Bivariate Analysis

Numerical Numerical

For Numerical Numerical exploration, we used Scatterplot to plot Frequency of Search Terms against Different intervals of time. As Google tends to get tons of search queries, plotting all of them isn't feasible. As our goal is to find the highest grossing search query, we first set an interval to get the highest frequency for each query by splitting time into intervals of month.

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
# %matplotlib inline
import warnings
warnings.filterwarnings('ignore')


df = pd.read_csv('./trends.csv')
df = df.drop(columns = ['region_name', 'region_code', 'score', 'refresh_date'], axis=1)
df['week'] = df['week'].str[:7]
df = df.rename(columns={'week':'year'})
```

As splitting time according to year would make our data sparse, we avoid that. Then we group our data according to Date interval and our search term to get the frequencies.

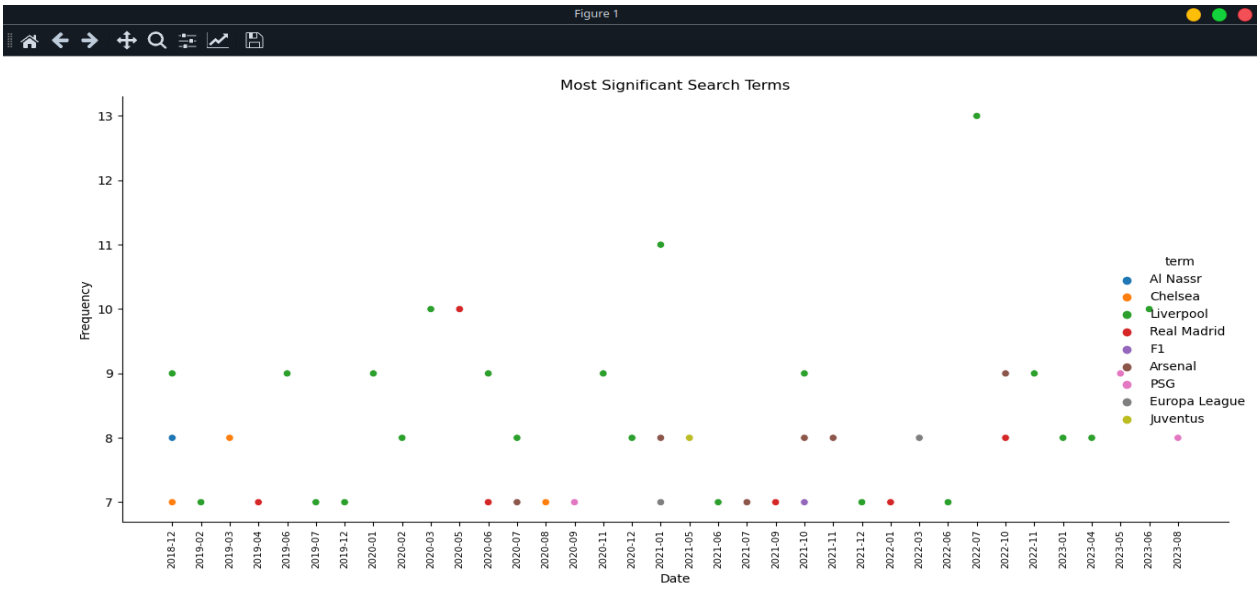
```
newDf = df.groupby(['year', 'term']).size().reset_index(name='Frequency')
```

As our target is to get data with high frequency, we filter our dataset with setting the limit of Frequency > 6.

```
newDf = newDf[newDf['Frequency'] > 6]
```

Now, we plot our scatterplot with Search frequency and date interval.

```
sns.relplot( data= newDf[['year','term', 'Frequency']], x = 'year', y = 'Frequency', hue = 'term')
plt.xlabel('Date')
plt.ylabel('Frequency')
plt.title('Most Significant Search Terms')
plt.xticks(rotation='vertical', fontsize=8)
plt.tight_layout()
plt.show()
```



For determining correlation between search term frequency and time, we isolate our dataset with data concerning only one search term suppose ‘Liverpool’.

```
liverpoolDf = newDf[newDf['term'] == 'Liverpool']
print(liverpoolDf)
```

	year	term	Frequency
1910	2018-12	Liverpool	9
3430	2019-02	Liverpool	7
6413	2019-06	Liverpool	9
7122	2019-07	Liverpool	7
10737	2019-12	Liverpool	7
11466	2020-01	Liverpool	9
12156	2020-02	Liverpool	8
12939	2020-03	Liverpool	10
15250	2020-06	Liverpool	9
15958	2020-07	Liverpool	8
19010	2020-11	Liverpool	9
19791	2020-12	Liverpool	8
20471	2021-01	Liverpool	11

24171	2021-06	Liverpool	7
27133	2021-10	Liverpool	9
28650	2021-12	Liverpool	7
29409	2022-01	Liverpool	7
33161	2022-06	Liverpool	7
33942	2022-07	Liverpool	13
36230	2022-10	Liverpool	8
37023	2022-11	Liverpool	9
38441	2023-01	Liverpool	8
40624	2023-04	Liverpool	8
42114	2023-06	Liverpool	10

Then we convert our date interval into numeric value I.e. timestamp.

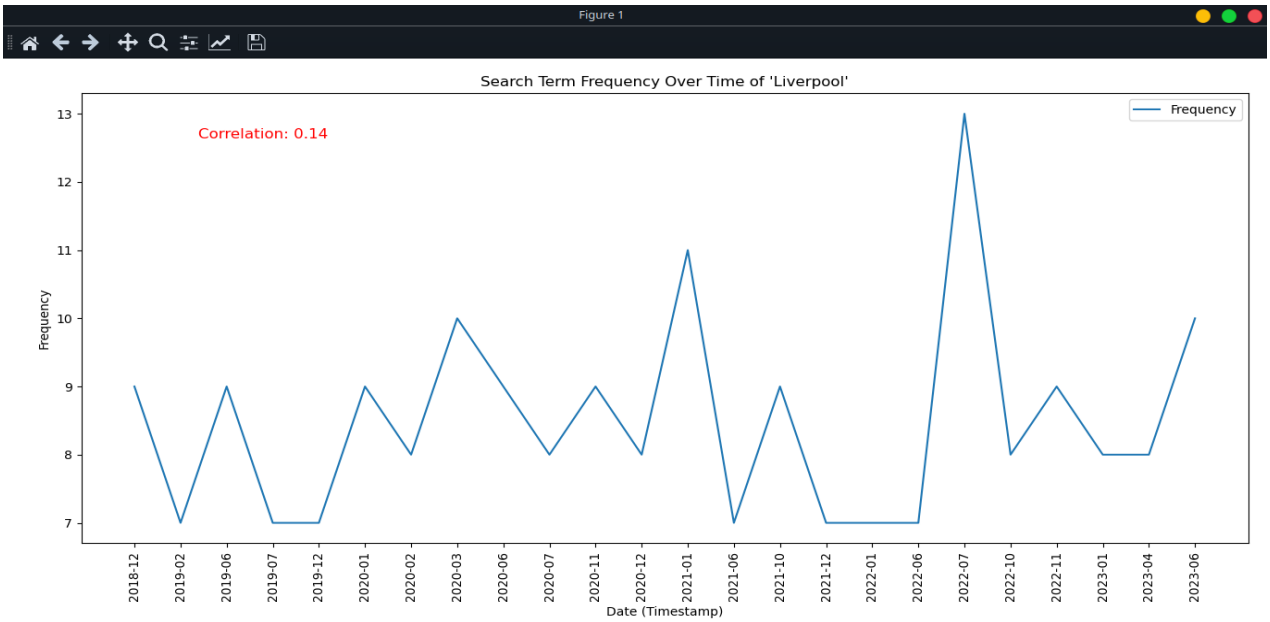
```
liverpoolDf['Numeric_date'] = pd.to_datetime(liverpoolDf['year']).apply(lambda x: x.timestamp())
```

After that we find our correlation and plot a linegraph of search term frequency over interval of time.

```
correlation = liverpoolDf['Numeric_date'].corr(liverpoolDf['Frequency'])
print(f"Correlation: {correlation: .2f} ")

plt.figure(figsize=(10, 6))
plt.plot(liverpoolDf['year'], liverpoolDf['Frequency'], label='Frequency')
plt.xlabel('Date (Timestamp)')
plt.ylabel('Frequency')
plt.title('Search Term Frequency Over Time of \'Liverpool\'')
plt.legend()

# Display the plot
plt.text(0.1, 0.9, f'Correlation: {correlation:.2f}', transform=plt.gca().transAxes, fontsize=12, color='red')
plt.xticks(rotation='vertical')
plt.tight_layout()
plt.show()
```



Since, the correlation between search term frequency and date interval is 0.14, it seems less likely to be related.