Website_Performance_Analysis_Project

January 13, 2026

[1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:

```
df= pd.read_csv("data-export.csv")
```

[3]:

```
df.head()
```

[3]:

```
          # ----------------------------------------  \
0  Session primary channel group (Default channel…
1                                              Direct
2                                      Organic Social
3                                              Direct
4                                      Organic Social

                 Unnamed: 1 Unnamed: 2 Unnamed: 3        Unnamed: 4  \
0  Date + hour (YYYYMMDDHH)      Users   Sessions  Engaged sessions
1                2024041623        237        300               144
2                2024041719        208        267               132
3                2024041723        188        233               115
4                2024041718        187        256               125

                            Unnamed: 5                 Unnamed: 6  \
0  Average engagement time per session  Engaged sessions per user
1                   47.526666666666700         0.6075949367088610
2                    32.09737827715360         0.6346153846153850
3                    39.93991416309010         0.6117021276595740
4                          32.16015625         0.6684491978609630

           Unnamed: 7           Unnamed: 8   Unnamed: 9
0  Events per session      Engagement rate  Event count
1   4.673333333333330                 0.48         1402
2   4.295880149812730   0.4943820224719100         1147
3   4.587982832618030  0.49356223175965700         1069
4            4.078125           0.48828125         1044
```

[4]:

```
df.columns=df.iloc[0]
```

[5]:

```
df.head()
```

[5]:

```
0 Session primary channel group (Default channel group)  \
0  Session primary channel group (Default channel…
1                                             Direct
2                                     Organic Social
3                                             Direct
4                                     Organic Social

0  Date + hour (YYYYMMDDHH)  Users  Sessions  Engaged sessions  \
0  Date + hour (YYYYMMDDHH)  Users  Sessions  Engaged sessions
1                2024041623    237       300               144
2                2024041719    208       267               132
3                2024041723    188       233               115
4                2024041718    187       256               125

0  Average engagement time per session  Engaged sessions per user  \
0  Average engagement time per session  Engaged sessions per user
1                   47.526666666666700         0.6075949367088610
2                    32.09737827715360         0.6346153846153850
3                    39.93991416309010         0.6117021276595740
4                          32.16015625         0.6684491978609630

0  Events per session      Engagement rate  Event count
0  Events per session      Engagement rate  Event count
1   4.673333333333330                 0.48         1402
2   4.295880149812730   0.4943820224719100         1147
3   4.587982832618030  0.49356223175965700         1069
4            4.078125           0.48828125         1044
```

[6]:

```
df.columns=df.iloc[0]
df=df.drop(index = 0).reset_index(drop=True)
df.columns = ["Channel Group","DateHour","Users","Sessions","Engaged␣
 ↪Sessions","Average Engagement Time Per Session","Engaged Sessions Per␣
 ↪User'","Eveant Per session","Engagement Rate","Event Count"]
```

[7]:

```
df.head()
```

[7]:

```
    Channel Group    DateHour Users Sessions Engaged Sessions  \
0          Direct  2024041623   237      300              144
1  Organic Social  2024041719   208      267              132
2          Direct  2024041723   188      233              115
3  Organic Social  2024041718   187      256              125
4  Organic Social  2024041720   175      221              112

  Average Engagement Time Per Session Engaged Sessions Per User'  \
0                  47.526666666666700         0.6075949367088610
```

```
1                   32.09737827715360         0.6346153846153850
2                   39.93991416309010         0.6117021276595740
3                         32.16015625         0.6684491978609630
4                  46.918552036199100                       0.64

  Eveant Per session      Engagement Rate Event Count
0  4.673333333333330                 0.48        1402
1  4.295880149812730   0.4943820224719100        1147
2  4.587982832618030  0.49356223175965700        1069
3           4.078125           0.48828125        1044
4  4.529411764705880   0.5067873303167420        1001
```

[8]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3182 entries, 0 to 3181
Data columns (total 10 columns):
 #   Column                                Non-Null Count  Dtype
---  ------                                --------------  -----
 0   Channel Group                         3182 non-null   object
 1   DateHour                              3182 non-null   object
 2   Users                                 3182 non-null   object
 3   Sessions                              3182 non-null   object
 4   Engaged Sessions                      3182 non-null   object
 5   Average Engagement Time Per Session  3182 non-null   object
 6   Engaged Sessions Per User'            3182 non-null   object
 7   Eveant Per session                    3182 non-null   object
 8   Engagement Rate                       3182 non-null   object
 9   Event Count                           3182 non-null   object
dtypes: object(10)
memory usage: 248.7+ KB
```

1 Cleaning data and Validation for the above dataset

[9]:
```python
df["DateHour"] = pd.
 ↪to_datetime(df["DateHour"],format="%Y%m%d%H",errors="coerce")
numeric_cols = df.columns.drop(["Channel Group","DateHour"])
df[numeric_cols] = df[numeric_cols].apply(pd.to_numeric,errors="coerce")
df["Hour"] = df["DateHour"].dt.hour
```

[10]:
```python
df.head()
```

[10]:
```
    Channel Group            DateHour  Users  Sessions  Engaged Sessions  \
0          Direct 2024-04-16 23:00:00    237       300               144
1  Organic Social 2024-04-17 19:00:00    208       267               132
2          Direct 2024-04-17 23:00:00    188       233               115
3  Organic Social 2024-04-17 18:00:00    187       256               125
```

```
4  Organic Social 2024-04-17 20:00:00    175       221               112

   Average Engagement Time Per Session  Engaged Sessions Per User'  \
0                            47.526667                    0.607595
1                            32.097378                    0.634615
2                            39.939914                    0.611702
3                            32.160156                    0.668449
4                            46.918552                    0.640000

   Eveant Per session  Engagement Rate  Event Count  Hour
0            4.673333         0.480000         1402    23
1            4.295880         0.494382         1147    19
2            4.587983         0.493562         1069    23
3            4.078125         0.488281         1044    18
4            4.529412         0.506787         1001    20
```

[11]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3182 entries, 0 to 3181
Data columns (total 11 columns):
 #   Column                               Non-Null Count  Dtype
---  ------                               --------------  -----
 0   Channel Group                        3182 non-null   object
 1   DateHour                             3182 non-null   datetime64[ns]
 2   Users                                3182 non-null   int64
 3   Sessions                             3182 non-null   int64
 4   Engaged Sessions                     3182 non-null   int64
 5   Average Engagement Time Per Session  3182 non-null   float64
 6   Engaged Sessions Per User'           3182 non-null   float64
 7   Eveant Per session                   3182 non-null   float64
 8   Engagement Rate                      3182 non-null   float64
 9   Event Count                          3182 non-null   int64
 10  Hour                                 3182 non-null   int32
dtypes: datetime64[ns](1), float64(4), int32(1), int64(4), object(1)
memory usage: 261.2+ KB
```

[12]:
```
df.describe()
```

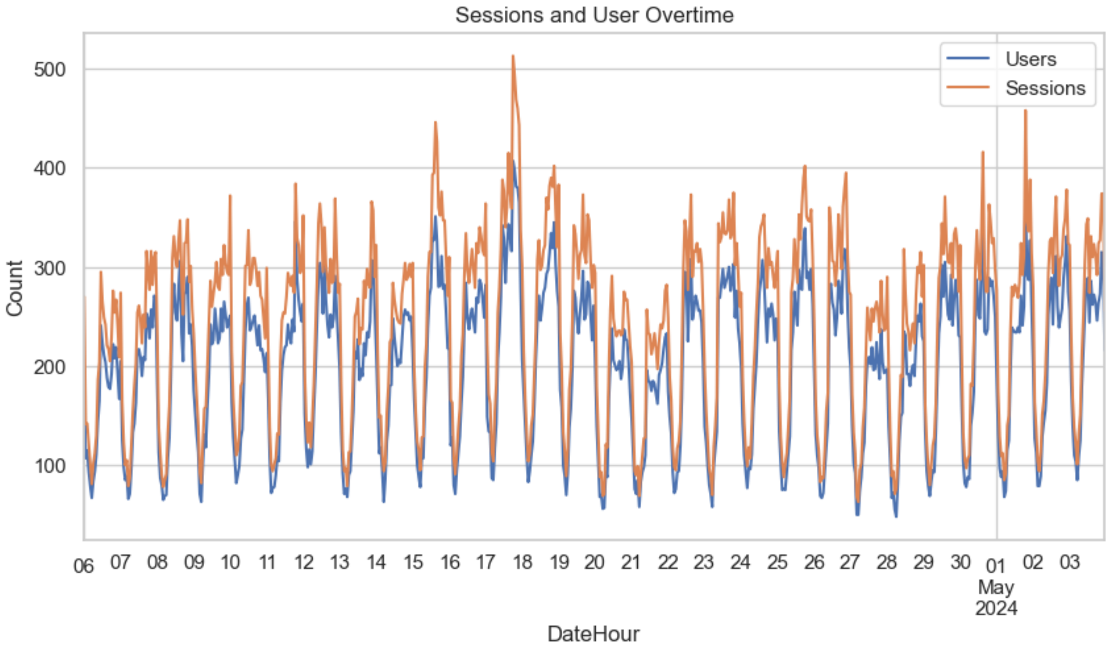
[12]:
```
                            DateHour        Users     Sessions  \
count                           3182  3182.000000  3182.000000
mean   2024-04-20 01:17:07.278441216    41.935889    51.192646
min              2024-04-06 00:00:00     0.000000     1.000000
25%              2024-04-13 02:15:00    20.000000    24.000000
50%              2024-04-20 02:00:00    42.000000    51.000000
75%              2024-04-26 22:00:00    60.000000    71.000000
max              2024-05-03 23:00:00   237.000000   300.000000
std                              NaN    29.582258    36.919962
```

```
       Engaged Sessions  Average Engagement Time Per Session  \
count       3182.000000                          3182.000000
mean          28.325581                            66.644581
min            0.000000                             0.000000
25%           13.000000                            32.103034
50%           27.000000                            49.020202
75%           41.000000                            71.487069
max          144.000000                          4525.000000
std           20.650569                           127.200659

       Engaged Sessions Per User'  Eveant Per session  Engagement Rate  \
count                 3182.000000         3182.000000      3182.000000
mean                     0.606450            4.675969         0.503396
min                      0.000000            1.000000         0.000000
25%                      0.561404            3.750000         0.442902
50%                      0.666667            4.410256         0.545455
75%                      0.750000            5.217690         0.633333
max                      2.000000           56.000000         1.000000
std                      0.264023            2.795228         0.228206

       Event Count         Hour
count  3182.000000  3182.000000
mean    242.272470    11.807040
min       1.000000     0.000000
25%     103.000000     6.000000
50%     226.000000    12.000000
75%     339.000000    18.000000
max    1402.000000    23.000000
std     184.440313     6.886686
```

2 1. Session and User Overtime

[13]:

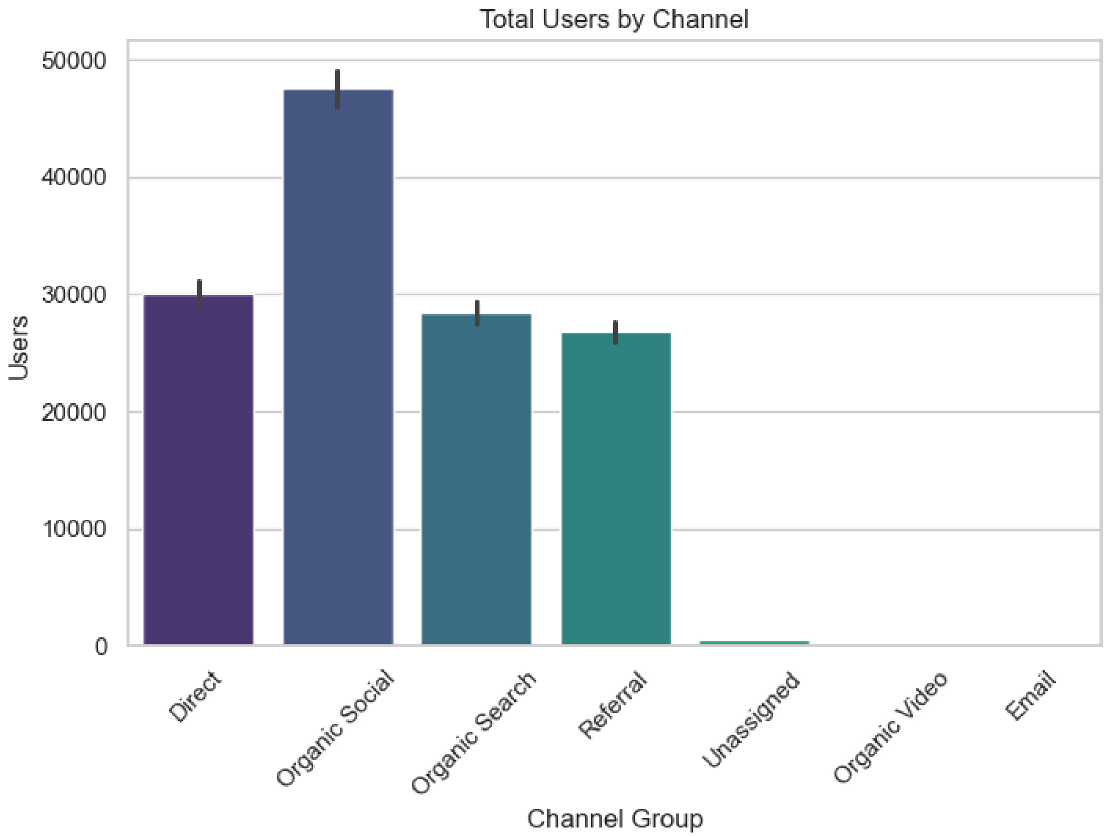
```
sns.set(style="whitegrid")
plt.figure(figsize=(10,5))
df.groupby("DateHour")[["Users","Sessions"]].sum().plot(ax=plt.gca())
plt.title("Sessions and User Overtime")
plt.xlabel("DateHour")
plt.ylabel("Count")
plt.show()
```



3 2. Total Users by Channel

[14]:

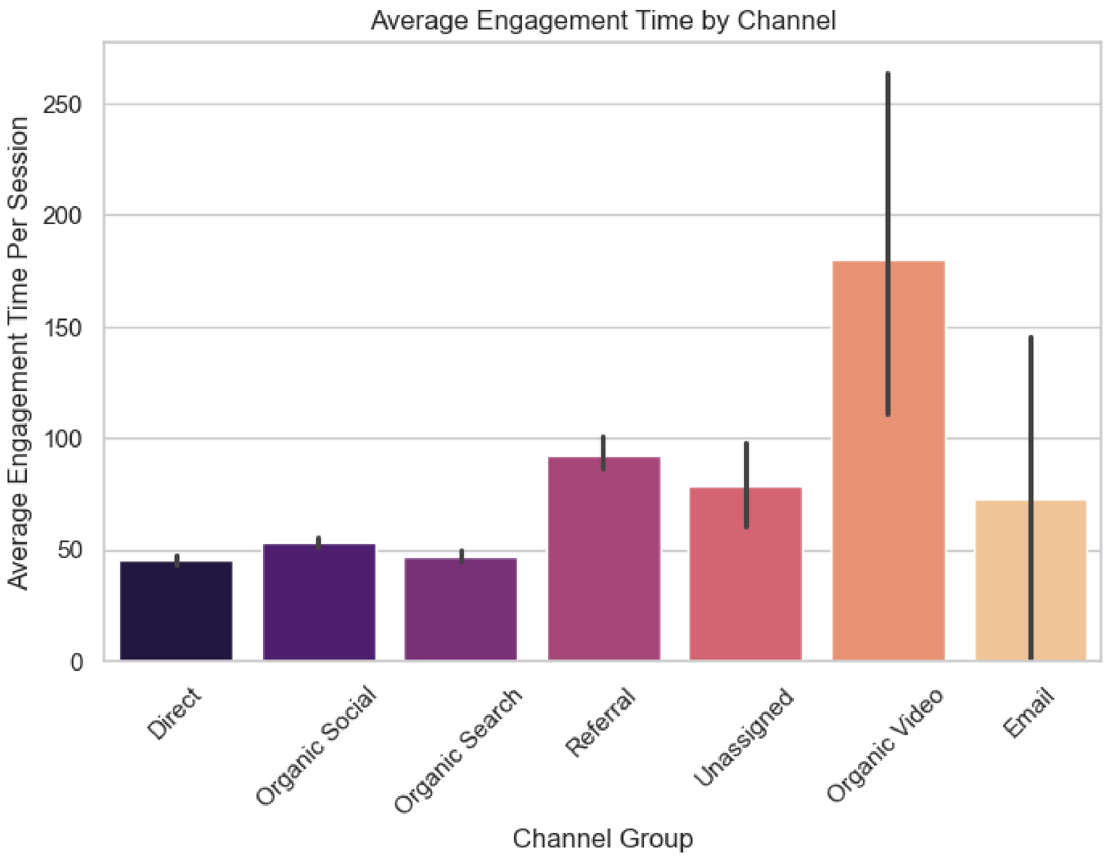
```
plt.figure(figsize=(8,5))
sns.barplot(data=df,x ="Channel Group",y ="Users",hue="Channel␣
 ↪Group",estimator=np.sum,palette="viridis")
plt.title("Total Users by Channel")
plt.xticks(rotation=45)
plt.show()
```



4 3. Average Engagement Time by Channel

[42]:

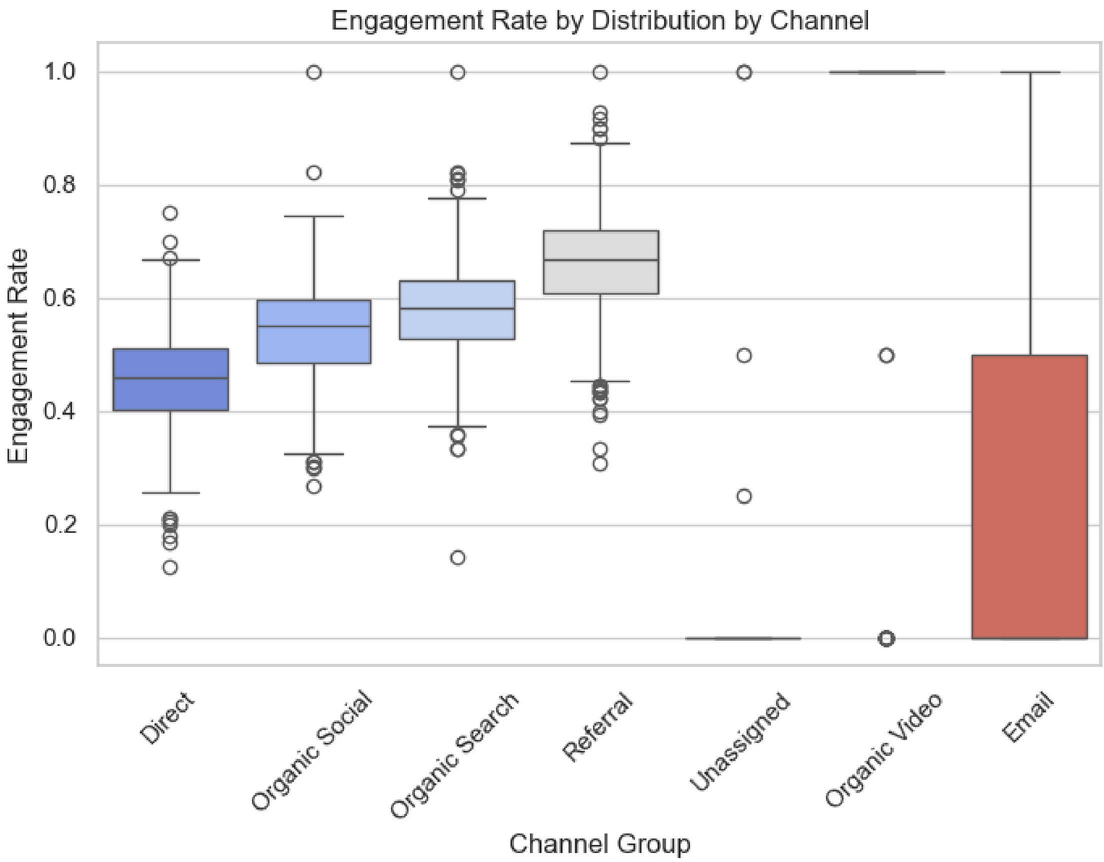
```
plt.figure(figsize=(8,5))
sns.barplot(data=df,x= "Channel Group",y= "Average Engagement Time Per␣
 ↪Session",hue= "Channel Group",estimator= np.mean,palette="magma")
plt.title("Average Engagement Time by Channel")
plt.xticks(rotation=45)
plt.show()
```



5 4. Engagement Rate Distribution by Channel

[47]:

```
plt.figure(figsize=(8,5))
sns.boxplot(data=df,x= "Channel Group",y= "Engagement Rate",hue= "Channel␣
↪Group",palette="coolwarm")
plt.title("Engagement Rate by Distribution by Channel")
plt.xticks(rotation=45)
plt.show()
```



6 5. Engaged and Non-Engaged Channel

[15]:

```
df.head()
```

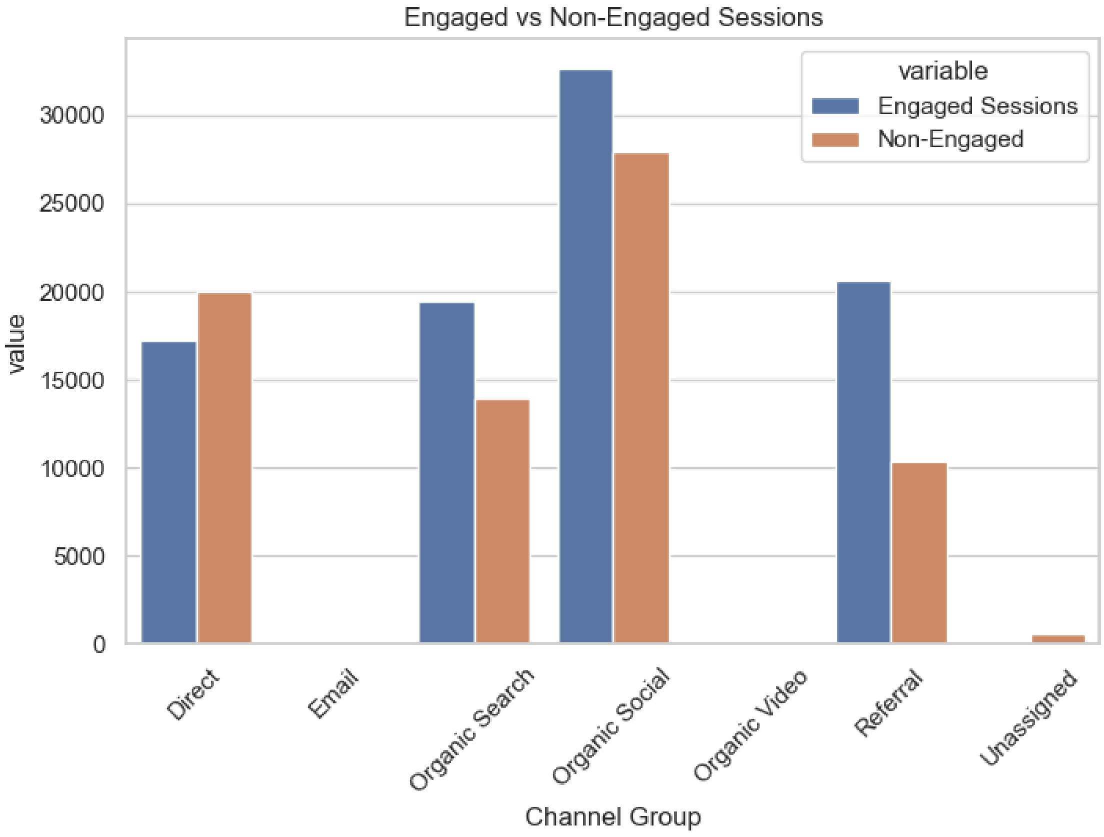
[15]:

```
     Channel Group            DateHour  Users  Sessions  Engaged Sessions  \
0           Direct 2024-04-16 23:00:00    237       300               144
1   Organic Social 2024-04-17 19:00:00    208       267               132
2           Direct 2024-04-17 23:00:00    188       233               115
3   Organic Social 2024-04-17 18:00:00    187       256               125
4   Organic Social 2024-04-17 20:00:00    175       221               112

   Average Engagement Time Per Session  Engaged Sessions Per User'  \
0                            47.526667                    0.607595
1                            32.097378                    0.634615
2                            39.939914                    0.611702
3                            32.160156                    0.668449
4                            46.918552                    0.640000

   Eveant Per session  Engagement Rate  Event Count  Hour
```

```
0             4.673333         0.480000         1402    23
1             4.295880         0.494382         1147    19
2             4.587983         0.493562         1069    23
3             4.078125         0.488281         1044    18
4             4.529412         0.506787         1001    20
```

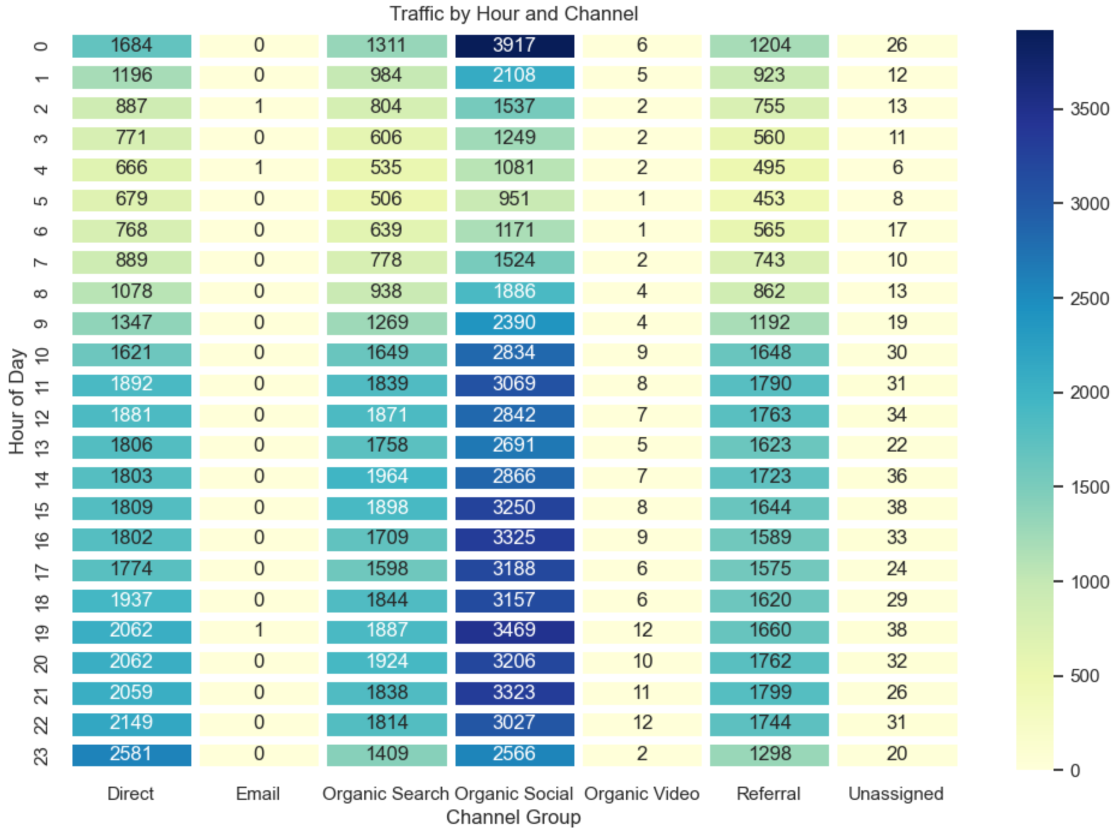
[18]:

```
session_df = df.groupby("Channel Group")[["Sessions","Engaged Sessions"]].sum().
  ↪reset_index()
session_df["Non-Engaged"] = session_df["Sessions"] - session_df["Engaged␣
  ↪Sessions"]
session_df_melted = session_df.melt(id_vars="Channel Group",␣
  ↪value_vars=["Engaged Sessions", "Non-Engaged"])

plt.figure(figsize=(8,5))
sns.barplot(data=session_df_melted, x="Channel Group", y="value",␣
  ↪hue="variable")
plt.title("Engaged vs Non-Engaged Sessions")
plt.xticks(rotation=45)
plt.show()
```



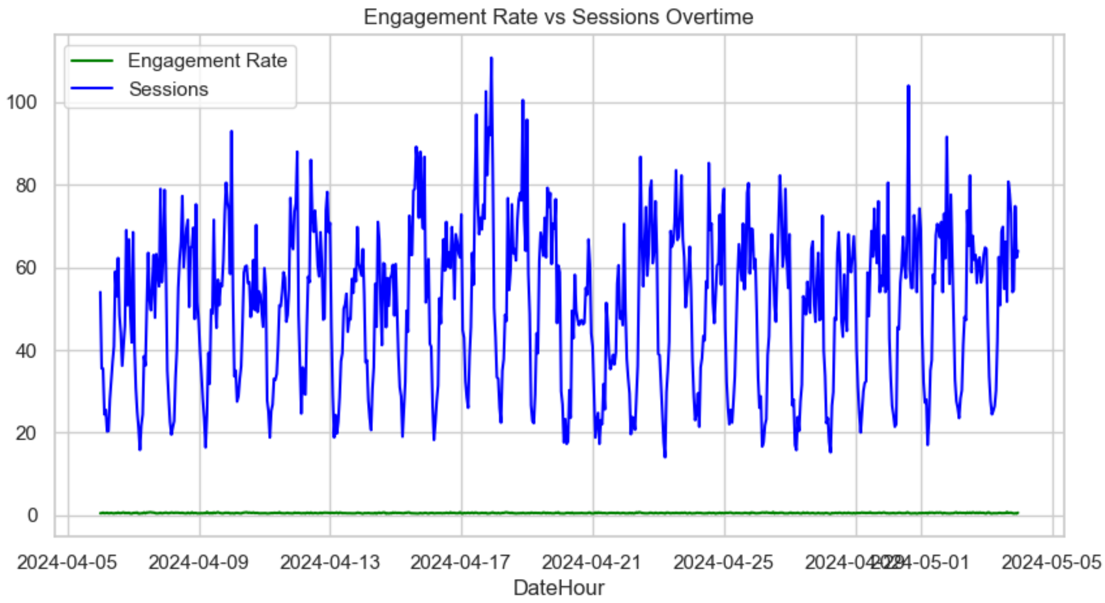
7 6. Traffic by Hour and Channel

```
[21]: heatmap_data = df.groupby(["Hour","Channel Group"])["Sessions"].sum().unstack().
       ↪fillna(0)

      plt.figure(figsize=(12,8))
      sns.heatmap(heatmap_data,cmap="YlGnBu", linewidths=5, annot=True, fmt=".0f")
      plt.title("Traffic by Hour and Channel")
      plt.xlabel("Channel Group")
      plt.ylabel("Hour of Day")
      plt.show()
```



8 7. Engagement rate vs Sessions Overtime

```
[22]: df_plot = df.groupby("DateHour")[["Engagement Rate","Sessions"]].mean().
       ↪reset_index()

      plt.figure(figsize=(10,5))
      plt.plot(df_plot["DateHour"],df_plot["Engagement Rate"],label="Engagement␣
       ↪Rate",color="green")
      plt.plot(df_plot["DateHour"],df_plot["Sessions"],label="Sessions",color="blue")
      plt.title("Engagement Rate vs Sessions Overtime")
      plt.xlabel("DateHour")
      plt.legend()
      plt.grid(True)
      plt.show()
```



[ ]:

```

```