DIAGNOSTIC ANALYSIS USING PYTHON – NATIONAL HEALTH SERVICES

Author: Neetu Thomas

24 October 2022

This page intentionally left blank

Contents

Abbreviations

AD	Actual Duration
AR	Appointments regional
DNA	Did Not Attend
NC	National Category

1 Introduction

NHS has a high number of missed appointments resulting in significant costs to them. They want to better understand the reasons for this and eliminate this by data driven solutions.

Questions to be answered

- Has there been adequate staff and capacity in the Networks?
- What was the actual utilisation of resources?

Below 3 datasets and Twitter data were used for the analysis

- actual_duration.csv
- apointments_regional.csv
- national_categories.xlsx

2 Analytical Approach

2.1 Import & Explore the data

Import

New Python3 file in Jupyter notebook launched.

Necessary libraries imported into notebook. Universally used aliases used for libraries for ease of use

Libraries imported include:

- Pandas as pd
- numpy as np
- Matplotlib.pyplot as plt
- Seaborn as sns
- Datetime as dt

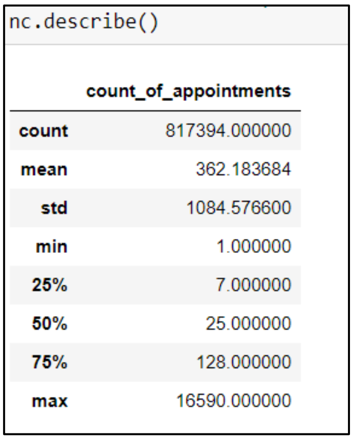
Loaded datasets into Jupyter using **pd_read_csv** – for csv files and **pd.read_excel** – for excel files.

Ensured datasets are in the current working directory as the notebook.

Sense check

The DataFrames (df) are sense checked using:

- df.shape
- df.columns
- df.dtypes
- df.head()
- info()
- describe()
  Only the **'**count_of_appointments' column is integer datatype showing descriptive statistics



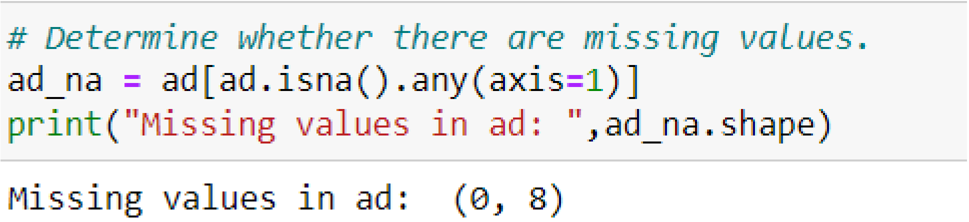
```
nc.describe()
```

	count_of_appointments
count	817394.000000
mean	362.183684
std	1084.576600
min	1.000000
25%	7.000000
50%	25.000000
75%	128.000000
max	16590.000000

The nc is a very big file and takes longer time to load.

Missing values

All 3 DF checked for missing values using isna() function- NO missing values found.

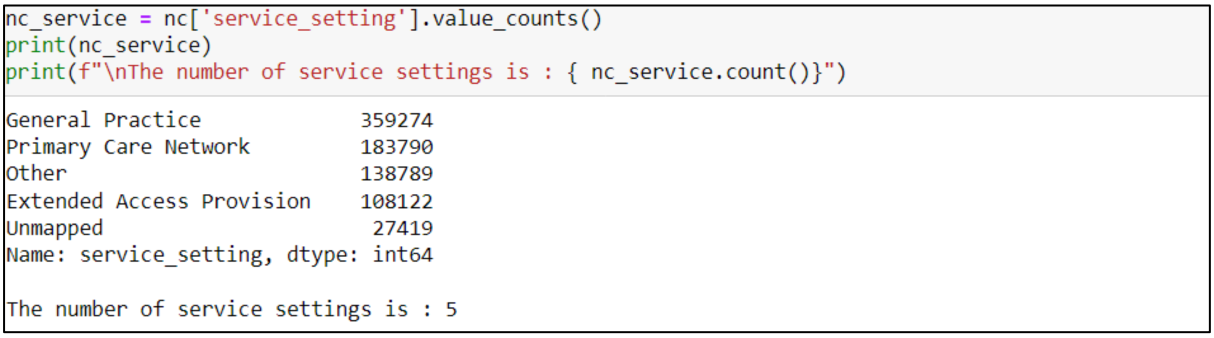


```
# Determine whether there are missing values.
ad_na = ad[ad.isna().any(axis=1)]
print("Missing values in ad: ",ad_na.shape)
```

```
Missing values in ad:  (0, 8)
```

2.1.1 Initial Exploration

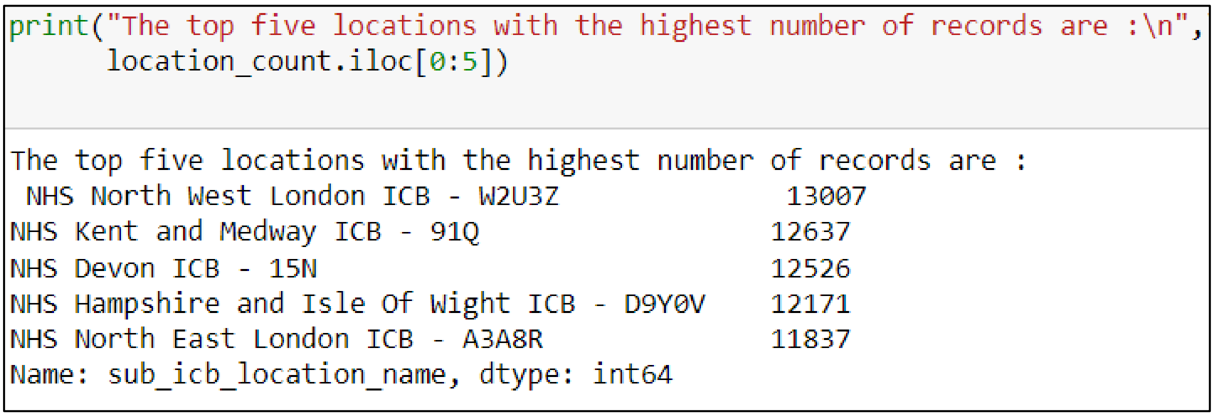
To determine the number of locations, service settings, context types, national categories and appointment statuses – value_counts() used.



```
nc_service = nc['service_setting'].value_counts()
print(nc_service)
print(f"\nThe number of service settings is : { nc_service.count()}")
```

```
General Practice             359274
Primary Care Network         183790
Other                        138789
Extended Access Provision    108122
Unmapped                      27419
Name: service_setting, dtype: int64

The number of service settings is : 5
```

iloc() – used to select the required number of rows/columns.



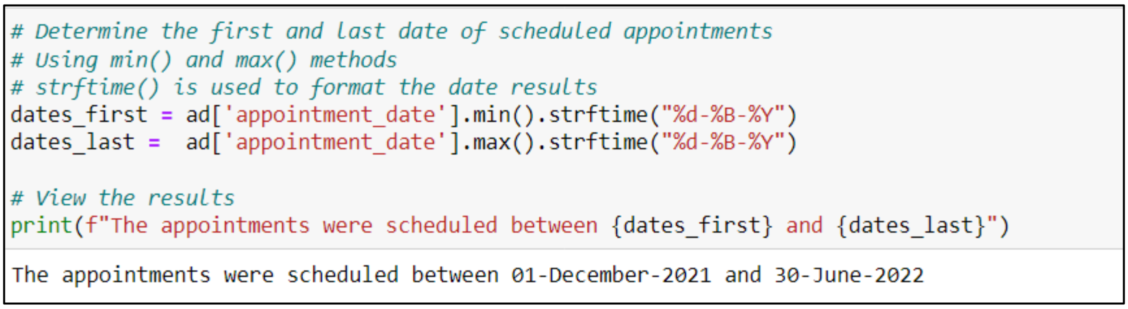
```
print("The top five locations with the highest number of records are :\n",
      location_count.iloc[0:5])
```

```
The top five locations with the highest number of records are :
 NHS North West London ICB - W2U3Z                13007
NHS Kent and Medway ICB - 91Q                    12637
NHS Devon ICB - 15N                              12526
NHS Hampshire and Isle Of Wight ICB - D9Y0V      12171
NHS North East London ICB - A3A8R                11837
Name: sub_icb_location_name, dtype: int64
```

2.2 Analyse the Data

The earliest and latest scheduled appointments obtained from date column.

- ad
    - 'appointment_date' in ad is in object(str) dtype.
    - Converted to datetime format using ***pd.to_datetime().***
    - First and last dates found using the min() and max().
    - strftime() formats the date in "%d-%B-%Y"



```
# Determine the first and last date of scheduled appointments
# Using min() and max() methods
# strftime() is used to format the date results
dates_first = ad['appointment_date'].min().strftime("%d-%B-%Y")
dates_last =  ad['appointment_date'].max().strftime("%d-%B-%Y")

# View the results
print(f"The appointments were scheduled between {dates_first} and {dates_last}")
```

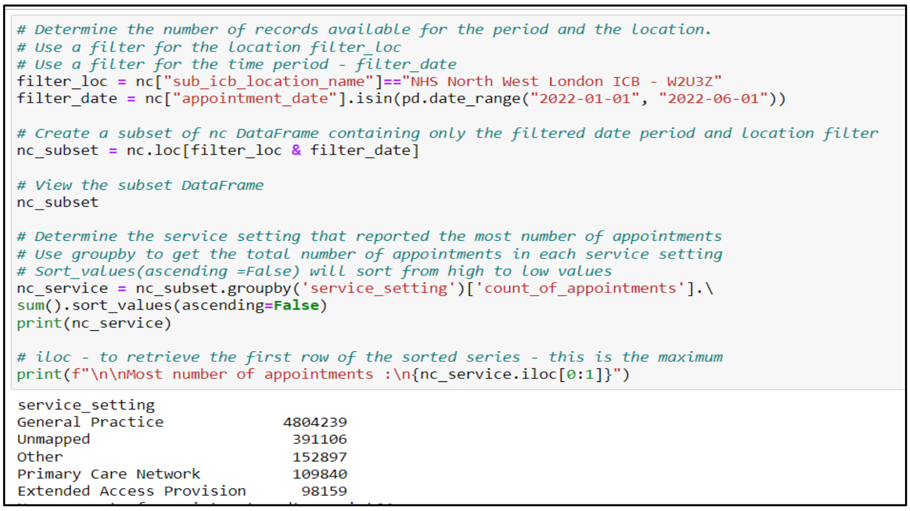
```
The appointments were scheduled between 01-December-2021 and 30-June-2022
```

- nc
    - The appointments were scheduled between 01 August 2021 and 30 June 2022.
- ar
    - 'appointment_month' - earliest and latest months January 2020 and June 2022.

- Groupby()

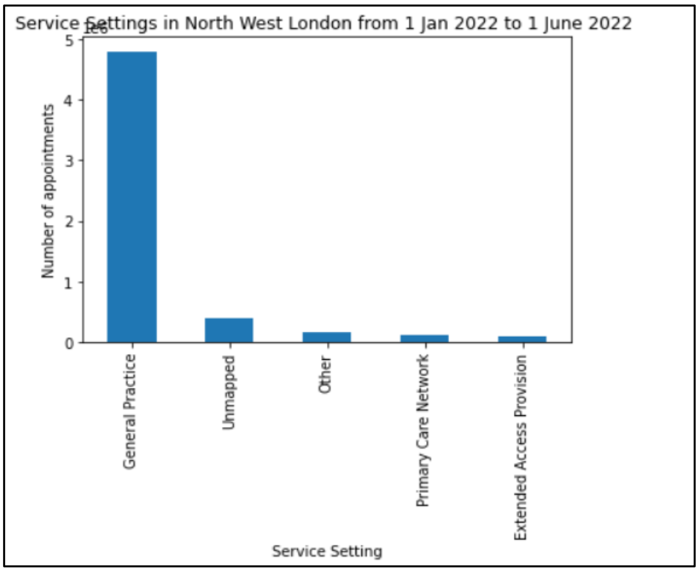
nc dataframe analysed to know the number of appointments between 1 January and 1 June 2022 in location –' NHS North West London ICB - W2U3Z'

- loc[] used to subset the data
- '&' – to filter based on condition
- groupby() applied to group the 'service_setting' and the sum of 'count of appointments' calculated using sum().
- sort_values(ascending=False) – sorts in descending order

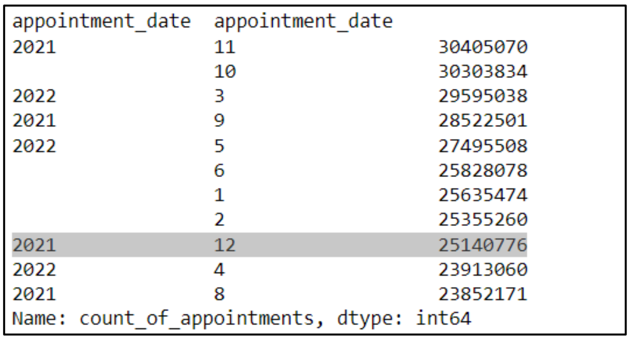


```
# Determine the number of records available for the period and the location.
# Use a filter for the location filter_loc
# Use a filter for the time period - filter_date
filter_loc = nc["sub_icb_location_name"]=="NHS North West London ICB - W2U3Z"
filter_date = nc["appointment_date"].isin(pd.date_range("2022-01-01", "2022-06-01"))

# Create a subset of nc DataFrame containing only the filtered date period and location filter
nc_subset = nc.loc[filter_loc & filter_date]

# View the subset DataFrame
nc_subset

# Determine the service setting that reported the most number of appointments
# Use groupby to get the total number of appointments in each service setting
# Sort_values(ascending =False) will sort from high to low values
nc_service = nc_subset.groupby('service_setting')['count_of_appointments'].\
sum().sort_values(ascending=False)
print(nc_service)

# iloc - to retrieve the first row of the sorted series - this is the maximum
print(f"\n\nMost number of appointments :\n{nc_service.iloc[0:1]}")
```

```
service_setting
General Practice              4804239
Unmapped                       391106
Other                          152897
Primary Care Network           109840
Extended Access Provision       98159
```



- Service_Setting GP has the highest number of appointments in the given period.
- groupby() applied on the appointment date



```
appointment_date  appointment_date
2021              11                  30405070
                  10                  30303834
2022              3                   29595038
2021              9                   28522501
2022              5                   27495508
                  6                   25828078
                  1                   25635474
                  2                   25355260
2021              12                  25140776
2022              4                   23913060
2021              8                   23852171
Name: count_of_appointments, dtype: int64
```

2.3 Initial Findings

- November 2021 has the highest number of appointments - This is probably with the onset of winter and the winter flu.
- December 2021 has one of fewer number of appointments – This can be as it is holiday period. This period needs to be analysed further to see if there have been a lot of missed appointments and or if there has been staff shortage.

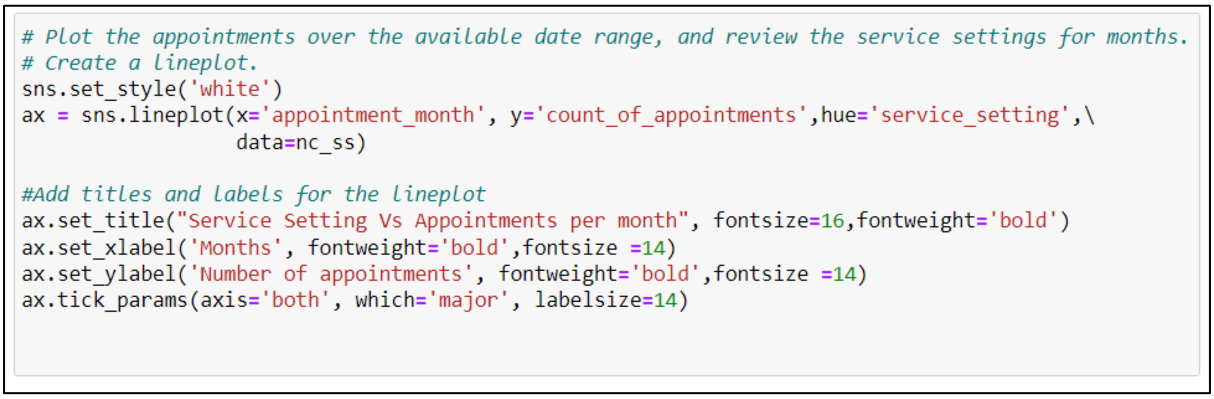
3 Visualisation and Insights

3.1 Initial trends

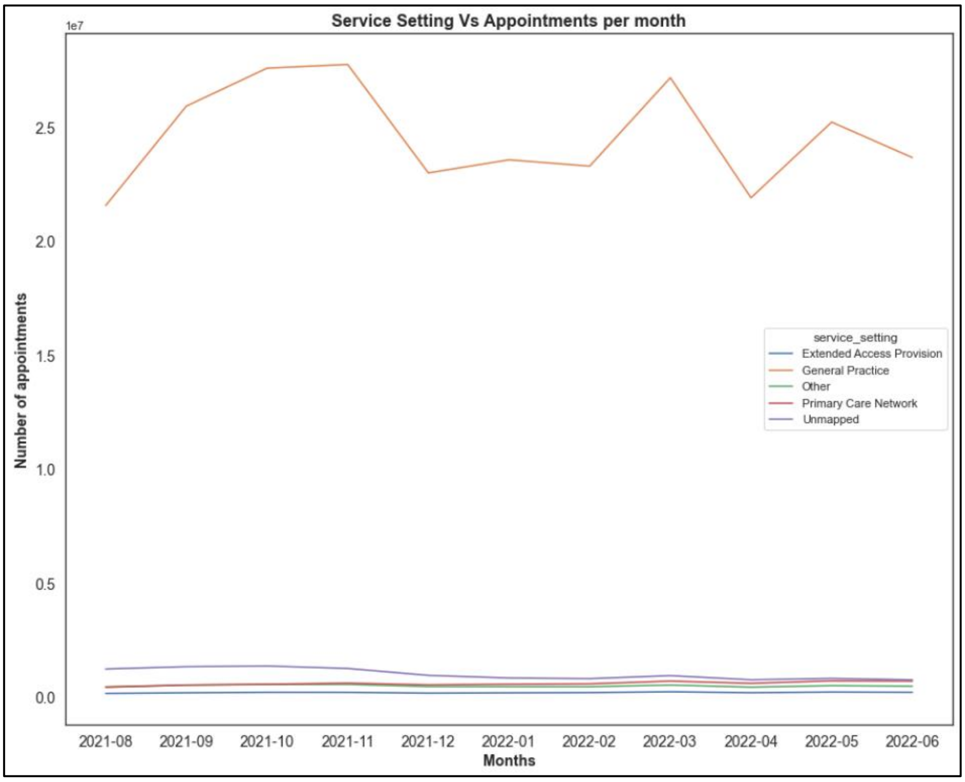
Number of appointments analysed basis service settings, context types and national categories to see the monthly and seasonal trends.

Service settings

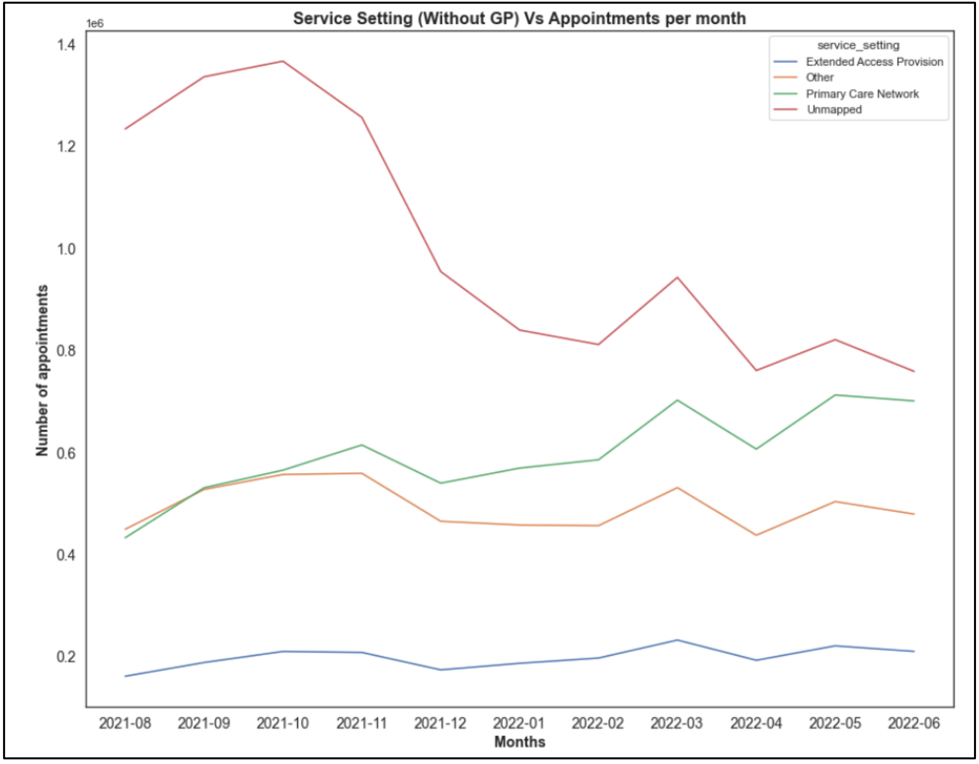
- The appointment months and service settings are grouped
- Seaborn lineplot used to visualize results.
- Background style set to white.
- x -axis shows the months, y axis takes the count of appointments. The different service settings viewed using 'hue'
- The label and titles included.
- 'General Practice' has the highest number of appointments



```
# Plot the appointments over the available date range, and review the service settings for months.
# Create a lineplot.
sns.set_style('white')
ax = sns.lineplot(x='appointment_month', y='count_of_appointments',hue='service_setting',\
                  data=nc_ss)

#Add titles and labels for the lineplot
ax.set_title("Service Setting Vs Appointments per month", fontsize=16,fontweight='bold')
ax.set_xlabel('Months', fontweight='bold',fontsize =14)
ax.set_ylabel('Number of appointments', fontweight='bold',fontsize =14)
ax.tick_params(axis='both', which='major', labelsize=14)
```



- Removed GP from the plot to remove skewness
- Significant number of appointments under the unmapped service category

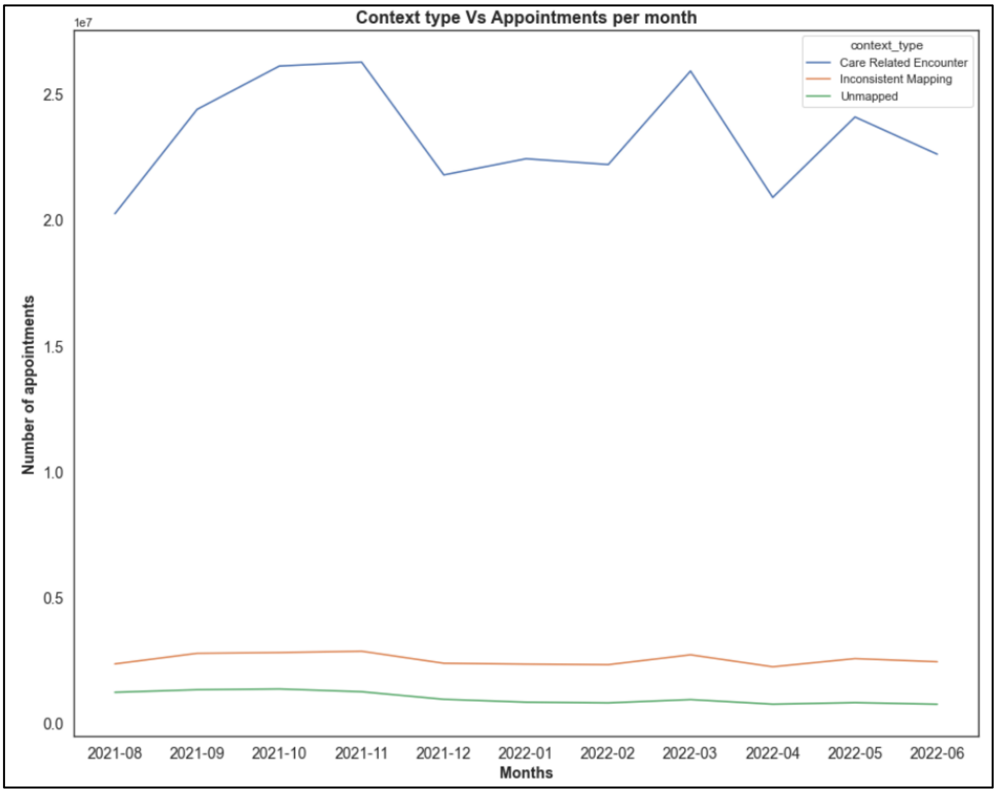


○

Context types

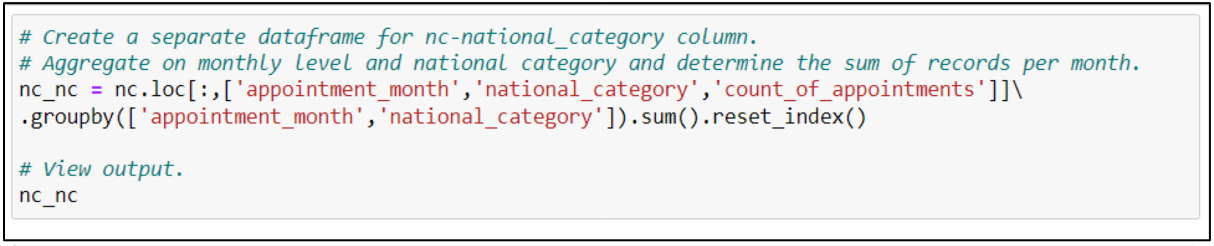
- The monthly appointments in various context types grouped

- ‘Care Related Encounters’ highest

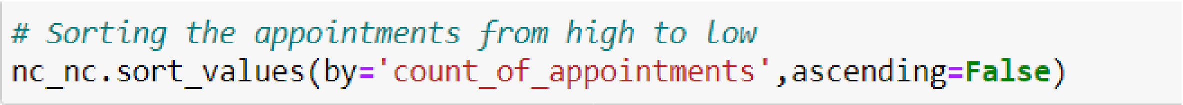


National Category

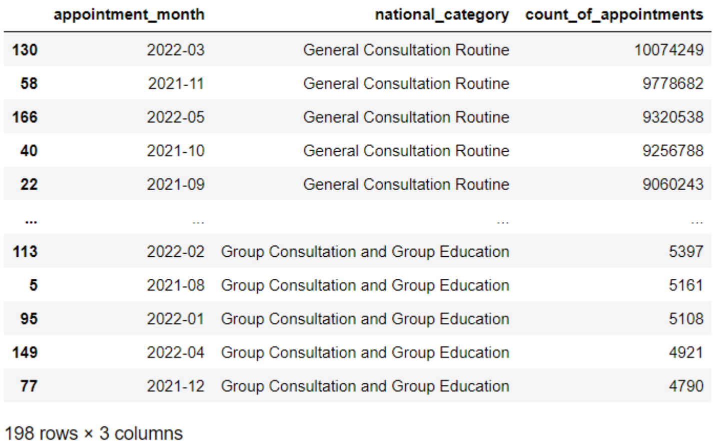
- General Consultation Routine highest.
- Planned clinic visits spike in October.



```
# Create a separate dataframe for nc-national_category column.
# Aggregate on monthly level and national category and determine the sum of records per month.
nc_nc = nc.loc[:,['appointment_month','national_category','count_of_appointments']]\
.groupby(['appointment_month','national_category']).sum().reset_index()

# View output.
nc_nc
```

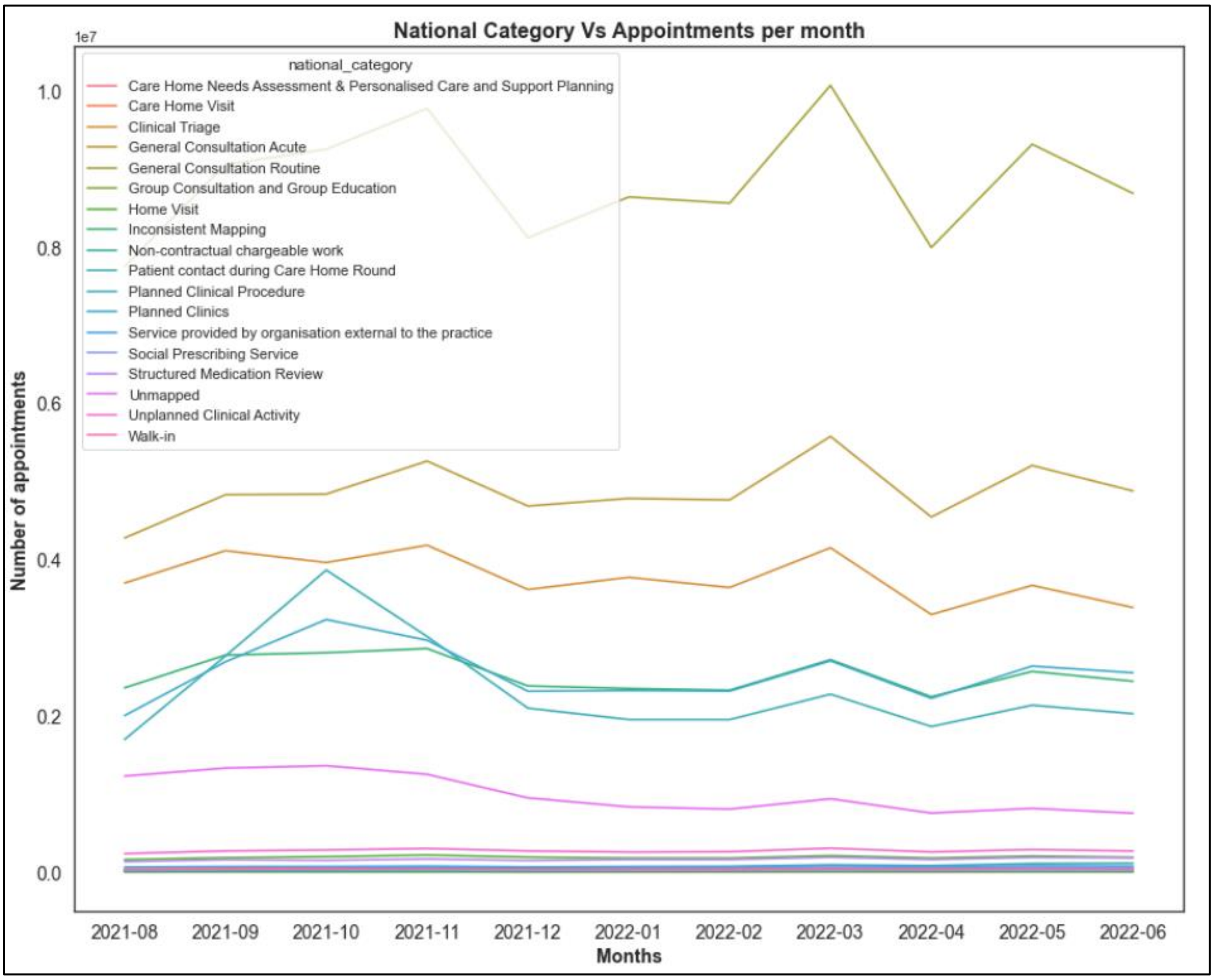


```
# Sorting the appointments from high to low
nc_nc.sort_values(by='count_of_appointments',ascending=False)
```



	appointment_month	national_category	count_of_appointments
130	2022-03	General Consultation Routine	10074249
58	2021-11	General Consultation Routine	9778682
166	2022-05	General Consultation Routine	9320538
40	2021-10	General Consultation Routine	9256788
22	2021-09	General Consultation Routine	9060243
...	...	...	...
113	2022-02	Group Consultation and Group Education	5397
5	2021-08	Group Consultation and Group Education	5161
95	2022-01	Group Consultation and Group Education	5108
149	2022-04	Group Consultation and Group Education	4921
77	2021-12	Group Consultation and Group Education	4790

198 rows × 3 columns

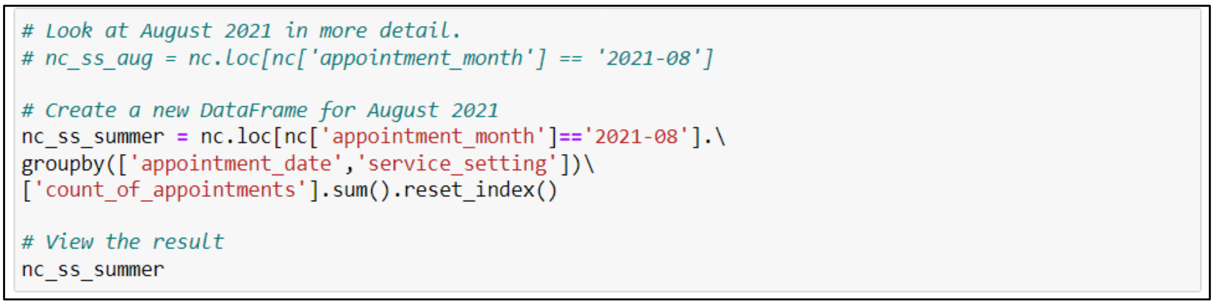


3.2 Seasonal Trends

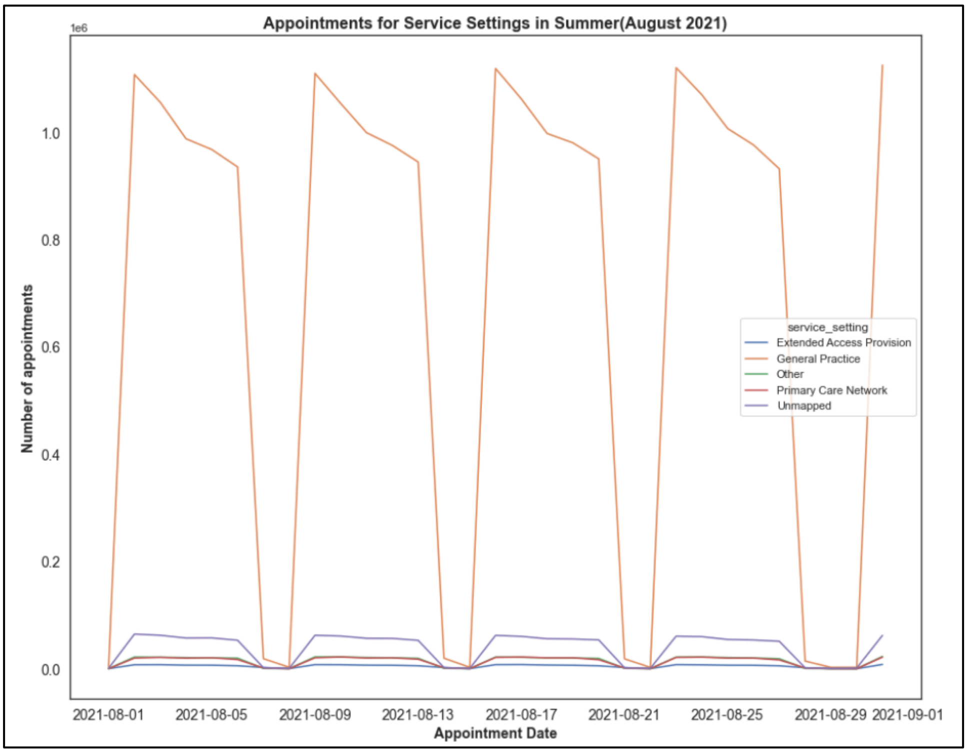
- Summer- August 2021
- Autumn – October 2021
- Winter – January 2022
- Spring – April 2022

Summer

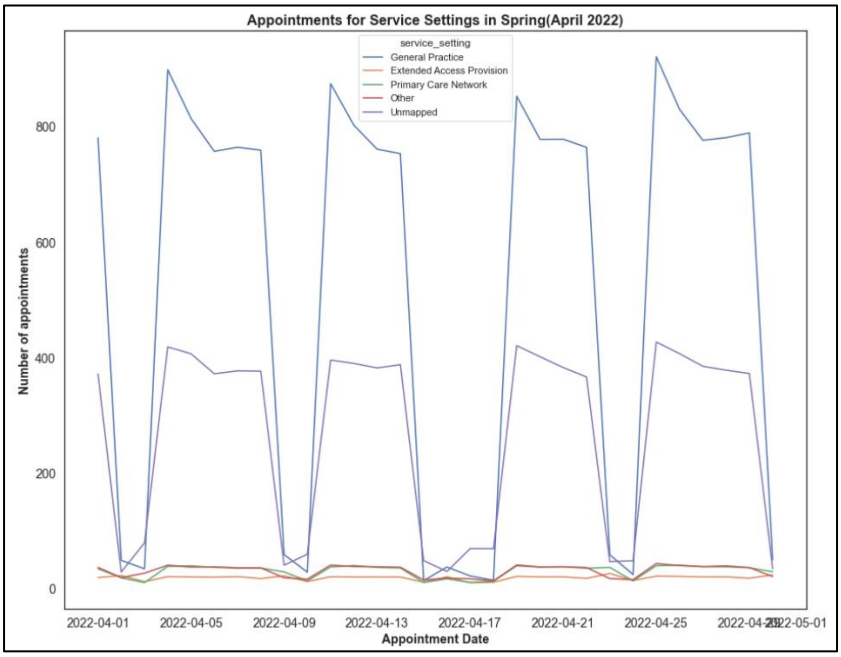
- Subset for August created
- Appointments across all the service settings is as expected peaking on a Monday and zero on weekends
- A similar trend observed in autumn and winter



```
# Look at August 2021 in more detail.
# nc_ss_aug = nc.loc[nc['appointment_month'] == '2021-08']

# Create a new DataFrame for August 2021
nc_ss_summer = nc.loc[nc['appointment_month']=='2021-08'].\
groupby(['appointment_date','service_setting'])\
['count_of_appointments'].sum().reset_index()

# View the result
nc_ss_summer
```



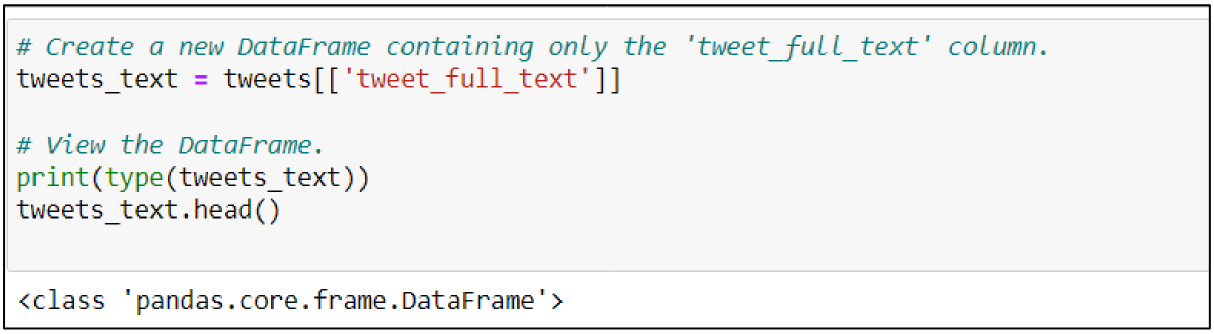
- In spring similar pattern but lower number of appointments



3.3 Twitter analysis and Trends

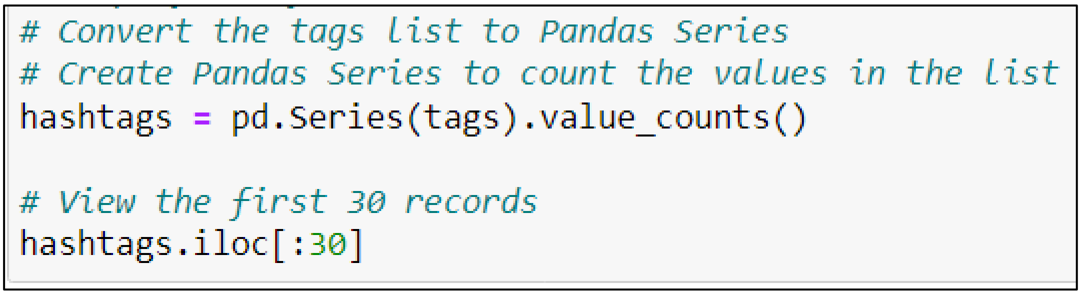
- The tweets.csv file imported into jupyter and loaded into a dataframe.
- 'tweet_retweet_count' and 'tweet_favorite_count' columns are explored
- Value_counts() to get the number of unique values.
- From the result we can check for tweets that have retweet count over 100 to understand the tweets in depth.

To review the hashtags, a separate dataframe created containing only the 'tweet_full_text' column



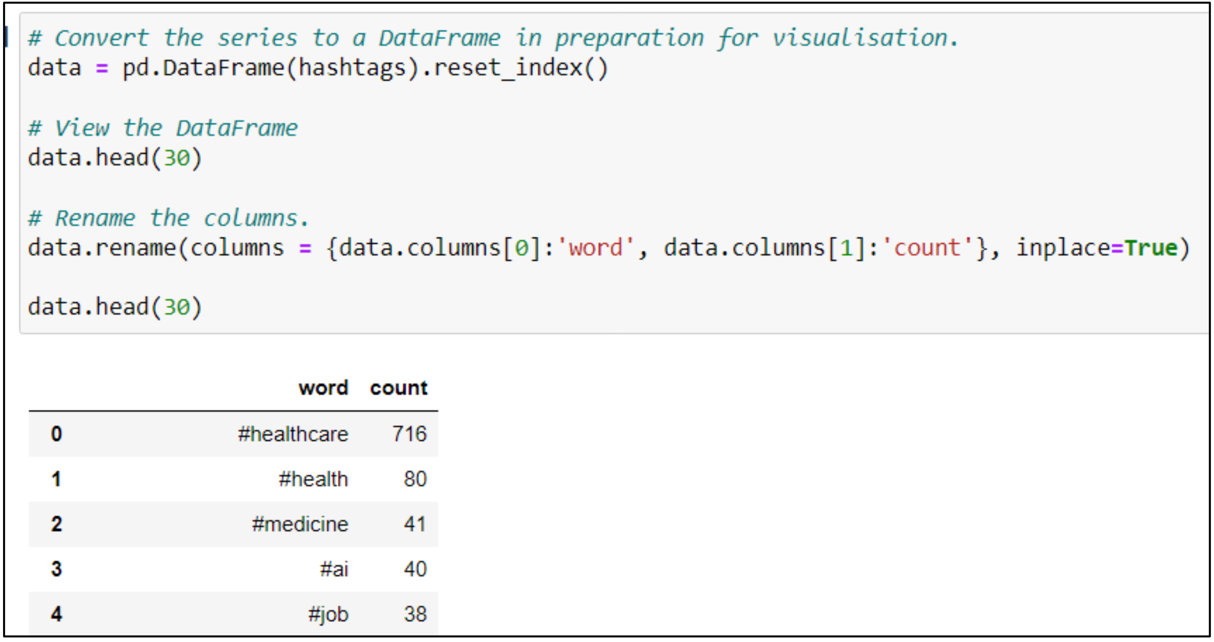
```
# Create a new DataFrame containing only the 'tweet_full_text' column.
tweets_text = tweets[['tweet_full_text']]

# View the DataFrame.
print(type(tweets_text))
tweets_text.head()

<class 'pandas.core.frame.DataFrame'>
```

- List created that contains only the values with # symbol. – tags[]
- tags[] list converted to a Pandas Series - hashtags



```
# Convert the tags list to Pandas Series
# Create Pandas Series to count the values in the list
hashtags = pd.Series(tags).value_counts()

# View the first 30 records
hashtags.iloc[:30]
```

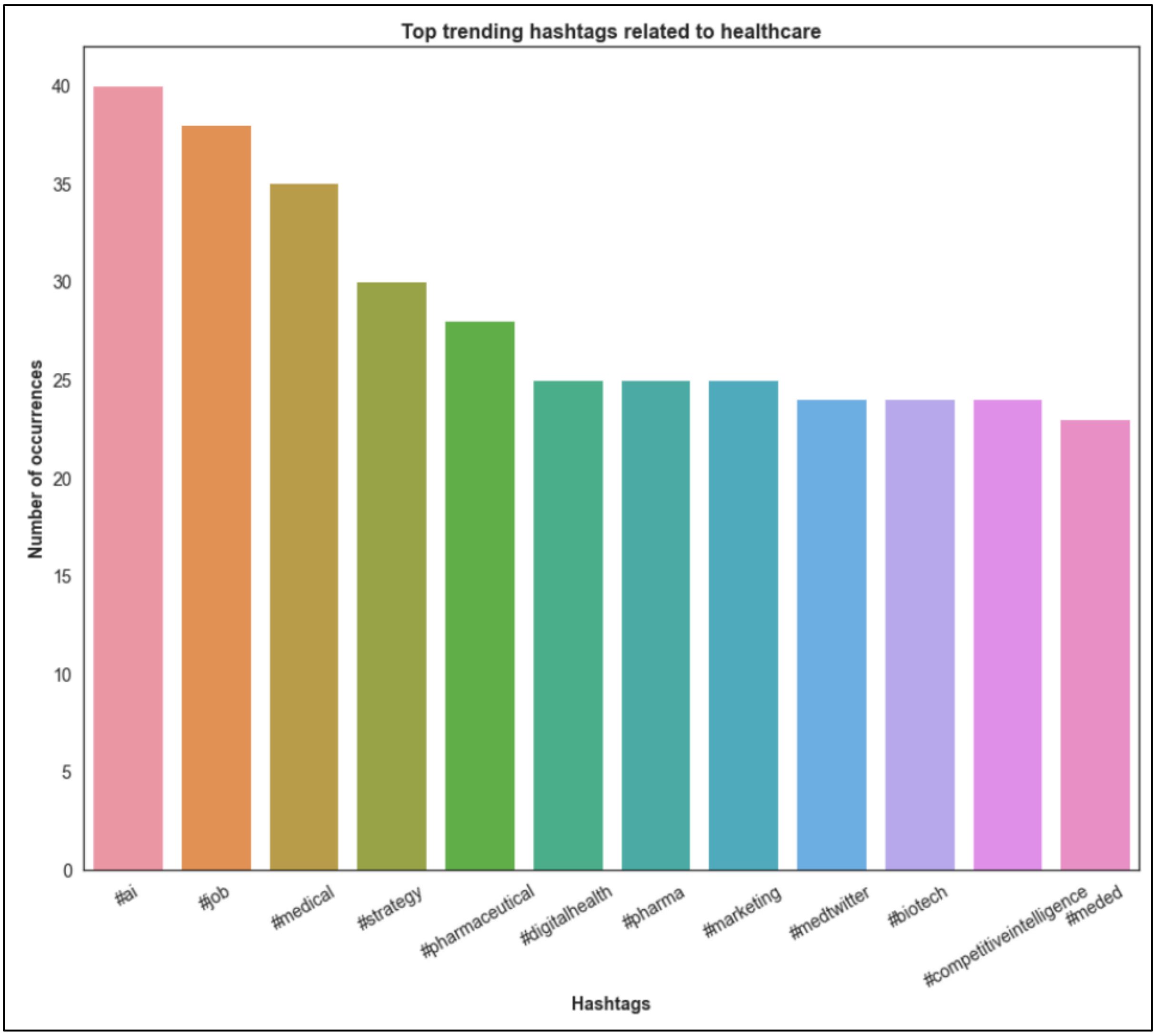
- Value_counts() – used
- #healthcare with a count of 716 is an outlier that skews the barplot.



```
# Convert the series to a DataFrame in preparation for visualisation.
data = pd.DataFrame(hashtags).reset_index()

# View the DataFrame
data.head(30)

# Rename the columns.
data.rename(columns = {data.columns[0]:'word', data.columns[1]:'count'}, inplace=True)

data.head(30)
```

	word	count
0	#healthcare	716
1	#health	80
2	#medicine	41
3	#ai	40
4	#job	38

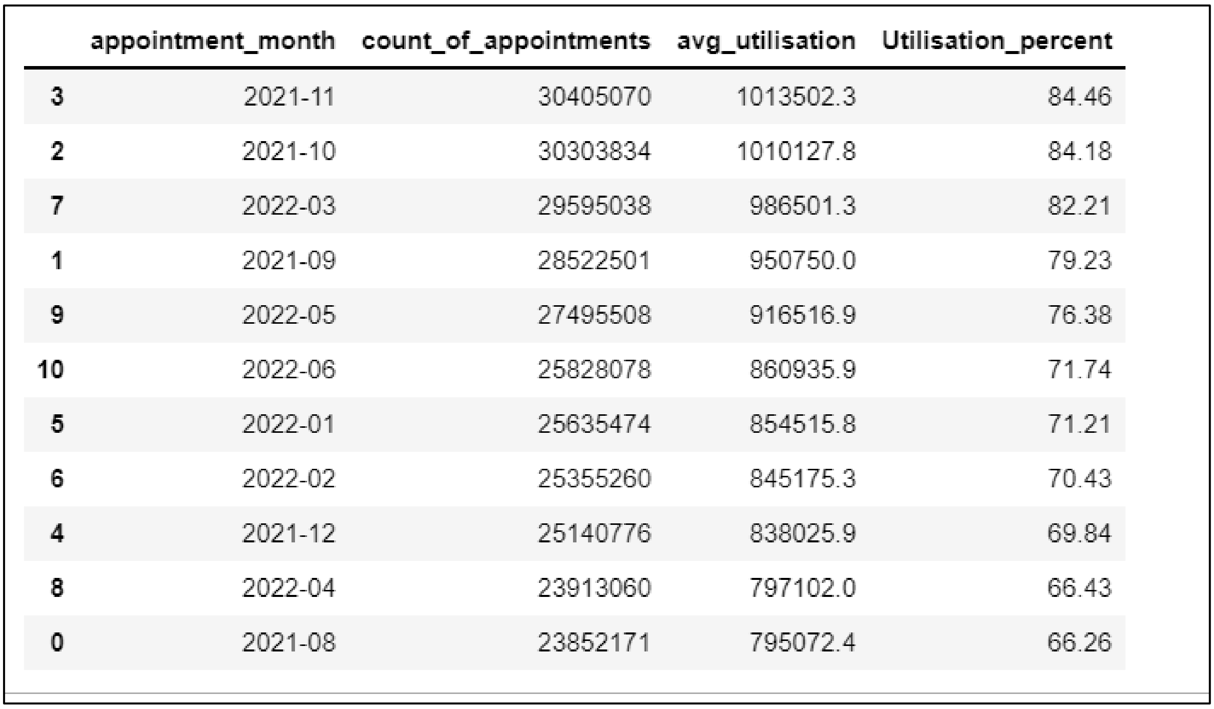
- Overrepresented hashtags removed using the quartiles and IQR for the 'count' column.
- Visualisation improved by using only the hashtags with a count>20



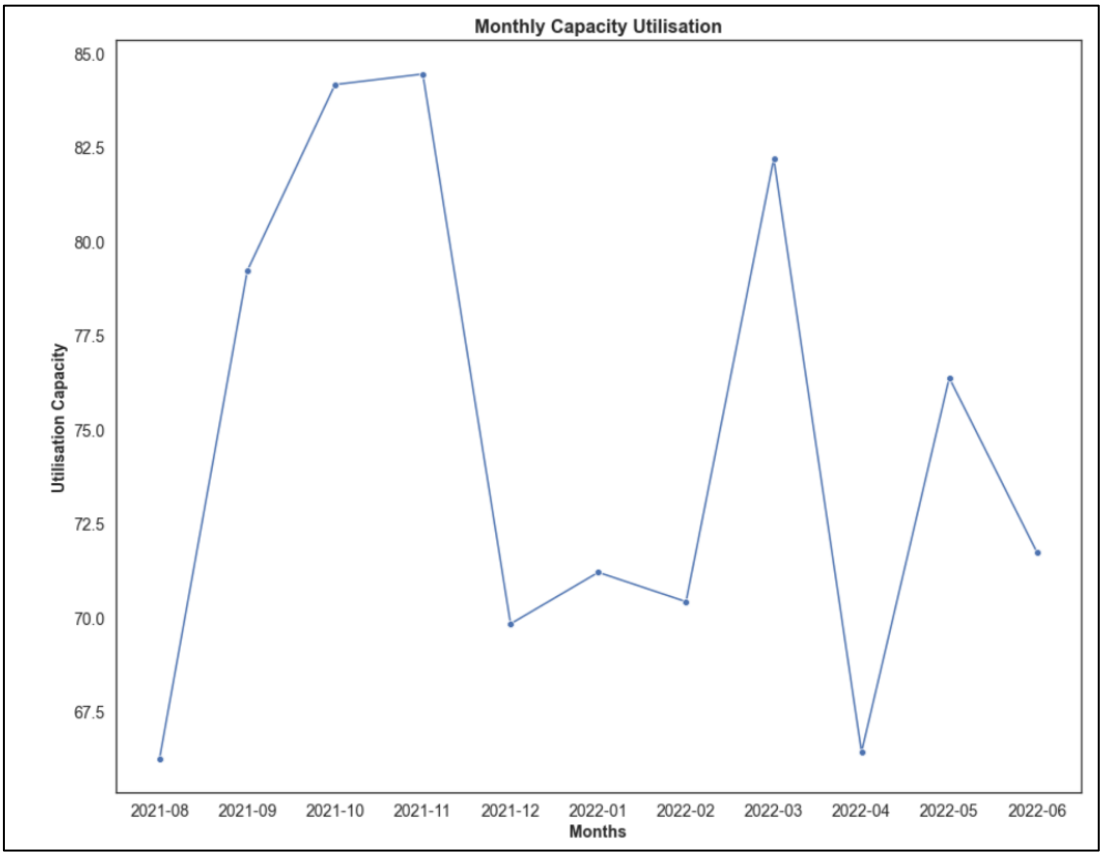
4 Patterns and Predictions

NHS Utilisation Capacity

- To check if NHS should increase staff levels, NHS utilization evaluated.
- Total number of appointments grouped for each month and compared with the maximum NHS capacity of 1,200,000 appointments per day.
- There is adequate staff and capacity in the NHS networks.
- The maximum utilization percentage was approximately 85 % in November

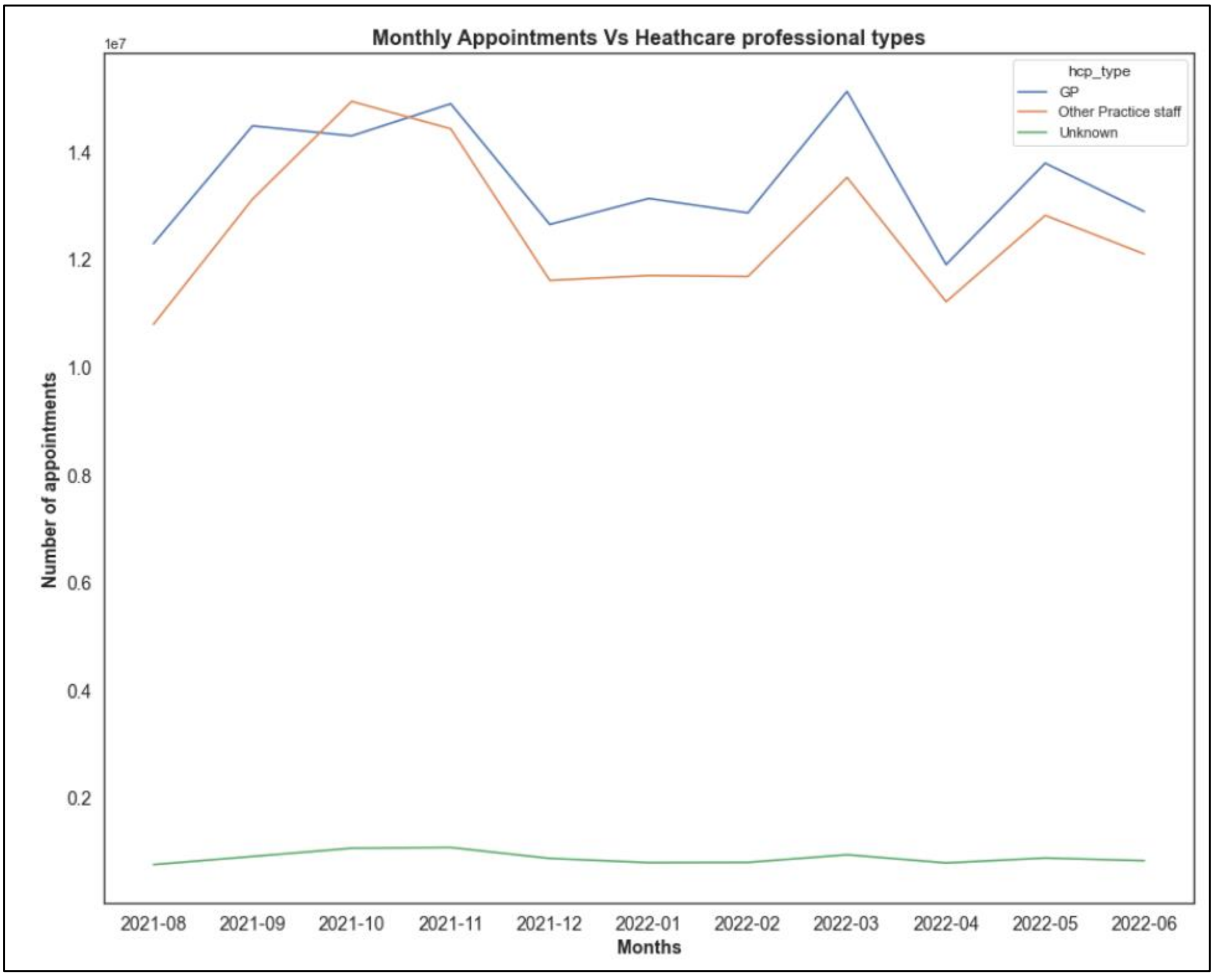


	appointment_month	count_of_appointments	avg_utilisation	Utilisation_percent
3	2021-11	30405070	1013502.3	84.46
2	2021-10	30303834	1010127.8	84.18
7	2022-03	29595038	986501.3	82.21
1	2021-09	28522501	950750.0	79.23
9	2022-05	27495508	916516.9	76.38
10	2022-06	25828078	860935.9	71.74
5	2022-01	25635474	854515.8	71.21
6	2022-02	25355260	845175.3	70.43
4	2021-12	25140776	838025.9	69.84
8	2022-04	23913060	797102.0	66.43
0	2021-08	23852171	795072.4	66.26



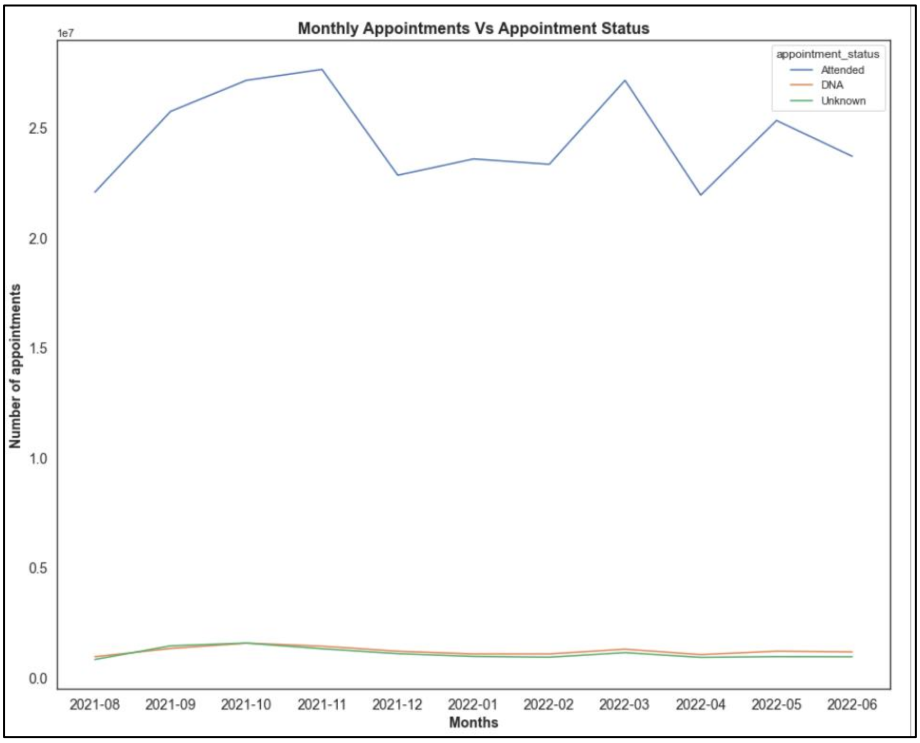
Healthcare Professionals

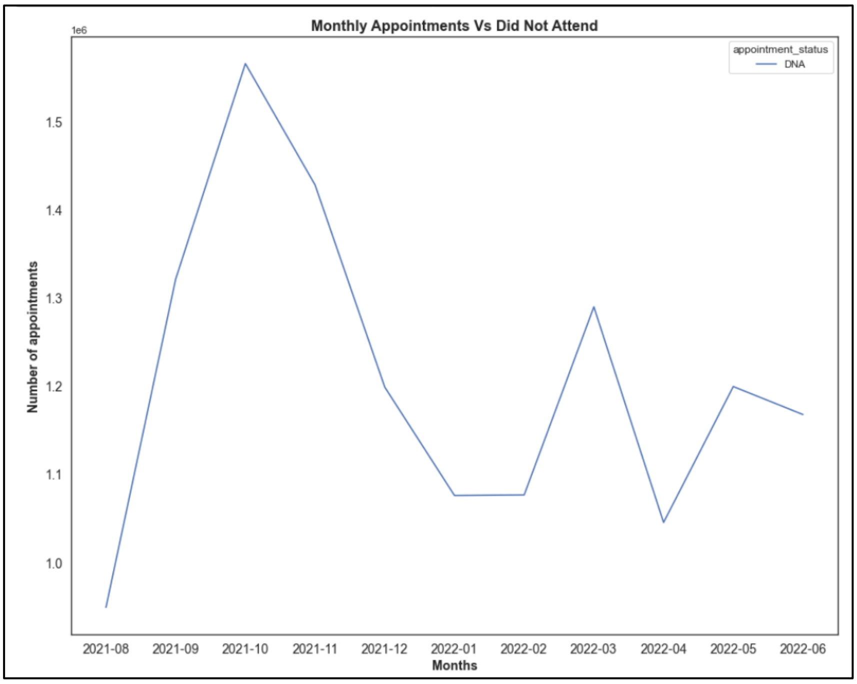
- Monthly appointments for different hcp_types plotted
- The 'Other Practise Staff' has a greater number of appointments in October & November- busiest months.
- Both GP and Other Practise Staff are well utilised.



Appointment Status

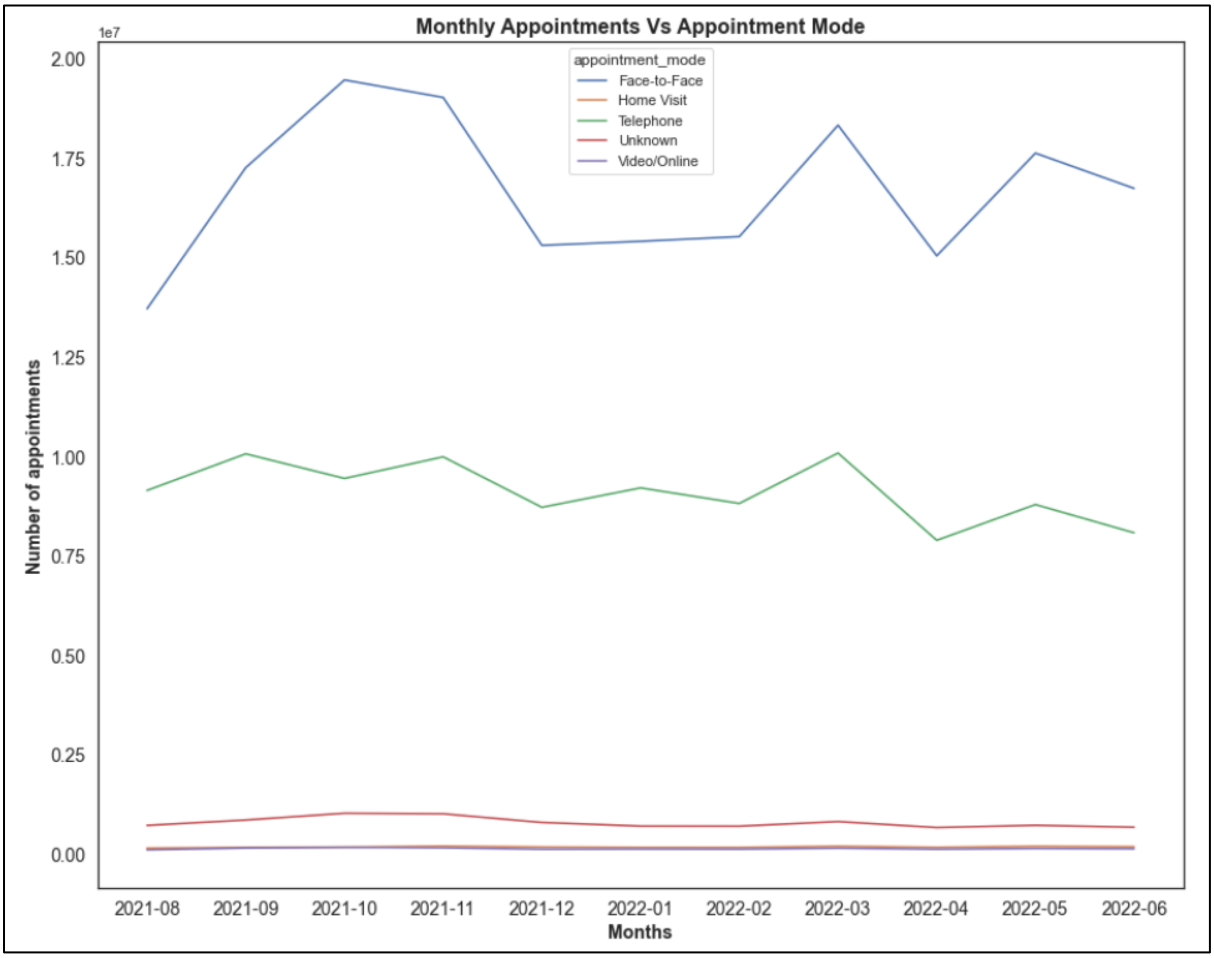
- Appointment status Vs monthly appointments plotted
- Number of appointments that were not attended (DNA status) follows the total number of appointment in that month –highest number of DNA in October and a lower second peak in March
- The Attended appointments follow similar trend across the months as with hcp_typ and service settings.





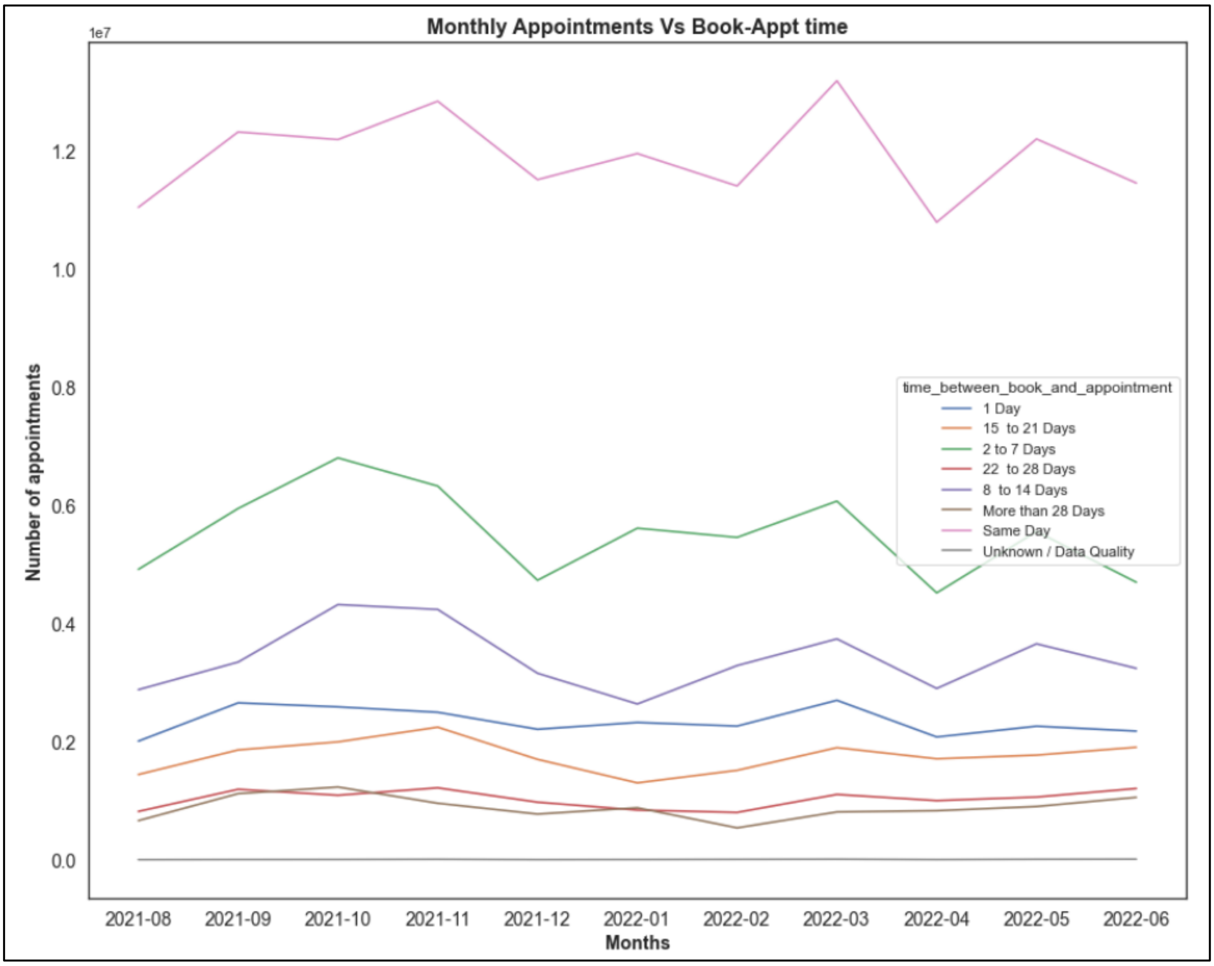
Appointment types

- Face-to-face appointments are the highest following a similar rise in Oct-Nov and decrease in Dec-Jan months. They peak again in March 2022.
- Followed by telephonic appointments



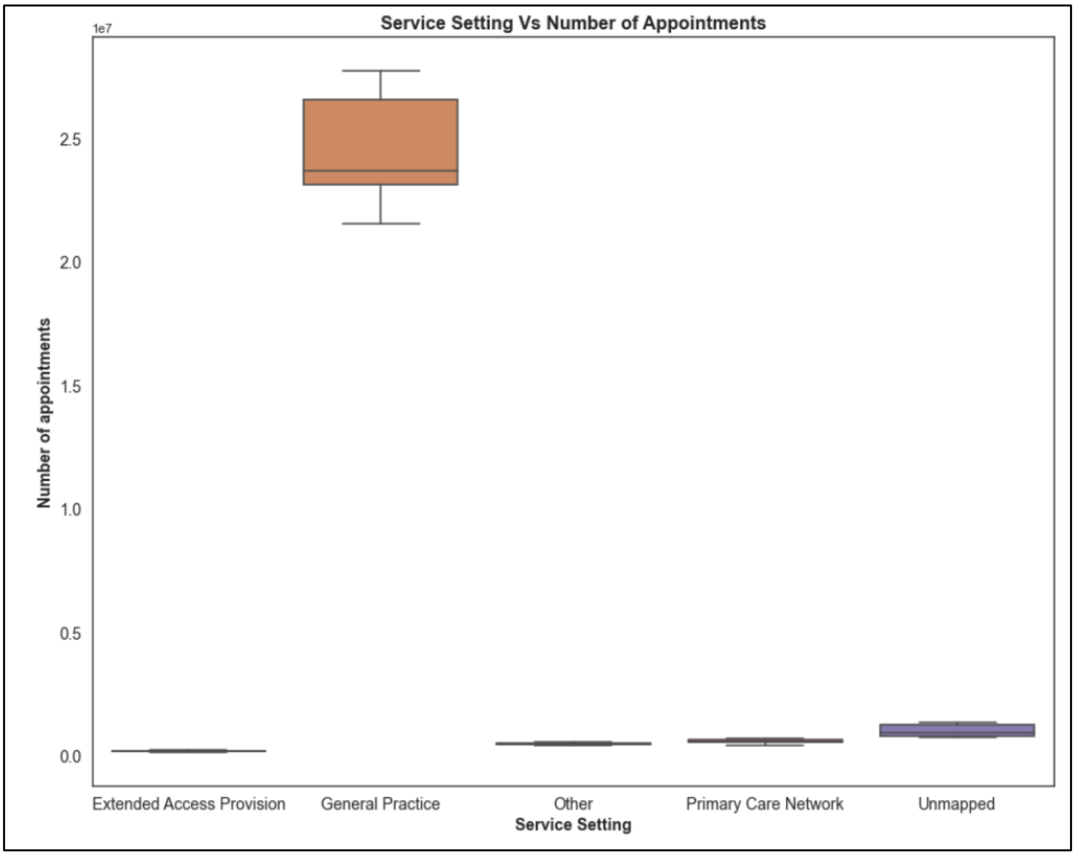
Time between booking and appointment

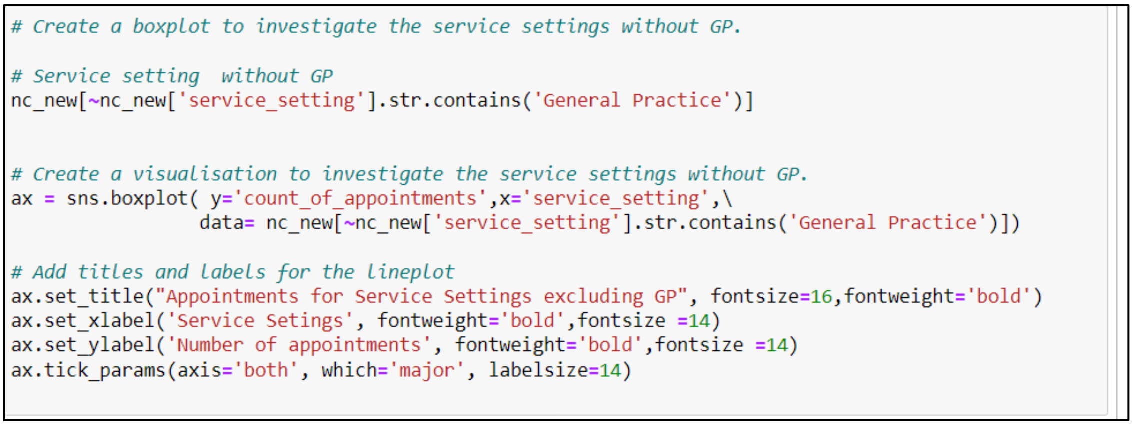
- Same day appointments are highest followed by appointments within week (2-7 days)
- 1 day appointments are lower than 2 weeks appointments



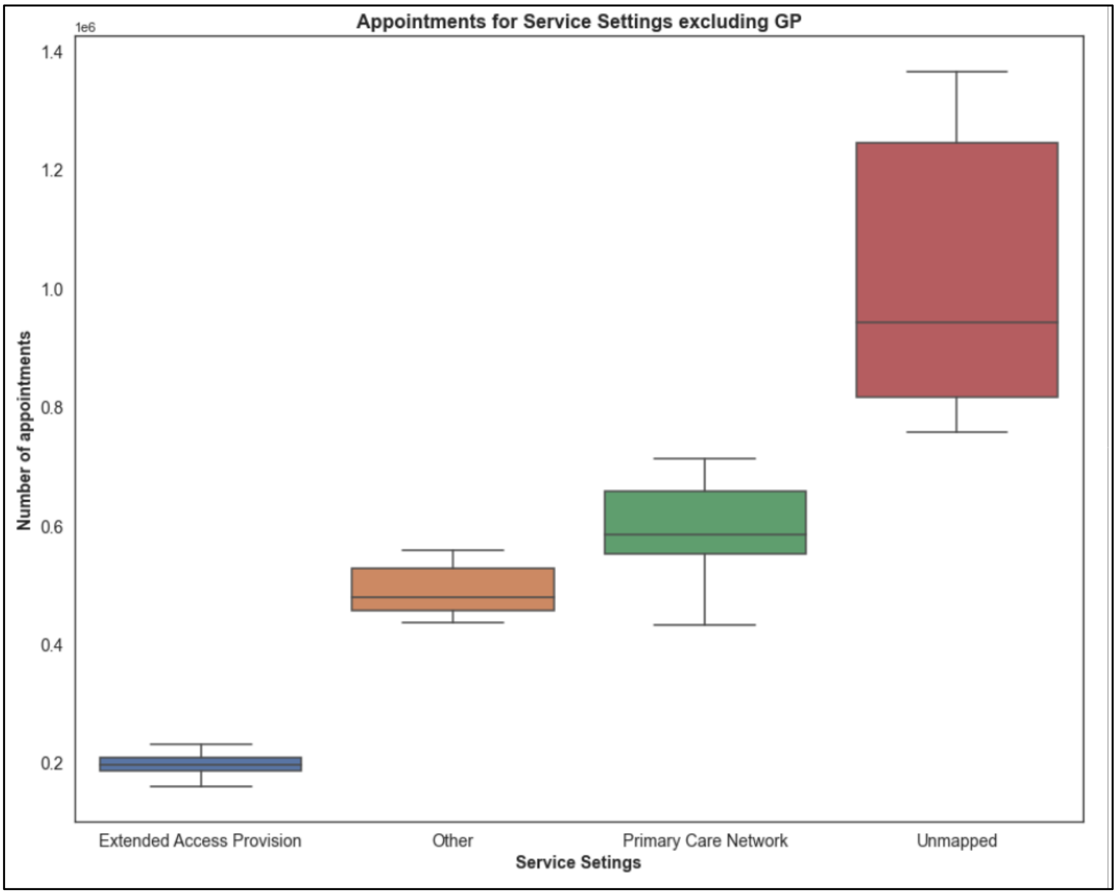
Service setting spread

- Boxplot used to analyse the spread of service settings across the appointments
- General Practice is the most utilised
- Removing General Practice, we can visualise the spread better





```
# Create a boxplot to investigate the service settings without GP.

# Service setting  without GP
nc_new[~nc_new['service_setting'].str.contains('General Practice')]


# Create a visualisation to investigate the service settings without GP.
ax = sns.boxplot( y='count_of_appointments',x='service_setting',\
                 data= nc_new[~nc_new['service_setting'].str.contains('General Practice')])

# Add titles and labels for the lineplot
ax.set_title("Appointments for Service Settings excluding GP", fontsize=16,fontweight='bold')
ax.set_xlabel('Service Setings', fontweight='bold',fontsize =14)
ax.set_ylabel('Number of appointments', fontweight='bold',fontsize =14)
ax.tick_params(axis='both', which='major', labelsize=14)
```



- A high number of appointments are unmapped.
- Data Entry needs improvement so that accurate data is available for analysis

5 Summary of Findings and Recommendations

- NHS resources are adequately utilised.
- November has the highest number of appointments, then October, March.
  -About 15 % buffer available before NHS reaches its maximum utilisation capacity
  -This includes DNA status appointments
  -There is enough buffer available for capacity utilisation.
- August, April, December - months with fewer appointments
  -About 35% buffer available to reach maximum capacity
- NHS need not consider increasing staff level, although more staffing data is required for in-depth study of adequate staff capacity in the network
- Significant high number of DNA appointments – to be further investigated to understand the cancelled / missed appointments
- General Practice Service setting is the most utilised category.
- Unmapped Service setting – high number of appointments - potential area for improvement in future analysis.