COVID-19 is a highly infectious disease caused by the SARS-CoV-2 virus that was found in Wuhan, China in December 2019. The disease quickly spread over the world, resulting in a pandemic. [illegible] The dataset contains data on the number of persons who received the COVID-19 vaccination in different parts of the United Kingdom. [illegible]

```
In [1]: # importing the required libraries
        import numpy as np
        import pandas as pd
```

```
In [2]: # Reading the dataset into a pandas dataframe
        df = pd.read_csv([illegible])

        # Displaying the first three rows of the dataset
        df.head(3)
```

	areaCode	areaName	year	month	Quarter	day	WorkingDay	FirstDose	SecondDose	ThirdDose
0	E92000001	England	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
1	E92000001	England	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
2	E92000001	England	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Descriptive statistics for the dataset

```
In [3]: # Generating descriptive statistics for the dataset with describe () function
        print(df.describe())
```

[illegible]

Evaluating any records that have missing values

```
In [4]: # Checking for missing values
        print(df.isnull().sum())
```

[illegible]

```
In [5]: # Checking for duplicate rows
        duplicate_rows=df[df.duplicated()]
        print("number of duplicate rows: ", duplicate_rows.shape)
```

[illegible]

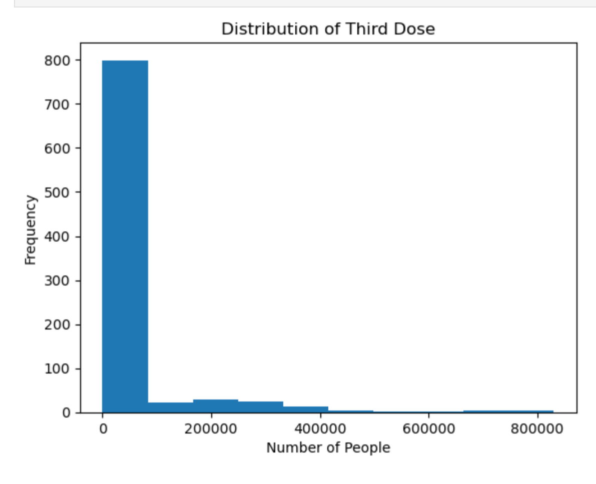
HANDLING MISSING VALUES

[illegible]

GRAPHS

CONTINUOUS/NUMERICAL VARIABLE'S DISTRIBUTION GRAPH

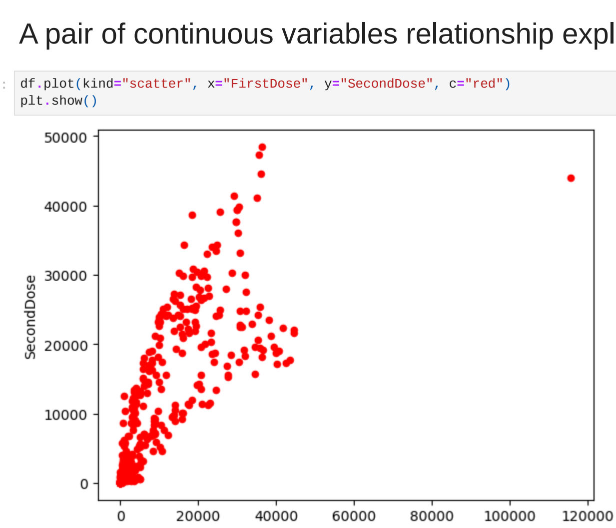
```
# Histogram of Third dose
plt.hist(df["ThirdDose"])
plt.title("Distribution of Third Dose")
plt.xlabel("Number of People")
plt.ylabel("Frequency")
plt.show()
```



The above code generates a histogram for the distribution of the number of people who have taken the third dose of the vaccine. [illegible]

A pair of continuous variables relationship explained through scatter plot

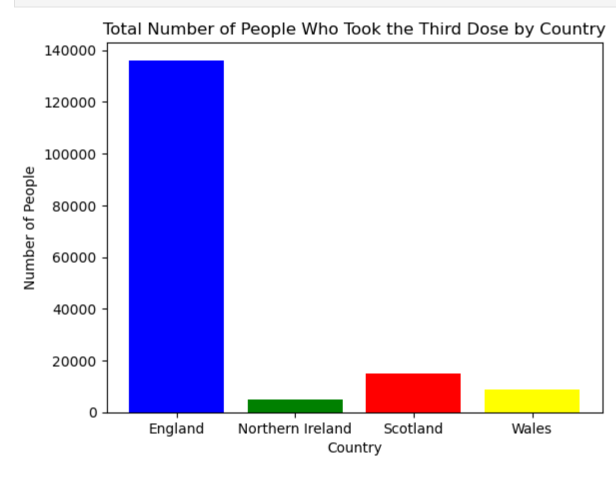
```
df.plot(kind="scatter", x="ThirdDose", y="SecondDose", c="red")
plt.show()
```



[illegible]

A categorical variable and a continuous variable's relationship explained through bar graph

[illegible]



The code organises the data by the areaName column, which represents the various regions or nations in the UK, and then uses the sum() method on the ThirdDose column to get the total number of persons who took the third dose in each country.

The code's output shows a graphic depiction of the distribution of Third dosage vaccinations across the UK's regions. The map allows us to compare the number of people who received the Third dosage in other countries and discover any areas with greater or lower immunisation rates.

Showing unique values of a categorical variable and their frequencies

```
In [14]: # Display unique values of the 'areaName' variable and their frequencies
         region_counts = df['areaName'].value_counts()
         print(region_counts)
```

```
England             236
Northern Ireland    235
Scotland            222
Wales               210
Name: areaName, dtype: int64
```

```
In [15]: # Display unique values of the 'year' variable and their frequencies
         year_counts = df['year'].value_counts()
         print(year_counts)
```

```
2022.0    566
2021.0    337
Name: year, dtype: int64
```

```
In [16]: # Display unique values of the 'Quarter' variable and their frequencies
         quarter_counts = df['Quarter'].value_counts()
         print(quarter_counts)
```

```
Q1    360
Q4    335
Q2    206
Q3      2
Name: Quarter, dtype: int64
```

```
In [17]: # Display unique values of the 'month' variable and their frequencies
         month_counts = df['month'].value_counts()
         print(month_counts)
```

```
3     124
1     124
12    124
4     120
11    120
2     112
10     91
5      86
9       2
Name: month, dtype: int64
```

```
In [18]: # Display unique values of the 'areaCode' variable and their frequencies
         areaCode_counts = df['areaCode'].value_counts()
         print(areaCode_counts)
```

```
E92000001    236
N92000002    235
S92000003    222
W92000004    210
Name: areaCode, dtype: int64
```

Table of two potentially linked categorical variables and conducting a statistical test for determining their independence and interpreting the results.

To create a contingency table we can use the cross_tab function, supplying the two variables which are areaName and Quarter.

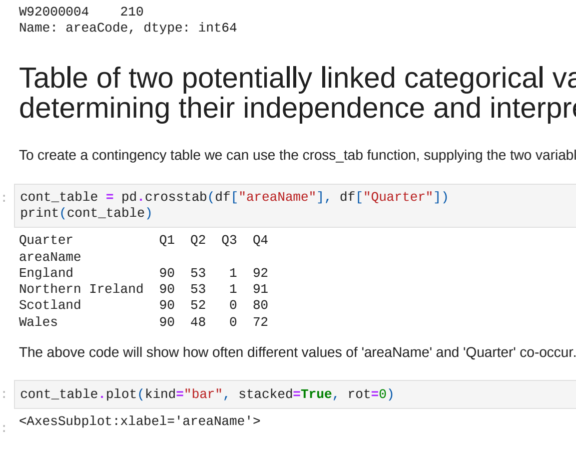
```
In [19]: cont_table = pd.crosstab(df["areaName"], df["Quarter"])
         print(cont_table)
```

```
Quarter           Q1  Q2  Q3  Q4
areaName
England           90  53   1  92
Northern Ireland  90  53   1  91
Scotland          90  52   0  80
Wales             90  48   0  72
```

The above code will show how often different values of 'areaName' and 'Quarter' co-occur.

```
In [20]: cont_table.plot(kind="bar", stacked=True, rot=0)
```

```
Out[20]: <AxesSubplot:xlabel='areaName'>
```



The plot suggests that there were very few vaccination done in the quarter 3 which were done in England and Northen Ireland and not in Scotland and Wales

To perform a statistical test of the independence between them and interpret the results we shall use the Chi-square test .The chi-square test is a statistical hypothesis test that compares the observed frequencies of categorical data to the expected frequencies. It is used to determine whether there is a significant dependancy between two categorical variables.

Our null hypothesis is that there is no dependancy between the two catagorical values. The alternative hypothesis is that there is dependancy.

$H0$:μ=0

HA:μ≠0

We will use the α=0.05 as significance level.

To run a chi-square test on the contingency table, we input it directly into the chi2_contingency function.

from scipy import stats #importing the required library chi2, p_val, dof, expected = stats.chi2_contingency(cont_table) print(f"p-value: {p_val}")

```
In [21]: from scipy import stats #importing the required library
         chi2, p_val, dof, expected = stats.chi2_contingency(cont_table)
         print(f"p-value: {p_val}")
```

```
p-value: 0.9346962560335181
```

The p-value is greater than the significance level of 0.05. Therefore, we fail to reject the null hypothesis. In other words, we found evidence that there is no dependency between the areaName and Quarter in which the doses were taken.

Creating subsets based on two or more criteria and to present descriptive statistics on those subsets.

```
In [22]: subset1 = df[(df['month'] > 5) & (df['day'] == "Mon")]
         subset1.describe()
         #subset of data is created based on two criteria: "month" being greater than 5 and "day" being equal to 'Mon'
```

Out[22]:

	year	month	FirstDose	SecondDose	ThirdDose
count	47.0	47.000000	47.000000	47.000000	47.000000
mean	2021.0	11.106383	8058.297872	6873.489362	87614.914894
std	0.0	0.758547	12059.142229	10035.937306	138325.113486
min	2021.0	10.000000	0.000000	279.000000	1519.000000
25%	2021.0	11.000000	840.500000	899.000000	13976.000000
50%	2021.0	11.000000	1273.000000	1318.000000	28328.000000
75%	2021.0	12.000000	16651.500000	15546.000000	122038.000000
max	2021.0	12.000000	36092.000000	44655.000000	753793.000000

```
In [23]: subset2 = df[(df['areaName'] =="England") & (df['month'] >3)]
         subset2.describe()
         #subset of data is created based on two criteria: "areaName" being equal to 'England' and "month" being greater than '3'
```

Out[23]:

	year	month	FirstDose	SecondDose	ThirdDose
count	146.000000	146.000000	146.000000	146.000000	146.000000
mean	2021.363014	8.602740	21278.815068	18096.849315	194358.113494
std	0.482524	3.242986	13766.879726	10584.163870	194278.205850
min	2021.000000	4.000000	955.000000	916.000000	2287.000000
25%	2021.000000	5.000000	9950.250000	9218.750000	13372.500000
50%	2021.000000	10.000000	20409.000000	18007.500000	176389.000000
75%	2022.000000	11.000000	30364.000000	24241.750000	305387.250000
max	2022.000000	12.000000	115551.000000	48491.000000	830403.000000

HYPOTHESIS TESTING

The null hypothesis is that there is no difference between number of Third dose taken in England and Scotland:

$H0$:μ=0

The alternative hypothesis is that there is a difference between number of Third dose taken in England and Scotland:

HA:μ≠0

where µ is the mean difference in doses taken in England and Scotland

```
In [24]: # select the third dose of "England"
         eng = df[df['areaName'] == 'England']['ThirdDose']

         #mean ThirdDoses in England
         eng.mean()
```

```
Out[24]: 136112.4854667623
```

```
In [25]: # select the third dose  of "Scotland"
         scot = df[df['areaName'] == 'Scotland']['ThirdDose']

         # mean ThirdDoses in Scotland
         scot.mean()
```

```
Out[25]: 14923.583649350909
```

We use the paired sample t-test. We will use the α=0.05 significance level.

The ttest_rel function takes two arguments, where one is "England" values and the other one is "Scotland values in pairs. The function always returns the p-value for a two-tail test.

```
In [26]: t_val, p_val = stats.ttest_ind(eng, scot)

         print(f"t-value: {t_val}, p-value: {p_val}")

         if p_val < 0.05:
             print('There is evidence of a significant Difference.')
         else:
             print('There is no evidence of a significant Difference.')
```

```
t-value: 10.444696488385551, p-value: 4.857097866858563e-23
There is evidence of a significant Difference.
```

The p-value is less than the significance level (α=0.05), thus we reject null hypothisis. Therefore, we can conclude that there is a significant difference betweeen the means of Third dose taken in England and Scotland

Creating one or more tables that group the data by a certain categorical variable and display summarized information for each group

```
In [27]: group = df.groupby("areaCode")#using groupby function to form groups.
         m = group["month"].mean()
         m
```

```
Out[27]: areaCode
         E92000001    6.084746
         N92000002    6.068085
         S92000003    5.864865
         W92000004    5.723810
         Name: month, dtype: float64
```

```
In [28]: group = df.groupby("areaName")
         fd= group["FirstDose"].mean()
         fd
```

```
Out[28]: areaName
         England             16869.427966
         Northern Ireland      497.302128
         Scotland             1188.974429
         Wales                 726.514143
         Name: FirstDose, dtype: float64
```

```
In [29]: group = df.groupby("month")
         sd= group["SecondDose"].mean()
         sd
```

```
Out[29]: month
         1      8157.178431
         2      5122.776786
         3      3498.016129
         4      2631.266667
         5      1998.988372
         9     15591.000000
         10     8423.131868
         11     5730.608333
         12     8436.235894
         Name: SecondDose, dtype: float64
```

```
In [30]: group = df.groupby("year")
         td= group["ThirdDose"].mean()
         td
```

```
Out[30]: year
         2021.0    97584.876620
         2022.0     9823.102473
         Name: ThirdDose, dtype: float64
```

LINEAR REGRESSION MODEL

```
In [31]: # importing required library
         import statsmodels.api as sm
```

```
In [32]: #creating linear regression model
         model=sm.OLS.from_formula('ThirdDose ~ SecondDose ',data=df).fit()
```

ThirdDose is the dependant variable and the SecondDose is the independant variable. By taking into consideration ThirdDose and SecondDose we can perform regression analysis.

```
In [33]: #information about the linear regression
         print(model.summary())
```

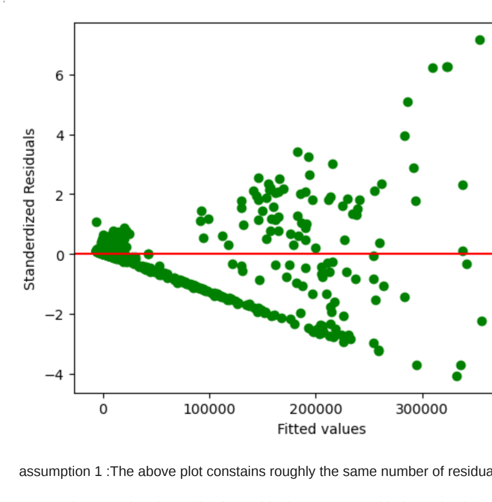
```
                            OLS Regression Results
==============================================================================
Dep. Variable:              ThirdDose   R-squared:                       0.588
Model:                            OLS   Adj. R-squared:                  0.588
Method:                 Least Squares   F-statistic:                     1287.
Date:                Fri, 31 Mar 2023   Prob (F-statistic):          9.79e-176
Time:                        04:58:18   Log-Likelihood:                -11317.
No. Observations:                 903   AIC:                         2.264e+04
Df Residuals:                     901   BIC:                         2.265e+04
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept  -6259.4642   2616.774     -2.392      0.017   -1.14e+04   -1123.783
SecondDose     8.7529      0.244     35.869      0.000       8.274       9.232
==============================================================================
Omnibus:                      358.599   Durbin-Watson:                   0.134
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             5791.497
Skew:                           1.368   Prob(JB):                         0.00
Kurtosis:                      15.101   Cond. No.                     1.26e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.26e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

The Model summary of our regression model says that as there is no difference between R Square and Adjusted R Square, so we take Adjusted R square into consideration which depicts that our model is 58.8%.According to the model, SecondDose has a statistically significant and favourable effect on ThirdDose.

```
In [34]: st_resids=model.get_influence().resid_studentized_internal #getting the standardized residuals (error) for the model
```

```
In [35]: plt.scatter(model.fittedvalues, st_resids, color="green")
         plt.xlabel("Fitted values")
         plt.ylabel("Standerdized Residuals")
         plt.axhline(y=0,color="r",linestyle="-")
```

```
Out[35]: <matplotlib.lines.Line2D at 0x242a4b54e50>
```

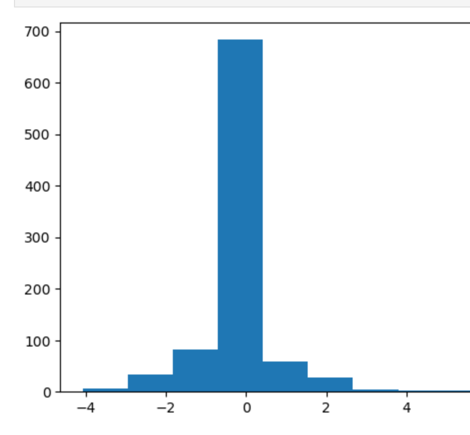


assumption 1 :The above plot constains roughly the same number of residuals above and below 0 so it passes the assumption.

assumption 2 :In the above plot the residuals are scattered independantly so it passes the assumption 2.

assumption 3 : In the above plot there is a pattern of increasing and decreasing values so it fails the assumption 3.

```
In [36]: plt.hist(st_resids, bins=10)
         plt.show()
```



Assumption 4 :In the above histogram residuals are normally distributed forming a bell shabed curve so it passes assumption 4.