Trivago_Case_Study_Task1_Al_Ameen

October 22, 2023

```
[1]: import pandas as pd
     import numpy as np
     import seaborn as sns
     import matplotlib.pyplot as plt
```

```
[ ]: df_task1 = pd.read_excel('/Users/ameen/Downloads/202303_Task1_Sessions.xlsx')
```

```
[ ]: df_task2 = pd.read_excel('/Users/ameen/Downloads/202303_Task2_Actions.xlsx')
```

```
[ ]: df_task1.head()
```

```
[ ]:   ymd,session_id,tracking_id,platform,is_app,is_repeater,traffic_type,country_na
     me,agent_id,clickouts,bookings,session_duration,entry_page,total_ctp,arrival_day
     ,departure_day
     0  20220626,2022062620046057322,FA6JXA8TAJ,UK,0,1…
     1  20220518,2022051821943006017,OX7RLU6KF7,BR,0,0…
     2  20220508,2022050821020053928,0I59VWLQW0,UK,0,0…
     3  20220507,2022050706015039122,JXNHOBQL50,CH,0,0…
     4  20220523,2022052320052048087,W24I0V5Z2L,IT,0,0…
```

```
[ ]: columns = df_task1.columns[0]
```

0.0.1 Data Cleaning Task 1

```
[ ]: split_data = df_task1[columns].str.split(',', expand=True)
```

```
[ ]: split_data.drop([16,17],axis = 1,inplace = True)
```

```
[ ]: split_data.columns = ['ymd', 'session_id', 'tracking_id', 'platform', 'is_app',␣
      ↪'is_repeater', 'traffic_type', 'country_name', 'agent_id', 'clickouts',␣
      ↪'bookings', 'session_duration', 'entry_page', 'total_ctp', 'arrival_day',␣
      ↪'departure_day']
```

```
[ ]: df_task1 = split_data.copy()
```

```
[ ]: df_task1.head()
```

[ ]:

```
        ymd           session_id tracking_id platform is_app is_repeater  \
0  20220626  2022062620046057322  FA6JXA8TAJ       UK      0           1
1  20220518  2022051821943006017  0X7RLU6KF7       BR      0           0
2  20220508  2022050821020053928  0I59VWLQW0       UK      0           0
3  20220507  2022050706015039122  JXNH0BQL50       CH      0           0
4  20220523  2022052320052048087  W24I0V5Z2L       IT      0           0

  traffic_type    country_name agent_id clickouts bookings session_duration  \
0            2  United Kingdom       16         0        0               29
1            2          Brazil        2         3        0             1485
2            2  United Kingdom       20         0        0              143
3            2     Switzerland       28         0        0               69
4            2           Italy       20         6        0              887

  entry_page total_ctp arrival_day departure_day
0       2111         0          \N            \N
1       2100        27    20220530      20220531
2       2100         0          \N            \N
3       2100         0          \N            \N
4       2100       100    20220609      20220610
```

[ ]:

```
# df_task1[df_task1['arrival_day']=='\\N']
```

[ ]:

```
df_task1['date'] = df_task1['ymd'].str[0:4]+'-'+df_task1['ymd'].str[4:
 ↪6]+'-'+df_task1['ymd'].str[6:8]
```

[ ]:

```
## There are some non numeric values in the column clickouts converting them to␣
 ↪0
df_task1['clickouts'] = np.where(df_task1['clickouts'].str.
 ↪isnumeric(),df_task1['clickouts'],'0')
```

[ ]:

```
df_task1['clickouts'] = df_task1['clickouts'].astype(int)
```

[ ]:

```
df_task1['is_repeater'] = df_task1['is_repeater'].astype(int)
df_task1['bookings'] = df_task1['bookings'].astype(int)
df_task1['session_duration'] = df_task1['session_duration'].astype(int)
df_task1['total_ctp'] = df_task1['total_ctp'].astype(int)
```

Aggregating Data at Date Level

[ ]:

```
df_agg1 = df_task1.groupby(['date']).agg({'session_id':'nunique','tracking_id':
 ↪'nunique','is_repeater':'sum','clickouts':'sum','bookings':
 ↪'sum','session_duration':'mean','total_ctp':'sum'}).reset_index()
```

[ ]:

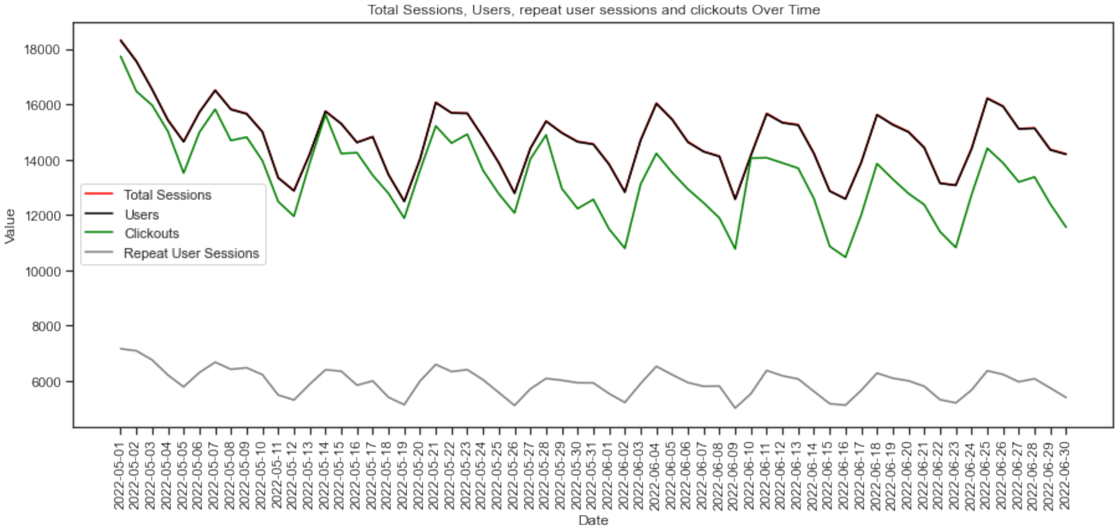
```
df_agg1.rename({'session_id':'total_sessions','tracking_id':
 ↪'users','is_repeater':'repeat_user_sessions','session_duration':
 ↪'average_session_duration'},inplace=True,axis=1)
```

0.1 Task1 - Descriptive Analysis

0.1.1 Plotting Total Sessions, Users, and Clickouts over time

[ ]:

```
sns.set(style="whitegrid")
sns.set_style("ticks")
# Create the line plot for 'total_sessions' and 'users'.
plt.figure(figsize=(15, 6))
ax = sns.lineplot(data=df_agg1, x='date', y='total_sessions', label='Total␣
  ↪Sessions', color='red')
sns.lineplot(data=df_agg1, x='date', y='users', label='Users', color = 'black')
sns.lineplot(data=df_agg1, x='date', y='clickouts', label='Clickouts', color =␣
  ↪'green')
sns.lineplot(data=df_agg1, x='date', y='repeat_user_sessions', label='Repeat␣
  ↪User Sessions', color = 'grey')

plt.title('Total Sessions, Users, repeat user sessions and clickouts Over Time')
ax.set_xlabel('Date')
ax.set_ylabel('Value')

# Set the x-axis tick positions and labels for all dates.
x_ticks = range(len(df_agg1))
x_labels = df_agg1['date']  # Format the date labels as desired.

ax.set_xticks(x_ticks)
ax.set_xticklabels(x_labels, rotation=90)

# Add a legend for the lines.
plt.legend(loc='best')

plt.show()
```

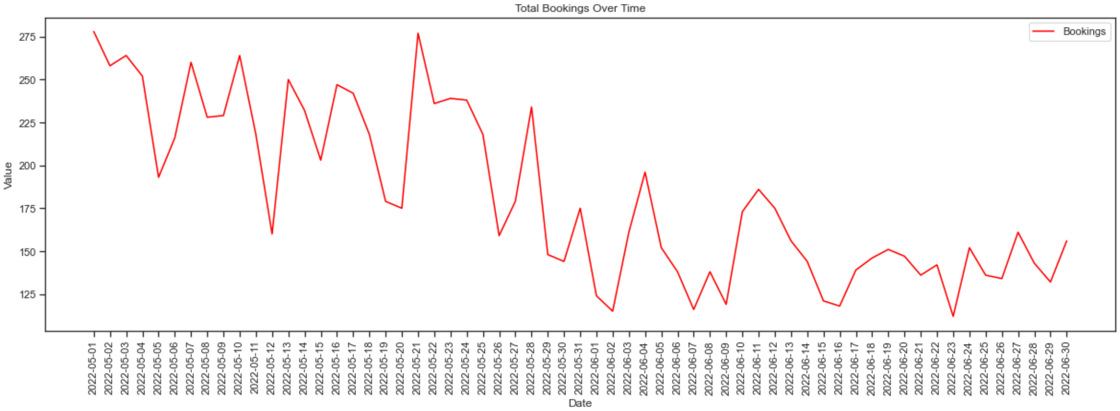


The presented data illustrates that the overall count of visitors to Trivago, the total number of sessions, and sessions initiated by repeat users have displayed a degree of steadiness when comparing May'22 to June'22. In contrast, the number of Clickouts is showing a decreasing trend over time. Notably, the graph portrays a distinctive zigzag pattern, hinting at weekly peaks in user visits followed by subsequent declines.

0.1.2 Plotting bookings over time

[ ]:

```
sns.set(style="whitegrid")
sns.set_style("ticks")
# Create the line plot for 'total_sessions' and 'users'.
plt.figure(figsize=(20, 6))
ax = sns.lineplot(data=df_agg1, x='date', y='bookings', label='Bookings',␣
 ↪color='red')
# sns.lineplot(data=df_agg1, x='date', y='users', label='Users', color =␣
 ↪'black')

plt.title('Total Bookings Over Time')
ax.set_xlabel('Date')
ax.set_ylabel('Value')

# Set the x-axis tick positions and labels for all dates.
x_ticks = range(len(df_agg1))
x_labels = df_agg1['date']  # Format the date labels as desired.

ax.set_xticks(x_ticks)
ax.set_xticklabels(x_labels, rotation=90)

# Add a legend for the lines.
plt.legend(loc='best')

plt.show()
```



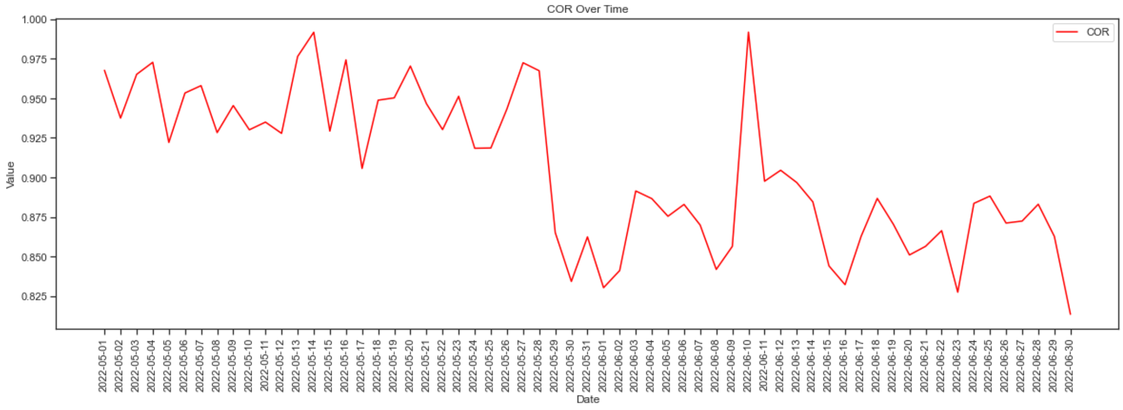
The line plot above suggests that the total number of bookings has been on a declining trend from May'22 to June'22.

```
[ ]: ## defining the Clickouts Ratio (COR)
```

```
[ ]: df_agg1['cor'] = df_agg1['clickouts']/df_agg1['total_sessions']
```

0.1.3 Plotting COR

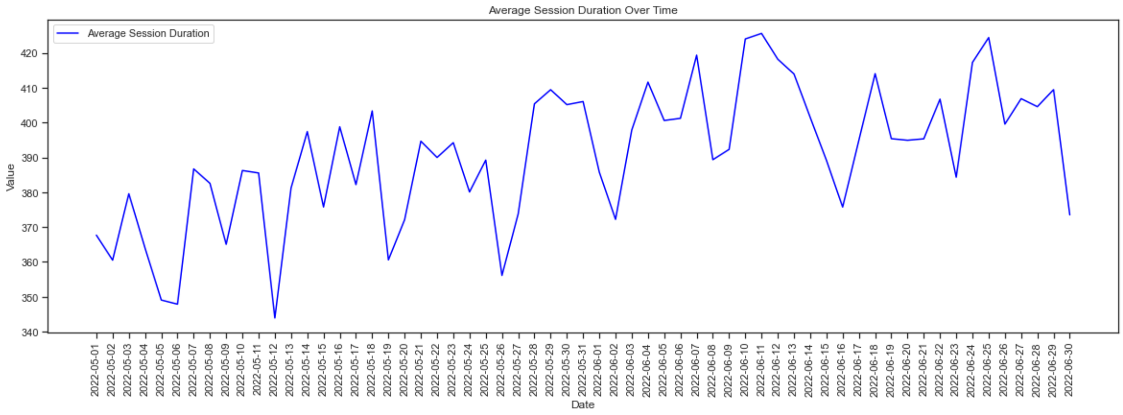
```
[ ]: sns.set(style="whitegrid")
     sns.set_style("ticks")
     # Create the line plot for 'total_sessions' and 'users'.
     plt.figure(figsize=(20, 6))
     ax = sns.lineplot(data=df_agg1, x='date', y='cor', label='COR', color='red')

     plt.title('COR Over Time')
     ax.set_xlabel('Date')
     ax.set_ylabel('Value')

     # Set the x-axis tick positions and labels for all dates.
     x_ticks = range(len(df_agg1))
     x_labels = df_agg1['date']  # Format the date labels as desired.

     ax.set_xticks(x_ticks)
     ax.set_xticklabels(x_labels, rotation=90)

     # Add a legend for the lines.
     plt.legend(loc='best')

     plt.show()
```



The line plot above suggests that the COR has been on a declining trend from May'22 to June'22.

0.1.4 Plotting Average Session Duration over time

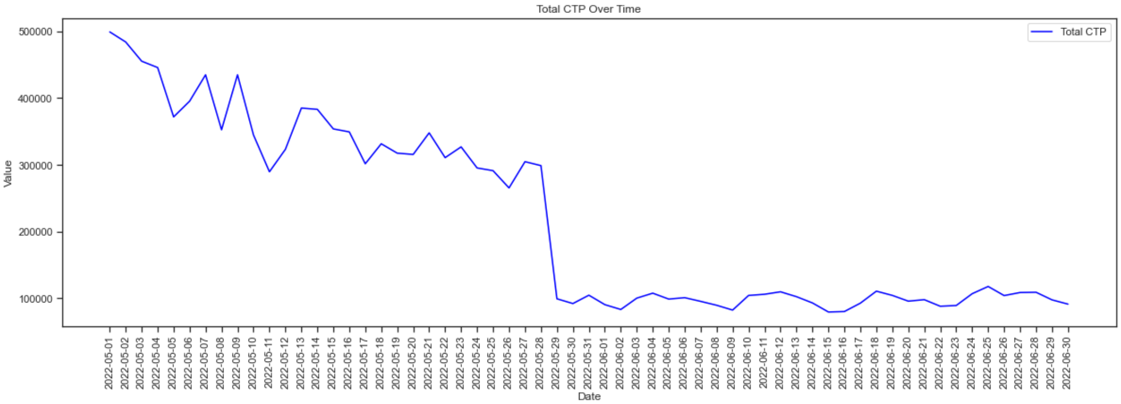
[ ]:
```
sns.set(style="whitegrid")
sns.set_style("ticks")
# Create the line plot for 'total_sessions' and 'users'.
plt.figure(figsize=(20, 6))
ax = sns.lineplot(data=df_agg1, x='date', y='average_session_duration',␣
 ↪label='Average Session Duration', color='blue')

plt.title('Average Session Duration Over Time')
ax.set_xlabel('Date')
ax.set_ylabel('Value')

# Set the x-axis tick positions and labels for all dates.
x_ticks = range(len(df_agg1))
x_labels = df_agg1['date']  # Format the date labels as desired.

ax.set_xticks(x_ticks)
ax.set_xticklabels(x_labels, rotation=90)

# Add a legend for the lines.
plt.legend(loc='best')

plt.show()
```



In contrast to the declining trend in total bookings, the average session duration has seen an increasing trend from May 2022 to June 2022.

0.1.5 Plotting Total Ctp over time

```
[ ]: sns.set(style="whitegrid")
     sns.set_style("ticks")
     # Create the line plot for 'total_sessions' and 'users'.
     plt.figure(figsize=(20, 6))
     ax = sns.lineplot(data=df_agg1, x='date', y='total_ctp', label='Total CTP',␣
      ↪color='blue')

     plt.title('Total CTP Over Time')
     ax.set_xlabel('Date')
     ax.set_ylabel('Value')

     # Set the x-axis tick positions and labels for all dates.
     x_ticks = range(len(df_agg1))
     x_labels = df_agg1['date']  # Format the date labels as desired.

     ax.set_xticks(x_ticks)
     ax.set_xticklabels(x_labels, rotation=90)

     # Add a legend for the lines.
     plt.legend(loc='best')

     plt.show()
```



The Total CTP has notably decreased starting from May 29, 2023, indicating a decline in user intent to participate in transactions during June 2023 as compared to May 2023.

Aggregating Data at Date X Country level

```
[ ]: df_agg2 = df_task1.groupby(['date','country_name']).agg({'session_id':
      ↪'nunique','tracking_id':'nunique','is_repeater':'sum','clickouts':
      ↪'sum','bookings':'sum','session_duration':'mean','total_ctp':'sum'}).
      ↪reset_index()
```

[ ]:

```
df_agg2.rename({'country_name':'country','session_id':
 ↪'total_sessions','tracking_id':'users','is_repeater':
 ↪'repeat_user_sessions','session_duration':
 ↪'average_session_duration'},axis=1,inplace = True)
```

[ ]:

```
df_agg2['date'] = pd.to_datetime(df_agg2['date'])
df_agg2['month'] = df_agg2['date'].dt.strftime('%B')
```

[ ]:

```
df_temp = df_agg2.groupby('country')['total_sessions'].sum().reset_index()
```

[ ]:

```
df_temp.sort_values('total_sessions',ascending = False, inplace=True)
```

[ ]:

```
## 58% of the total sessions are originated from 10 countries
df_temp.sort_values('total_sessions',ascending = False).head(10).total_sessions.
 ↪sum()/df_temp.total_sessions.sum()
```

[ ]:

```
0.5779577777777778
```

[ ]:

```
top_10_countries = list(df_temp.head(10)['country'].unique())
```

[ ]:

```
#changing the country names of countries which are not in top 10 countries list␣
 ↪basis total sessions
df_agg2['country_tag'] = np.where(df_agg2['country'].
 ↪isin(top_10_countries),df_agg2['country'],'others')
```

[ ]:

```
df_agg3 = df_agg2.groupby(['month','country_tag']).agg({'total_sessions':
 ↪'sum','users':'sum','repeat_user_sessions':'sum','clickouts':
 ↪'sum','bookings':'sum','total_ctp':'sum'}).reset_index()
```

[ ]:

```
top_10_countries
```

[ ]:

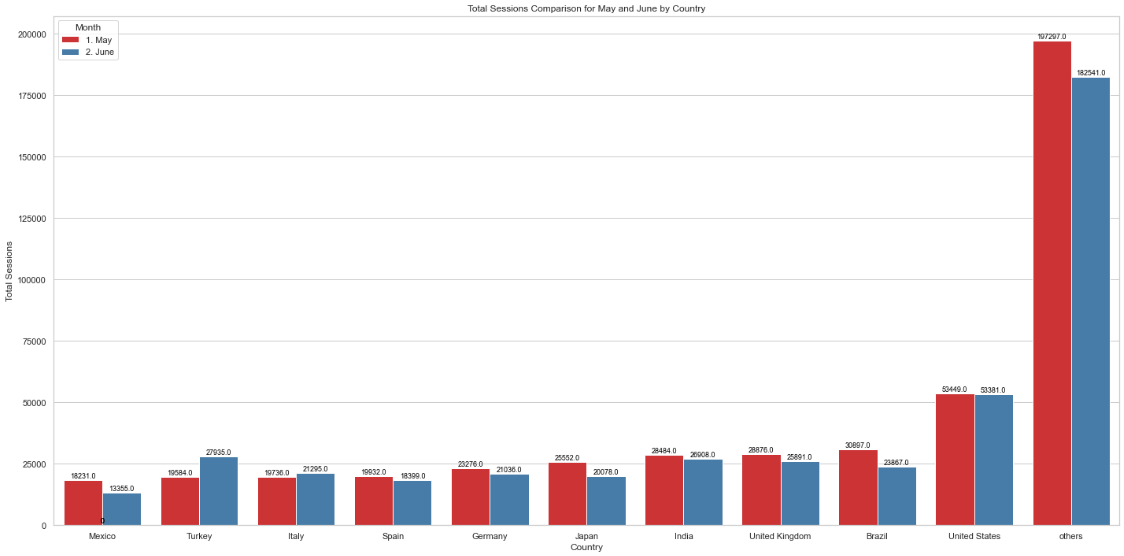
```
['United States',
 'India',
 'United Kingdom',
 'Brazil',
 'Turkey',
 'Japan',
 'Germany',
 'Italy',
 'Spain',
 'Mexico']
```

0.1.6 Total Sessions Comparison for May and June by Country

[ ]:

```
import seaborn as sns
import matplotlib.pyplot as plt

# Set the style of the plot (optional but can improve aesthetics)
```

```python
sns.set(style="whitegrid")

# Create a bar chart comparing May and June data for every country
df_agg3['month'] = df_agg3['month'].replace({'May': '1. May', 'June': '2. June'})
df_agg3.sort_values(['month','total_sessions'],inplace=True)
plt.figure(figsize=(20, 10))
ax = sns.barplot(x='country_tag', y='total_sessions', hue='month', data=df_agg3, palette='Set1')

# Set the title and labels
plt.title('Total Sessions Comparison for May and June by Country')
ax.set_xlabel('Country')
ax.set_ylabel('Total Sessions')

# Customize the legend and change the order of the legend labels
handles, labels = ax.get_legend_handles_labels()
ax.legend(handles=handles, labels=labels, title='Month', loc='best')

# Annotate each bar with its corresponding y-value
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()), ha='center', va='center', fontsize=9, color='black', xytext=(0, 5), textcoords='offset points')

plt.tight_layout()
plt.show()
```



```
[ ]: may_total_sessions = df_agg3[df_agg3['month'] == '1. May'].
      ↪groupby('country_tag')['total_sessions'].sum().reset_index()
     may_total_sessions.rename({'total_sessions':'May total sessions'},inplace=True,␣
      ↪axis = 1)
     june_total_sessions = df_agg3[df_agg3['month'] == '2. June'].
      ↪groupby('country_tag')['total_sessions'].sum().reset_index()
     june_total_sessions.rename({'total_sessions':'June total␣
      ↪sessions'},inplace=True, axis = 1)
     df_change = pd.merge(may_total_sessions,june_total_sessions, on ='country_tag',␣
      ↪how='inner')
     df_change['Perc. Change in Sessions'] = (df_change['June total sessions'] -␣
      ↪df_change['May total sessions'])*100/df_change['May total sessions']
     # df_change.sort_values('May total sessions',ascending = False).
      ↪reset_index(drop=True)
     # Calculate the overall total for May and June sessions and percentage change
     overall_total_may = df_change['May total sessions'].sum()
     overall_total_june = df_change['June total sessions'].sum()
     overall_percentage_change = ((overall_total_june - overall_total_may) /␣
      ↪overall_total_may) * 100

     # Add a new row to the DataFrame
     df_change.loc['Total'] = ['Overall', overall_total_may, overall_total_june,␣
      ↪overall_percentage_change]

     # Format the 'Perc. Change in Sessions' column with a percentage symbol and␣
      ↪rounding
     df_change['Perc. Change in Sessions'] = df_change['Perc. Change in Sessions'].
      ↪apply(lambda x: f'{x:.1f}%')
     df_change.sort_values('May total sessions',ascending = False,inplace=True)

     # Reset the index to have a proper DataFrame
     df_change = df_change.reset_index(drop=True)
     df_change
```

```
[ ]:        country_tag  May total sessions  June total sessions  \
     0          Overall              465314               434686
     1           others              197297               182541
     2    United States               53449                53381
     3           Brazil               30897                23867
     4   United Kingdom               28876                25891
     5            India               28484                26908
     6            Japan               25552                20078
     7          Germany               23276                21036
     8            Spain               19932                18399
     9            Italy               19736                21295
     10          Turkey               19584                27935
```

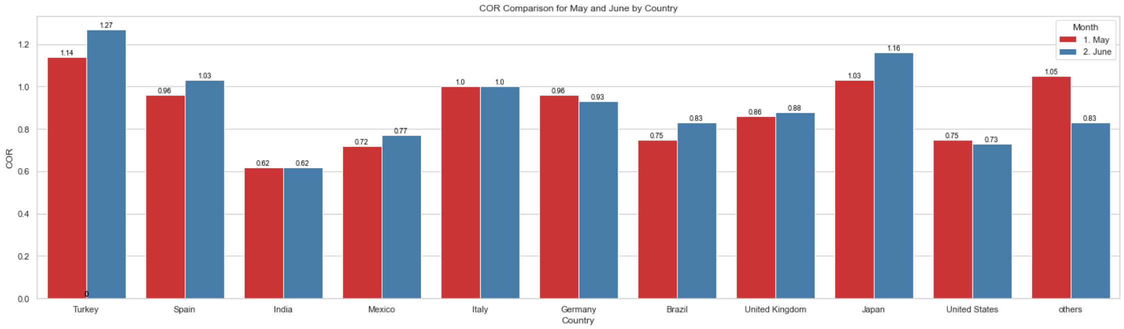
```
11           Mexico                18231                13355

   Perc. Change in Sessions
0                     -6.6%
1                     -7.5%
2                     -0.1%
3                    -22.8%
4                    -10.3%
5                     -5.5%
6                    -21.4%
7                     -9.6%
8                     -7.7%
9                      7.9%
10                    42.6%
11                   -26.7%
```

Total sessions worldwide experienced a 6.6% decline in June when compared to May

0.1.7 COR Comparison for May and June by Country

[ ]:
```
df_agg3['cor'] = df_agg3['clickouts']/df_agg3['total_sessions']
df_agg3['cor'] = df_agg3['cor'].round(2)
```

[ ]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Set the style of the plot (optional but can improve aesthetics)
sns.set(style="whitegrid")
df_agg3.sort_values(['month','bookings'],inplace=True)


# Create a bar chart comparing May and June data for every country
plt.figure(figsize=(20, 6))
ax = sns.barplot(x='country_tag', y='cor', hue='month', data=df_agg3,␣
 ↪palette='Set1')

# Set the title and labels
plt.title('COR Comparison for May and June by Country')
ax.set_xlabel('Country')
ax.set_ylabel('COR')

# Customize the legend and change the order of the legend labels
handles, labels = ax.get_legend_handles_labels()
ax.legend(handles=handles, labels=labels, title='Month', loc='best')

# Annotate each bar with its corresponding y-value
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.
 ↪get_height()), ha='center', va='center', fontsize=9, color='black',␣
 ↪xytext=(0, 5), textcoords='offset points')

plt.tight_layout()
plt.show()
```



```
[ ]: ##Calculating the Overall COR change
     may_cor = df_agg3[df_agg3['month']=='1. May']['clickouts'].sum()/
      ↪df_agg3[df_agg3['month']=='1. May']['total_sessions'].sum()
     june_cor = df_agg3[df_agg3['month']=='2. June']['clickouts'].sum()/
      ↪df_agg3[df_agg3['month']=='2. June']['total_sessions'].sum()
```

```
[ ]: print("May COR  : {}".format(round(may_cor,2)))
     print("June COR : {}".format(round(june_cor,2)))
```

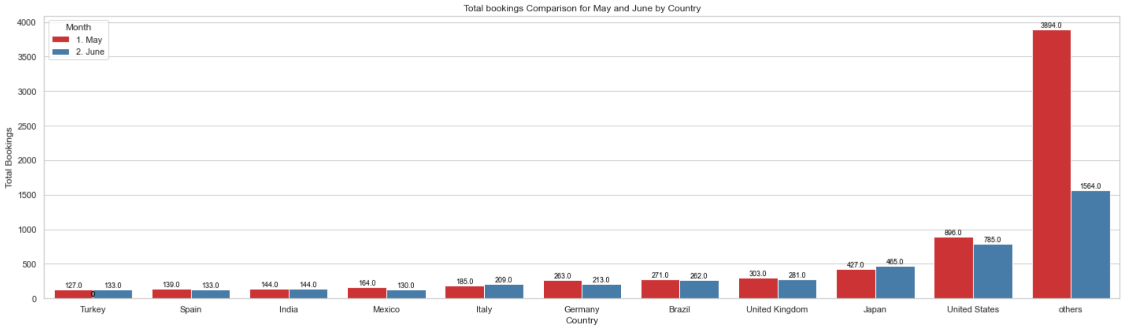
```
May COR  : 0.94
June COR : 0.87
```

The Clickout Ratio (COR) has exhibited stability or growth in most of the top countries, excluding the United States and Germany, where it witnessed a slight decline in June compared to May. Conversely, for countries outside the top 10 in total session numbers, the COR has experienced a substantial decrease, dropping from 1.05 to 0.83. Overall, the COR ratio decreased from 0.94 in May to 0.87.

0.1.8 Total Bookings Comparison for May and June by Country

```
[ ]: import seaborn as sns
     import matplotlib.pyplot as plt

     # Set the style of the plot (optional but can improve aesthetics)
     sns.set(style="whitegrid")
     df_agg3.sort_values(['month','bookings'],inplace=True)
```

```
# Create a bar chart comparing May and June data for every country
plt.figure(figsize=(20, 6))
ax = sns.barplot(x='country_tag', y='bookings', hue='month', data=df_agg3,␣
↪palette='Set1')

# Set the title and labels
plt.title('Total bookings Comparison for May and June by Country')
ax.set_xlabel('Country')
ax.set_ylabel('Total Bookings')

# Customize the legend and change the order of the legend labels
handles, labels = ax.get_legend_handles_labels()
# Change the order of the labels
ax.legend(handles=handles, labels=labels, title='Month', loc='best')

# Annotate each bar with its corresponding y-value
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.
↪get_height()), ha='center', va='center', fontsize=9, color='black',␣
↪xytext=(0, 5), textcoords='offset points')

plt.tight_layout()
plt.show()
```



[ ]:
```
# Calculate May total bookings and rename the column
may_total_bookings = df_agg3[df_agg3['month'] == '1. May'].
↪groupby('country_tag')['bookings'].sum().reset_index()
may_total_bookings.rename(columns={'bookings': 'May total bookings'},␣
↪inplace=True)

# Calculate June total bookings and rename the column
june_total_bookings = df_agg3[df_agg3['month'] == '2. June'].
↪groupby('country_tag')['bookings'].sum().reset_index()
```

```
june_total_bookings.rename(columns={'bookings': 'June total bookings'},␣
 ↪inplace=True)

# Merge the DataFrames
df_change_bookings = pd.merge(may_total_bookings, june_total_bookings,␣
 ↪on='country_tag', how='inner')

# Calculate the percentage change in bookings
df_change_bookings['Perc. Change in Bookings'] = (df_change_bookings['June␣
 ↪total bookings'] - df_change_bookings['May total bookings']) * 100 /␣
 ↪df_change_bookings['May total bookings']

# Calculate the overall total for May and June bookings and percentage change
overall_total_may_bookings = df_change_bookings['May total bookings'].sum()
overall_total_june_bookings = df_change_bookings['June total bookings'].sum()
overall_percentage_change_bookings = ((overall_total_june_bookings -␣
 ↪overall_total_may_bookings) / overall_total_may_bookings) * 100

# Add a new row for overall bookings
df_change_bookings = df_change_bookings.append({'country_tag': 'Overall', 'May␣
 ↪total bookings': overall_total_may_bookings, 'June total bookings':␣
 ↪overall_total_june_bookings, 'Perc. Change in Bookings':␣
 ↪overall_percentage_change_bookings}, ignore_index=True)

# Format the 'Perc. Change in Bookings' column with a percentage symbol and␣
 ↪rounding
df_change_bookings['Perc. Change in Bookings'] = df_change_bookings['Perc.␣
 ↪Change in Bookings'].apply(lambda x: f'{x:.1f}%')

# Sort the DataFrame by 'May total bookings' in descending order
df_change_bookings.sort_values('May total bookings', ascending=False,␣
 ↪inplace=True)

# Reset the index for a proper DataFrame
df_change_bookings.reset_index(drop=True, inplace=True)

df_change_bookings
```

```
[ ]:        country_tag  May total bookings  June total bookings  \
     0          Overall                6813                 4319
     1           others                3894                 1564
     2    United States                 896                  785
     3            Japan                 427                  465
     4   United Kingdom                 303                  281
     5           Brazil                 271                  262
     6          Germany                 263                  213
```

```
7            Italy                 185                  209
8           Mexico                 164                  130
9            India                 144                  144
10           Spain                 139                  133
11          Turkey                 127                  133

   Perc. Change in Bookings
0                    -36.6%
1                    -59.8%
2                    -12.4%
3                      8.9%
4                     -7.3%
5                     -3.3%
6                    -19.0%
7                     13.0%
8                    -20.7%
9                      0.0%
10                    -4.3%
11                     4.7%
```

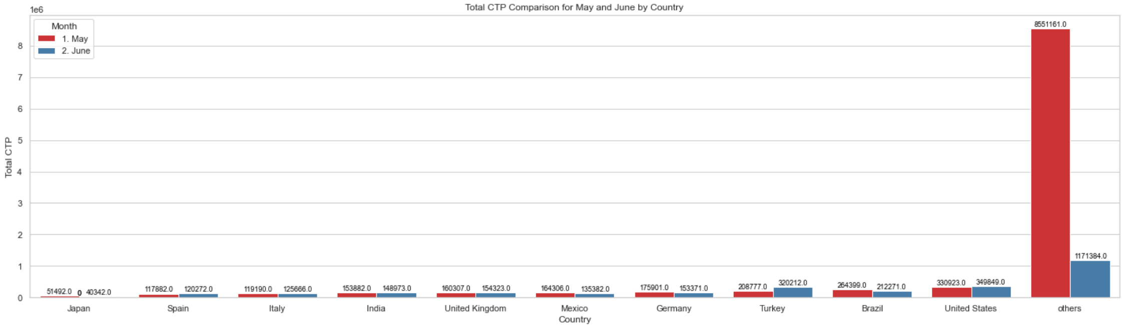
Total bookings in June demonstrate a notable decrease, with a reduction of approximately 37% when compared to May. The decrease is particularly prominent among countries not ranked in the top 10 by session count, while the top 10 countries also experience a decline. This significant drop is directly linked to the decrease in Clickout Ratio (COR) for non-top 10 countries.

0.1.9 Total CTP Comparison for May and June by Country

[ ]:

```
import seaborn as sns
import matplotlib.pyplot as plt

# Set the style of the plot (optional but can improve aesthetics)
sns.set(style="whitegrid")
df_agg3.sort_values(['month','total_ctp'],inplace=True)


# Create a bar chart comparing May and June data for every country
plt.figure(figsize=(20, 6))
ax = sns.barplot(x='country_tag', y='total_ctp', hue='month', data=df_agg3,␣
 ↪palette='Set1')

# Set the title and labels
plt.title('Total CTP Comparison for May and June by Country')
ax.set_xlabel('Country')
ax.set_ylabel('Total CTP')

# Customize the legend and change the order of the legend labels
handles, labels = ax.get_legend_handles_labels()
```

```
# Change the order of the labels
ax.legend(handles=handles, labels=labels, title='Month', loc='best')

# Annotate each bar with its corresponding y-value
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.
 ↪get_height()), ha='center', va='center', fontsize=9, color='black',␣
 ↪xytext=(0, 5), textcoords='offset points')

plt.tight_layout()
plt.show()
```



```
[ ]: # Calculate May total CTP and rename the column
may_total_ctp = df_agg3[df_agg3['month'] == '1. May'].
 ↪groupby('country_tag')['total_ctp'].sum().reset_index()
may_total_ctp.rename(columns={'total_ctp': 'May total CTP'}, inplace=True)

# Calculate June total CTP and rename the column
june_total_ctp = df_agg3[df_agg3['month'] == '2. June'].
 ↪groupby('country_tag')['total_ctp'].sum().reset_index()
june_total_ctp.rename(columns={'total_ctp': 'June total CTP'}, inplace=True)

# Merge the DataFrames
df_change_ctp = pd.merge(may_total_ctp, june_total_ctp, on='country_tag',␣
 ↪how='inner')

# Calculate the percentage change in CTP
df_change_ctp['Perc. Change in CTP'] = (df_change_ctp['June total CTP'] -␣
 ↪df_change_ctp['May total CTP']) * 100 / df_change_ctp['May total CTP']

# Calculate the overall total for May and June CTP and percentage change
overall_total_may_ctp = df_change_ctp['May total CTP'].sum()
overall_total_june_ctp = df_change_ctp['June total CTP'].sum()
```

```
overall_percentage_change_ctp = ((overall_total_june_ctp -␣
 ↪overall_total_may_ctp) / overall_total_may_ctp) * 100

# Add a new row for overall CTP
df_change_ctp = df_change_ctp.append({'country_tag': 'Overall', 'May total CTP':
 ↪ overall_total_may_ctp, 'June total CTP': overall_total_june_ctp, 'Perc.␣
 ↪Change in CTP': overall_percentage_change_ctp}, ignore_index=True)

# Format the 'Perc. Change in CTP' column with a percentage symbol and rounding
df_change_ctp['Perc. Change in CTP'] = df_change_ctp['Perc. Change in CTP'].
 ↪apply(lambda x: f'{x:.1f}%')

# Sort the DataFrame by 'May total CTP' in descending order
df_change_ctp.sort_values('May total CTP', ascending=False, inplace=True)

# Reset the index for a proper DataFrame
df_change_ctp.reset_index(drop=True, inplace=True)

df_change_ctp
```

[ ]:

```
       country_tag  May total CTP  June total CTP Perc. Change in CTP
0          Overall       10298220         2932045              -71.5%
1           others        8551161         1171384              -86.3%
2    United States         330923          349849                5.7%
3           Brazil         264399          212271              -19.7%
4           Turkey         208777          320212               53.4%
5          Germany         175901          153371              -12.8%
6           Mexico         164306          135382              -17.6%
7   United Kingdom         160307          154323               -3.7%
8            India         153882          148973               -3.2%
9            Italy         119190          125666                5.4%
10           Spain         117882          120272                2.0%
11           Japan          51492           40342              -21.7%
```

The total content items(Total CTP) viewed has dropped by a whooping 71.5% in June compared to May. Though for most countries in the top 10 list by total sessions have this metrics dropped, the significant drop happened in countries where this ratio

[ ]:

```
df_agg2
```

[ ]:

```
            date        country  total_sessions  users  repeat_user_sessions  \
0     2022-05-01  Aland Islands               1      1                     0
1     2022-05-01        Albania               2      2                     1
2     2022-05-01        Algeria              16     16                     6
3     2022-05-01        Andorra               2      2                     0
4     2022-05-01         Angola               1      1                     1
…            …              …             …    …                   …
```

```
8434 2022-06-30      Venezuela             15     15                     2
8435 2022-06-30        Vietnam             82     82                    14
8436 2022-06-30         Zambia              1      1                     0
8437 2022-06-30       Zimbabwe              1      1                     0
8438 2022-06-30             \N              7      7                     2

      clickouts  bookings  average_session_duration  total_ctp month  \
0             1         0                515.000000          0   May
1             0         0                 15.000000          0   May
2             5         0                152.125000        130   May
3             0         0                  3.000000          0   May
4             0         0                 53.000000          0   May
…           …       …                       …        …    …
8434         12         1                312.133333          3  June
8435         29         0                140.768293        121  June
8436          0         0                176.000000          1  June
8437          2         0                614.000000         43  June
8438          4         0                170.571429          3  June

     country_tag
0         others
1         others
2         others
3         others
4         others
…            …
8434      others
8435      others
8436      others
8437      others
8438      others

[8439 rows x 11 columns]
```

[ ]:

```
df_agg4 = df_agg2.groupby(['month','country']).agg({'total_sessions':
 ↪'sum','users':'sum','repeat_user_sessions':'sum','clickouts':
 ↪'sum','bookings':'sum','total_ctp':'sum'}).reset_index()
```

[ ]:

```
# Calculate May total CTP and rename the column
may_total_ctp = df_agg4[df_agg4['month'] == 'May'].
 ↪groupby('country')['total_ctp'].sum().reset_index()
may_total_ctp.rename(columns={'total_ctp': 'May total CTP'}, inplace=True)

# Calculate June total CTP and rename the column
june_total_ctp = df_agg4[df_agg4['month'] == 'June'].
 ↪groupby('country')['total_ctp'].sum().reset_index()
june_total_ctp.rename(columns={'total_ctp': 'June total CTP'}, inplace=True)
```

```
# Merge the DataFrames
df_change_ctp = pd.merge(may_total_ctp, june_total_ctp, on='country',␣
 ↪how='outer')
df_change_ctp['May total CTP'] = df_change_ctp['May total CTP'].fillna(0)
df_change_ctp['June total CTP'] = df_change_ctp['June total CTP'].fillna(0)
# Calculate the percentage change in CTP
df_change_ctp['Perc. Change in CTP'] = (df_change_ctp['June total CTP'] -␣
 ↪df_change_ctp['May total CTP']) * 100 / df_change_ctp['May total CTP']

# Calculate the overall total for May and June CTP and percentage change
overall_total_may_ctp = df_change_ctp['May total CTP'].sum()
overall_total_june_ctp = df_change_ctp['June total CTP'].sum()
overall_percentage_change_ctp = ((overall_total_june_ctp -␣
 ↪overall_total_may_ctp) / overall_total_may_ctp) * 100

# Add a new row for overall CTP
df_change_ctp = df_change_ctp.append({'country': 'Overall', 'May total CTP':␣
 ↪overall_total_may_ctp, 'June total CTP': overall_total_june_ctp, 'Perc.␣
 ↪Change in CTP': overall_percentage_change_ctp}, ignore_index=True)

# Format the 'Perc. Change in CTP' column with a percentage symbol and rounding
df_change_ctp['Perc. Change in CTP'] = df_change_ctp['Perc. Change in CTP'].
 ↪apply(lambda x: f'{x:.1f}%')

# Sort the DataFrame by 'May total CTP' in descending order
df_change_ctp.sort_values('May total CTP', ascending=False, inplace=True)

# Reset the index for a proper DataFrame
df_change_ctp.reset_index(drop=True, inplace=True)

df_change_ctp.head(25)
```

```
[ ]:            country  May total CTP  June total CTP Perc. Change in CTP
     0           Overall     10298220.0       2932045.0              -71.5%
     1             Korea      7003146.0             0.0             -100.0%
     2     United States       330923.0        349849.0                5.7%
     3            Brazil       264399.0        212271.0              -19.7%
     4            Turkey       208777.0        320212.0               53.4%
     5           Moldova       181904.0             0.0             -100.0%
     6           Germany       175901.0        153371.0              -12.8%
     7              Iran       165618.0          1498.0              -99.1%
     8            Mexico       164306.0        135382.0              -17.6%
     9    United Kingdom       160307.0        154323.0               -3.7%
     10            India       153882.0        148973.0               -3.2%
     11            Italy       119190.0        125666.0                5.4%
     12            Spain       117882.0        120272.0                2.0%
```

```
13            Australia         96219.0         90729.0                 -5.7%
14               Canada         70238.0         71520.0                  1.8%
15               France         64660.0         65739.0                  1.7%
16            Argentina         62467.0         55023.0                -11.9%
17               Greece         54433.0         59271.0                  8.9%
18                Japan         51492.0         40342.0                -21.7%
19          Netherlands         47209.0         43544.0                 -7.8%
20             Portugal         44103.0         41713.0                 -5.4%
21             Malaysia         42219.0         56860.0                 34.7%
22   Russian Federation         40365.0             0.0               -100.0%
23          Switzerland         37308.0         32577.0                -12.7%
24             Tanzania         35916.0           134.0                -99.6%
```

```
[ ]: outlier_countries = ['Korea','Moldova','Iran','Russian Federation','Tanzania']
     df_change_ctp[df_change_ctp['country'].isin(outlier_countries)]['May total␣
      ↪CTP'].sum()/df_change_ctp[df_change_ctp['country']=='Overall']['May total␣
      ↪CTP'].sum()
```

```
[ ]: 0.7211876421362138
```

The table demonstrates a significant decline of almost 100% in the Total Content Page Items Viewed (CTP) from May to June for countries like Korea, Moldova, Iran, the Russian Federation, and Tanzania. Collectively, these countries contributed to 72% of the overall Total CTP.

```
[ ]: df_agg4[df_agg4['country'].isin(outlier_countries)]
```

```
[ ]:     month             country  total_sessions  users  repeat_user_sessions  \
     90   June                Iran             106    106                    28
     192  June            Tanzania              37     37                     8
     315   May                Iran              89     89                    27
     328   May               Korea            3312   3304                  1142
     354   May             Moldova              86     86                    23
     391   May  Russian Federation            8004   7996                  2202
     427   May            Tanzania              20     20                     4

          clickouts  bookings  total_ctp
     90         103         0       1498
     192         20         2        134
     315       1018        62     165618
     328      56058      2118    7003146
     354       1100        30     181904
     391       7569        12      40365
     427        258        10      35916
```

```
[ ]: Overall_bookings_may = df_agg4[df_agg4['month']=='May']['bookings'].sum()
     korea_bookings_may = 2118 #Obtained from amove table
```

```
#Calculating the percentage of bookings contributed by Korea relative to the␣
 ↪overall total.
Korea_bookings_perc = korea_bookings_may/Overall_bookings_may
print(Korea_bookings_perc*100)
```

```
31.08762659621312
```

```
[ ]: Overall_clickouts_may = df_agg4[df_agg4['month']=='May']['clickouts'].sum()
     korea_clickouts_may = 56058 #Obtained from amove table
     # Calculating the percentage of clickouts contributed by Korea relative to the␣
      ↪overall total.
     Korea_clickouts_perc = korea_clickouts_may/Overall_clickouts_may
     print(Korea_clickouts_perc*100)
```

```
12.827359908836916
```

```
[ ]: Overall_sessions_may = df_agg4[df_agg4['month']=='May']['total_sessions'].sum()
     korea_sessions_may = 3312 #Obtained from amove table
     # Calculating the percentage of sessions contributed by Korea relative to the␣
      ↪overall total.
     korea_sessions_perc = korea_sessions_may/Overall_sessions_may
     print(korea_sessions_perc*100)
```

```
0.7117774234173053
```

```
[ ]: Overall_ctp_may = df_agg4[df_agg4['month']=='May']['total_ctp'].sum()
     korea_total_ctp_may = 7003146 #Obtained from amove table

     #Calculating the percentage of Total CTP contributed by Korea relative to the␣
      ↪overall total.
     Korea_total_ctp_perc = korea_total_ctp_may/Overall_ctp_may
     print(Korea_total_ctp_perc*100)
```

```
68.00346079225342
```

The decline in Total Bookings, COR, and Total CTP can be linked to the absence of traffic on the Trivago site from Korea in June 2022. In May, Korea contributed just **0.71%** to the total number of sessions, yet it played a significant role, accounting for **68%** of the overall Total CTP, **31%** of the overall bookings and **13%** of the overall clickouts.

0.1.10 Summary of the Descriptive Analysis

- Plotted the metrics for total sessions, users, Clickout Ratio (COR), Bookings, and Total CTP with the date on the X axis. The visual analysis revealed that total sessions and user counts remained relatively stable from May to June, while there was a noticeable decline in COR, Total CTP, and Bookings over the same period.
- Upon a more detailed examination at the country level, it was observed that the top 10 countries, which contribute the most to total session volume, did not significantly influence

the decline in COR, Total CTP, and Bookings. For these top 10 countries, the respective metrics either remained constant or exhibited slight changes from May to June.

- Further investigation into countries outside the top 10 list revealed that there was either no traffic or very minimal traffic from countries such as Korea, Moldova, Iran, the Russian Federation, and Tanzania in June. However, these countries had a significant volume of traffic in May.
- Notably, Korea alone accounted for 68% of the Total CTP, 31% of the Bookings, and 13% of the Clickouts, but it contributed only 0.7% of the total sessions in May. Consequently, while the total sessions and user counts remained relatively unchanged due to the limited contribution from Korea, the significant drop in CTP, Bookings, and COR was primarily attributed to their higher involvement in May.

0.2 Q1)

Calculate the clickout ratio per platform and device type - what platform has the highest COR? What device has the lowest COR? Are there differences by traffic type? Can you draw any conclusions from the ratios about the coded values for traffic type?

[ ]:

```
df_task1.head()
```

[ ]:

```
        ymd           session_id tracking_id platform is_app  is_repeater  \
0  20220626  2022062620046057322  FA6JXA8TAJ       UK      0            1
1  20220518  2022051821943006017  0X7RLU6KF7       BR      0            0
2  20220508  2022050821020053928  0I59VWLQW0       UK      0            0
3  20220507  2022050706015039122  JXNHOBQL50       CH      0            0
4  20220523  2022052320052048087  W24I0V5Z2L       IT      0            0

  traffic_type    country_name agent_id  clickouts  bookings  \
0            2  United Kingdom       16          0         0
1            2          Brazil        2          3         0
2            2  United Kingdom       20          0         0
3            2     Switzerland       28          0         0
4            2           Italy       20          6         0

   session_duration entry_page  total_ctp arrival_day departure_day  \
0                29        2111          0          \N            \N
1              1485        2100         27    20220530      20220531
2               143        2100          0          \N            \N
3                69        2100          0          \N            \N
4               887        2100        100    20220609      20220610

         date
0  2022-06-26
1  2022-05-18
2  2022-05-08
3  2022-05-07
4  2022-05-23
```

```
[ ]:
```

```
[ ]:
```