Covid_EDA

2024-05-13

Introduction

This EDA examines a dataset that chronicles the daily number of COVID-19 vaccines for each stage of doses administered in the UK from a period between 2021 and 2022. By charting the ebb and flow of doses administered, we aim to understand how temporal factors influenced vaccination trends. These insights will reveal when the shift in the course of the pandemic occurred.

1. Pre-processing Data

```r
#Loads the relevant packages for this EDA
library(readxl)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v readr     2.1.4
## v ggplot2   3.4.3     v stringr   1.5.0
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
suppressWarnings(library(moments))
library(lmtest)
```

```
## Loading required package: zoo
##
## Attaching package: ’zoo’
##
## The following objects are masked from ’package:base’:
##
##     as.Date, as.Date.numeric
```

```
#Reads the Excel sheet into a dataframe and prints the first few rows
df = read_excel('UK_VaccinationsData.xlsx')
head(df)
```

```
## # A tibble: 6 x 10
##   areaName areaCode   year month Quarter day   WorkingDay FirstDose SecondDose
##   <chr>    <chr>     <dbl> <dbl> <chr>   <chr> <chr>          <dbl>      <dbl>
## 1 England  E92000001  2022     5 Q2      Mon   Yes             3034       3857
## 2 England  E92000001  2022     5 Q2      Sun   No              5331       3330
## 3 England  E92000001  2022     5 Q2      Sat   No             13852       9759
## 4 England  E92000001  2022     5 Q2      Fri   Yes             5818       5529
## 5 England  E92000001  2022     5 Q2      Thu   Yes             8439       6968
## 6 England  E92000001  2022     5 Q2      Wed   Yes             4955       5247
## # i 1 more variable: ThirdDose <dbl>
```

```
#Prints the number of rows and columns in the dataframe
cat('There are',nrow(df),'rows and',ncol(df), 'columns in the dataframe')
```

```
## There are 904 rows and 10 columns in the dataframe
```

2. Descriptive Statistics

To get a better idea of the data, we look at the descriptive statistics of the dataframe and its structure

```
summary(df)
```

```
##    areaName           areaCode              year          month       
##  Length:904         Length:904         Min.   :2021   Min.   : 1.000  
##  Class :character   Class :character   1st Qu.:2021   1st Qu.: 2.000  
##  Mode  :character   Mode  :character   Median :2022   Median : 4.000  
##                                        Mean   :2022   Mean   : 5.947  
##                                        3rd Qu.:2022   3rd Qu.:11.000  
##                                        Max.   :2022   Max.   :12.000  
##                                        NA's   :1                      
##    Quarter              day             WorkingDay          FirstDose       
##  Length:904         Length:904         Length:904         Min.   :     0.0  
##  Class :character   Class :character   Class :character   1st Qu.:   338.5  
##  Mode  :character   Mode  :character   Mode  :character   Median :   876.5  
##                                                           Mean   :  4994.3  
##                                                           3rd Qu.:  3653.2  
##                                                           Max.   :115551.0  
##                                                           NA's   :4         
##    SecondDose      ThirdDose
##  Min.   :    0   Min.   :     0
##  1st Qu.:  478   1st Qu.:  1314
##  Median :  971   Median :  6992
##  Mean   : 5574   Mean   : 42530
##  3rd Qu.: 5770   3rd Qu.: 23465
##  Max.   :48491   Max.   :830403
##  NA's   :3       NA's   :6
```

```
str(df)
```

```
## tibble [904 x 10] (S3: tbl_df/tbl/data.frame)
##  $ areaName  : chr [1:904] "England" "England" "England" "England" ...
##  $ areaCode  : chr [1:904] "E92000001" "E92000001" "E92000001" "E92000001" ...
##  $ year      : num [1:904] 2022 2022 2022 2022 2022 ...
##  $ month     : num [1:904] 5 5 5 5 5 5 5 5 5 5 ...
##  $ Quarter   : chr [1:904] "Q2" "Q2" "Q2" "Q2" ...
##  $ day       : chr [1:904] "Mon" "Sun" "Sat" "Fri" ...
##  $ WorkingDay: chr [1:904] "Yes" "No" "No" "Yes" ...
##  $ FirstDose : num [1:904] 3034 5331 13852 5818 8439 ...
##  $ SecondDose: num [1:904] 3857 3330 9759 5529 6968 ...
##  $ ThirdDose : num [1:904] 8747 4767 12335 10692 11701 ...
```

AreaName and AreaCode: These are character columns representing the name and corresponding code of the area/country e.g England/E92000001

Year and Month: These are numeric columns with the year and month as integers. The min and max values show the data spans from 2021 to 2022, with most entries being in 2022 indicated by the median. There is however a missing value in the year column. The descriptive stats for the month column are less telling as it's not clear what months are recorded.

Quarter, Day and WorkingDay: These are character columns representing the quarter of the year (e.g Q2), the day of the week (e.g Mon, Sun) and whether it was a working day or not. The day column goes backwards in the week from Sunday to Monday.

FirstDose, SecondDose, and ThirdDose: These columns are numeric and represent vaccination doses. On average, the third dose was taken the most by a large distance. This indicates that for the recorded period, the vast majority of people were on the third dose. There are missing values in each column.

The descriptive stats for all three doses seem to tell the same story that the distribution is positively skewed as the median is significantly smaller than the mean for each dose. This means the majority of the data is clustered towards the lower values on the left. Furthermore, the max values are notably higher than the third quartiles indicating there are huge outliers which we will investigate.

3. Data Preparation

Before any further analysis, we must check where and how many missing values there are.

```
missing_values = sum(is.na(df))
if(missing_values > 1){
  cat('There are',missing_values,'missing values', '\n')
}else if(missing_values == 0){
    print("There is 1 missing value")
}else{
  print("There are no missing values")
}
```

```
## There are 18 missing values
```

```
colSums(is.na(df))
```

```
##   areaName   areaCode       year      month    Quarter        day WorkingDay 
##          0          0          1          0          1          1          2 
##  FirstDose SecondDose  ThirdDose 
##          4          3          6
```

A. Dealing With Missing Data

Since the month column spans until May 2022 and has no missing values, we impute 2022 into missing values in the months recorded in 2022, and 2021 into all other missing values.

```
#Creates a vector that holds the number values for the months recorded in 2022
months_22 = c(1, 2, 3, 4, 5)
#Accesses year values that are missing and have a month in the 2022 month vector and imputes 2022
df$year[is.na(df$year) & df$month %in% months_22] = 2022
#Accesses all other missing year values and imputes 2021
df$year[is.na(df$year)] = 2021

#Sums the missing values in the column and stores it in the variable
missing_year_values = sum(is.na(df$year))
#Prints how many missing values are in the column
cat('There are now', missing_year_values, 'missing values in this column')
```

```
## There are now 0 missing values in this column
```

Similarly, we impute Q1, Q2, Q3 or Q4 into the missing Quarter value based on which vector the corresponding month is in as there are no missing month values.

```
#Creates vectors that holds the months that fall into each quarter
Q1_months = c(1:3)
Q2_months = c(4:6)
Q3_months = c(7:9)
Q4_months = c(10:12)
#Imputes Q1, Q2, Q3, Q4 into the missing value based on what month the recording was in
df$Quarter[is.na(df$Quarter) & df$month %in% Q1_months] = 'Q1'
df$Quarter[is.na(df$Quarter) & df$month %in% Q2_months] = 'Q2'
df$Quarter[is.na(df$Quarter) & df$month %in% Q3_months] = 'Q3'
df$Quarter[is.na(df$Quarter) & df$month %in% Q4_months] = 'Q4'

#Sums the missing values in the column and stores it in the variable
missing_Q_values = sum(is.na(df$Quarter))
#Prints how many missing values are in the column
cat('There are now', missing_Q_values, 'missing values in this column')
```

```
## There are now 0 missing values in this column
```

Creates vector of weekend days and imputes ‘No’ to missing working day values if the corresponding day is in the vector and ‘Yes’ for any other missing value.

```
#Creates vector that holds the weekend days
weekend = c('Sat', 'Sun')
#Imputes No if the missing values day is in the weekend vector
df$WorkingDay[is.na(df$WorkingDay) & df$day %in% weekend] = 'No'
#Imputes Yes for all other missing values
df$WorkingDay[is.na(df$WorkingDay)] = 'Yes'

#Sums the missing values in the column and stores it in the variable
missing_WD_values = sum(is.na(df$WorkingDay))
#Prints how many missing values are in the column
cat('There are now', missing_WD_values, 'missing values in the WorkingDay column')
```

```
## There are now 0 missing values in the WorkingDay column
```

To fill in the missing day values, I created a vector containing the days of the week and a function that returns the next day in the vector based on the last non-missing day value. The data is grouped by areaName accounting for the date resetting for each country.

```
#Creates a vector for days of the week going from Sun-Mon
days = c('Sun', 'Sat', 'Fri', 'Thu', 'Wed', 'Tue', 'Mon')
#Creates a function to return the next day in the days vector
next_day = function(day) {
  next_index = (match(day, days) %% 7) + 1
  return(days[next_index])
}
#Starts chain of operations to be done on the dataframe
df = df %>%
#Groups by areaName
  group_by(areaName) %>%
#Calls the next day function to fill missing values
#Using the last non-missing value as the input to the function
  mutate(day = ifelse(is.na(day), next_day(lag(day)), day)) %>%
#Ungroups the dataframe
  ungroup()
#Sums the missing values in the column and stores it in the variable
missing_day_values = sum(is.na(df$day))
#Prints how many missing values are in the column
cat('There are now', missing_day_values, 'missing values in this column')
```

```
## There are now 0 missing values in this column
```

In filling the missing dose values, I grouped the data by area since each country could have vastly different daily doses and imputed the mean to minimise the effect on the data.

```
#Starts chain of operations to be done on the dataframe
df = df %>%
#Groups by areaName
  group_by(areaName) %>%
#Imputes the mean into the missing values excluding the missing value
  mutate(FirstDose = ifelse(is.na(FirstDose), mean(FirstDose, na.rm = T), FirstDose),
    SecondDose = ifelse(is.na(SecondDose), mean(SecondDose, na.rm = T), SecondDose),
    ThirdDose = ifelse(is.na(ThirdDose), mean(ThirdDose, na.rm = T), ThirdDose)) %>%
```

```r
#Ungroups the dataframe
  ungroup()
#Sums the missing values in each column and stores it in each variable
missing_FD_values = sum(is.na(df$FirstDose))
missing_SD_values = sum(is.na(df$SecondDose))
missing_TD_values = sum(is.na(df$ThirdDose))
#Sets if statement to print if the condition of each variable being 0 is met
if (missing_FD_values == 0 && missing_SD_values == 0 && missing_TD_values == 0){
  print('There are now 0 missing values in the vaccine dose columns')
#Sets else statement to print if the condition is not met
}else{
  print('There are still missing values')
}
```

```
## [1] "There are now 0 missing values in the vaccine dose columns"
```

Checks all columns again.

```r
#Sums up the missing values in each column and prints the result for each
colSums(is.na(df))
```

```
##   areaName   areaCode       year      month    Quarter        day WorkingDay 
##          0          0          0          0          0          0          0 
##  FirstDose SecondDose  ThirdDose 
##          0          0          0
```

B. Re-formatting categorical variables

The initial summary statistics only tells us that the categorical variables are characters which is not very useful so we will convert them to factors to get better insights when we re-run the summary statistics with no missing data. We will also be treating year and month as categorical values here to see the number of observations from each year and month.

```r
#Creates a vector for the categorical value columns
categorical_cols = c('areaName', 'areaCode', 'year', 'month', 'Quarter', 'day', 'WorkingDay')
#Converts columns to factors
df[categorical_cols] = lapply(df[categorical_cols], as.factor)
#Prints descriptive statistics
summary(df)
```

```
##               areaName        areaCode     year           month     Quarter  
##  England         :236   E92000001:236   2021:338   1      :124   Q1:360  
##  Northern Ireland:236   N92000002:236   2022:566   3      :124   Q2:206  
##  Scotland        :222   S92000003:222              12     :124   Q3:  2  
##  Wales           :210   W92000004:210              4      :120   Q4:336  
##                                                    11     :120           
##                                                    2      :112           
##                                                    (Other):180           
##   day      WorkingDay   FirstDose        SecondDose        ThirdDose   
##  Fri:130   No :260    Min.   :     0   Min.   :    0.0   Min.   :     0  
##  Mon:129   Yes:644    1st Qu.:   342   1st Qu.:  481.8   1st Qu.:  1317
##  Sat:130              Median :   871   Median :  970.0   Median :  7394
##  Sun:130              Mean   :  4976   Mean   : 5558.2   Mean   : 42447
##  Thu:130              3rd Qu.:  3634   3rd Qu.: 5739.2   3rd Qu.: 23360
##  Tue:127              Max.   :115551   Max.   :48491.0   Max.   :830403
##  Wed:128
```

```
#Counts the frequency of unique values in the month column
unique_months = df %>% count(df$month)
print(unique_months)
```

```
## # A tibble: 9 x 2
##   ‘df$month‘     n
##   <fct>      <int>
## 1 1            124
## 2 2            112
## 3 3            124
## 4 4            120
## 5 5             86
## 6 9              2
## 7 10            92
## 8 11           120
## 9 12           124
```

AreaName and AreaCode: England and Northern Ireland have the most entries with 236, followed by Scotland (222) and Wales (210).

Year: There are 338 entries in 2021 and 566 in 2022.

Month: January, March and December have the most entries with 124 each while April and November have 120. February has 112 entries while September(2), October(92) and May(86) have 180 entries combined.

Quarter: Q1 (360) and Q4 (336) have the most entries followed by Q2 (206) while Q3 only has 2 entries as data was only collected from late September.

Day: The day data ranges from 127-130 entries per day.

WorkingDay: Most of the data was recorded on a working day as expected and the split is proportional.

3. Visualising Data

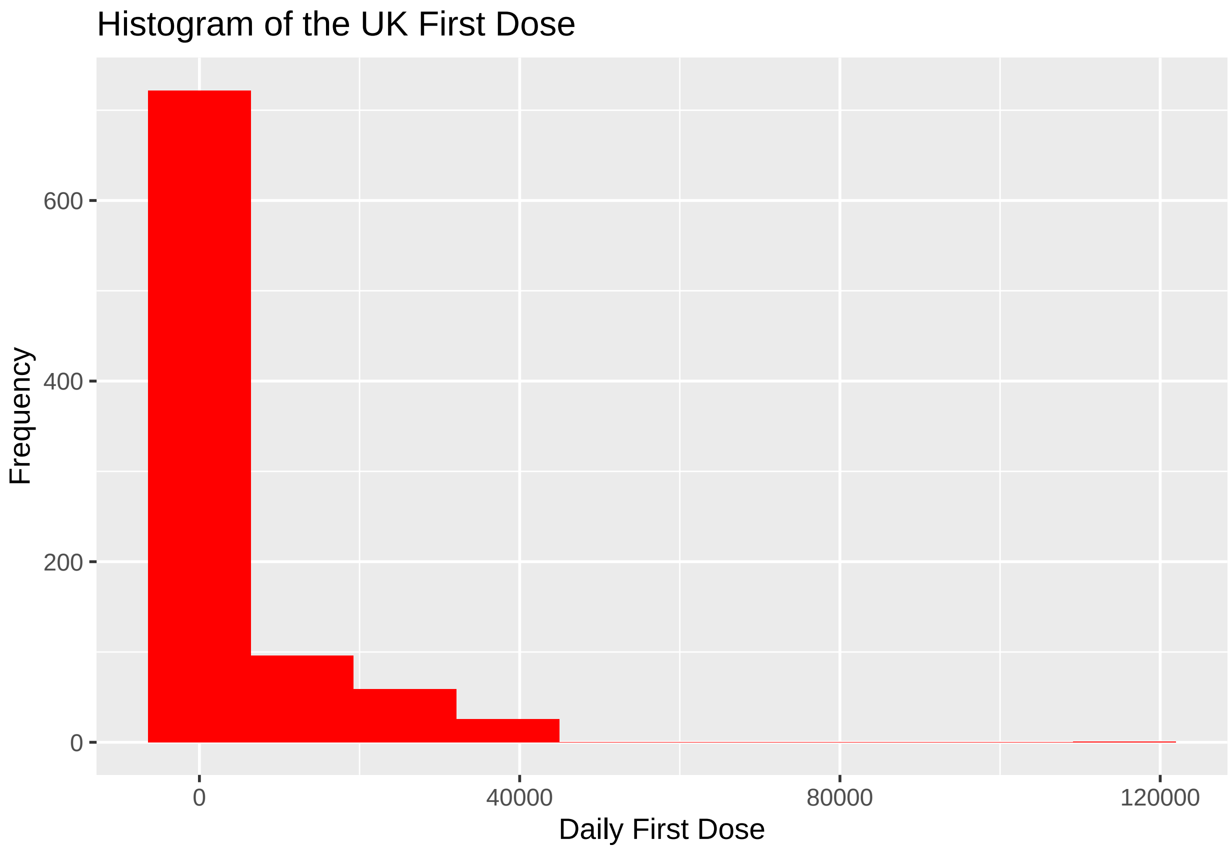
A. Distribution Of Each Dose

These histograms and boxplots visualize the distribution of data for the three distinct vaccine doses, providing clearer insights into the distribution observed from the descriptive statistics.

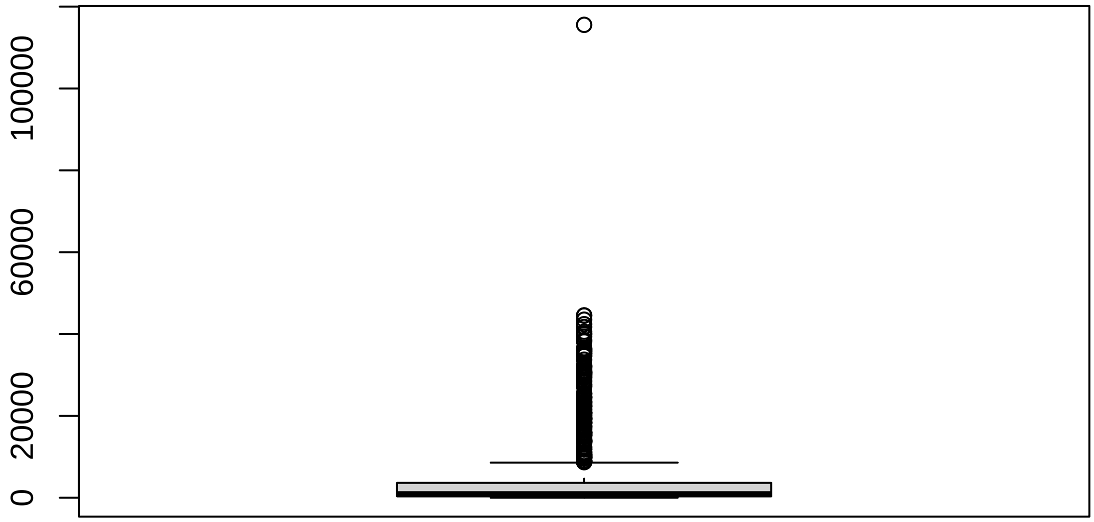
The first dose histogram is significantly skewed to the right, meaning the tail extends to the right increasing the mean. The vast majority of observations are concentrated in the first bucket, then the distribution consistently decreases across the following three buckets before reaching the outlier bucket at the end. The boxplot also depicts all the outliers in the data that we may need to exclude later in the analysis.

```
#Creates histogram of the First Dose column
ggplot(df, aes(x = FirstDose)) +
#Sets the bins to 10 and colour to red
  geom_histogram(bins = 10, fill = 'red') +
  labs(title = 'Histogram of the UK First Dose',
```

```r
#Labels the axis and the title
       x = 'Daily First Dose',
       y = 'Frequency')
```

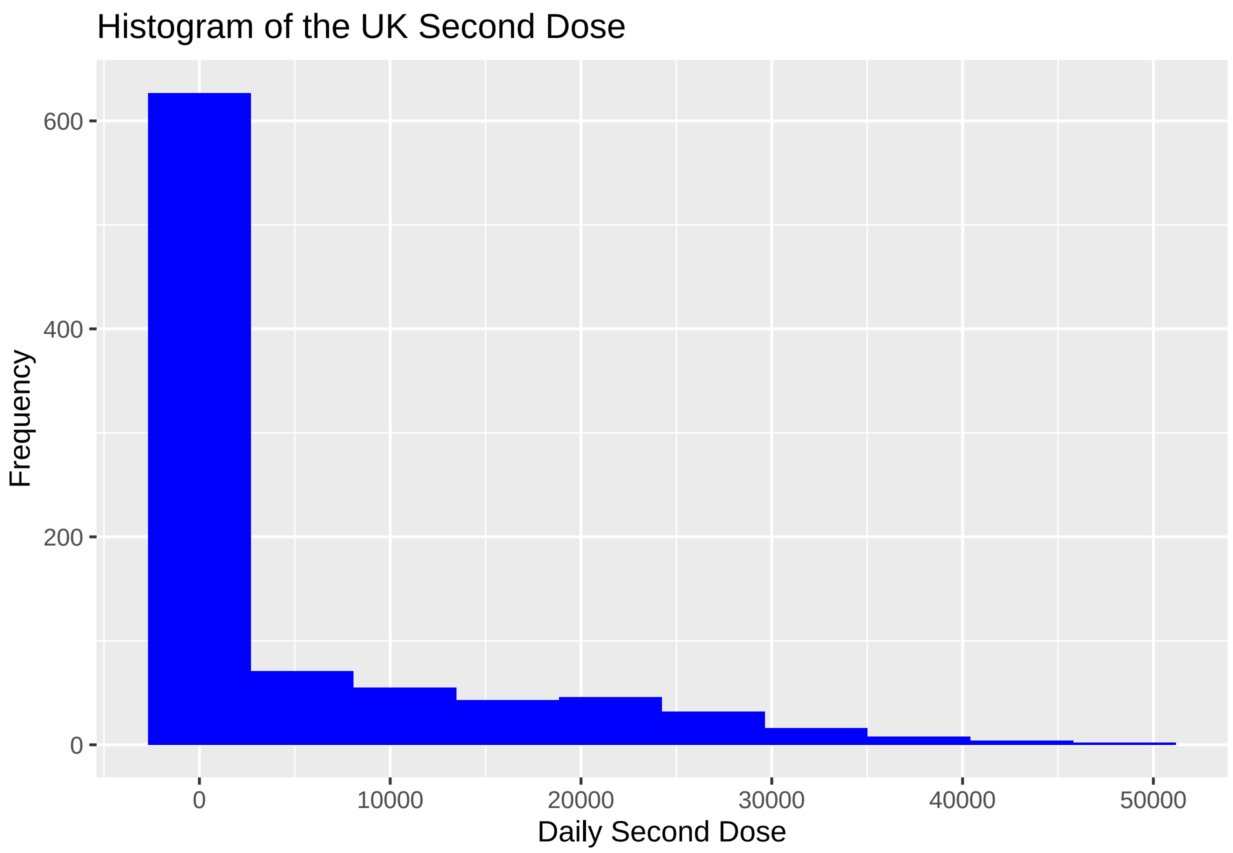


```r
boxplot(df$FirstDose)
```

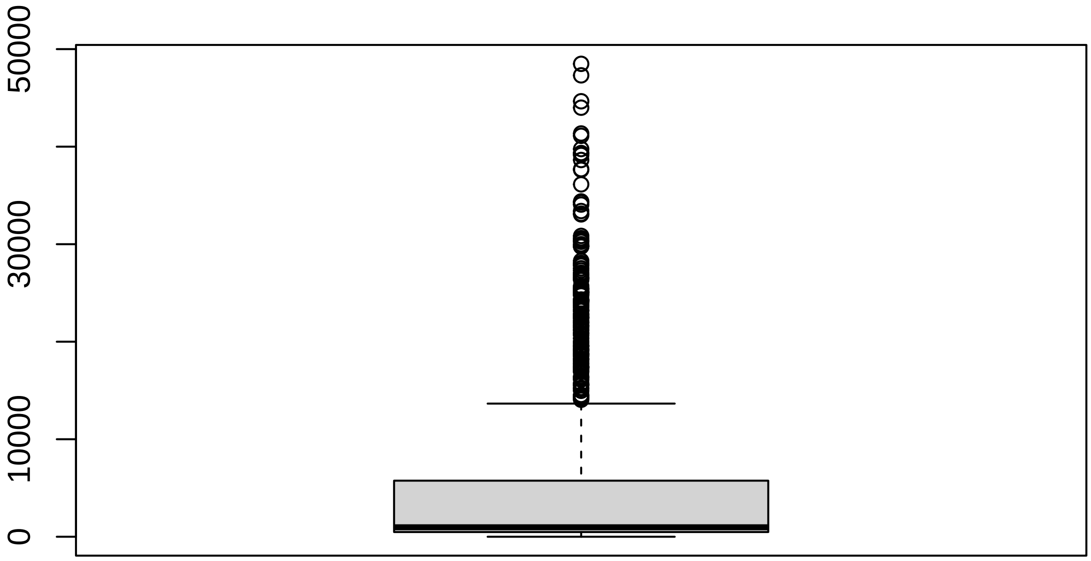


This second dose histogram also displays positive skewness with the majority of observations but the subsequent buckets remain relatively consistent in their distributions before decreasing across the last five buckets. As the the max value for the second dose is not as big an outlier as with the first does, we get a more normal looking boxplot.

```
#Creates histogram of the Second Dose column
ggplot(df, aes(x = SecondDose)) +
#Sets the bins to 10 and colour to blue
  geom_histogram(bins = 10, fill = 'blue') +
#Labels the axis and the title
  labs(title = 'Histogram of the UK Second Dose',
       x = 'Daily Second Dose',
       y = 'Frequency')
```

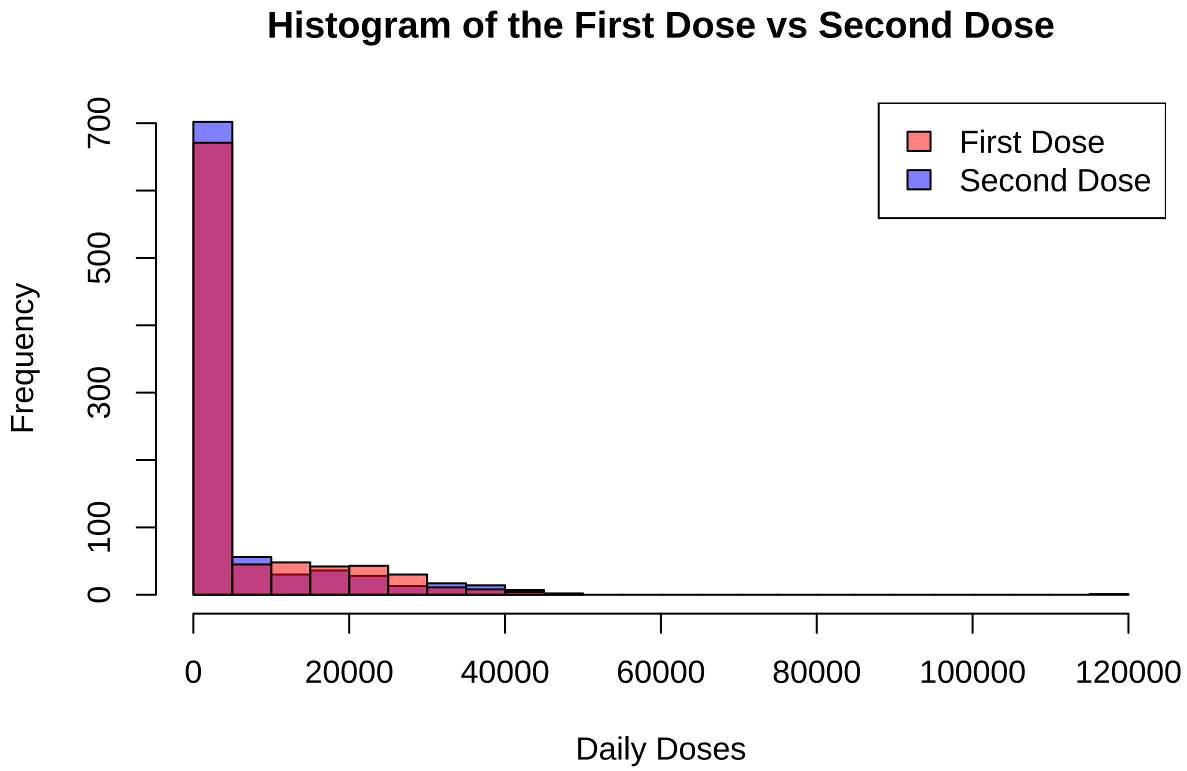


```
boxplot(df$SecondDose)
```

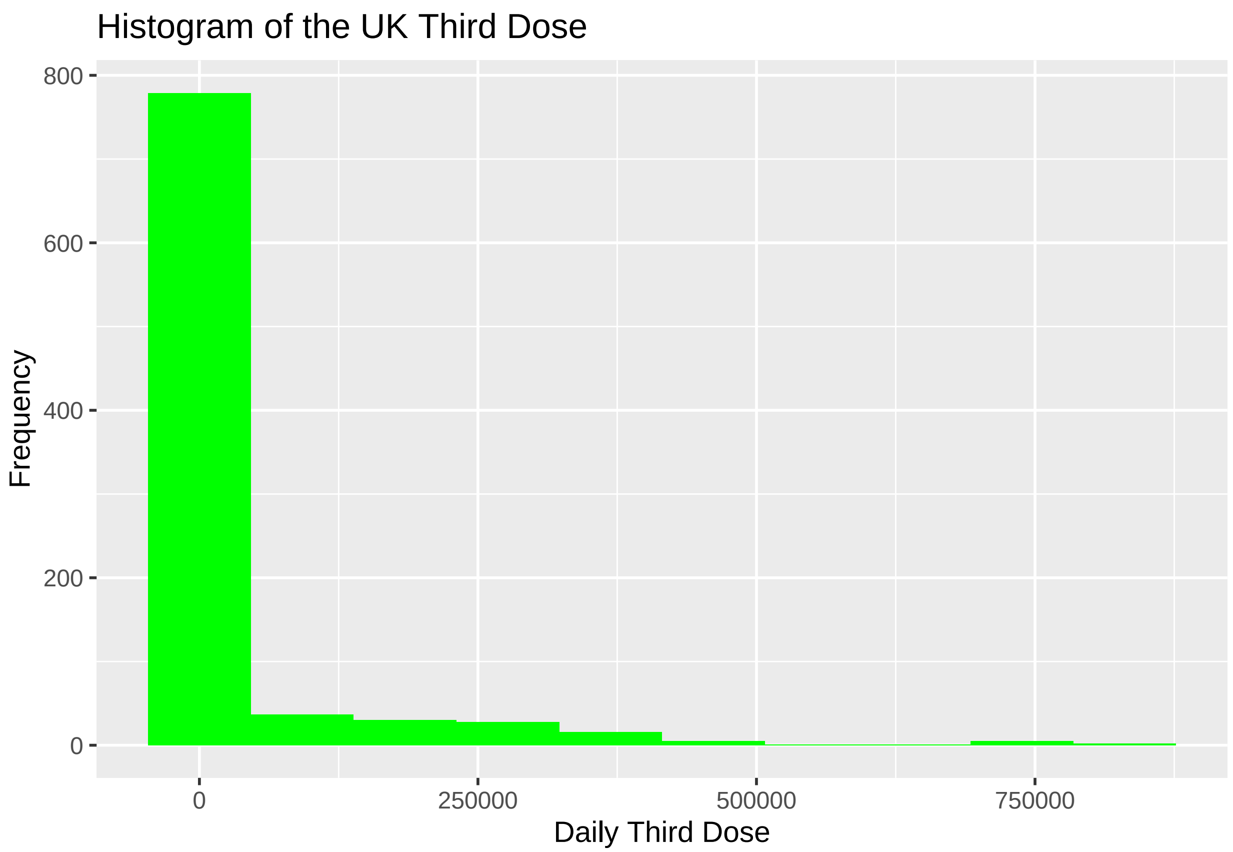


After observing that most of the first and second dose data was in the 0-50,000 range, the histograms were overlapped on the same plot to gain further insights. The two histograms exhibit striking similarities with minor differences between the bucket frequencies.

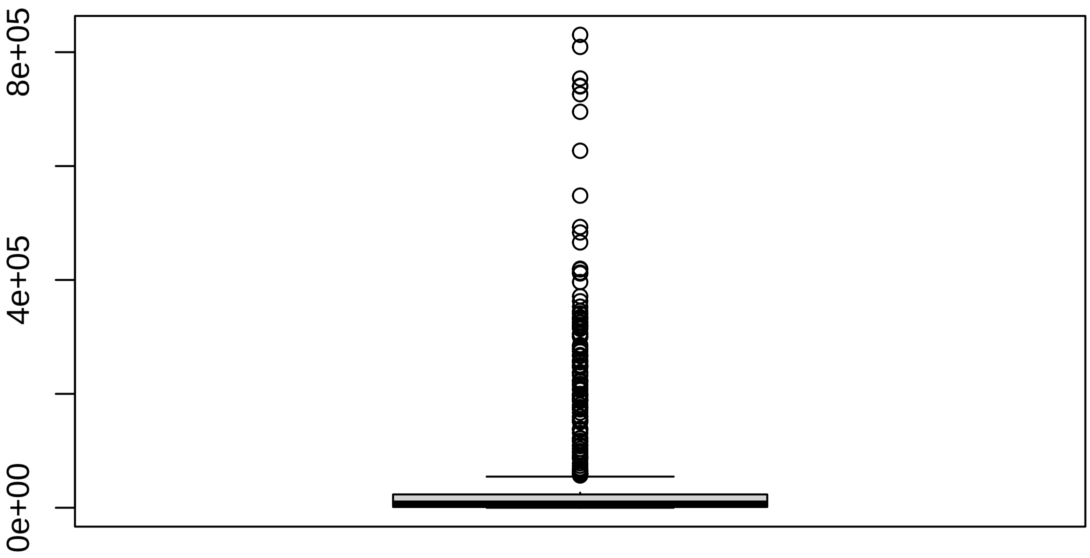
```
#Plots histogram of the First dose
hist(df$FirstDose,
     main ='Histogram of the First Dose vs Second Dose',
#Labels the x-axis
     xlab = 'Daily Doses',
#Uses 20 breaks for the plot
     breaks = 20,
#Sets the colour to blue and makes it semi-transparent
     col = rgb(0, 0, 1, alpha = 0.5))
#Creates histogram of the Second dose
hist(df$SecondDose,
#Uses 10 breaks so each bucket represents the same range as the First dose plot
     breaks = 10,
#Sets the colour to red and makes it semi-transparent
     col = rgb(1, 0, 0, alpha = 0.5),
#Overlaps the second dose histogram with the first dose histogram plot
     add = TRUE)
#Adds a legend in the top right of the plot indicating which colour represents each dose
legend("topright",
       legend = c("First Dose", "Second Dose"),
       fill = c(rgb(1, 0, 0, alpha = 0.5), rgb(0, 0, 1, alpha = 0.5)))
```



```
#Creates histogram of the Third Dose column
ggplot(df, aes(x = ThirdDose)) +
#Sets the bins to 10 and colour to green
  geom_histogram(bins = 10, fill = 'green') +
#Labels the axis and the title
  labs(title = 'Histogram of the UK Third Dose',
       x = 'Daily Third Dose',
       y = 'Frequency')
```



```
boxplot(df$ThirdDose)
```



B. Relationship Between The Total Doses Administered And The Month

```
#Creates new dataframe and enters grouped monthly total doses
monthly_totals = df %>%
  group_by(month) %>%
  summarise(TotalDoses = sum(FirstDose + SecondDose + ThirdDose))
#Prints new dataframe
print(monthly_totals)
```

```
## # A tibble: 9 x 2
##   month TotalDoses
##   <fct>      <dbl>
## 1 1       4830496.
## 2 2       1741644
## 3 3       1198705
## 4 4       1211268
## 5 5        738072.
## 6 9        201962.
## 7 10      8449652.
## 8 11     12191156.
## 9 12     17332175.
```

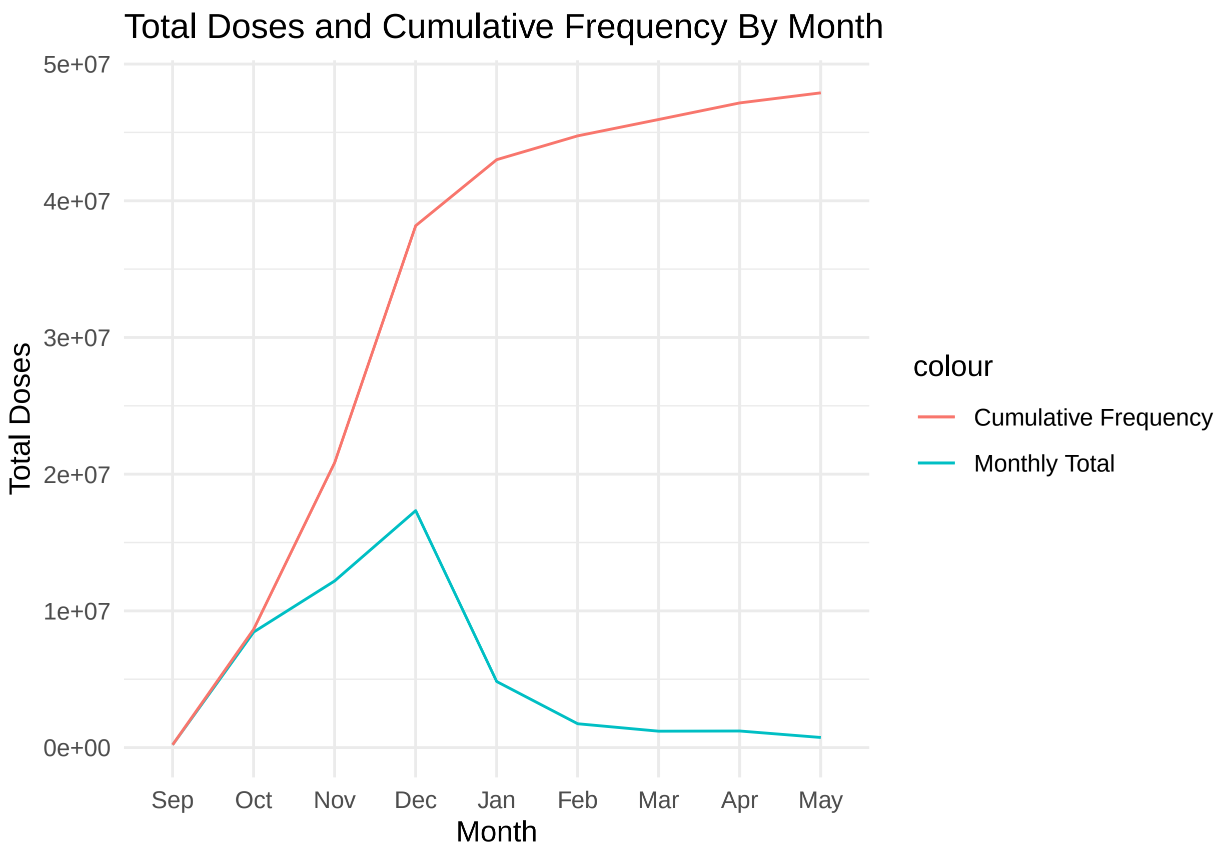
```r
#Sets custom order for the index
custom_index_order = c(6, 7, 8, 9, 1, 2, 3, 4, 5)
#Re-orders the dataframe by this custom order
monthly_totals = monthly_totals[custom_index_order, ]
#Adds new column with the cumulative of total doses
monthly_totals$cumulative_total = cumsum(monthly_totals$TotalDoses)
#Creates vector with names of the months
month_names = c('Sep', 'Oct', 'Nov', 'Dec', 'Jan', 'Feb', 'Mar', 'Apr', 'May')
#Updates the month column with the names of the month and sets that as the levels
monthly_totals$month = factor(month_names, levels = month_names)
#Prints updated dataframe
print(monthly_totals)
```

```
## # A tibble: 9 x 3
##   month TotalDoses cumulative_total
##   <fct>      <dbl>            <dbl>
## 1 Sep      201962.          201962.
## 2 Oct     8449652.         8651614.
## 3 Nov    12191156.        20842770.
## 4 Dec    17332175.        38174945.
## 5 Jan     4830496.        43005442.
## 6 Feb     1741644         44747086.
## 7 Mar     1198705         45945791.
## 8 Apr     1211268         47157059.
## 9 May      738072.        47895130.
```

This graph depicting the change in total doses by month shows the monthly total doses consistently increase until peaking in December. There's then a steep decline in January before a more gradual decline after February. This suggests a concerted effort was made to be fully vaccinated by Christmas/New Years.

The cumulative frequency line has an S-curve shape which shows the total doses gradually increase and then accelerate from November to December before tapering off in the new year. This shape reflects the progress of the vaccination campaign with the total doses reaching a plateau as more people were vaccinated.

```r
#Sets x-axis to be the month
ggplot(monthly_totals, aes(x = month)) +
#Plots a line of the Total Doses and Cumulative Frequency with a legend
  geom_line(aes(y = TotalDoses, color = "Monthly Total"), group = 1) +
  geom_line(aes(y = cumulative_total, color = "Cumulative Frequency"), group = 1) +
#Labels the axis
  labs(x = 'Month', y = 'Total Doses') +
#Adds a title
  ggtitle('Total Doses and Cumulative Frequency By Month') +
#Sets the theme as minimal
  theme_minimal()
```



C. Total Doses Administered By Country

As we can see from the dataframe and the barchart below, there is a large gap between the total doses administered in England compared to the rest of the UK. This is presumably due to its greater population as England has about 84% of the UK's population(per ons.gov.uk 2021 census data). The percentage split of total doses administered closely mirrors that of the population split in the UK so we can not say that any country was necessarily better or faster than the others in getting their population vaccinated.

```
#Creates new dataframe and enters total doses grouped by country
country_totals = df %>%
  group_by(areaName) %>%
  summarise(TotalDoses = sum(FirstDose + SecondDose + ThirdDose))%>%
  mutate(Percentage = (TotalDoses / sum(TotalDoses)) * 100)
#Prints new dataframe
print(country_totals)
```

```
## # A tibble: 4 x 3
##   areaName         TotalDoses Percentage
##   <fct>                 <dbl>      <dbl>
## 1 England           40556401.      84.7
## 2 Northern Ireland   1386816.       2.90
## 3 Scotland           3893781.       8.13
## 4 Wales              2058132.       4.30
```

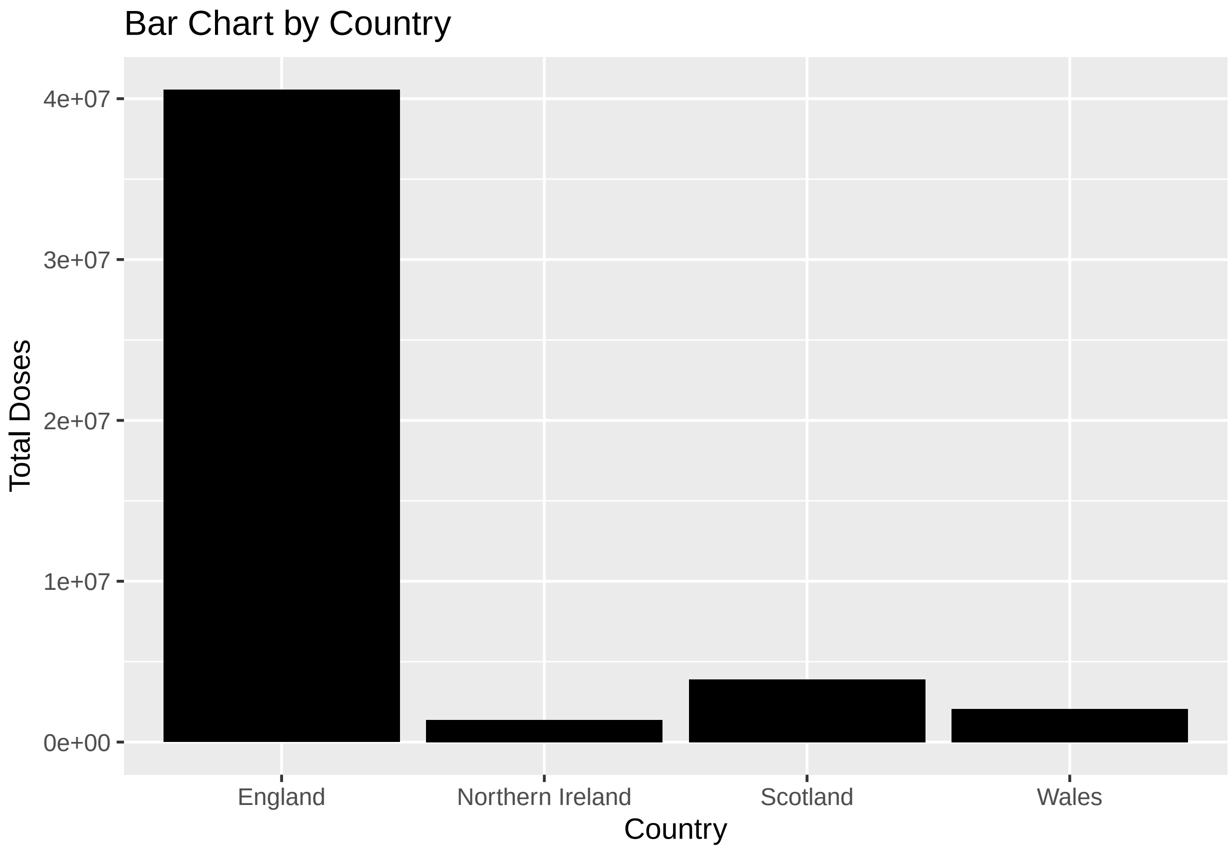
```r
#UK Population data per the ons.gov.uk 2021 census
UK_population = data.frame(
  country = c("England", "Northern Ireland", "Scotland", "Wales"),
  population = c(56536000, 1905000, 5480000, 3105000)
)

#Calculates percentage of population for each country
UK_population = UK_population %>%
  mutate(percentage = (population/sum(population)) * 100)

print(UK_population)
```

```
##            country population percentage
## 1          England   56536000  84.349357
## 2 Northern Ireland    1905000   2.842181
## 3         Scotland    5480000   8.175932
## 4            Wales    3105000   4.632531
```

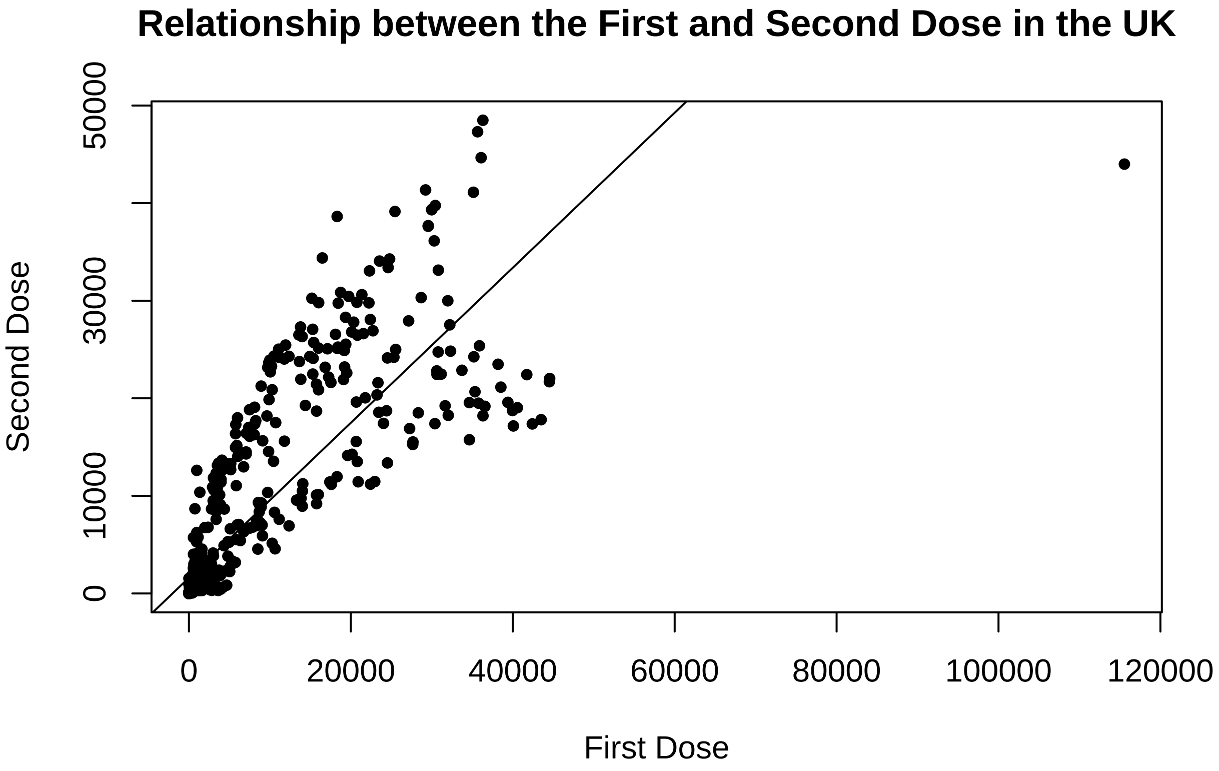
```r
#Creates a bar chart
ggplot(country_totals, aes(x = areaName, y = TotalDoses)) +
  geom_bar(stat = "identity", fill = "black") +
  labs(title = "Bar Chart by Country", x = "Country", y = "Total Doses")
```



D. The Relationship Between Two Doses

This scatter plot with a fitted line suggests a strong positive correlation between the first and second doses. While most data points are clumped together in the bottom left, the points spread out and deviate from the fitted line as we move along the x-axis. We can also clearly see the outlier from the FirstDose.

```
#Creates line of best fit from the First and Second dose data point
model = lm(SecondDose~FirstDose, data=df)
#Plots the data points on a graph including the fitted line
plot(x=df$FirstDose, y=df$SecondDose,
     xlab ='First Dose', ylab = 'Second Dose',
     abline(model),
     main='Relationship between the First and Second Dose in the UK',
     cex=0.8, pch=16)
```



Conducting a statistical test will provide a better understanding of the relationship between the doses.

```
#Step 1: The H0 is there is no significant correlation between the First and Second Dose
#Step 2: To measure the relationship between the doses we will use Pearson's correlation coefficient
#Step 3: The significance level to reject the H0 is 0.05

#Prints the pearson correlation coefficient of the two variables
cor.test(df$FirstDose, df$SecondDose, method = 'pearson')
```

```
##
##  Pearson’s product-moment correlation
```

```
##
## data:  df$FirstDose and df$SecondDose
## t = 45.59, df = 902, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.8142139 0.8537970
## sample estimates:
##       cor
## 0.8350831
```

Interpretation: The p-value of 2.2e-16 is less than 0.05, meaning there is a statistically significant relationship between the doses, so we reject the H0. The correlation coefficient of 0.8350831 indicates a strong positive correlation meaning as the FirstDose value increases, so will the SecondDose. This is interesting as my initial expectation was to see an inverse relationship as people progress to the second dose, but the observed positive correlation may be attributed to increased vaccine availability and phased vaccination campaigns, ensuring timely administration of both doses.

4. Descriptive Statistics Of Data Subsets

Analysis of Q4 and Q1 data for England to see changes across the quarters

```r
#Filters the data for England in Q4 and Q1
england_subset1 = df %>%
  filter(areaName == 'England', Quarter == 'Q4')
england_subset2 = df %>%
  filter(areaName == 'England', Quarter == 'Q1')
#Prints descriptive stats for noth subsets
summary(england_subset1)
```

```
##              areaName       areaCode    year        month    Quarter  day
##  England         :92   E92000001:92   2021:92   10     :31   Q1: 0   Fri:14
##  Northern Ireland: 0   N92000002: 0   2022: 0   12     :31   Q2: 0   Mon:13
##  Scotland        : 0   S92000003: 0             11     :30   Q3: 0   Sat:13
##  Wales           : 0   W92000004: 0             1      : 0   Q4:92   Sun:13
##                                                 2      : 0           Thu:13
##                                                 3      : 0           Tue:13
##                                                 (Other): 0           Wed:13
##  WorkingDay   FirstDose        SecondDose      ThirdDose
##  No :26     Min.   :   955   Min.   :  916   Min.   : 10477
##  Yes:66     1st Qu.: 19703   1st Qu.:18549   1st Qu.:192994
##             Median : 26337   Median :22435   Median :267121
##             Mean   : 27551   Mean   :23783   Mean   :301325
##             3rd Qu.: 34640   3rd Qu.:27967   3rd Qu.:340742
##             Max.   :115551   Max.   :48491   Max.   :830403
##
```

```r
summary(england_subset2)
```

```
##              areaName       areaCode    year        month    Quarter  day
##  England         :90   E92000001:90   2021: 0   1      :31   Q1:90   Fri:12
##  Northern Ireland: 0   N92000002: 0   2022:90   3      :31   Q2: 0   Mon:13
##  Scotland        : 0   S92000003: 0             2      :28   Q3: 0   Sat:13
##  Wales           : 0   W92000004: 0             4      : 0   Q4: 0   Sun:13
##                                                 5      : 0           Thu:13
##                                                 9      : 0           Tue:13
##                                                 (Other): 0           Wed:13
##  WorkingDay   FirstDose       SecondDose      ThirdDose
##  No :26     Min.   : 2203   Min.   : 2684   Min.   :  6366
##  Yes:64     1st Qu.: 4092   1st Qu.:12756   1st Qu.: 16093
##             Median : 8118   Median :17851   Median : 23614
##             Mean   : 9716   Mean   :19073   Mean   : 41625
##             3rd Qu.:13946   3rd Qu.:24320   3rd Qu.: 45082
##             Max.   :29231   Max.   :41351   Max.   :206676
##
```

Subset1: This subset further reflects the strong vaccination push in Q4, interestingly there were more first doses administered on average than second doses.

Subset2: Q1 shows substantially reduced vaccination efforts especially for the first and third doses but the second doses seems to have retained a similar rate as in Q4.

The dramatic change in doses administered across the two quarters indicates a change in public health priorities or a successful vaccination campaign.

5. Statistical Significance Of The Mean Difference Between Subsets

It was harder to tell if there is a significant difference between the administration of the Second Dose in Q4 compared to Q1, so conducting a statistical test will provide a better understanding.

```
#Step 1: The H0 is that there is no mean difference between the subsets
#Step 2: To determine the significance of the mean difference between subsets
#we will use an Independent Two-Sample t-test
#Step 3: The significance level to reject the H0 is 0.05

#Tests for if the mean difference of subset 1 is greater than subset 2
t.test(england_subset1$SecondDose, england_subset2$SecondDose, alternative='greater')
```

```
##
##  Welch Two Sample t-test
##
## data:  england_subset1$SecondDose and england_subset2$SecondDose
## t = 3.796, df = 177.92, p-value = 0.0001007
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  2658.626      Inf
## sample estimates:
## mean of x mean of y
##  23783.14  19072.76
```

Interpretation: The p-value of 0.0001007 is less than 0.05 so we must reject the H0. This suggests the mean of the Second Doses administered in Q4 for England was significantly more than in Q1.

6. Grouping Data By Working Day And Quarter

```
#Creates a subset of data for England
england_data = subset(df, areaName == 'England')
#Creates table from the england data subset
summary_table = england_data %>%
#Groups by year and Quarter
  group_by(Quarter, WorkingDay) %>%
#Creats columns for total doses with the sum for each dose
  summarise(
    Average_FD = mean(FirstDose),
    Average_SD = mean(SecondDose),
    Average_TD = mean(ThirdDose))
```

```
## ‘summarise()‘ has grouped output by ’Quarter’. You can override using the
## ‘.groups‘ argument.
```

```
#prints the summary table
print(summary_table)
```

```
## # A tibble: 7 x 5
## # Groups:   Quarter [4]
##   Quarter WorkingDay Average_FD Average_SD Average_TD
##   <fct>   <fct>           <dbl>      <dbl>      <dbl>
## 1 Q1      No             10037.     19271.     39144.
## 2 Q1      Yes             9586.     18992.     42633.
## 3 Q2      No             13136.      9129.     10415.
## 4 Q2      Yes             9003.      7506.     12033.
## 5 Q3      Yes            28689      30318     136511.
## 6 Q4      No             20131.     21668     282463.
## 7 Q4      Yes            30475.     24616.    308755.
```

This table provides a snapshot of average vaccine doses administered on weekdays compared to weekends in England over time. The average number of doses administered during Q3 and Q4 is noticeably higher than in Q1 and Q2, particularly evident in Third Dose data. Interestingly, working days seem to influence vaccination rates differently across quarters and doses. For the third dose, more people were vaccinated during working days than weekends across all quarters but for the first and second dose, the average is higher during the weekends for Q1 and Q2. This could be due to people being more relaxed about getting vaccinated after Christmas and New Years so waiting until the weekend when they have time. Q4 on the other hand has a higher average during working days across all three doses.

7. Linear Regression Model

This regression model attempts to predict the ThirdDose value based on the other variables, excluding month, areaCode and day to avoid multicollinearity. Dummy variables are used to encode the year and WorkingDay with ‘Yes’ and 2022 being set as 0 since they have the most entries with ‘No’ and 2021 set as 1. Quarter and areaName were hot encoded as there are four categories each.

```
#Encoding the year and WorkingDay columns
df$WorkingDay = ifelse(df$WorkingDay == 'Yes', 0, 1)
df$year = ifelse(df$year == 2022, 0, 1)
#Creates dummy country variables
dummy_country = model.matrix(~ areaName - 1, data = df)
```

```r
#Creates dataframe with dummy countries
dummy_country_df = as.data.frame(dummy_country)
#Renames columns
colnames(dummy_country_df) = c("England","NI","Scotland","Wales")
#Creates dummy quarter variables
dummy_quarter = model.matrix(~ Quarter - 1, data = df)
#Creates dataframe with dummy quarters
dummy_quarter_df = as.data.frame(dummy_quarter)
#Renames columns
colnames(dummy_quarter_df) = c("Q1","Q2","Q3","Q4")
#Adds dummy variables to the main dataframe
df = cbind(df, dummy_country_df, dummy_quarter_df)
#Printing first few rows of the combined dataframe
head(df)
```

```
##   areaName  areaCode year month Quarter day WorkingDay FirstDose SecondDose
## 1  England E92000001    0     5      Q2 Mon          0      3034       3857
## 2  England E92000001    0     5      Q2 Sun          1      5331       3330
## 3  England E92000001    0     5      Q2 Sat          1     13852       9759
## 4  England E92000001    0     5      Q2 Fri          0      5818       5529
## 5  England E92000001    0     5      Q2 Thu          0      8439       6968
## 6  England E92000001    0     5      Q2 Wed          0      4955       5247
##   ThirdDose England NI Scotland Wales Q1 Q2 Q3 Q4
## 1      8747       1  0        0     0  0  1  0  0
## 2      4767       1  0        0     0  0  1  0  0
## 3     12335       1  0        0     0  0  1  0  0
## 4     10692       1  0        0     0  0  1  0  0
## 5     11701       1  0        0     0  0  1  0  0
## 6     11219       1  0        0     0  0  1  0  0
```

A. Initial Model

As England and Q1 have the most entries, they were excluded from the initial model to act as a reference for the other categories to be compared to.

```r
#Creates a regression model to predict the ThirdDose value based on these variables
model1 = lm(ThirdDose~year+WorkingDay+FirstDose+SecondDose+NI+Scotland+Wales+Q2+Q3+Q4,
            data=df)
summary(model1)
```

```
##
## Call:
## lm(formula = ThirdDose ~ year + WorkingDay + FirstDose + SecondDose +
##     NI + Scotland + Wales + Q2 + Q3 + Q4, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -296261  -21171    1824   19279  401771
##
## Coefficients: (1 not defined because of singularities)
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.118e+05  8.383e+03 -13.338  < 2e-16 ***
## year         4.928e+04  4.366e+03  11.289  < 2e-16 ***
## WorkingDay  -2.052e+03  3.947e+03  -0.520 0.603316    
## FirstDose    3.809e+00  3.613e-01  10.542  < 2e-16 ***
## SecondDose   8.741e+00  4.704e-01  18.584  < 2e-16 ***
## NI           8.706e+04  8.537e+03  10.198  < 2e-16 ***
## Scotland     8.685e+04  8.249e+03  10.528  < 2e-16 ***
## Wales        8.938e+04  8.502e+03  10.513  < 2e-16 ***
## Q2           1.888e+04  4.953e+03   3.811 0.000148 ***
## Q3          -1.027e+05  3.812e+04  -2.695 0.007171 ** 
## Q4                  NA         NA      NA       NA    
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
## 
## Residual standard error: 53630 on 894 degrees of freedom
## Multiple R-squared:  0.7398, Adjusted R-squared:  0.7372 
## F-statistic: 282.4 on 9 and 894 DF,  p-value: < 2.2e-16
```

Negative Coefficients: The WorkingDay(when it is a weekend day) and Q3 variables are associated with a decrease in the ThirdDose.

Positive Coefficients: The year, FirstDose and SecondDose, country and Q2 variables are associated with an increase in the ThirdDose with the countries and SecondDose having the most significant impact.

Variable Significance: Only the WorkingDay variable has a statistically insignificant p-value meaning it is not a key factor for the ThirdDose value. The Q4 variable returns NA most likely due to multicollinearity with the other quarters.

Adjusted R-squared: At 0.7372, this suggests that a high proportion of the ThirdDose is explained by the model making it a good fit.

B. Parsimonious Model

Based on this we can reduce the variables to find the most parsimonious model with only the statistically significant variables and we can see the R-squared marginally increase.

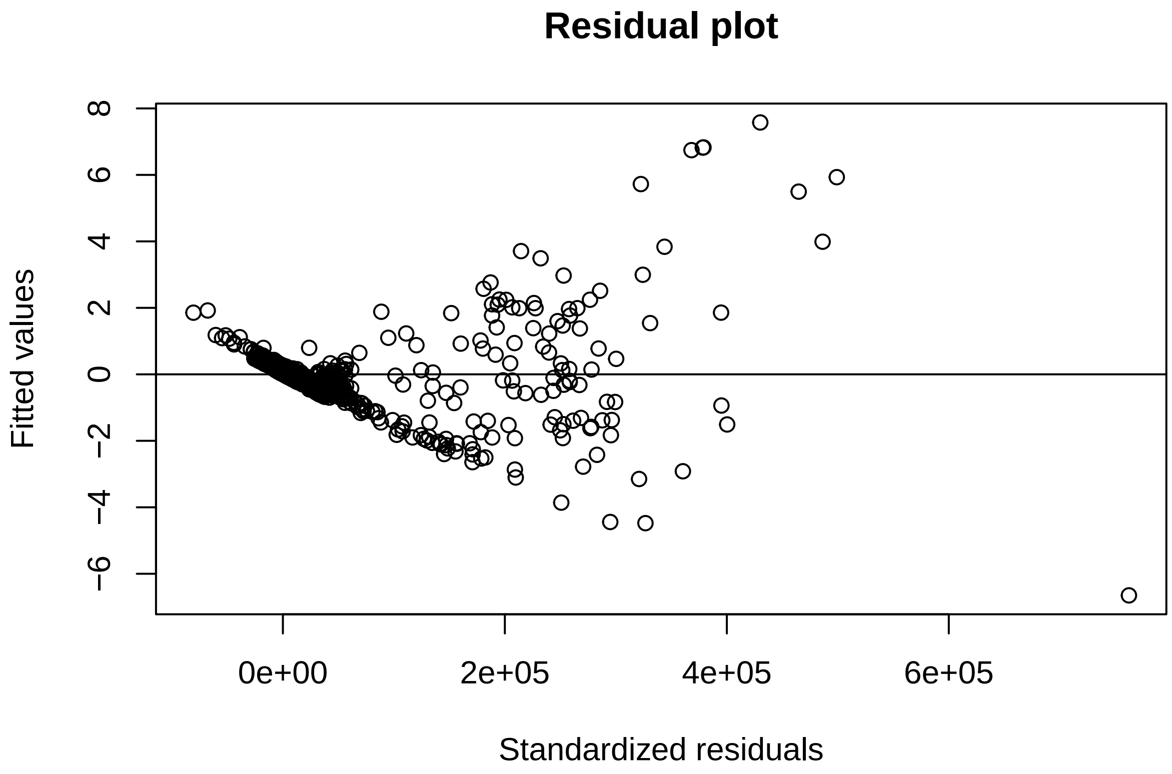
```
#creating new regression model
model2 = lm(ThirdDose~year+FirstDose+SecondDose+NI+Scotland+Wales+Q2+Q3,
            data=df)
summary(model2)
```

```
## 
## Call:
## lm(formula = ThirdDose ~ year + FirstDose + SecondDose + NI + 
##     Scotland + Wales + Q2 + Q3, data = df)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -296384  -21033    1773   19513  400261 
## 
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -1.125e+05  8.290e+03 -13.566  < 2e-16 ***
## year         4.926e+04  4.364e+03  11.290  < 2e-16 ***
## FirstDose    3.817e+00  3.608e-01  10.580  < 2e-16 ***
## SecondDose   8.737e+00  4.701e-01  18.585  < 2e-16 ***
## NI           8.712e+04  8.532e+03  10.210  < 2e-16 ***
## Scotland     8.691e+04  8.245e+03  10.541  < 2e-16 ***
## Wales        8.944e+04  8.497e+03  10.526  < 2e-16 ***
## Q2           1.884e+04  4.950e+03   3.805 0.000151 ***
## Q3          -1.022e+05  3.809e+04  -2.682 0.007448 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 53600 on 895 degrees of freedom
## Multiple R-squared:  0.7397, Adjusted R-squared:  0.7374
## F-statistic:   318 on 8 and 895 DF,  p-value: < 2.2e-16
```

C. Diagnostics

This plot indicates that errors are not evenly scattered around 0

```
#Creates variables to hold residual and fitted values
st_resids = rstandard(model2)
fitted_v = model2$fitted.values
#Tests for zero mean of errors
plot(x=fitted_v, y=st_resids, abline(h=0),
     xlab="Standardized residuals", ylab="Fitted values",
     main = "Residual plot")
```



The p-value is less than 0.05 so we reject the H0 meaning the data is not normally distributed

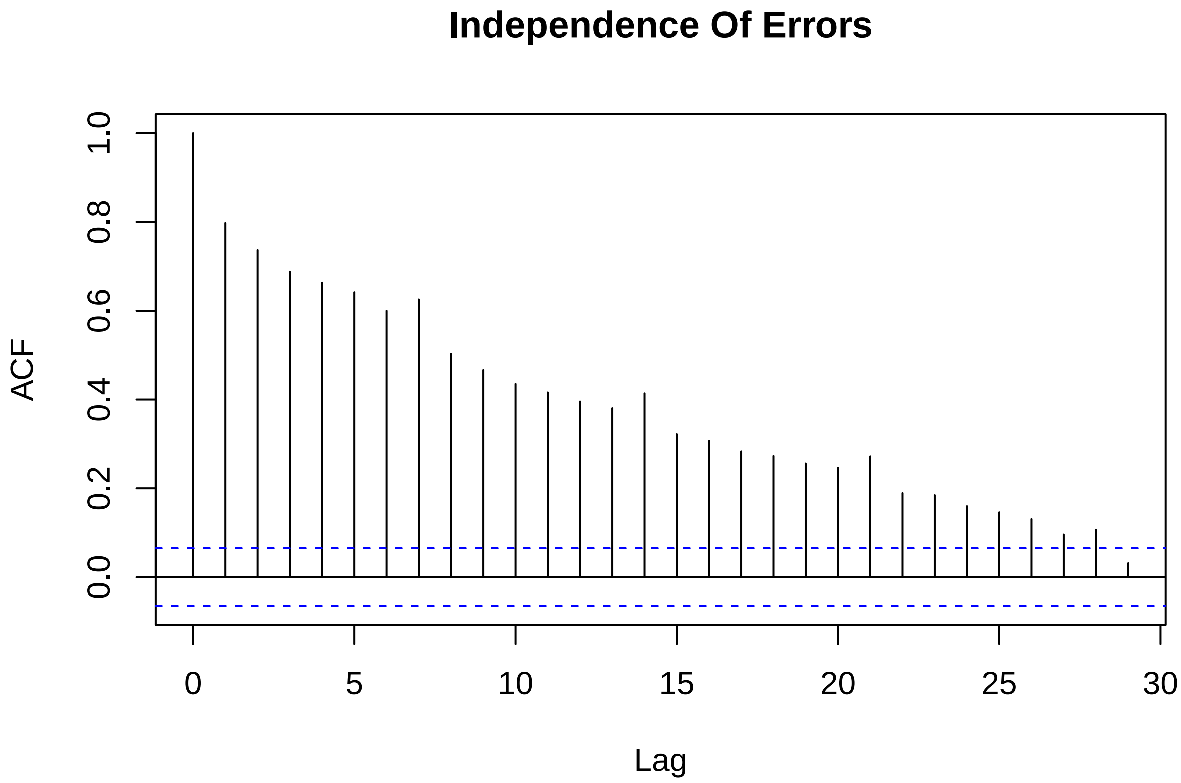
```
#Tests for normality of errors
#Step 1: The HO is that the data is normally distributed
#Step 2: To determine normality we will use a Jarque-Bera test
#Step 3: The significance level to reject the HO is 0.05

jarque.test(st_resids)
```

```
##
##  Jarque-Bera Normality Test
##
## data:  st_resids
## JB = 10487, p-value < 2.2e-16
## alternative hypothesis: greater
```

The autocorrelation plot shows that most coefficients are outside the confidence band indicating that the errors are not independent

```
#Tests for independence of errors
acf(st_resids, main='Independence Of Errors')
```



The p-value is less than 0.05 so we reject the H0 meaning the model is not homoscedastic

```
#Tests for homoscedasticity
#Step 1: The HO is that the data is homoscedastic
#Step 2: To determine homoscedasticity we will use a Breusch-Pagan test
```

```
#Step 3: The significance level to reject the H0 is 0.05
bptest(model2)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  model2
## BP = 397.49, df = 8, p-value < 2.2e-16
```

D. Removing Outliers

With the diagnostics failing, I decided to remove outliers to see if this would improve the model which led to even more variables being excluded as they were not statistically significant. This saw a reduction in the adjusted R-squared meaning the model is not as good a fit as before, and after re-running the diagnostics this new model still failed all the key assumptions.

```
columns = c("FirstDose", "SecondDose", "ThirdDose")

#Loops over columns to remove outliers
for (col in columns) {
  outliers <- boxplot.stats(df[[col]])$out
  no_outliers <- df[!df[[col]] %in% outliers, ]
}
summary(no_outliers)
```

```
##              areaName        areaCode         year            month     Quarter
##  England         :128   E92000001:128   Min.   :0.0000   3      :124   Q1:343
##  Northern Ireland:236   N92000002:236   1st Qu.:0.0000   4      :120   Q2:206
##  Scotland        :212   S92000003:212   Median :0.0000   2      :112   Q3:  1
##  Wales           :210   W92000004:210   Mean   :0.3015   1      :107   Q4:236
##                                         3rd Qu.:1.0000   11     : 89
##                                         Max.   :1.0000   5      : 86
##                                                          (Other):148
##   day        WorkingDay       FirstDose         SecondDose        ThirdDose
##  Fri:112   Min.   :0.0000   Min.   :    0.0   Min.   :    0.0   Min.   :    0
##  Mon:113   1st Qu.:0.0000   1st Qu.:  286.2   1st Qu.:  434.0   1st Qu.: 1171
##  Sat:114   Median :0.0000   Median :  692.0   Median :  793.5   Median : 4864
##  Sun:115   Mean   :0.2913   Mean   : 2034.8   Mean   : 2936.3   Mean   : 9973
##  Thu:112   3rd Qu.:1.0000   3rd Qu.: 1635.8   3rd Qu.: 2015.0   3rd Qu.:15087
##  Tue:110   Max.   :1.0000   Max.   :27649.0   Max.   :34382.0   Max.   :54649
##  Wed:110
##     England             NI            Scotland          Wales
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.0000   Median :0.0000   Median :0.0000   Median :0.0000
##  Mean   :0.1628   Mean   :0.3003   Mean   :0.2697   Mean   :0.2672
##  3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##
##        Q1               Q2               Q3                 Q4
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.000000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.000000   1st Qu.:0.0000
##  Median :0.0000   Median :0.0000   Median :0.000000   Median :0.0000
##  Mean   :0.4364   Mean   :0.2621   Mean   :0.001272   Mean   :0.3003
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:0.000000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.000000   Max.   :1.0000
##
```

```r
model3 = lm(ThirdDose~year+SecondDose+Scotland+Wales,
            data=no_outliers)
summary(model3)
```

```
##
## Call:
## lm(formula = ThirdDose ~ year + SecondDose + Scotland + Wales,
##     data = no_outliers)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -20812  -3684   -791   2770  31171
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.914e+03  4.467e+02  -4.285 2.05e-05 ***
## year         1.521e+04  5.151e+02  29.521  < 2e-16 ***
## SecondDose   1.510e+00  4.689e-02  32.200  < 2e-16 ***
## Scotland     7.003e+03  5.812e+02  12.050  < 2e-16 ***
## Wales        3.666e+03  5.917e+02   6.196 9.37e-10 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 6475 on 781 degrees of freedom
## Multiple R-squared:  0.6776, Adjusted R-squared:  0.6759
## F-statistic: 410.3 on 4 and 781 DF,  p-value: < 2.2e-16
```

Interpretation: While these models may have predictive capability, the reliability and validity are compromised due to all four key linear regression assumptions being violated. Alternative modelling techniques may be required for improved validity.

Conclusion

This exploratory analysis has identified that there was a major shift in the pandemic after Q4 of 2021, most notably in December when the majority of the population had received the third and final dose. The doses administered significantly declined in the early months of 2022 presumably because most people were now fully vaccinated but also possibly due to people being less worried about Covid and not being in as much of a rush to get vaccinated.