NHS UK – Appointments Data: Diagnostic Analysis using Python

Sujith Kumaar K C

28th October 2024

1. Problem Statement:

The NHS faces challenges in understanding **service utilisation** and **missed appointments**, which incur significant costs. There is a need to investigate missed appointment patterns, evaluate resource capacity, and explore potential improvements.

2. Business Objective Analysis:

Through a 5-why structured analysis approach, we identified that the NHS is facing budget constraints and needs to optimize resource allocation while improving service delivery. The analysis focuses on

- Understanding utilization trends
- Assessing staff and capacity adequacy
- Analysing missed appointments.
- Evaluate the value of external data sources for decision-making.

3. Data Assumptions:

Data assumptions defined based on metadata and data exploration to avoid ambiguity:

1. The data does not fully capture demand or capacity, as not all GP activities or practices in England are included. The datasets (ad, ar, nc) are assumed to accurately represent NHS appointments.
2. Data is collected from GP systems for everyday use, leading to variations in quality since it's not designed for analysis.
3. Data limitations rising from collected practices cannot be definitively determined or resolved, for example 'Unknown' appointment status cannot be determined if they were attended or missed.
4. Misrepresentation of appointment modes (e,g., video calls marked as face-to-face) exists. The provided data is used without estimates for misrepresentation.

Time Context: The COVID-19 pandemic disrupted NHS services from March 2020. Appointment backlogs and staff shortages have affected service normalization.

4. Analysis

4.1 Data Understanding and Preparation

Data Sources used for the analysis are:

- **actual_duration.csv:** Appointment duration data
- **appointments_regional.csv:** monthly aggregated ICB-level appointment data
- **national_categories.xlsx:** Categorized appointment data daily data.
- **Tweets.csv**: Mined twitter data for scoping out external data analytics.
- **LocationBreakup.xlsx:** Mapping information for regions, ICB locations, and sub-ICB location codes.

  Source: https://geoportal.statistics.gov.uk/datasets/2bca16d4f8e4426d80137213fce90bbd_0/explore

4.2 Data Ingestion, Wrangling and Manipulation

The necessary libraries – pandas, NumPy, Seaborn, Matplotlib, Datetime and relativedelta were imported at the start of the analysis. Data from each source was loaded into pandas Dataframes using `pd.read_csv()` and `pd.read_excel()` functions.

- ar – Appointments regional monthly data (Jan 2020 – June 2022)
- nc – National categories daily data (Aug 2021 – June 2022)
- ad – Appointments duration daily data (Dec 2021 – June 2022)

A basic sense check of the data using `.info()` `.describe`, and `.shape()` functions were performed.

Missing values: Dataframes were checked for missing values using `.isna().sum()`. Since there were no missing values, no action was taken for filling.

Duplicates: All datasets were checked for duplicates using `duplicated()` method. 21,604 duplicates identified in AR(`appointments_regional`) was analysed with a flag column along with monthly aggregations for a subset of AR and NC to validate appointment counts. The duplicates were retained to keep the same counts across both datasets.

Data type conversions: `appointment_date` and `appointment_month` columns were converted to datetime objects as required using `pd.to_datetime()`.

Location mapping was implemented with `pd.merge()` against externally sourced Locations dataframe on reference code columns (`region_ons_code, icb_ons_code, sub_icb_ons_code`).

- Region, ICB and Sub-ICB location information were added to the NC and AD dataframes.
- Region and ICB information were added to the AR Dataframe.

Data consistency and validation:

- Data validation included cross-dataset appointment count verification between `appointments_regional` and `national_categories` for the period August 2021 to June 2022, and comparison of 'Attended' status between `appointments_regional` and `actual_duration` from December 2021 to June 2022.'

Data merging opportunities: Full integration between appointment-level datasets wasn't possible due to aggregation differences

Additional wrangling and manipulations were performed as part of the specific analysis with copies of the dataframes as and when necessary.

Example:

1. Weekday column added to nc_copy based on appointment_date as part of the utilisation analysis:

```
nc_util = nc_merged.copy()
nc_util['weekday'] = nc_util['appointment_date'].dt.strftime('%A')

# Create a categorical column with days in order
day_order = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday']
nc_util['weekday'] = pd.Categorical(nc_util['weekday'], categories=day_order, ordered=True)
```

2. Multiple dataframes created from AR dataframe using groupby functions for 1/2/3 factors for Missed appointments analysis.

```
#1. Subsets of Missed and Attended appointments
DNA = ar_merged[ar_merged['appointment_status']=='DNA']
Attended = ar_merged[ar_merged['appointment_status']=='Attended']
Total_DNA_appts = DNA['count_of_appointments'].sum()

# 2. DNA Appointments grouped by different factors within AR Dataframe - Individual groupby
DNA1 = DNA.groupby(['appointment_mode']).sum('count_of_appointments').sort_values(by='count_of_appointments',ascending = False).reset_index()
DNA2 = DNA.groupby(['hcp_type']).sum('count_of_appointments').sort_values(by='count_of_appointments',ascending = False).reset_index()
DNA3 = DNA.groupby(['time_between_book_and_appointment']).sum('count_of_appointments').sort_values(by='count_of_appointments',ascending = False).reset_index()
DNA4 = DNA.groupby(['region_name']).sum('count_of_appointments').sort_values(by='count_of_appointments',ascending = False).reset_index()
DNA5 = DNA.groupby(['icb_name']).sum('count_of_appointments').sort_values(by='count_of_appointments',ascending = False).reset_index()
```

4.3 Exploratory Data Analysis

EDA was performed to gain an initial understanding of NHS appointment data, its structure, distributions, and potential relationships between variables.

1. **Location breakdown:** Locations were broken down at region, icb and sub-icb level using `nunique()` function. `groupby()` functions were used to group and get the sum of appointments at each level and to identify the top locations. Formatted tables were used to quickly identify patterns based on which visualizations were created for business presentations.
2. **Categorical Breakdown:** An analysis was conducted to examine appointment distributions across various categories:
   - Service settings
   - Context types
   - National categories
   - Appointment status
   - Healthcare provider (HCP) type
   - Appointment mode
   - Time between booking and appointment duration

   To streamline this analysis, a custom function - `analyze_category()` was developed. This function:
   - Aggregates appointment counts on a category using groupby operations
   - Calculates percentage of total appointments
   - Generates formatted tables for easy interpretation
   - Returns a grouped dataframe for future analysis

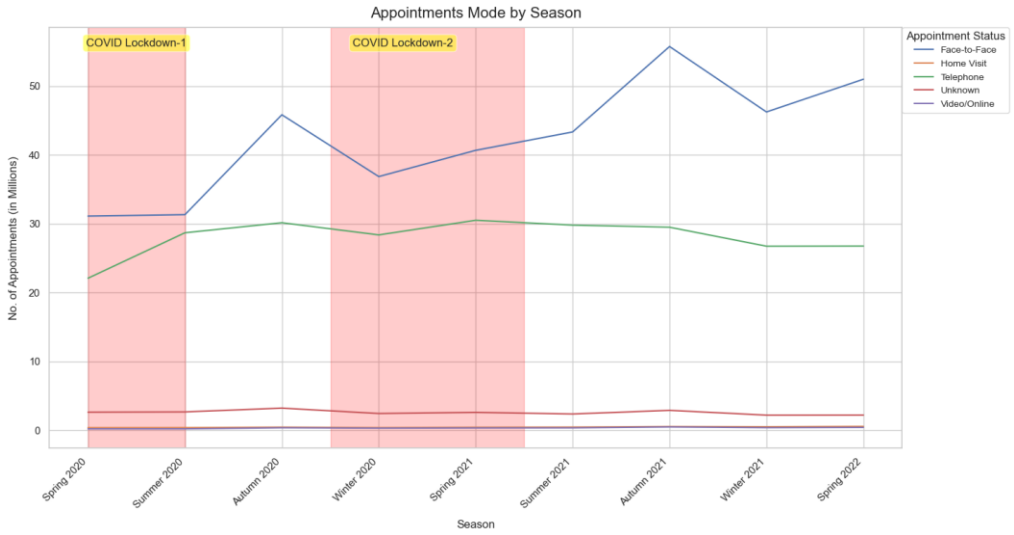
   This approach enabled efficient analysis across multiple categories, providing valuable insights into appointment distribution patterns.
3. **Appointments and Records by Month:** `groupby()` on appointment_month data with `sum()` and `count()` aggregations applied on `count_of_appointments` were performed on all 3 dataframes, flattened and concatenated to compare and analyse.
4. **Monthly and Seasonal Trends:** Copies of original dataframes were created using `.copy()`. Appointment_month column was changed to string using `.astype(str)` for easier chart creations. Dataframes were created for different categories (service settings, context type, national categories, HCP type, appointment mode, time between booking and appointment) using `groupby()` functions with appointment_month and category, with `sum()` on count_of_appointments. Where necessary, subsets of the resultant dataframes were created to visualize and analyse monthly trends individually.

   Example:

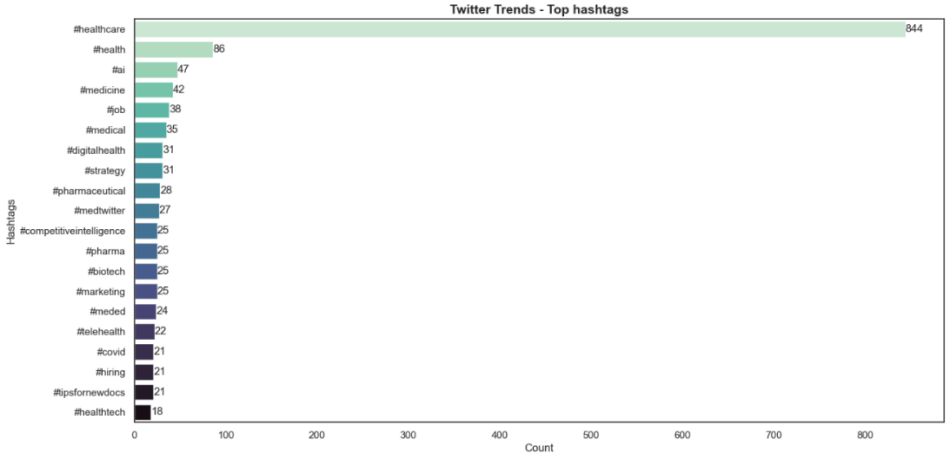
```
# Dataset creation for charts
#Service setting
nc_ss = nc_copy.groupby(['appointment_month','service_setting']).sum(['count_of_appointments']).reset_index()
# subsets of nc_ss into GP and Other service settings as GP counts are high
gp_service_setting_monthly = (nc_ss[nc_ss['service_setting'] == 'General Practice'])
other_service_setting_monthly = (nc_ss[nc_ss['service_setting']!= 'General Practice'])
print ('\nService Settings Monthly-Sample data')
print (nc_ss.head())
```

   Seasonal Trends were analysed by adding a season column to the dataframes using a custom assign_season function which assigned the season based on the month value from appointment_month column that was passed. `Season_order` list was created and applied and used to order the seasons chronologically using `pd.Categorical()` function. `groupby()` was called on different categorical data elements to identify seasonal trends and plot them.

Seasonal chart: Appointments by Mode



5. **Twitter analysis:** Hashtags with their counts were obtained from `tweet_full_text` column. The hashtag entities and their counts were double checked against tweet_entities_hashtags column using `str.split(',').explode().str.strip()` functions. The top 20 trending hashtags were plotted using a barplot().



4.4 Data Analysis and Patterns

4.4.1. Utilisation Analysis:

The NHS provided maximum capacity utilisation rate of 1.2Million appointments / day was used on NC (daily-data) to begin the analysis. The daily utilisation rate showed certain days to have very low rates of 2% and 0.2% while others showed 90%+. This was further investigated by adding a weekday column using `.dt.strftime('%A')` on `appointment_day` and appointments grouped based on weekday. Subset dataframes were created for weekdays and weekends to check the overall trends of `daily_utilisation_rate` on weekdays and weekends with basic statistics like `min, max, mean and count.`

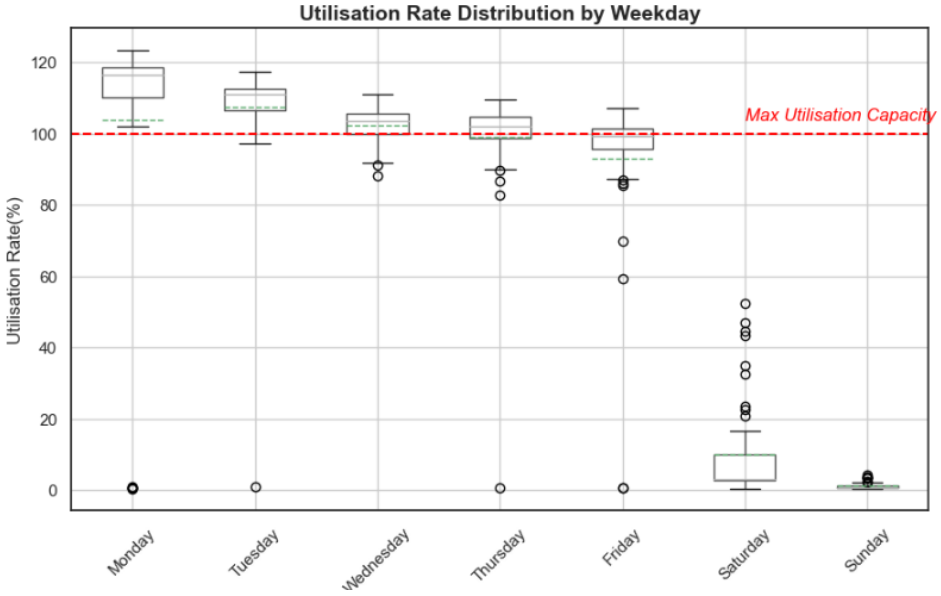
Code for subset creation

```
# Create weekdays dataframe
weekdays_df = nc_agg[~nc_agg['appointment_date'].dt.weekday.isin([5, 6])]

# Create weekends dataframe
weekends_df = nc_agg[nc_agg['appointment_date'].dt.weekday.isin([5, 6])]
```

Utilisation Statistics by weekday

	daily_utilisation_rate			
	min	mean	max	count
weekday				
Monday	0.45	103.64	123.35	48
Tuesday	0.85	107.30	117.29	48
Wednesday	88.22	102.40	110.99	48
Thursday	0.54	98.92	109.46	48
Friday	0.45	92.90	106.97	47
Saturday	0.41	9.88	52.36	47
Sunday	0.31	1.07	4.07	48

A boxplot was created to check for outliers on weekdays. Weekday outliers identified were individually analysed to confirm that they were bank holidays – mostly found on Mondays.



Average utilisation rates on weekdays and weekends were calculated. Based on the findings, the Monthly average values for the entire timeframe (30-month) in AR dataframe was calculated using a factor of 22.15 days in a month to identify actual utilization rates.

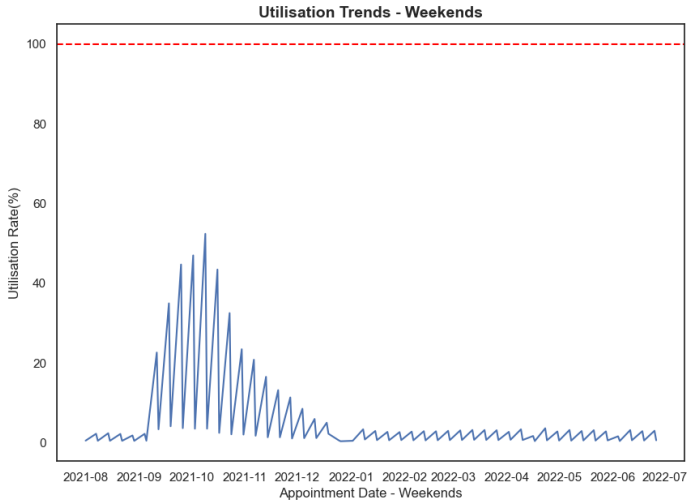
```
[110]: ar_df = ar_merged.groupby('appointment_month').sum('count_of_appointments')

       #Workingdays
       wkgdays_month = 22.15
       monthly_util = max_daily_util*wkgdays_month

       ar_df['weekday_utilisation']=(ar_df['count_of_appointments']/wkgdays_month).round(1)
       ar_df['monthly_utilisation_rate'] = (ar_df['count_of_appointments']/monthly_util*100).round(1)
       # View the DataFrame.
       ar_df
```

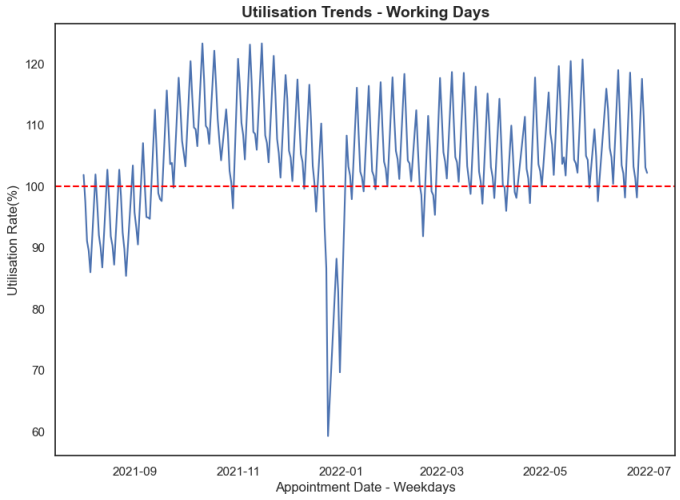
[110]:

appointment_month	count_of_appointments	weekday_utilisation	monthly_utilisation_rate
2020-01-01	27199296	1227959.2	102.3
2020-02-01	24104621	1088244.7	90.7
2020-03-01	24053468	1085935.3	90.5
2020-04-01	16007881	722703.4	60.2
2020-05-01	16417212	741183.4	61.8
2020-06-01	20690805	934122.1	77.8
2020-07-01	22491437	1015414.8	84.6
2020-08-01	20150520	909730.0	75.8

Line charts were created for the monthly utilisation rates with annotations at 100% for the max capacity utilisation rate to compare and identify trends.

Utilisation trend charts – Weekends Vs Weekdays





Note: Simple moving average was calculated and plotted but was dropped from further analysis as the data could be biased due to COVID related spikes and post COVID over-utilisation phase that was seen to be coming down in the recent couple of months.

4.4.2. Missed Appointments Analysis:

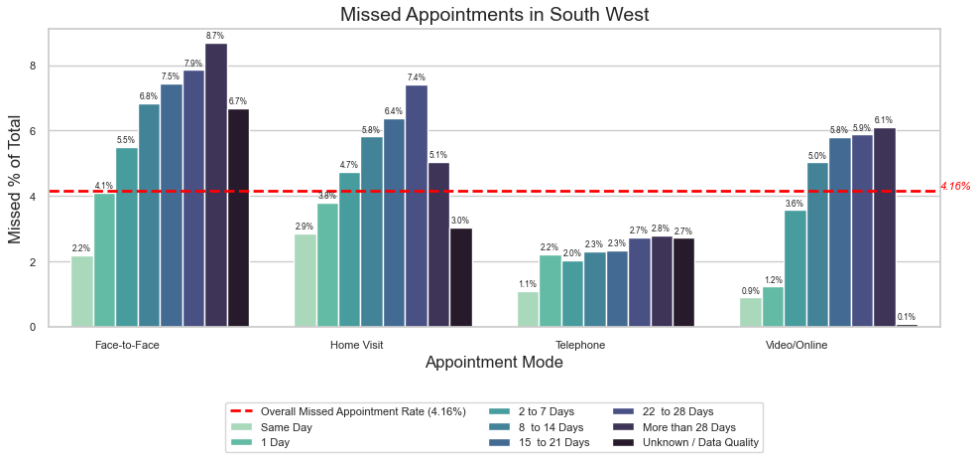
`ar_merged` with 30 months of data was used to create subset dataframes based on appointment status as DNA and Attended. The DNA subset was aggregated using the `groupby()` function for each categorical variable – appointment mode, HCP type, region, ICB, time between booking and appointment. This analysis revealed strong correlations between specific appointment modes and booking times, contributing significantly to missed appointments. Based on these **univariate findings**, below **multivariate analysis** was conducted to explore these relationships further:

1. Appointment mode + Appointment months
2. (Appointment Mode + Time between booking and appointment) AND Appointment month
3. (Region+Appointment Mode+ Time between booking and appointment) AND Appointment month

- `groupby()` was used extensively to group data by different factors and calculate aggregates.
- `pivot_table()` was used to restructure data for specific analysis, like comparing missed and total appointments monthly.

Patterns identified were charted using bar charts and histograms with annotations (4.16% for the 30-month period) and showcased.

Multivariate Analysis chart: Missed appointments by Appointment Mode & Booking to Appointment time, filtered for South West regional level



4.5 Visualisations and Insights:

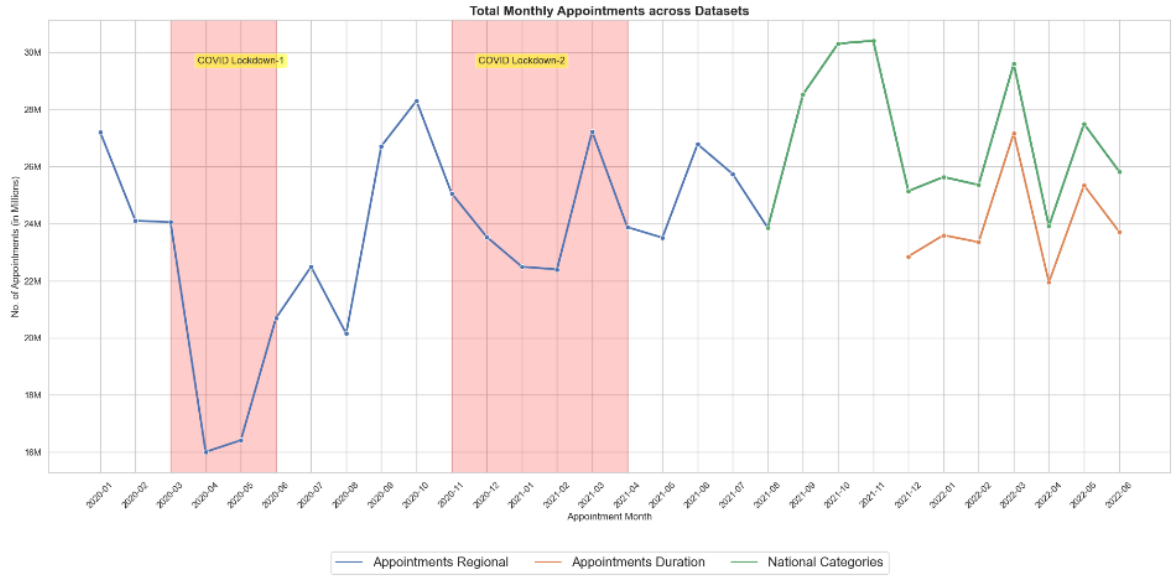
`Style.format()` and `background.gradient()` functions were used to display groupby results in a formatted view that can be easily consumed, analysed and initial insights derived.

```
nc_appts_region.style.format({'count_of_appointments': my_format, '% of Total': '{:.1f}'}).background_gradient(subset=['count_of_appointments','% of Total'], cmap='BuGn')
```

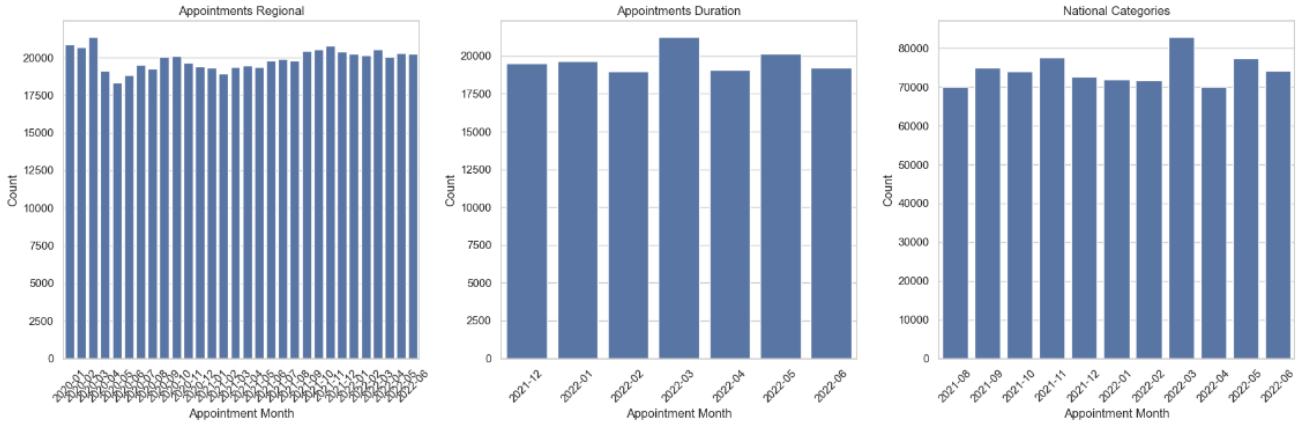
Region Wise Appointments count [NC Dataset] - 11 months

	region_name	count_of_appointments	% of Total
0	Midlands	57,352,329	19.4
1	North East and Yorkshire	47,915,966	16.2
2	South East	45,003,534	15.2
3	London	43,484,439	14.7
4	North West	35,865,459	12.1
5	East of England	34,055,047	11.5
6	South West	32,369,996	10.9

Lineplots [`sns.lineplot()`] were created to showcase monthly appointment trend patterns. COVID lockdown periods on the chart using `plt.axvspan()` and annotations using `plt.annotate()` for clearer context.



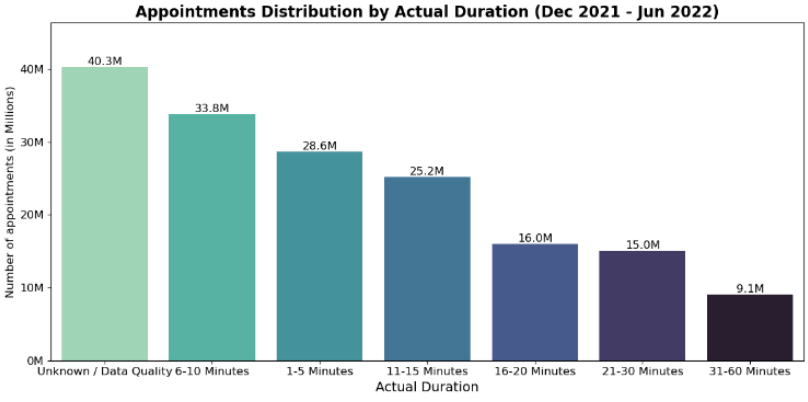
Barplots within 3 subplots were generated to check the record counts for skewness across months



Bar charts were generated using seaborn (`sns.barplot`) and customized using matplotlib functions. Bar labels added using `plt.text()` and appointment counts x-axis formatted using `xaxis.set_major_formatter` and `plt.FuncFormatter()` functions.

Sample Categorical distribution bar chart:

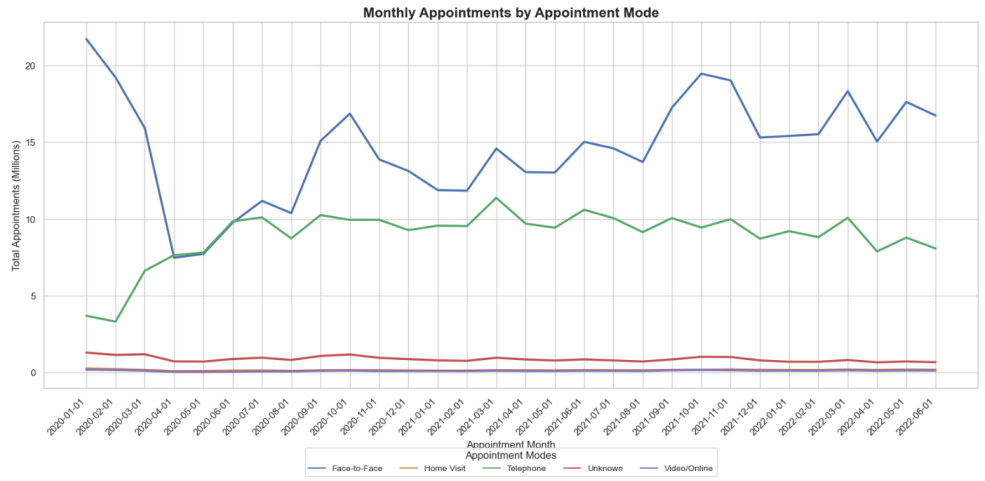
```
# Actual Duration Distribution plots - Actual Duration
plt.figure(figsize=(12, 6))
ax = sns.barplot(x='actual_duration', y='Appointments', data=ad_result_dfs['time_between_book_and_appointment'],palette='mako_r')
plt.title('Appointments Distribution by Actual Duration (Dec 2021 - Jun 2022)', fontsize = 16, fontweight = 'bold')
plt.xlabel('Actual Duration',fontsize = 14)
plt.ylabel('Number of appointments (in Millions)',fontsize = 12)
plt.gca().yaxis.set_major_formatter(plt.FuncFormatter(lambda x, p: f'{x/1e6:.0f}M'))
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)
plt.ylim(0, ad_result_dfs['time_between_book_and_appointment']['Appointments'].max() * 1.15)
for index, value in enumerate(ad_result_dfs['time_between_book_and_appointment']['Appointments']):
    plt.text(index,value, f'{value/1e6:.1f}M', va='bottom',ha='center', fontsize=12)
plt.tight_layout()
plt.savefig('appointments_by_actual_duration.png', dpi=300, bbox_inches='tight')
plt.show()
```



Monthly and Seasonal insights were obtained with line plots where multiple matplotlib functions were utilized to make the visualisations presentable to the business audience.

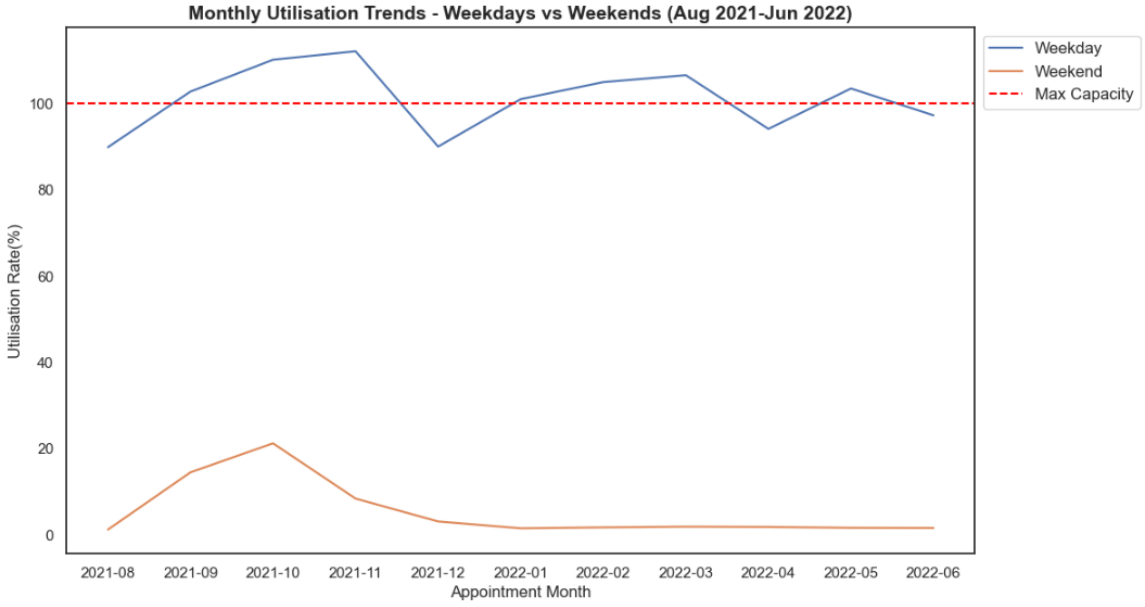
Example:

```
# --- Monthly appointments - Lineplot by Appointment mode ---
plt.figure(figsize=(16, 8))
sns.set_style("whitegrid")
sns.lineplot(data= appt_modes_monthly,x='appointment_month',y='count_of_appointments',hue='appointment_mode',errorbar=None, linewidth = 2.5)
plt.title('Monthly Appointments by Appointment Mode', fontsize=16, fontweight= 'bold')
plt.xlabel('Appointment Month', fontsize=12)
plt.ylabel('Total Appointments (Millions)', fontsize=12)
plt.legend(title = 'Appointment Modes',loc='center', bbox_to_anchor=(0.5,-0.2), ncols = 5, fontsize='small')
plt.xticks(rotation=45, ha='right')
# Format y-axis to show thousands
plt.gca().yaxis.set_major_formatter(plt.FuncFormatter(lambda x, p: format(int(x/1e6), ',')))
plt.tight_layout()
plt.show()
```



Dataframes were concatenated to obtain comparative charts wherever necessary.

```
# Concatenate the dataframes, adding a column to identify weekdays and weekends
combined_df = pd.concat([NC_Mutil_wkdys.assign(day_type='Weekday'), NC_Mutil_wkends.assign(day_type='Weekend')])

#Line chart showing monthly weekdays and weekends utilisation trends
plt.figure(figsize=(12, 7))
sns.lineplot(data=combined_df, x='appointment_month', y='avg_utilisation_rate', hue='day_type')

# Adding horizontal line for max capacity
plt.axhline(100, color='red', linestyle='--', label='Max Capacity')

# Customize the plot
plt.title('Monthly Utilisation Trends - Weekdays vs Weekends (Aug 2021-Jun 2022)', fontsize=14, fontweight='bold')
plt.xticks(rotation=0)
plt.ylabel('Utilisation Rate(%)')
plt.xlabel('Appointment Month')
plt.legend(loc='upper left', bbox_to_anchor=(1, 1), fontsize=12)
plt.show()
```



5. Insights and Recommendations

Key Exploratory Data Insights:

1. **Appointment Mode Distribution**: Face-to-face and telephone appointments dominate (95.3% of all appointments). Since COVID-19, telephone appointments have doubled.
2. **Time Between Booking and Appointment**: 87.7% of all appointments are booked within 0-14 days, which is also when most missed appointments occur.
3. **Regional Disparities**: Midlands handled the highest volume of appointments (57.4 million), while North West managed the fewest. London and Midlands also show higher missed appointment rates, especially for face-to-face.
4. **Service Setting**: General Practice represents 91% of appointments, with seasonal peaks in October-November and March-April.
5. **Impact of COVID-19**: Appointment volumes dropped sharply during lockdowns, but rebounded strongly post-lockdown, indicating postponed appointments being rescheduled.

Key Insights:

1. **Missed Appointments**: Face-to-face appointments account for the highest number of missed appointments (78%), followed by telephone appointments (17%). Most missed appointments occur within 0-14 days of booking.
2. **Appointment Mode Trends**: Telephone appointments have grown significantly since COVID-19, with the lowest missed rates. Face-to-face appointments, though reduced, still exhibit high missed rates.
3. **Regional Variation**: Midlands, London, North West, and North East & Yorkshire contribute to 62% of missed appointments.
4. **Capacity and Utilization**: Weekdays are overutilized (73% of days exceeding capacity), while weekends are underutilized.

Recommendations:

1. **Shorten Booking Intervals**: Increase same-day and next-day appointments to lower missed rates.
2. **Expand Remote Consultations**: Encourage telephone and video consultations to reduce missed face-to-face slots.
3. **Improve Weekend Utilization**: Increase weekend appointments to relieve weekday overuse.
4. **Region-Specific Interventions**: Tailor strategies to regions with higher missed appointments.
5. **Targeted Reminders**: Use automated reminders to reduce DNA for appointments with booking window greater than 8 days.

6. Data Limitations and Future Analysis

- Current analysis is limited by the inability to merge the three datasets at individual appointment level. A unified data model linking appointment-level data across all factors would enable more robust statistical analysis and predictive modelling.
- Population metrics with age, demography, major illnesses, geographical terrain, Healthcare providers (Doctors/Nurses/facilities) at sub-ICB level are vital to get a holistic understanding in terms of service.
- The post-COVID normalization period (approximately 7 months) provides insufficient temporal depth for high-confidence predictive modelling.
- Limited time series data affects the statistical significance of trend analysis and seasonal adjustments
- Granular geographic analysis could be performed on high performing / high risk locations to reveal localized patterns, pain points to be resolved.

References:

1. Location breakdown - Sub ICB Locations to Integrated Care Boards to NHSER (July 2022) Lookup in EN | Open Geography Portal (statistics.gov.uk)
2. ICBs explained - https://www.nhsconfed.org/publications/integrated-care-systems-ics
3. Average cost per appointment - https://www.england.nhs.uk/east-of-england/2023/12/08/quarter-of-a-million-more-seen-by-gps-in-the-east-of-england-during-october-as-costs-of-no-shows-revealed/
4. COVID UK Timeline - https://www.researchgate.net/figure/UK-COVID-19-infection-timeline-The-timelime-summarises-COVID-19-related-events-in-the-UK_fig1_370253143

5. Styling table outputs - https://pbpython.com/styling-pandas.html
6. Seaborn colour palettes - https://seaborn.pydata.org/tutorial/color_palettes.html
7. Breakdown pie charts - https://www.geeksforgeeks.org/how-to-create-a-pie-chart-in-seaborn/
8. Basic Pandas cheatsheet - Pandas_Cheat_Sheet.pdf (datacamp.com)
9. Custom sorting in Dataframe - python - Custom sorting in pandas dataframe - Stack Overflow