PRODIGY_DS_04

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

[2]:
```
cols=['ID', 'Topic', 'Sentiment', 'Text']
train = pd.read_csv(r"/content/twitter_training.csv",names=cols)
```

[3]:
```
train.head()
```

[3] :

	ID	Topic	Sentiment	\
0	2401	Borderlands	Positive	
1	2401	Borderlands	Positive	
2	2401	Borderlands	Positive	
3	2401	Borderlands	Positive	
4	2401	Borderlands	Positive	

	Text
0	im getting on borderlands and i will murder yo...
1	I am coming to the borders and I will kill you...
2	im getting on borderlands and i will kill you ...
3	im coming on borderlands and i will murder you...
4	im getting on borderlands 2 and i will murder ...

[4] :
```
train.shape
```

[4]: (46295, 4)

[5] :
```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 46295 entries, 0 to 46294
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   ID          46295 non-null  int64
 1   Topic       46295 non-null  object
```

```
 2   Sentiment  46295 non-null  object
 3   Text       45850 non-null  object
dtypes: int64(1), object(3)
memory usage: 1.4+ MB
```

[6]:
```
train.describe(include=object)
```

[6]:

	Topic	Sentiment	Text
count	46295	46295	45850
unique	20	4	42998
top	Microsoft	Positive	It is not the first time that the EU Commissio...
freq	2400	13710	109

[7]:
```
train['Sentiment'].unique()
```

[7]:
```
array(['Positive', 'Neutral', 'Negative', 'Irrelevant'], dtype=object)
```

[8]:
```
train.isnull().sum()
```

[8]:
```
ID             0
Topic          0
Sentiment      0
Text         445
dtype: int64
```

[9]:
```
train.dropna(inplace=True)
```

[10]:
```
train.isnull().sum()
```

[10]:
```
ID           0
Topic        0
Sentiment    0
Text         0
dtype: int64
```

[11]:
```
train.duplicated().sum()
```

[11]:
```
1501
```

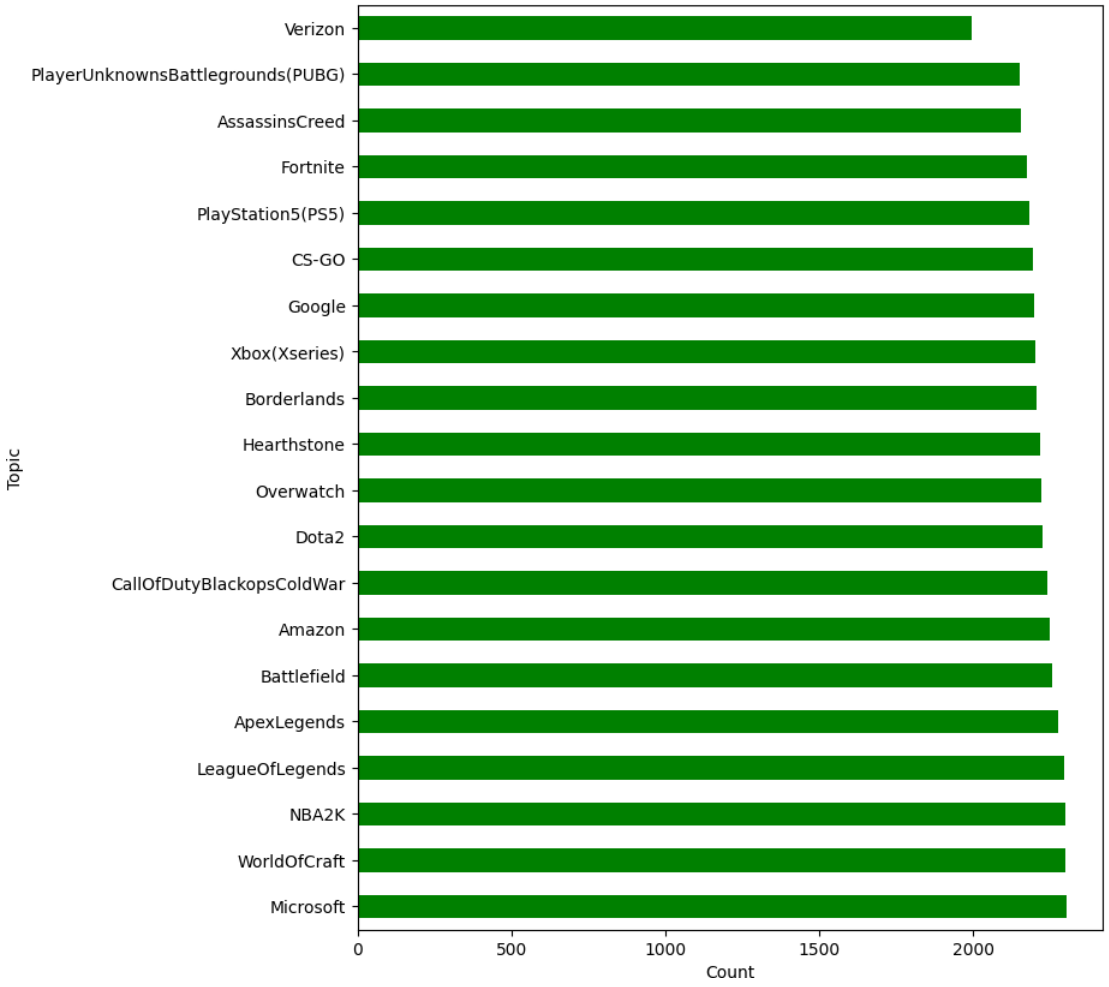
[12]:
```
train.drop_duplicates(inplace=True)
```

[13]:
```
train.duplicated().sum()
```

[13]:
```
0
```

[14]:
```
plt.figure(figsize=(8,10))
train['Topic'].value_counts().plot(kind='barh',color='g')
```

```
plt.xlabel("Count")
plt.show()
```

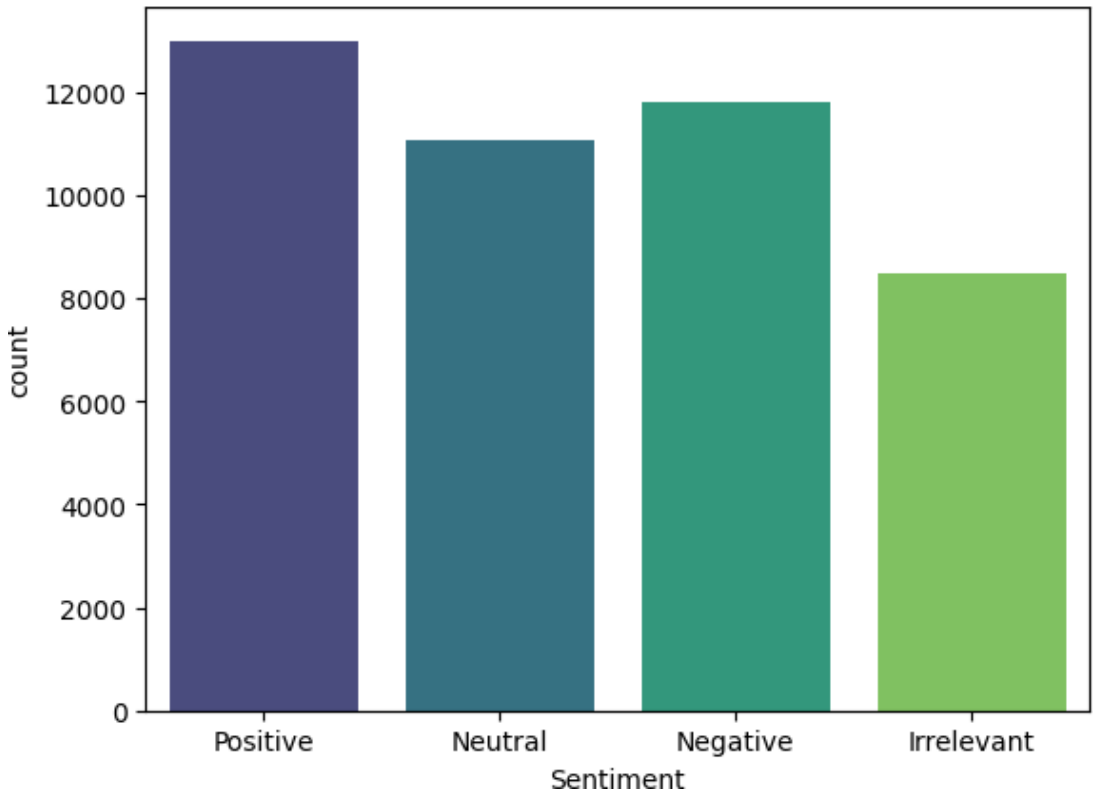


[15] :
```
sns.countplot(x = 'Sentiment',data=train,palette='viridis')
plt.show()
```

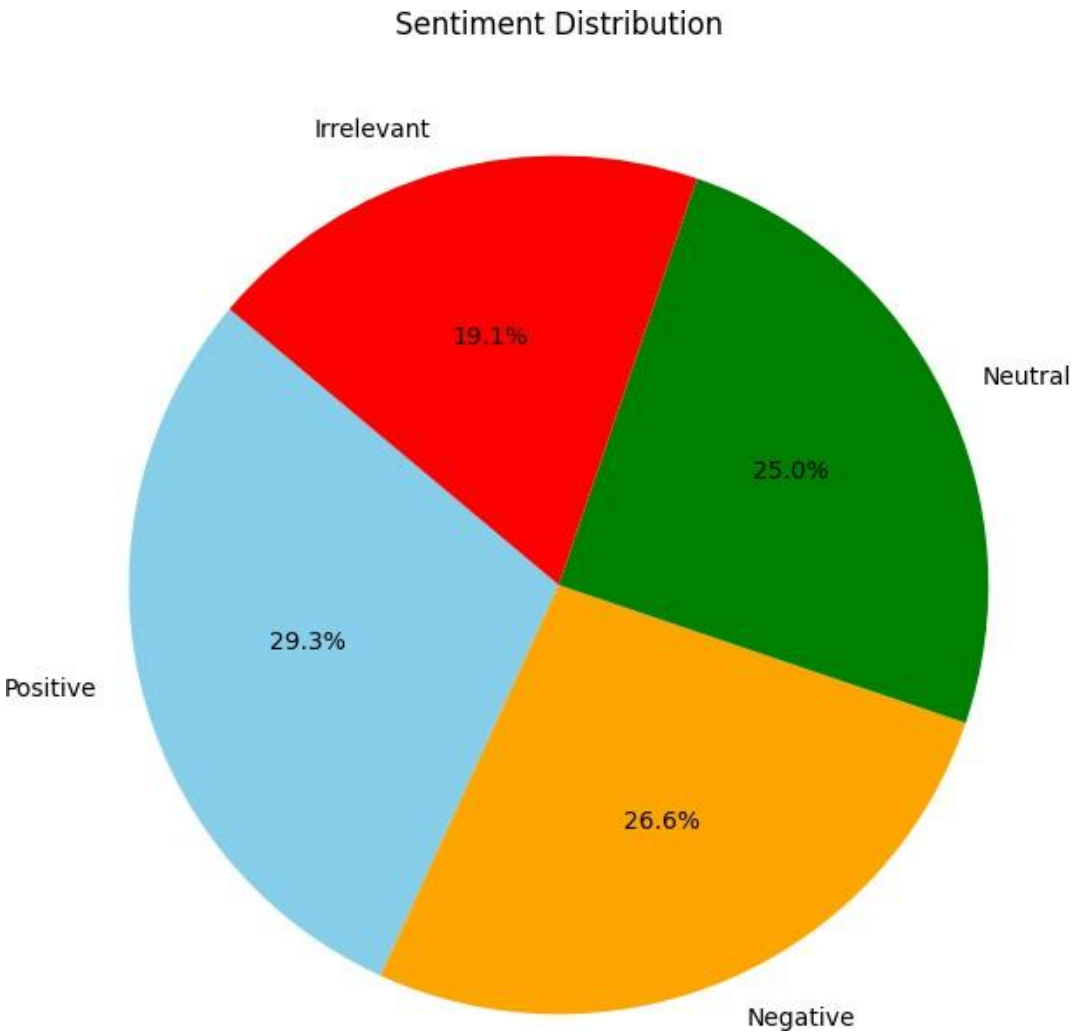
<ipython-input-15-0f5f2096c1d5>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

```
sns.countplot(x = 'Sentiment',data=train,palette='viridis')
```



[16]:
```
# Calculate the counts for each sentiment
sentiment_counts = train['Sentiment'].value_counts()

# Create the pie chart
plt.figure(figsize=(8, 8))
plt.pie(sentiment_counts, labels=sentiment_counts.index, autopct="%1.1f%%",
 startangle=140, colors=['skyblue', 'orange', 'green', 'red', 'purple'])

plt.title('Sentiment Distribution')

# Show the plot
plt.show()
```



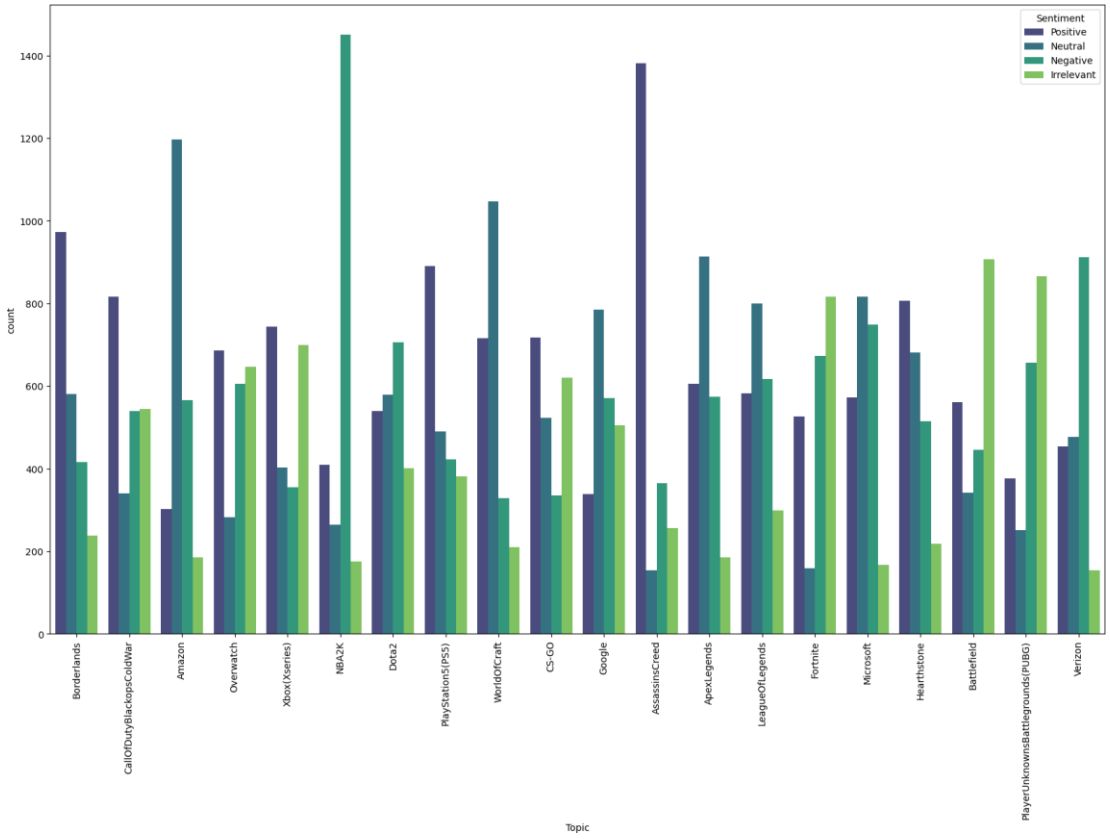
[17] :
```
train
```

[17]:

	ID	Topic	Sentiment	\
0	2401	Borderlands	Positive	
1	2401	Borderlands	Positive	
2	2401	Borderlands	Positive	
3	2401	Borderlands	Positive	
4	2401	Borderlands	Positive	
...	...	...	...	
46289	11943	Verizon	Neutral	
46290	11944	Verizon	Neutral	
46291	11944	Verizon	Neutral	

```
46292  11944       Verizon    Neutral
46294  11944       Verizon    Neutral

                                                    Text
0       im getting on borderlands and i will murder yo...
1       I am coming to the borders and I will kill you...
2       im getting on borderlands and i will kill you ...
3       im coming on borderlands and i will murder you...
4       im getting on borderlands 2 and i will murder ...
...                                                   ...
46289  some stocks play at peak interesting looking i...
46290 The last 3 August’s I have broken my phone. Th...
46291  The last 3 August's I've broken my phone. This...
46292  The last time I broke my phone was on August 3...
46294                                                   7

[44349 rows x 4 columns]
```

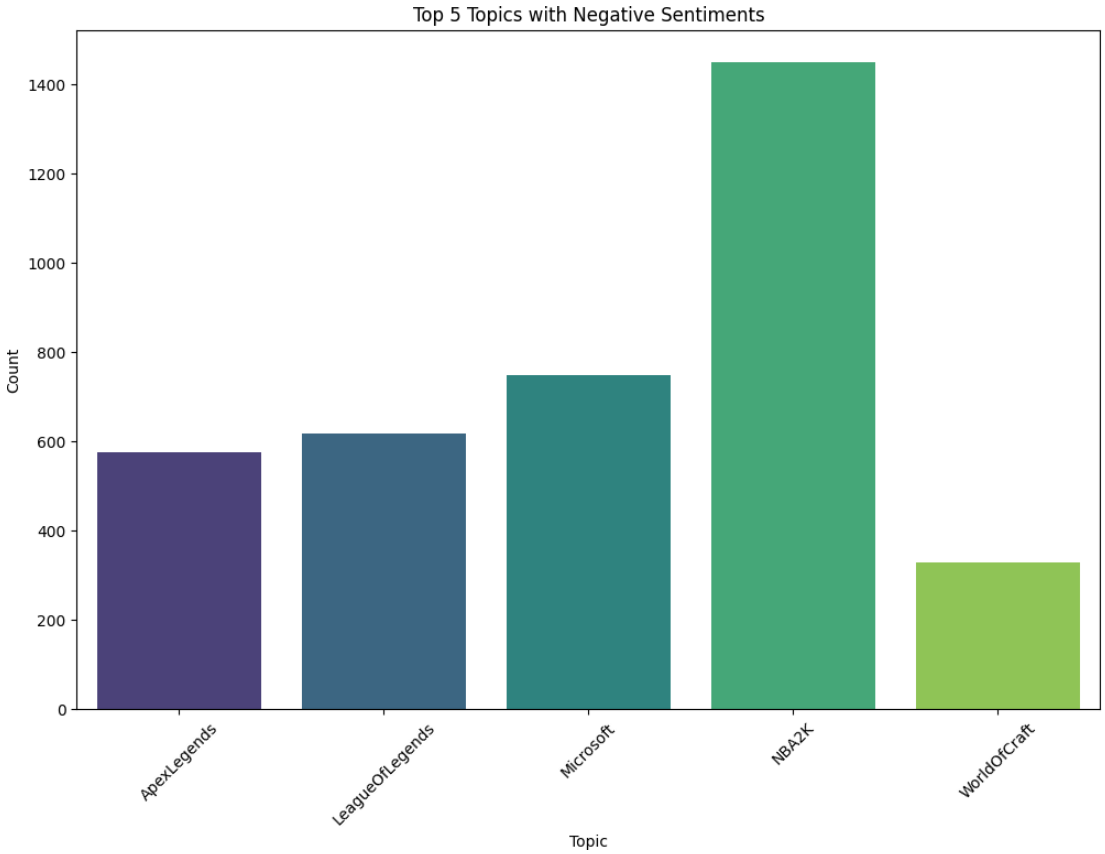
[18] :
```
plt.figure(figsize=(20,12))
sns.countplot(x='Topic',data=train,palette='viridis',hue='Sentiment')
plt.xticks(rotation=90)
plt.show()
```



[20]:
```
## Group by Topic and Sentiment
topic_wise_sentiment = train.groupby(["Topic", "Sentiment"]).size().
  reset_index(name='Count')

# Step 2: Select Top 5 Topics
topic_counts = train['Topic'].value_counts().nlargest(5).index
top_topics_sentiment = topic_wise_sentiment[topic_wise_sentiment['Topic'].
  isin(topic_counts)]
```

[21]:
```
plt.figure(figsize=(12, 8))
sns.barplot(data=top_topics_sentiment[top_topics_sentiment['Sentiment'] ==
  'Negative'], x='Topic', y='Count', palette='viridis')
plt.title('Top 5 Topics with Negative Sentiments')
plt.xlabel('Topic')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```

```
<ipython-input-21-7127521535d3>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(data=top_topics_sentiment[top_topics_sentiment['Sentiment'] ==
'Negative'], x='Topic', y='Count', palette='viridis')
```

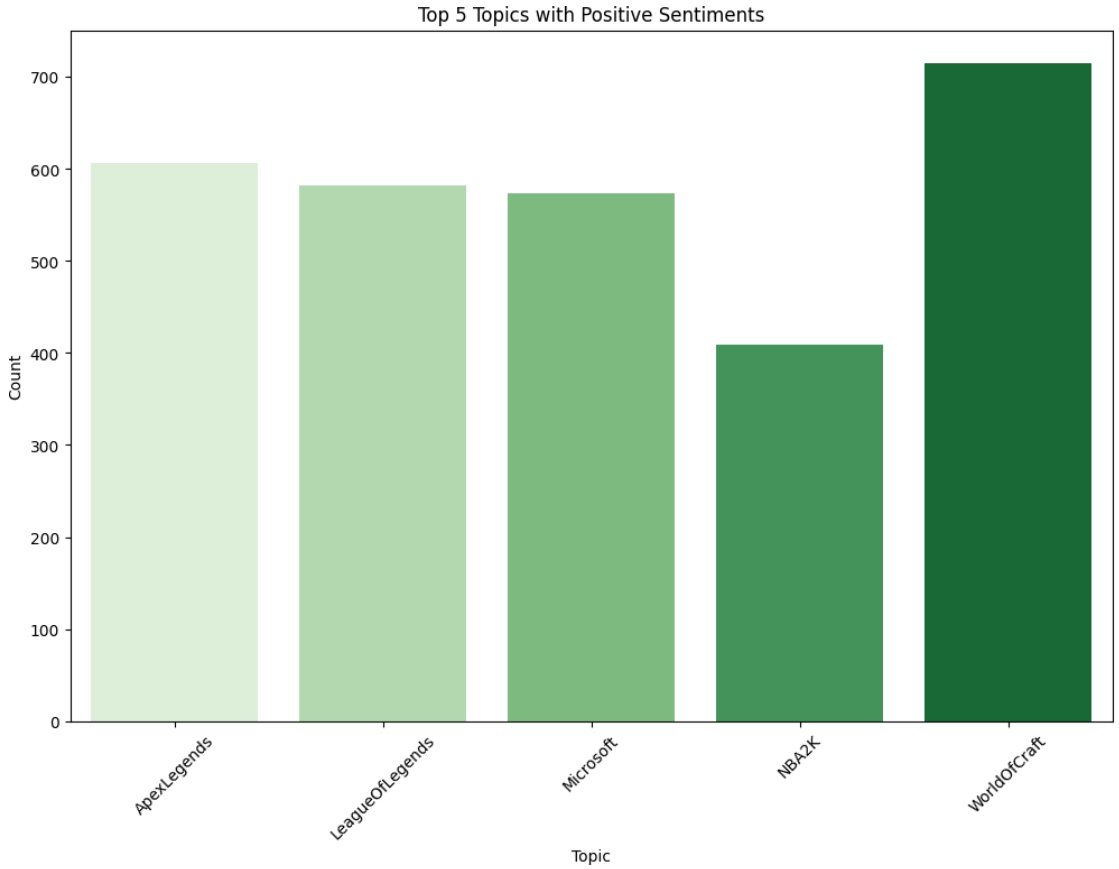


[22]:
```
plt.figure(figsize=(12, 8))
sns.barplot(data=top_topics_sentiment[top_topics_sentiment['Sentiment'] ==␣
 ↪'Positive'], x='Topic', y='Count', palette='Greens')
plt.title('Top 5 Topics with Positive Sentiments')
plt.xlabel('Topic')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```

<ipython-input-22-fa26222f4ed6>:2: FutureWarning:

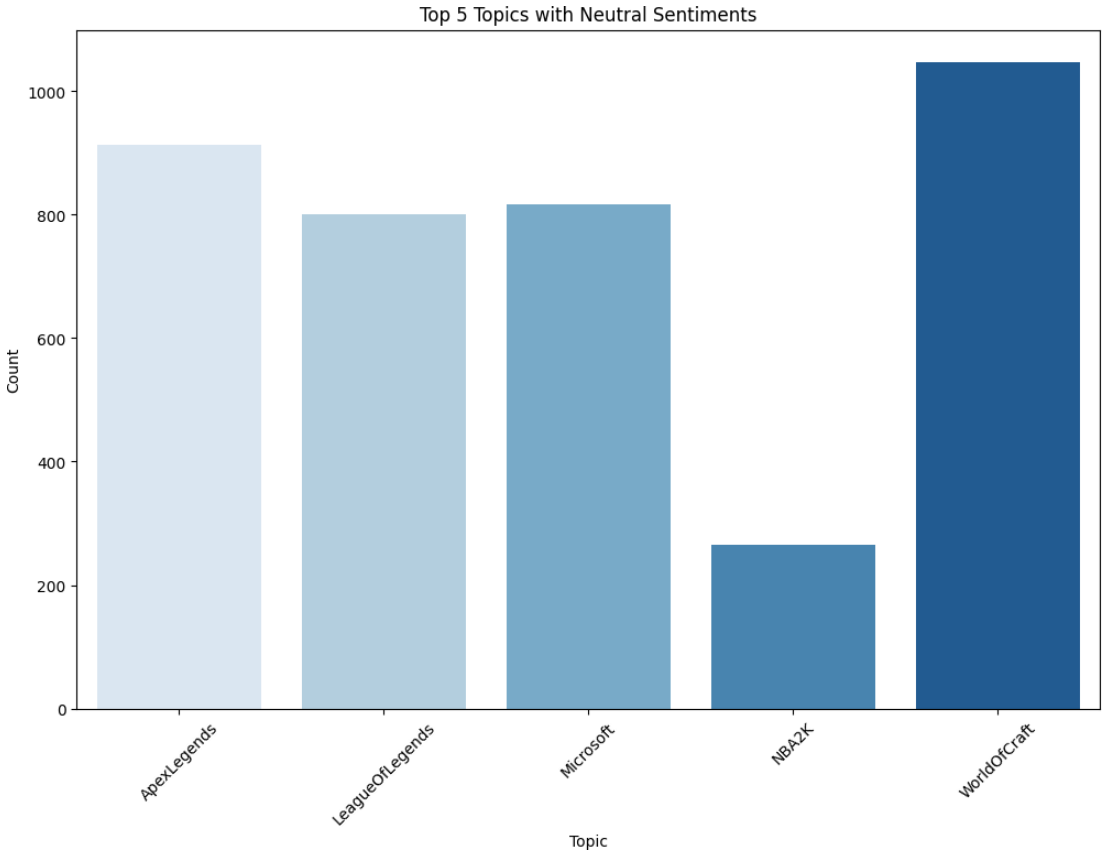
Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

sns.barplot(data=top_topics_sentiment[top_topics_sentiment['Sentiment'] == 'Positive'], x='Topic', y='Count', palette='Greens')



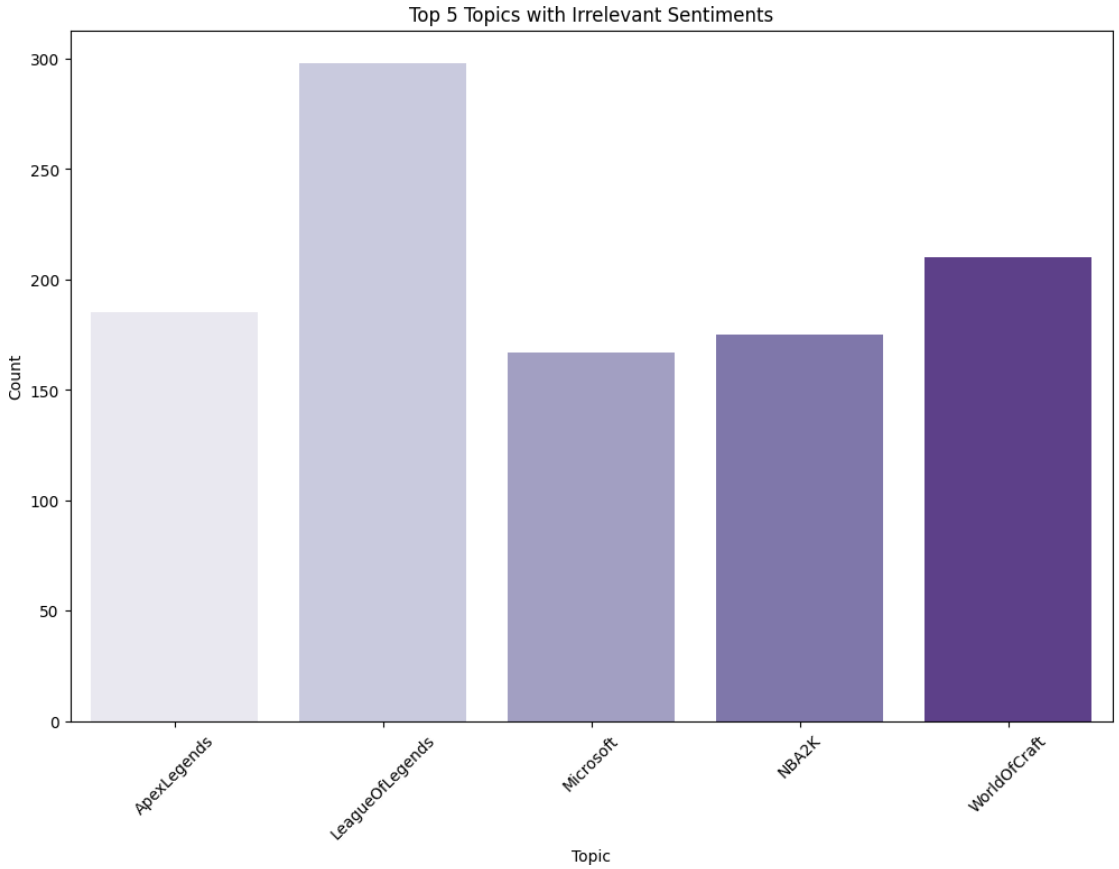
[23]:
```
plt.figure(figsize=(12, 8))
sns.barplot(data=top_topics_sentiment[top_topics_sentiment['Sentiment'] ==␣
 ↪'Neutral'], x='Topic', y='Count', palette='Blues')
plt.title('Top 5 Topics with Neutral Sentiments')
plt.xlabel('Topic')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```

```
<ipython-input-23-af01e1bcdbaa>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(data=top_topics_sentiment[top_topics_sentiment['Sentiment'] ==
'Neutral'], x='Topic', y='Count', palette='Blues')
```

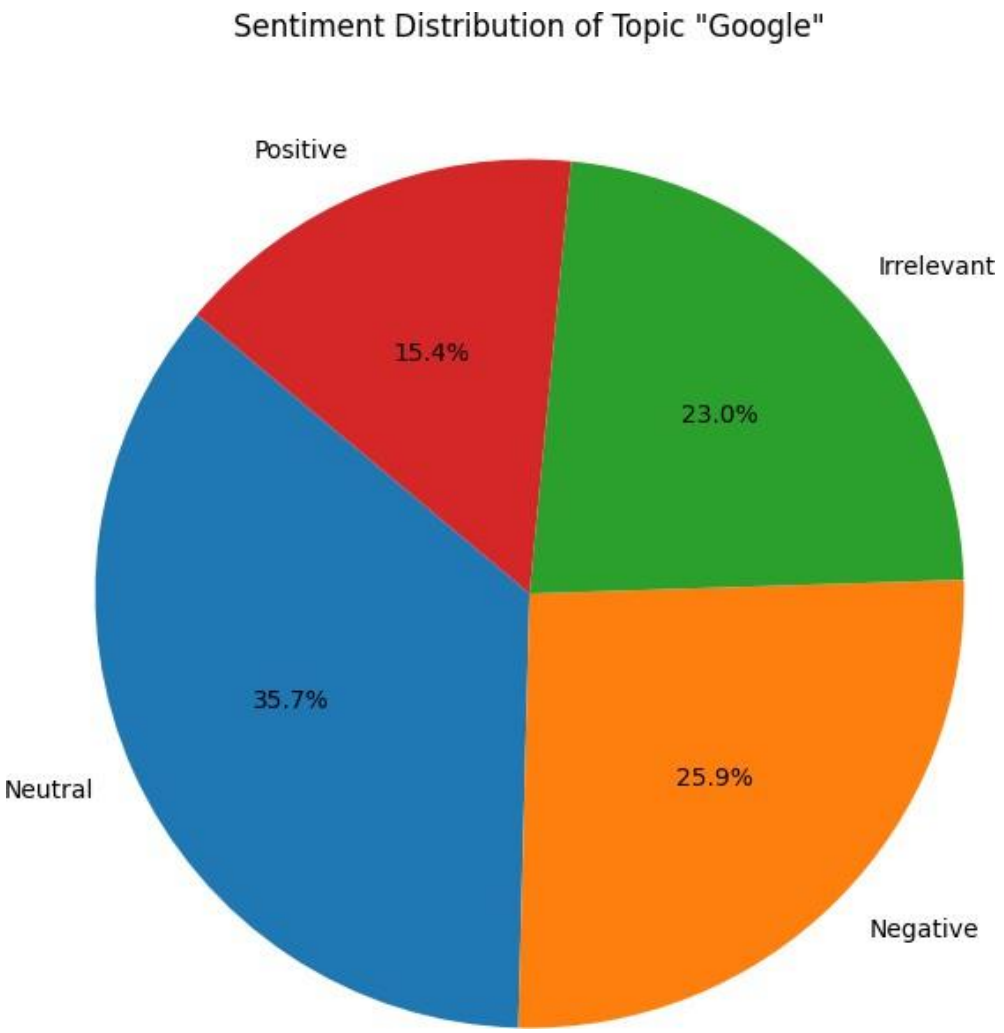


[24]:
```
plt.figure(figsize=(12, 8))
sns.barplot(data=top_topics_sentiment[top_topics_sentiment['Sentiment'] ==␣
 ↪'Irrelevant'], x='Topic', y='Count', palette='Purples')
plt.title('Top 5 Topics with Irrelevant Sentiments')
plt.xlabel('Topic')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```

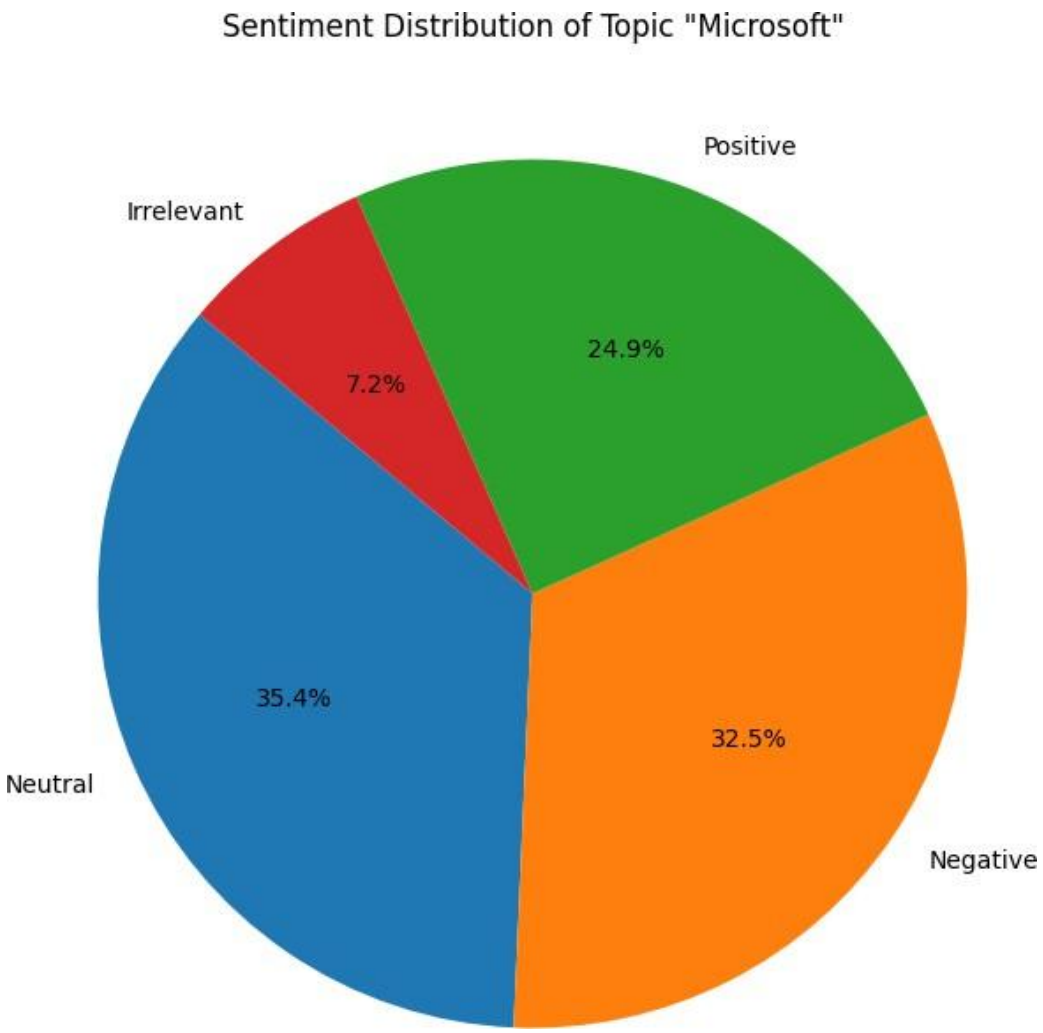
```
<ipython-input-24-7662d01b7d35>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(data=top_topics_sentiment[top_topics_sentiment['Sentiment'] ==
'Irrelevant'], x='Topic', y='Count', palette='Purples')
```



[25]:
```
# Filter the dataset to include only entries related to the topic 'Google'
google_data = train[train['Topic'] == 'Google']

# Count the occurrences of each sentiment within the filtered dataset
sentiment_counts = google_data['Sentiment'].value_counts()

# Plot the pie chart
plt.figure(figsize=(8, 8))
plt.pie(sentiment_counts, labels=sentiment_counts.index, autopct='%1.1f%%',
    startangle=140)
plt.title('Sentiment Distribution of Topic "Google"')
plt.show()
```



[26]:
```
# Filter the dataset to include only entries related to the topic 'Microsoft'
ms_data = train[train['Topic'] == 'Microsoft']

# Count the occurrences of each sentiment within the filtered dataset
sentiment_counts = ms_data['Sentiment'].value_counts()

# Plot the pie chart
plt.figure(figsize=(8, 8))
plt.pie(sentiment_counts, labels=sentiment_counts.index, autopct='%1.1f%%',
 startangle=140)
plt.title('Sentiment Distribution of Topic "Microsoft"')
plt.show()
```



[27]:
```
train['msg_len'] = train['Text'].apply(len)
```

[28]:
```
train
```

[28]:

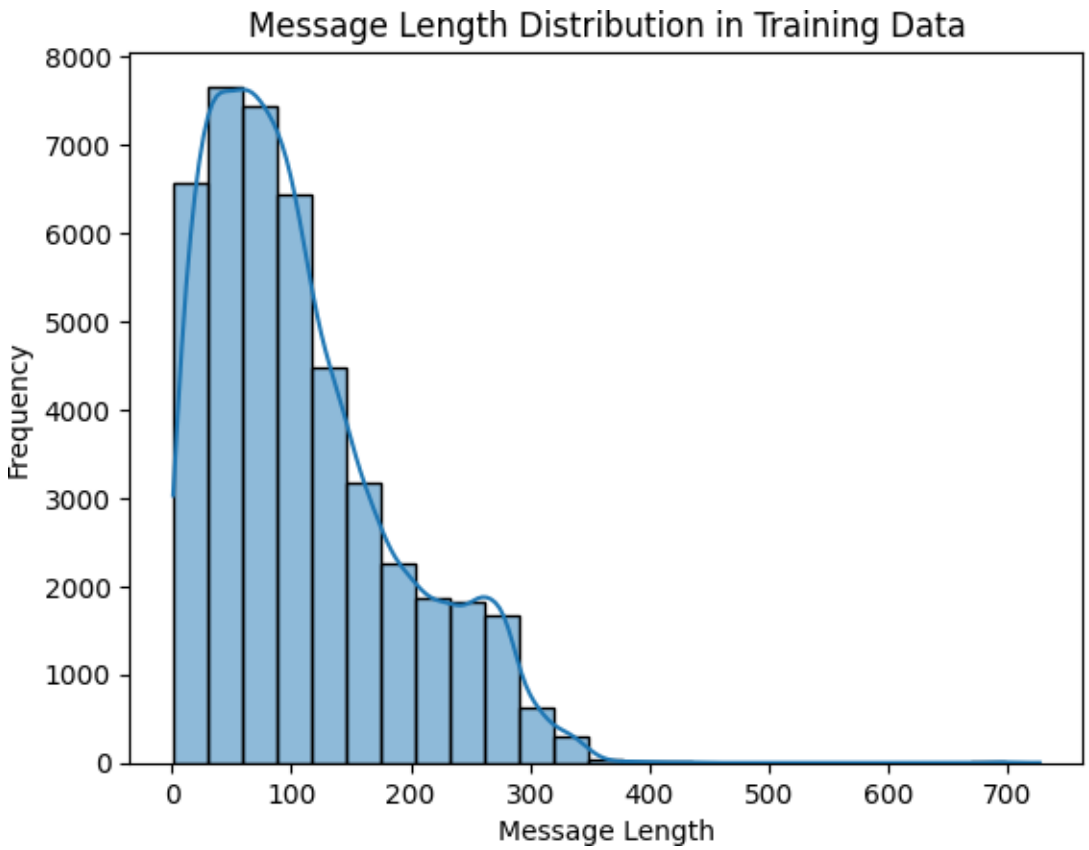
	ID	Topic	Sentiment	\
0	2401	Borderlands	Positive	
1	2401	Borderlands	Positive	
2	2401	Borderlands	Positive	
3	2401	Borderlands	Positive	
4	2401	Borderlands	Positive	
...	...	...	...	
46289	11943	Verizon	Neutral	

```
46290  11944       Verizon    Neutral
46291  11944       Verizon    Neutral
46292  11944       Verizon    Neutral
46294  11944       Verizon    Neutral
```

	Text	msg_len
0	im getting on borderlands and **i** will murder yo...	53
1	**I** am coming to the borders and **I will kill** you...	51
2	im getting on borderlands and **i** will **kill** you ...	50
3	im coming on borderlands and **i** will murder you...	51
4	im getting on borderlands 2 and **i** will murder ...	57
...	...	...
46289	some stocks play at peak interesting looking i...	138
46290	The last 3 August's **I** have broken my phone. Th...	203
46291	The last 3 August's I've broken my phone. This...	208
46292	The last time **I** broke my phone was on August 3...	168
46294	7	2

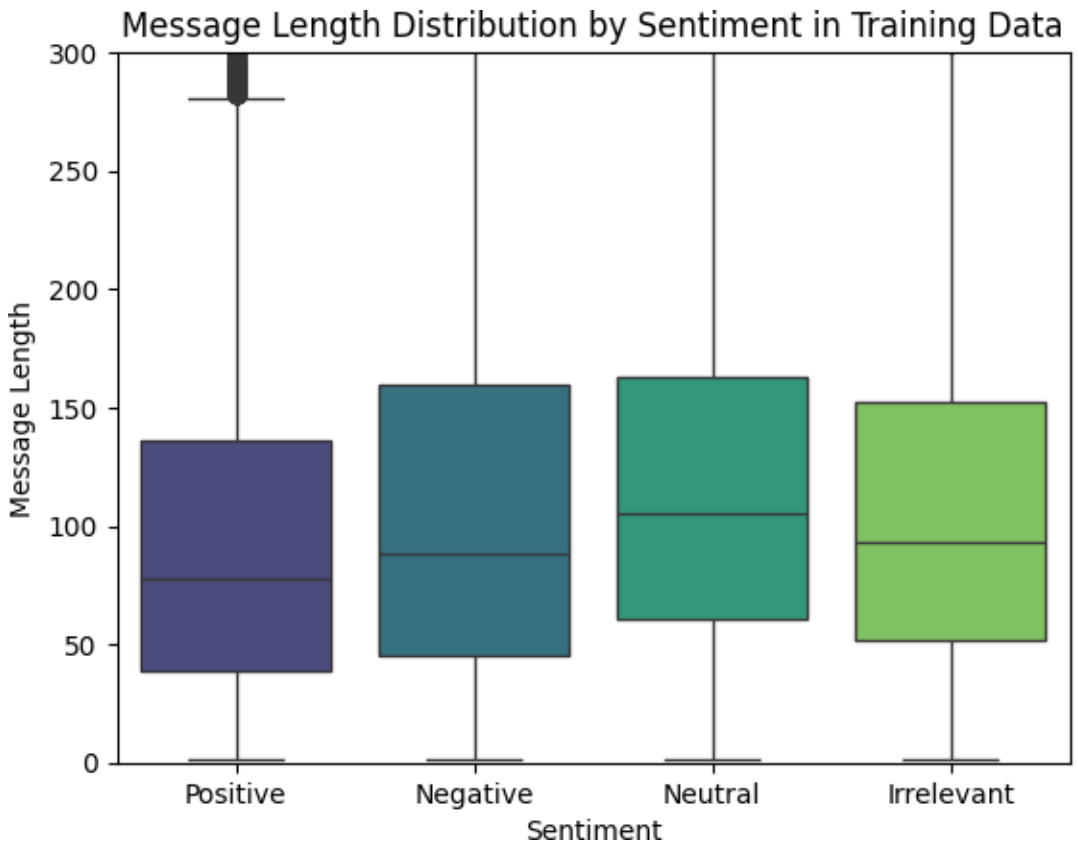
[44349 rows x 5 columns]

[29]:
```
sns.histplot(train['msg_len'], bins=25,kde=True)
plt.title('Message Length Distribution in Training Data')
plt.ylabel('Frequency')
plt.xlabel('Message  Length')
plt.show()
```

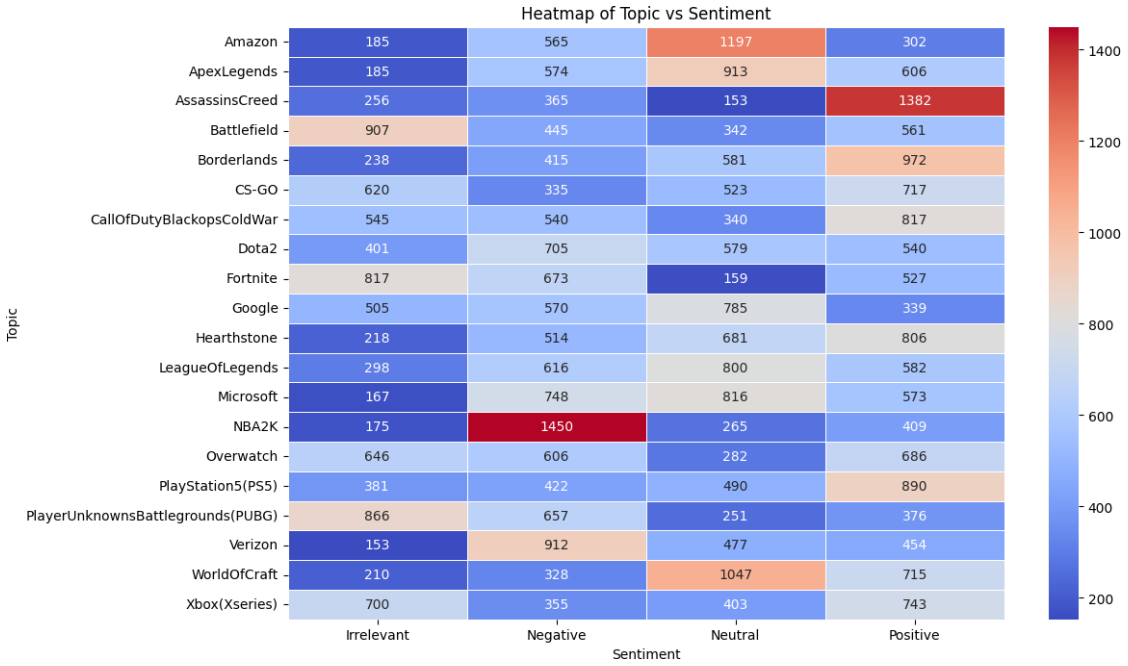


[30]:
```
sns.boxplot(data=train, x=train['Sentiment'], y='msg_len', palette='viridis',
 order=['Positive', 'Negative', 'Neutral', 'Irrelevant'])
plt.title('Message Length Distribution by Sentiment in Training Data')
plt.ylabel('Message Length')
plt.xlabel('Sentiment')
plt.ylim(0,300)
plt.show()
```

```
<ipython-input-30-ab60571ae2bf>:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(data=train, x=train['Sentiment'], y='msg_len', palette='viridis', order=['Positive', 'Negative', 'Neutral', 'Irrelevant'])
```



[31]:
```
# Create the crosstab
crosstab = pd.crosstab(index=train['Topic'], columns=train['Sentiment'])

# Plot the heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(crosstab, cmap='coolwarm', annot=True, fmt='d', linewidths=.5)

# Add labels and title
plt.title('Heatmap of Topic vs Sentiment')
plt.xlabel('Sentiment')
plt.ylabel('Topic')

# Show the plot
plt.show()
```



[32]:
```
topic_list = ' '.join(crosstab.index)

wc = WordCloud(width=1000, height=500).generate(topic_list)

plt.imshow(wc,  interpolation='bilinear')
```

[32]: <matplotlib.image.AxesImage at 0x7a62f038f4d0>

[33]:
```
corpus = ' '.join(train['Text'])

wc2 = WordCloud(width=1200, height=500).generate(corpus)

plt.imshow(wc2,  interpolation='bilinear')
```

[33]: <matplotlib.image.AxesImage at 0x7a62efdf0750>