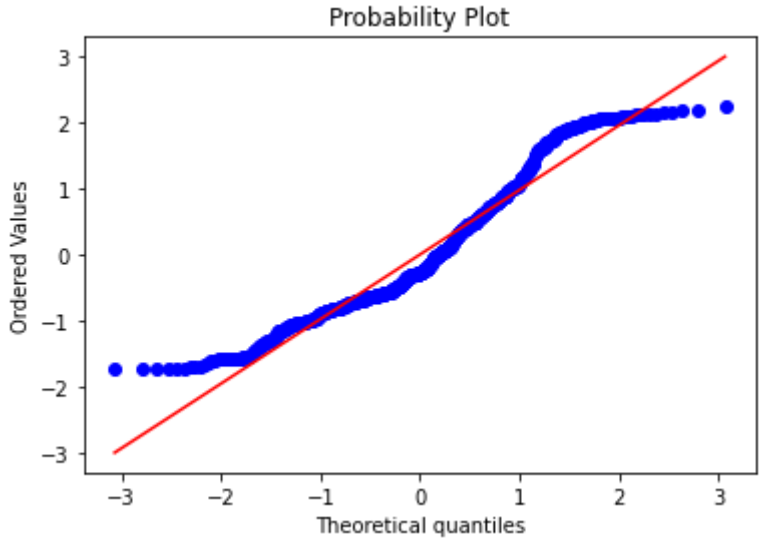
In [209…

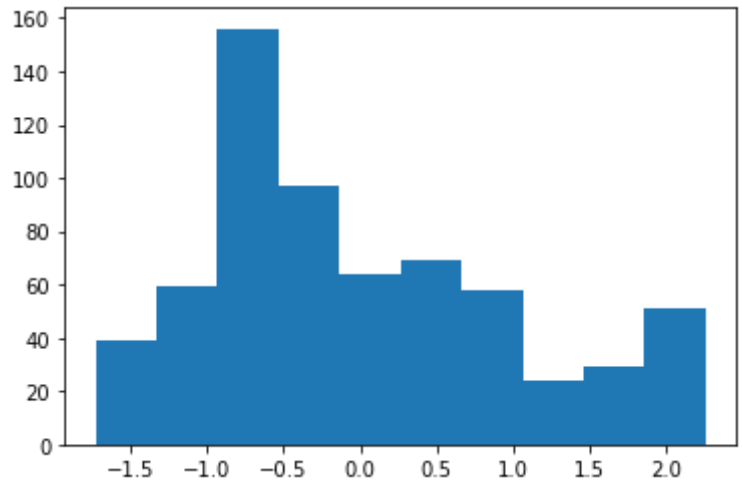
```
sns.regplot(x=model_fitted_vals,y=model_residuals,
 ci=False,lowess=True,
 line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8})
plt.xlabel("Fitted Values")
plt.ylabel("Residuals")
plt.show()
```



In [210…

```
stats.probplot(model_norm_residuals, plot=sns.mpl.pyplot)
plt.show()
plt.hist(model_norm_residuals)
plt.show()
```





In [211…

```
model_slr.summary()
```

Out[211…

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	**R-squared:**	0.072
Model:	OLS	**Adj. R-squared:**	0.070
Method:	Least Squares	**F-statistic:**	49.70
Date:	Sat, 11 Dec 2021	**Prob (F-statistic):**	4.62e-12
Time:	10:31:10	**Log-Likelihood:**	-6066.2
No. Observations:	646	**AIC:**	1.214e+04
Df Residuals:	644	**BIC:**	1.215e+04
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
const	3227.5018	172.836	18.674	0.000	2888.111	3566.893
total_vaccinations_per_hundred	16.0831	2.281	7.050	0.000	11.604	20.563

Omnibus:	47.439	**Durbin-Watson:**	0.335
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	43.335

Skew:	0.568	**Prob(JB):**	3.89e-10
Kurtosis:	2.434	**Cond. No.**	115.

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [212…
```
np.sqrt(model_slr.scale)
```

Out[212…
```
2901.5472635993583
```

MLR

In [213…
```
sel_data["year_21"]=pd.get_dummies(sel_data["year"])[2021]
```

```
<ipython-input-213-1d64f13d3fc7>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  sel_data["year_21"]=pd.get_dummies(sel_data["year"])[2021]
```

In [214…
```
sel_data["month_2"]=pd.get_dummies(sel_data["month"])["February"]
sel_data["month_3"]=pd.get_dummies(sel_data["month"])["March"]
sel_data["month_4"]=pd.get_dummies(sel_data["month"])["April"]
sel_data["month_5"]=pd.get_dummies(sel_data["month"])["May"]
sel_data["month_6"]=pd.get_dummies(sel_data["month"])["June"]
sel_data["month_7"]=pd.get_dummies(sel_data["month"])["July"]
sel_data["month_8"]=pd.get_dummies(sel_data["month"])["August"]
sel_data["month_9"]=pd.get_dummies(sel_data["month"])["September"]
sel_data["month_10"]=pd.get_dummies(sel_data["month"])["October"]
sel_data["month_11"]=pd.get_dummies(sel_data["month"])["November"]
sel_data["month_12"]=pd.get_dummies(sel_data["month"])["December"]
```

```
<ipython-input-214-6fa632f36b48>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  sel_data["month_2"]=pd.get_dummies(sel_data["month"])["February"]
<ipython-input-214-6fa632f36b48>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  sel_data["month_3"]=pd.get_dummies(sel_data["month"])["March"]
<ipython-input-214-6fa632f36b48>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  sel_data["month_4"]=pd.get_dummies(sel_data["month"])["April"]
<ipython-input-214-6fa632f36b48>:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  sel_data["month_5"]=pd.get_dummies(sel_data["month"])["May"]
<ipython-input-214-6fa632f36b48>:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  sel_data["month_6"]=pd.get_dummies(sel_data["month"])["June"]
<ipython-input-214-6fa632f36b48>:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  sel_data["month_7"]=pd.get_dummies(sel_data["month"])["July"]
<ipython-input-214-6fa632f36b48>:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  sel_data["month_8"]=pd.get_dummies(sel_data["month"])["August"]
<ipython-input-214-6fa632f36b48>:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  sel_data["month_9"]=pd.get_dummies(sel_data["month"])["September"]
<ipython-input-214-6fa632f36b48>:9: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  sel_data["month_10"]=pd.get_dummies(sel_data["month"])["October"]
<ipython-input-214-6fa632f36b48>:10: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  sel_data["month_11"]=pd.get_dummies(sel_data["month"])["November"]
<ipython-input-214-6fa632f36b48>:11: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  sel_data["month_12"]=pd.get_dummies(sel_data["month"])["December"]
```

In [215…

```
sel_data["country_ind"]=pd.get_dummies(sel_data["country"])["India"]
sel_data["country_irl"]=pd.get_dummies(sel_data["country"])["Ireland"]
sel_data["country_bah"]=pd.get_dummies(sel_data["country"])["Bahrain"]
```

```
<ipython-input-215-fdd7c0be6704>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy
  sel_data["country_ind"]=pd.get_dummies(sel_data["country"])["India"]
<ipython-input-215-fdd7c0be6704>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  sel_data["country_irl"]=pd.get_dummies(sel_data["country"])["Ireland"]
<ipython-input-215-fdd7c0be6704>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  sel_data["country_bah"]=pd.get_dummies(sel_data["country"])["Bahrain"]
```

In [216…

```
sel_data["conti_eur"]=pd.get_dummies(sel_data["continent"])["Europe"]
```

```
<ipython-input-216-4ddf0aea5024>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  sel_data["conti_eur"]=pd.get_dummies(sel_data["continent"])["Europe"]
```

In [217…

```
X = sel_data[["people_vaccinated_per_hundred","total_vaccinations_per_hundred","peop
X = sm.add_constant(X)
y = sel_data['daily_vaccinations_per_million']
X.head()
```

Out[217…

	const	people_vaccinated_per_hundred	total_vaccinations_per_hundred	people_fully_vaccinated_p
295	1.0	63.33	131.42	
296	1.0	63.39	131.86	
294	1.0	62.98	129.63	
293	1.0	62.93	129.42	
292	1.0	62.88	129.12	

In [218…

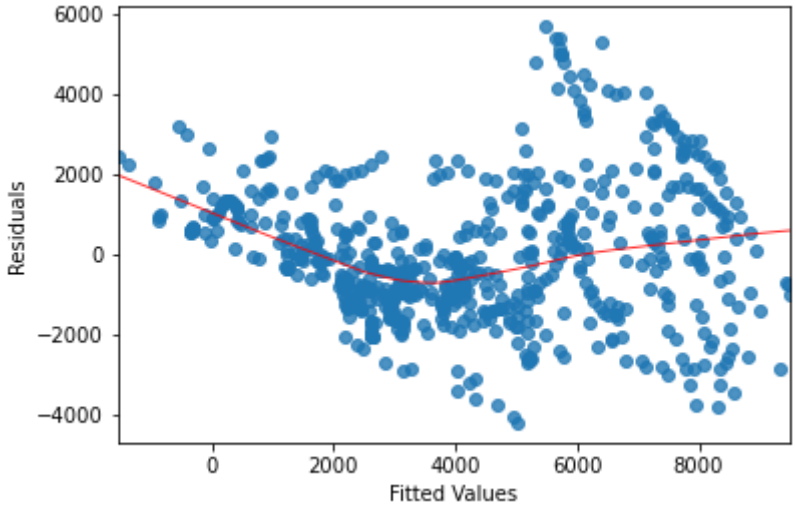
```
model_full_mlr = sm.OLS(y, X).fit()
```

In [219…

```
#fitted values
model_fitted_vals = model_full_mlr.fittedvalues
#model residuals
model_residuals = model_full_mlr.resid
#standardised residuals
model_norm_residuals = model_full_mlr.get_influence().resid_studentized_internal
```

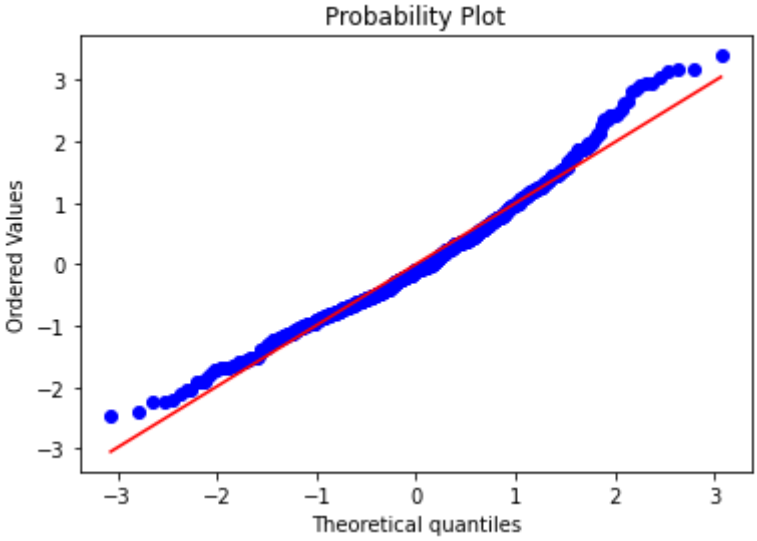
In [220…

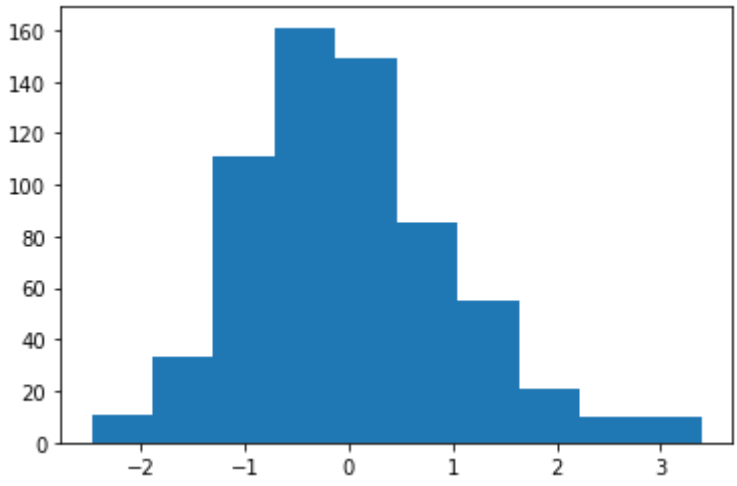
```
sns.regplot(x=model_fitted_vals,y=model_residuals,
 ci=False,lowess=True,
 line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8})
plt.xlabel("Fitted Values")
plt.ylabel("Residuals")
plt.show()
```



In [221…

```
stats.probplot(model_norm_residuals, plot=sns.mpl.pyplot)
plt.show()

plt.hist(model_norm_residuals)
plt.show()
```





The assumptions are roughly satisfied as there is equal variance and no strong devaition from normality.
the iid assumption is also met.

In [222…

```
model_full_mlr.summary()
```

Out[222...

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	**R-squared:**	0.684
Model:	OLS	**Adj. R-squared:**	0.675
Method:	Least Squares	**F-statistic:**	75.34
Date:	Sat, 11 Dec 2021	**Prob (F-statistic):**	2.08e-143
Time:	10:31:13	**Log-Likelihood:**	-5718.3
No. Observations:	646	**AIC:**	1.147e+04
Df Residuals:	627	**BIC:**	1.156e+04
Df Model:	18		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-968.4083	627.247	-1.544	0.123	-2200.168	263.351
people_vaccinated_per_hundred	374.6250	24.056	15.573	0.000	327.385	421.864
total_vaccinations_per_hundred	-147.5449	23.447	-6.293	0.000	-193.589	-101.500
people_fully_vaccinated_per_hundred	-68.8774	25.204	-2.733	0.006	-118.372	-19.383
year_21	3104.4078	817.007	3.800	0.000	1500.006	4708.809
month_2	-305.8513	316.524	-0.966	0.334	-927.427	315.725
month_3	-112.7019	321.460	-0.351	0.726	-743.970	518.567
month_4	359.9958	321.145	1.121	0.263	-270.654	990.646
month_5	626.7066	320.668	1.954	0.051	-3.007	1256.420
month_6	725.3279	330.047	2.198	0.028	77.197	1373.459
month_7	26.3157	343.437	0.077	0.939	-648.111	700.743
month_8	-1027.0043	362.896	-2.830	0.005	-1739.644	-314.365
month_9	-1447.2262	351.642	-4.116	0.000	-2137.764	-756.688
month_10	-1626.7927	370.579	-4.390	0.000	-2354.519	-899.067
month_11	-605.6963	439.439	-1.378	0.169	-1468.646	257.254
month_12	-232.8088	440.184	-0.529	0.597	-1097.223	631.605
country_ind	-2310.5079	269.354	-8.578	0.000	-2839.452	-1781.564
country_irl	1674.6371	222.018	7.543	0.000	1238.649	2110.625
country_bah	2678.5949	258.915	10.345	0.000	2170.150	3187.040
conti_eur	-1336.4953	260.943	-5.122	0.000	-1848.923	-824.068

Omnibus:	38.218	**Durbin-Watson:**	0.678
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	44.116
Skew:	0.585	**Prob(JB):**	2.63e-10
Kurtosis:	3.519	**Cond. No.**	1.34e+17

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 3.08e-28. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

The equation of line, y = -968.40 + 374.62 people_vaccinated_per_hundred - 147.54 total_vaccinations_per_hundred - 68.87 people_fully_vaccinated_per_hundred + 3104.4 year_21 -305.85 month_2 -112.70 month_3 + 359.99 month_4 + 626.7 month_5 + 725.32 month_6 + 26.31 month_7 -1027.00 month_8 -1447.22 month_9 -1447.22 month_10 -605.69 month_11 -605.69 month_12 -2310.50 country_ind + 1674.63 country_irl + 2678.59 country_bah - 1336.46 conti_eur

-968.40 is the estimated intercept, that is average y when all x are 0.

For every count increase in people_vaccinated_per_hundred, the average increase in daily_vaccinations_per_million will be 374.62 counts, keeping all other variables the same.

For every count increase in total_vaccinations_per_hundred, the average increase in daily_vaccinations_per_million will be - 147.54 counts, keeping all other variables the same.

For every count increase in people_fully_vaccinated_per_hundred, the average increase in daily_vaccinations_per_million will be - 68.87 counts, keeping all other variables the same.

For every count increase in year_21, the average increase in daily_vaccinations_per_million will be 3104.4 counts, keeping all other variables the same.

For every count increase in vaccinations in month_2, the average increase in daily_vaccinations_per_million will be -305.85 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_3, the average increase in daily_vaccinations_per_million will be 359.99 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_4, the average increase in daily_vaccinations_per_million will be -305.85 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_5, the average increase in daily_vaccinations_per_million will be 626.7 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_6, the average increase in daily_vaccinations_per_million will be 725.32 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_7, the average increase in daily_vaccinations_per_million will be 26.31 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_8, the average increase in daily_vaccinations_per_million will be -1027.00 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_9, the average increase in daily_vaccinations_per_million will be -1447.22 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_10, the average increase in daily_vaccinations_per_million will be -1447.22 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_11, the average increase in daily_vaccinations_per_million will be -605.69 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_12, the average increase in daily_vaccinations_per_million will be -605.69 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_ind, the average increase in daily_vaccinations_per_million will be -305.85 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_2, the average increase in daily_vaccinations_per_million will be -2310.50 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_irl, the average increase in daily_vaccinations_per_million will be 1674.63 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_bah, the average increase in daily_vaccinations_per_million will be 2678.59 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in conti_eur, the average increase in daily_vaccinations_per_million will be - 1336.46 vaccination counts, keeping all other variables the same.

68% variantion in hwt is being explained by this model.

1716.06 is RSE, the average spread of residuals around equation.

In [223…
```
np.sqrt(model_full_mlr.scale)
```

Out[223… 1716.069677663897

In [224…
```
from statsmodels.stats.outliers_influence import variance_inflation_factor
pd.Series([variance_inflation_factor(X.values, i)
```

```
  for i in range(X.shape[1])],
  index=X.columns)
```

```
C:\Users\Chikku\anaconda3\lib\site-packages\statsmodels\regression\linear_model.py:1
715: RuntimeWarning: divide by zero encountered in double_scalars
  return 1 - self.ssr/self.centered_tss
C:\Users\Chikku\anaconda3\lib\site-packages\statsmodels\stats\outliers_influence.py:
193: RuntimeWarning: divide by zero encountered in double_scalars
  vif = 1. / (1. - r_squared_i)
```

Out[224…

```
const                                  0.000000
people_vaccinated_per_hundred         82.112671
total_vaccinations_per_hundred        302.013712
people_fully_vaccinated_per_hundred    87.825520
year_21                                 1.347351
month_2                                 1.934285
month_3                                 1.995081
month_4                                 2.047339
month_5                                 2.151961
month_6                                 2.162413
month_7                                 2.081532
month_8                                 1.987035
month_9                                 2.353440
month_10                                2.032280
month_11                                1.393484
month_12                                1.762435
country_ind                                  inf
country_irl                             2.479098
country_bah                                  inf
conti_eur                                    inf
dtype: float64
```

In [225…

```
# removing insignificant variables and performing mlr again


X = sel_data[["people_vaccinated_per_hundred","people_fully_vaccinated_per_hundred",
X = sm.add_constant(X)
y = sel_data['daily_vaccinations_per_million']
X.head()
```

Out[225…

	const	people_vaccinated_per_hundred	people_fully_vaccinated_per_hundred	year_21	month_6
295	1.0	63.33	60.19	1	0
296	1.0	63.39	60.36	1	0
294	1.0	62.98	59.55	1	0
293	1.0	62.93	59.48	1	0
292	1.0	62.88	59.38	1	0

In [226…

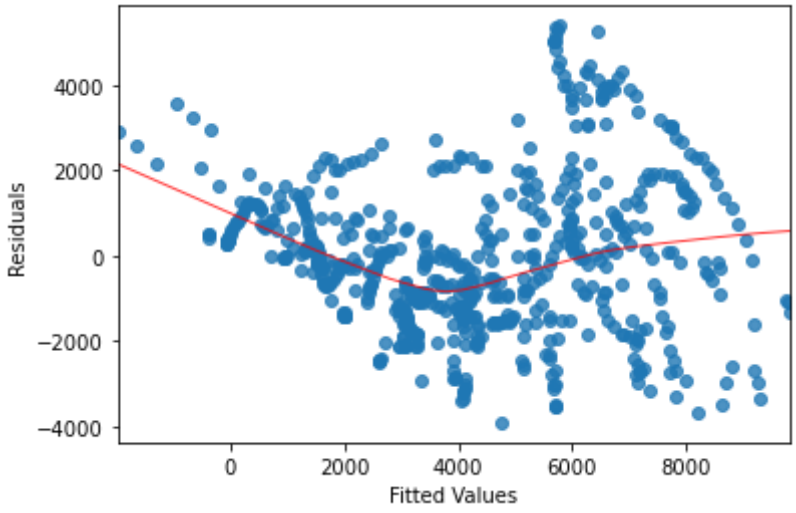
```
model_full_mlr1 = sm.OLS(y, X).fit()
```

In [227…

```
#fitted values
model_fitted_vals1 = model_full_mlr1.fittedvalues
#model residuals
model_residuals1 = model_full_mlr1.resid
#standardised residuals
model_norm_residuals1 = model_full_mlr1.get_influence().resid_studentized_internal
```

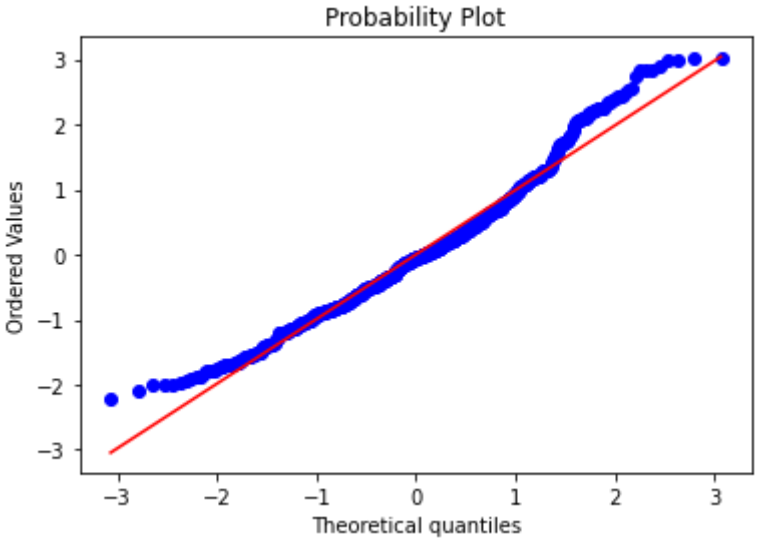
In [228…

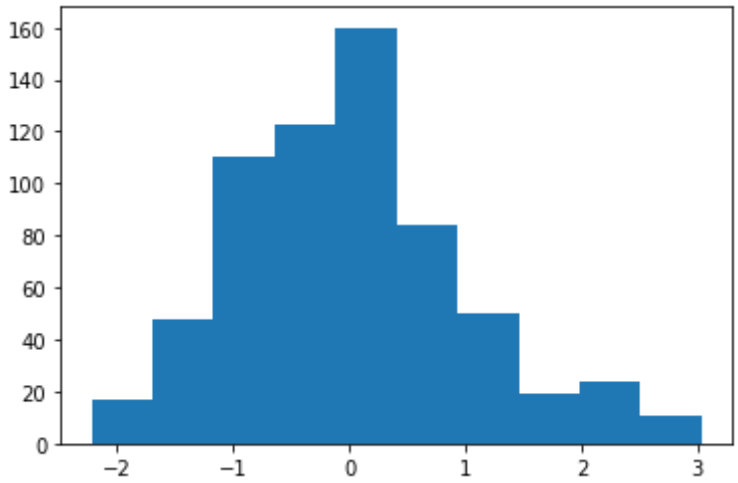
```
sns.regplot(x=model_fitted_vals1,y=model_residuals1,
 ci=False,lowess=True,
 line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8})
plt.xlabel("Fitted Values")
plt.ylabel("Residuals")
plt.show()
```



In [229…

```
stats.probplot(model_norm_residuals1, plot=sns.mpl.pyplot)
plt.show()

plt.hist(model_norm_residuals1)
plt.show()
```





i.i.d: Residuals are independently and identically distributed –random

scatter.

In [230…

```
model_full_mlr1.summary()
```

Out[230…

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	**R-squared:**	0.654
Model:	OLS	**Adj. R-squared:**	0.648
Method:	Least Squares	**F-statistic:**	119.8
Date:	Sat, 11 Dec 2021	**Prob (F-statistic):**	5.19e-139
Time:	10:31:15	**Log-Likelihood:**	-5747.8
No. Observations:	646	**AIC:**	1.152e+04
Df Residuals:	635	**BIC:**	1.157e+04
Df Model:	10		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-1160.6356	553.216	-2.098	0.036	-2246.989	-74.282
people_vaccinated_per_hundred	237.8681	8.455	28.134	0.000	221.265	254.471
people_fully_vaccinated_per_hundred	-222.6971	8.816	-25.261	0.000	-240.009	-205.386
year_21	2988.3757	749.171	3.989	0.000	1517.223	4459.529
month_6	669.9570	242.607	2.761	0.006	193.547	1146.367
month_8	-1352.1133	280.043	-4.828	0.000	-1902.036	-802.191
month_9	-1729.8410	261.092	-6.625	0.000	-2242.549	-1217.133
month_10	-2039.0891	288.155	-7.076	0.000	-2604.941	-1473.238
country_ind	-2035.7225	236.613	-8.604	0.000	-2500.360	-1571.085
country_irl	2024.8861	224.116	9.035	0.000	1584.787	2464.985
country_bah	2130.4219	231.057	9.220	0.000	1676.694	2584.150
conti_eur	-1255.3350	243.565	-5.154	0.000	-1733.625	-777.045

Omnibus:	35.237	**Durbin-Watson:**	0.697
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	39.557
Skew:	0.582	**Prob(JB):**	2.57e-09
Kurtosis:	3.340	**Cond. No.**	7.84e+16

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 3.03e-28. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

The equation of line, y = -1160.63 + 237.86 people_vaccinated_per_hundred - 222.69 people_fully_vaccinated_per_hundred + 2988.37 year_21 + 669.95 month_6 -1352.11 month_8 -1729.84 month_9 -2039.08 month_10 -2035.72 country_ind + 2024.88 country_irl + 2130.42 country_bah -1255.33 conti_eur

-1160.63 is the estimated intercept, that is average y when all x are 0.

For every count increase in people_vaccinated_per_hundred, the average increase in daily_vaccinations_per_million will be 237.86 counts, keeping all other variables the same.

For every count increase in people_fully_vaccinated_per_hundred, the average increase in daily_vaccinations_per_million will be - 222.69 , keeping all other variables the same.

For every count increase in year_21, the average increase in daily_vaccinations_per_million will be 2988.37 counts, keeping all other variables the same.

For every count increase in vaccinations in month_6, the average increase in daily_vaccinations_per_million will be 669.95 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_8, the average increase in daily_vaccinations_per_million will be -1352.11 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_9, the average increase in daily_vaccinations_per_million will be -1729.84 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_10, the average increase in daily_vaccinations_per_million will be -2039.08 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_ind, the average increase in daily_vaccinations_per_million will be -2035.72 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_irl, the average increase in daily_vaccinations_per_million will be 2024.88 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_bah, the average increase in daily_vaccinations_per_million will be 2130.42 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in conti_eur, the average increase in daily_vaccinations_per_million will be -1255.33 vaccination counts, keeping all other variables the same.

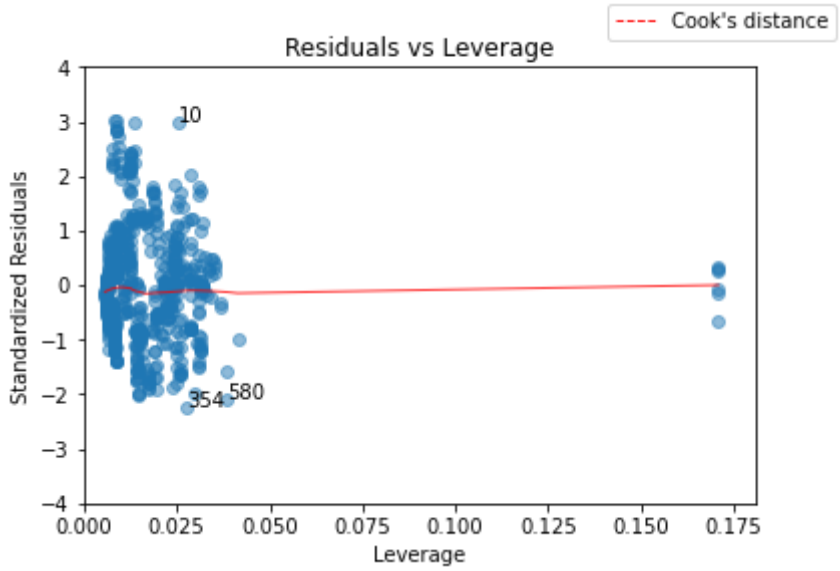
65% variation in daily vaccinations is being explained by this model.

-2035.72 is RSE, the average spread of residuals around the equation.

In [231…
```
np.sqrt(model_full_mlr1.scale)
```

Out[231…
```
1784.8850186108089
```

In [232…
```
def graph(formula, x_range, label=None):
 """
 Helper function for plotting cook's distance lines
 """
 x = x_range
 y = formula(x)
 plt.plot(x, y, label=label, lw=1, ls='--', color='red')
# leverage, from statsmodels internals
model_leverage1 = model_full_mlr1.get_influence().hat_matrix_diag

# cook's distance, from statsmodels internals
model_cooks1 = model_full_mlr1.get_influence().cooks_distance[0]
plot_cooks = plt.figure();
plt.scatter(model_leverage1, model_norm_residuals1, alpha=0.5);
sns.regplot(x=model_leverage1, y=model_norm_residuals1,
 scatter=False,
 ci=False,
 lowess=True,
 line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8});
plot_cooks.axes[0].set_xlim(0, max(model_leverage1)+0.01)
plot_cooks.axes[0].set_ylim(-4, 4)
plot_cooks.axes[0].set_title('Residuals vs Leverage')
plot_cooks.axes[0].set_xlabel('Leverage')
plot_cooks.axes[0].set_ylabel('Standardized Residuals');
# annotations
leverage_top_3 = np.flip(np.argsort(model_cooks1), 0)[:3]
for i in leverage_top_3:
 plot_cooks.axes[0].annotate(i,
 xy=(model_leverage1[i],
 model_norm_residuals1[i]));
p = len(model_full_mlr1.params) # number of model parameters
graph(lambda x: np.sqrt((0.5 * p * (1 - x)) / x),
 np.linspace(0.001, max(model_leverage1), 50),
 'Cook\'s distance') # 0.5 line
graph(lambda x: np.sqrt((1 * p * (1 - x)) / x),
 np.linspace(0.001, max(model_leverage1), 50)) # 1 line
plot_cooks.legend(loc='upper right');
```



In [233…
```
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

```
pd.Series([variance_inflation_factor(X.values, i)
 for i in range(X.shape[1])],
 index=X.columns)
```

```
C:\Users\Chikku\anaconda3\lib\site-packages\statsmodels\regression\linear_model.py:1
715: RuntimeWarning: divide by zero encountered in double_scalars
  return 1 - self.ssr/self.centered_tss
C:\Users\Chikku\anaconda3\lib\site-packages\statsmodels\stats\outliers_influence.py:
193: RuntimeWarning: divide by zero encountered in double_scalars
  vif = 1. / (1. - r_squared_i)
```

Out[233…

```
const                                 0.000000
people_vaccinated_per_hundred         9.376282
people_fully_vaccinated_per_hundred   9.932258
year_21                               1.047226
month_6                               1.080051
month_8                               1.093803
month_9                               1.199331
month_10                              1.135856
country_ind                                inf
country_irl                           2.335154
country_bah                                inf
conti_eur                                  inf
dtype: float64
```

Since 2 variables show higher multicollinearity, removing the highr one and running model again.

In [234…

```
# removing insignificant variables and performing mlr again


X = sel_data[["people_vaccinated_per_hundred", "year_21", "month_6","month_8","month
X = sm.add_constant(X)
y = sel_data['daily_vaccinations_per_million']
X.head()
```

Out[234…

	const	people_vaccinated_per_hundred	year_21	month_6	month_8	month_9	month_10	coun
295	1.0	63.33	1	0	0	0	0	
296	1.0	63.39	1	0	0	0	0	
294	1.0	62.98	1	0	0	0	0	
293	1.0	62.93	1	0	0	0	0	
292	1.0	62.88	1	0	0	0	0	

In [235…

```
model_full_mlr2 = sm.OLS(y, X).fit()
```

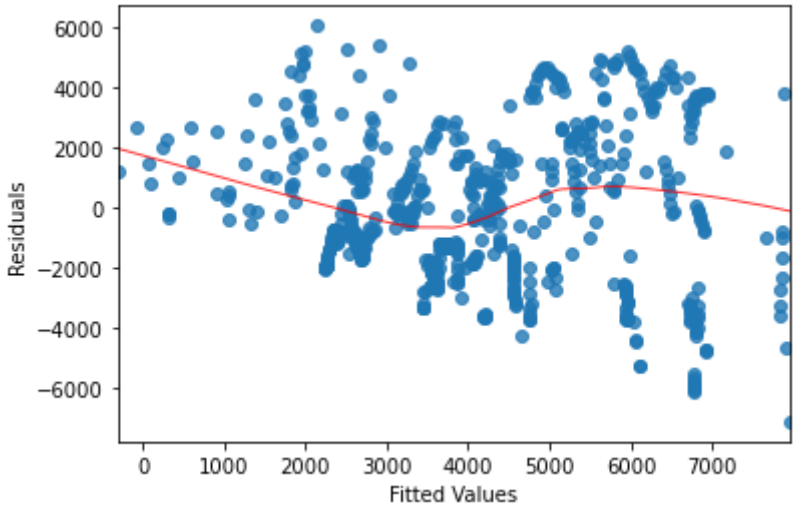
In [236…

```
#fitted values
model_fitted_vals2 = model_full_mlr2.fittedvalues
#model residuals
model_residuals2 = model_full_mlr2.resid
#standardised residuals
model_norm_residuals2 = model_full_mlr2.get_influence().resid_studentized_internal
```

In [237…

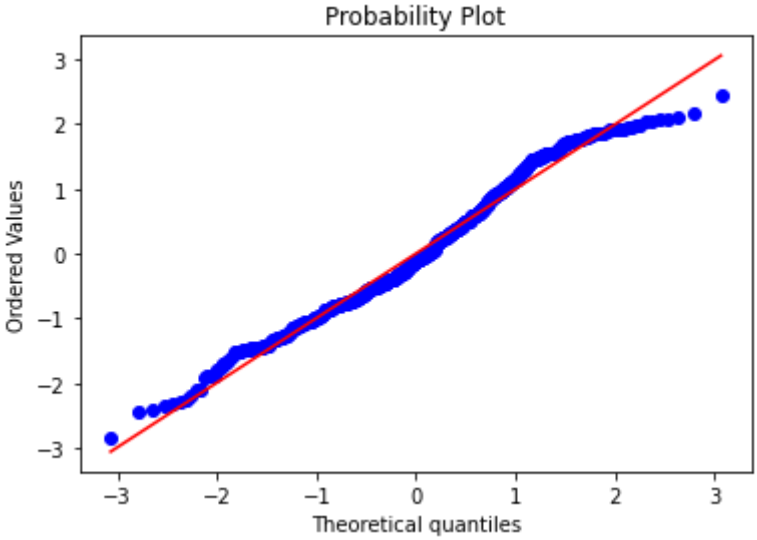
```
sns.regplot(x=model_fitted_vals2,y=model_residuals2,
 ci=False,lowess=True,
 line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8})
```

```
plt.xlabel("Fitted Values")
plt.ylabel("Residuals")
plt.show()
```



In [238...

```
stats.probplot(model_norm_residuals2, plot=sns.mpl.pyplot)
plt.show()
```



In [272...

```
model_full_mlr2.summary()
```

Out[272...

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	**R-squared:**	0.306
Model:	OLS	**Adj. R-squared:**	0.296
Method:	Least Squares	**F-statistic:**	31.09
Date:	Sat, 11 Dec 2021	**Prob (F-statistic):**	3.64e-45
Time:	19:12:19	**Log-Likelihood:**	-5972.4
No. Observations:	646	**AIC:**	1.196e+04
Df Residuals:	636	**BIC:**	1.201e+04
Df Model:	9		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-51.2541	780.246	-0.066	0.948	-1583.424	1480.916
people_vaccinated_per_hundred	43.7982	4.995	8.768	0.000	33.990	53.607
year_21	3135.7701	1059.932	2.958	0.003	1054.381	5217.160
month_6	1184.2395	342.042	3.462	0.001	512.572	1855.908
month_8	-948.7070	395.574	-2.398	0.017	-1725.496	-171.918
month_9	-2196.7951	368.479	-5.962	0.000	-2920.377	-1473.213
month_10	-2584.3614	406.550	-6.357	0.000	-3382.704	-1786.019
country_ind	-869.8043	328.340	-2.649	0.008	-1514.567	-225.042
country_irl	503.0906	305.421	1.647	0.100	-96.665	1102.846
country_bah	956.8360	320.234	2.988	0.003	327.992	1585.680
conti_eur	-138.2858	338.881	-0.408	0.683	-803.746	527.175

Omnibus:	21.962	**Durbin-Watson:**	0.451
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	13.042
Skew:	0.189	**Prob(JB):**	0.00147
Kurtosis:	2.415	**Cond. No.**	2.23e+17

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 2.35e-29. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

The equation of line, y = -51.25 + 43.79 people_vaccinated_per_hundred + 3135.77 year_21 + 1184.23 month_6 -948.70 month_8 -2196.79 month_9 -2584.36 month_10 -869.80 country_ind + 503.09 country_irl + 956.83 country_bah -138.28 conti_eur

-1160.63 is the estimated intercept, that is average y when all x are 0.

For every count increase in people_vaccinated_per_hundred, the average increase in daily_vaccinations_per_million will be 43.79 counts, keeping all other variables the same.

For every count increase in year_21, the average increase in daily_vaccinations_per_million will be 3135.77 counts, keeping all other variables the same.

For every count increase in vaccinations in month_6, the average increase in daily_vaccinations_per_million will be 1184.23 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_8, the average increase in daily_vaccinations_per_million will be -948.70 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_9, the average increase in daily_vaccinations_per_million will be -2196.79 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_10, the average increase in daily_vaccinations_per_million will be 2584.36 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_ind, the average increase in daily_vaccinations_per_million will be -869.80 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_irl, the average increase in daily_vaccinations_per_million will be 503.09 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_bah, the average increase in daily_vaccinations_per_million will be 956.83 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in conti_eur, the average increase in daily_vaccinations_per_million will be -138.28 vaccination counts, keeping all other variables the same.

30% variation in daily vaccinations is being explained by this model.

In [240…

```
# removing insignificant variables.

X = sel_data[["people_vaccinated_per_hundred","year_21", "month_6","month_8","month_
X = sm.add_constant(X)
y = sel_data['daily_vaccinations_per_million']
X.head()
```

Out[240…

	const	people_vaccinated_per_hundred	year_21	month_6	month_8	month_9	month_10	coun
295	1.0	63.33	1	0	0	0	0	
296	1.0	63.39	1	0	0	0	0	
294	1.0	62.98	1	0	0	0	0	
293	1.0	62.93	1	0	0	0	0	
292	1.0	62.88	1	0	0	0	0	

In [241…

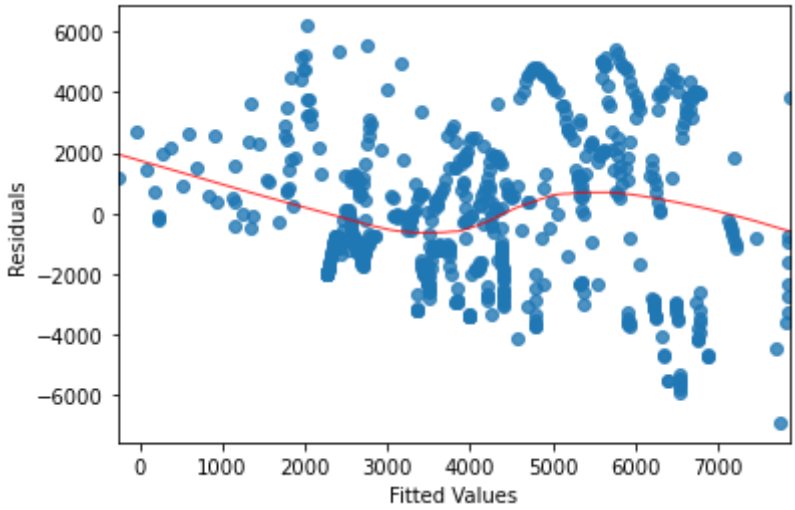
```
model_full_mlr3 = sm.OLS(y, X).fit()
```

In [242…

```
#fitted values
model_fitted_vals3 = model_full_mlr3.fittedvalues
#model residuals
model_residuals3 = model_full_mlr3.resid
#standardised residuals
model_norm_residuals3 = model_full_mlr3.get_influence().resid_studentized_internal
```

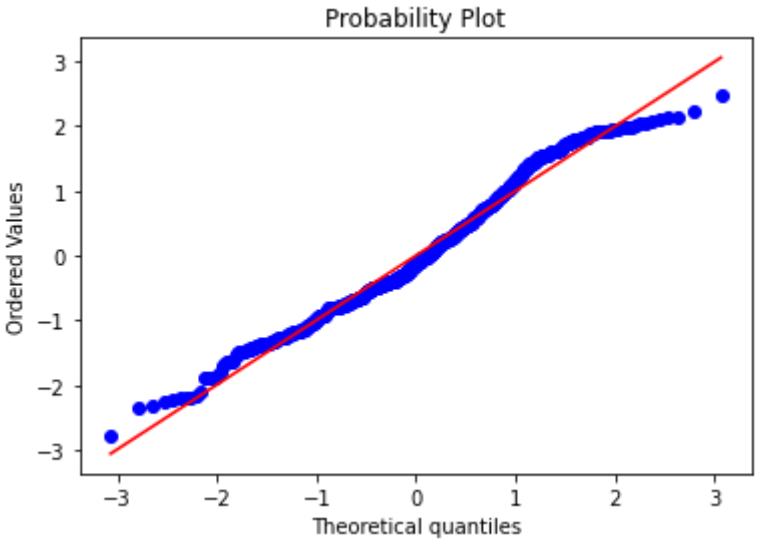
In [243…

```
sns.regplot(x=model_fitted_vals3,y=model_residuals3,
 ci=False,lowess=True,
 line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8})
plt.xlabel("Fitted Values")
plt.ylabel("Residuals")
plt.show()
```



In [244…

```
stats.probplot(model_norm_residuals3, plot=sns.mpl.pyplot)
plt.show()
```



In [245…

```
model_full_mlr3.summary()
```

Out[245…

OLS Regression Results

Dep. Variable:	daily_vaccinations_per_million	**R-squared:**	0.303
Model:	OLS	**Adj. R-squared:**	0.294
Method:	Least Squares	**F-statistic:**	34.54
Date:	Sat, 11 Dec 2021	**Prob (F-statistic):**	2.24e-45
Time:	10:31:21	**Log-Likelihood:**	-5973.8
No. Observations:	646	**AIC:**	1.197e+04
Df Residuals:	637	**BIC:**	1.201e+04

Df Model:	8
Covariance Type:	nonrobust

	coef	std err	t	P>|t|	[0.025	0.975]
const	229.5946	1043.210	0.220	0.826	-1818.952	2278.142
people_vaccinated_per_hundred	42.1816	4.904	8.601	0.000	32.551	51.812
year_21	3115.4994	1061.285	2.936	0.003	1031.459	5199.540
month_6	1212.7513	342.063	3.545	0.000	541.044	1884.458
month_8	-951.3802	396.102	-2.402	0.017	-1729.204	-173.556
month_9	-2133.5381	366.965	-5.814	0.000	-2854.145	-1412.931
month_10	-2565.1734	406.929	-6.304	0.000	-3364.258	-1766.089
country_ind	-1108.2326	244.216	-4.538	0.000	-1587.799	-628.667
country_bah	765.2781	302.640	2.529	0.012	170.986	1359.570

Omnibus:	21.412	**Durbin-Watson:**	0.447
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	14.393
Skew:	0.240	**Prob(JB):**	0.000749
Kurtosis:	2.448	**Cond. No.**	631.

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

The equation of line, y = 229.59 + 42.18 people_vaccinated_per_hundred + 3115.49 year_21 + 1212.75 month_6 -951.38 month_8 -2133.53 month_9 -2565.17 month_10 -1108.23 country_ind + 765.27 country_bah

229.59 is the estimated intercept, that is average y when all x are 0.

For every count increase in people_vaccinated_per_hundred, the average increase in daily_vaccinations_per_million will be 42.18counts, keeping all other variables the same.

For every count increase in year_21, the average increase in daily_vaccinations_per_million will be 115.49 counts, keeping all other variables the same.

For every count increase in vaccinations in month_6, the average increase in daily_vaccinations_per_million will be 1212.75 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_8, the average increase in daily_vaccinations_per_million will be -951.38 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_9, the average increase in daily_vaccinations_per_million will be -2133.53 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in month_10, the average increase in daily_vaccinations_per_million will be -2565.17 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_ind, the average increase in daily_vaccinations_per_million will be -1108.23 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_irl, the average increase in daily_vaccinations_per_million will be 503.09 vaccination counts, keeping all other variables the same.

For every count increase in vaccinations in country_bah, the average increase in daily_vaccinations_per_million will be 956.83 vaccination counts, keeping all other variables the same.

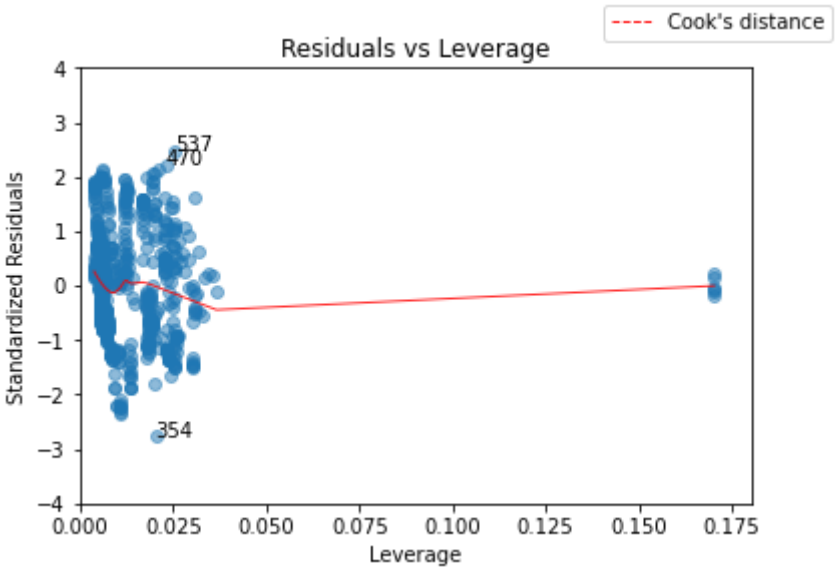
For every count increase in vaccinations in conti_eur, the average increase in daily_vaccinations_per_million will be -138.28 vaccination counts, keeping all other variables the same.

30% variation in daily vaccinations is being explained by this model.

In [246…

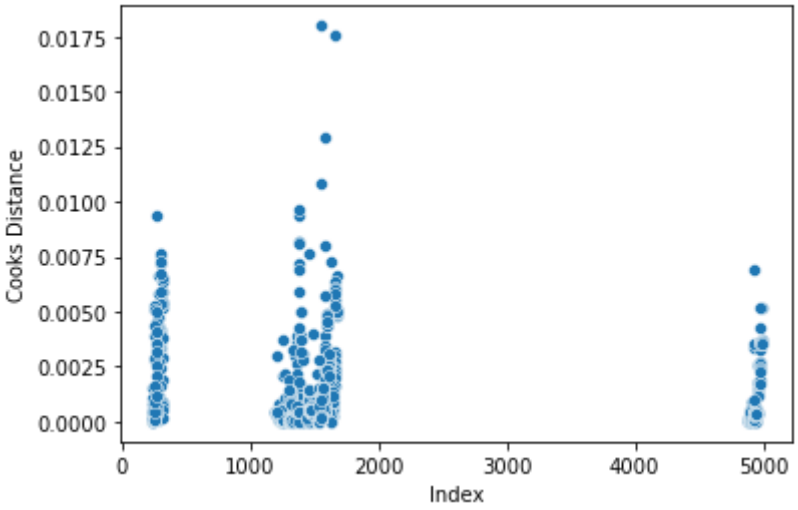
```
def graph(formula, x_range, label=None):
 """
 Helper function for plotting cook's distance lines
 """
 x = x_range
 y = formula(x)
 plt.plot(x, y, label=label, lw=1, ls='--', color='red')
# leverage, from statsmodels internals
model_leverage3 = model_full_mlr3.get_influence().hat_matrix_diag

# cook's distance, from statsmodels internals
model_cooks3 = model_full_mlr3.get_influence().cooks_distance[0]
plot_cooks = plt.figure();
plt.scatter(model_leverage3, model_norm_residuals3, alpha=0.5);
sns.regplot(x=model_leverage3, y=model_norm_residuals3,
 scatter=False,
 ci=False,
 lowess=True,
 line_kws={'color': 'red', 'lw': 1, 'alpha': 0.8});
plot_cooks.axes[0].set_xlim(0, max(model_leverage3)+0.01)
plot_cooks.axes[0].set_ylim(-4, 4)
plot_cooks.axes[0].set_title('Residuals vs Leverage')
plot_cooks.axes[0].set_xlabel('Leverage')
plot_cooks.axes[0].set_ylabel('Standardized Residuals');
# annotations
leverage_top_3 = np.flip(np.argsort(model_cooks3), 0)[:3]
for i in leverage_top_3:
 plot_cooks.axes[0].annotate(i,
 xy=(model_leverage3[i],
 model_norm_residuals3[i]));
p = len(model_full_mlr3.params) # number of model parameters
graph(lambda x: np.sqrt((0.5 * p * (1 - x)) / x),
 np.linspace(0.001, max(model_leverage3), 50),
 'Cook\'s distance') # 0.5 line
graph(lambda x: np.sqrt((1 * p * (1 - x)) / x),
```

```
 np.linspace(0.001, max(model_leverage3), 50)) # 1 line
plot_cooks.legend(loc='upper right');
```



In [247…

```
sns.scatterplot(x=sel_data.index, y=model_cooks3)
plt.xlabel('Index')
plt.ylabel('Cooks Distance')
plt.show()
```



no cooks distance greater than 0.5

In [248…

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
pd.Series([variance_inflation_factor(X.values, i)
 for i in range(X.shape[1])],
 index=X.columns)
```

Out[248…

```
const                            109.943415
people_vaccinated_per_hundred      1.571727
year_21                            1.047022
month_6                            1.069699
month_8                            1.090228
month_9                            1.180358
month_10                           1.128555
country_ind                        1.350147
country_bah                        1.140304
dtype: float64
```

All variables are significant.

In [249…
```
np.sqrt(model_full_mlr3.scale)
```

Out[249… 2528.7360968806156

In [250…
```
# assumptions ?? write here
```

In [251…
```
anovaResults = anova_lm(model_full_mlr3, model_full_mlr2)
anovaResults
```

Out[251…

	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)
0	637.0	4.073300e+09	0.0	NaN	NaN	NaN
1	636.0	4.055997e+09	1.0	1.730356e+07	2.713282	0.10001

In [252…
```
anovaResults = anova_lm(model_full_mlr2, model_full_mlr)
anovaResults
```

Out[252…

	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)
0	636.0	4.055997e+09	0.0	NaN	NaN	NaN
1	627.0	1.846449e+09	9.0	2.209548e+09	83.366397	4.178129e-101

In [253…
```
anovaResults = anova_lm(model_full_mlr3, model_full_mlr)
anovaResults
```

Out[253…

	df_resid	ssr	df_diff	ss_diff	F	Pr(>F)
0	637.0	4.073300e+09	0.0	NaN	NaN	NaN
1	627.0	1.846449e+09	10.0	2.226851e+09	75.617335	7.122837e-101

All the variables are significant

In [ ]:
```
```