```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')

cols= ['ID', 'Topic', 'Sentiment', 'Text']
df = pd.read_csv('/content/twitter_training.csv', names=cols, encoding
='latin-1')

df
```

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 74682,\n  \"fields\":
[\n    {\n      \"column\": \"ID\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 3740,\n        \"min\": 1,\n
\"max\": 13200,\n        \"num_unique_values\": 12447,\n
\"samples\": [\n          1616,\n          2660,\n          2335\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Topic\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
32,\n        \"samples\": [\n          \"Cyberpunk2077\",\n
\"Microsoft\",\n          \"TomClancysRainbowSix\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Sentiment\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 4,\n        \"samples\": [\n
\"Neutral\",\n          \"Irrelevant\",\n          \"Positive\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Text\",\n      \"properties\":
{\n        \"dtype\": \"string\",\n        \"num_unique_values\":
69489,\n        \"samples\": [\n          \"I \\u00e2\\u0080\\u0099 m
totally not gonna spend any more money trying on\",\n
\"Bernthal is great as Walker in Breakpoint.  \",\n          \"And
they're awesome\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df"}
```

```python
df.head(6)
```

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 74682,\n  \"fields\":
[\n    {\n      \"column\": \"ID\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 3740,\n        \"min\": 1,\n
\"max\": 13200,\n        \"num_unique_values\": 12447,\n
\"samples\": [\n          1616,\n          2660,\n          2335\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Topic\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
32,\n        \"samples\": [\n          \"Cyberpunk2077\",\n
\"Microsoft\",\n          \"TomClancysRainbowSix\"\n        ],\n
```

\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Sentiment\",\n
\"properties\": {\n   \"dtype\": \"category\",\n
\"num_unique_values\": 4,\n   \"samples\": [\n
\"Neutral\",\n   \"Irrelevant\",\n   \"Positive\"\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"Text\",\n   \"properties\":
{\n   \"dtype\": \"string\",\n   \"num_unique_values\":
69489,\n   \"samples\": [\n   \"I \\u00e2\\u0080\\u0099 m
totally not gonna spend any more money trying on\",\n
\"Bernthal is great as Walker in Breakpoint. \",\n   \"And
they're awesome\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   }\n]\
n}","type":"dataframe","variable_name":"df"}

```
df.tail()
```

{"summary":"{\n \"name\": \"df\",\n \"rows\": 5,\n \"fields\": [\n
{\n   \"column\": \"ID\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 0,\n   \"min\": 9200,\n
\"max\": 9200,\n   \"num_unique_values\": 1,\n
\"samples\": [\n   9200\n   ],\n   \"semantic_type\":
\"\",\n   \"description\": \"\"\n   }\n   },\n   {\n
\"column\": \"Topic\",\n   \"properties\": {\n   \"dtype\":
\"category\",\n   \"num_unique_values\": 1,\n   \"samples\":
[\n   \"Nvidia\"\n   ],\n   \"semantic_type\": \"\",\
n   \"description\": \"\"\n   }\n   },\n   {\n
\"column\": \"Sentiment\",\n   \"properties\": {\n
\"dtype\": \"category\",\n   \"num_unique_values\": 1,\n
\"samples\": [\n   \"Positive\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Text\",\n   \"properties\": {\n
\"dtype\": \"string\",\n   \"num_unique_values\": 5,\n
\"samples\": [\n   \"Just realized that my Mac window partition
is 6 years behind on Nvidia drivers and I have no idea how I didn't
notice\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe"}

```
df.shape
```

```
(74682, 4)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 74682 entries, 0 to 74681
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   ID         74682 non-null  int64
 1   Topic      74682 non-null  object
 2   Sentiment  74682 non-null  object
 3   Text       73996 non-null  object
dtypes: int64(1), object(3)
memory usage: 2.3+ MB
```

```
df.describe(include= 'object')
```

{"summary":"{\n \"name\": \"df\",\n \"rows\": 4,\n \"fields\": [\n   {\n   \"column\": \"Topic\",\n   \"properties\": {\n   \"dtype\": \"string\",\n   \"num_unique_values\": 4,\n   \"samples\": [\n   32,\n   \"2400\",\n   \"74682\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Sentiment\",\n   \"properties\": {\n   \"dtype\": \"string\",\n   \"num_unique_values\": 4,\n   \"samples\": [\n   4,\n   \"22542\",\n   \"74682\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Text\",\n   \"properties\": {\n   \"dtype\": \"string\",\n   \"num_unique_values\": 4,\n   \"samples\": [\n   69489,\n   \"172\",\n   \"73996\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe"}

```
df['Sentiment'].unique()
```

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

```
#checking for missing values in the dataset
df.isnull().sum()
```

```
ID             0
Topic          0
Sentiment      0
Text         686
dtype: int64
```

```
df.dropna(inplace=True)
```

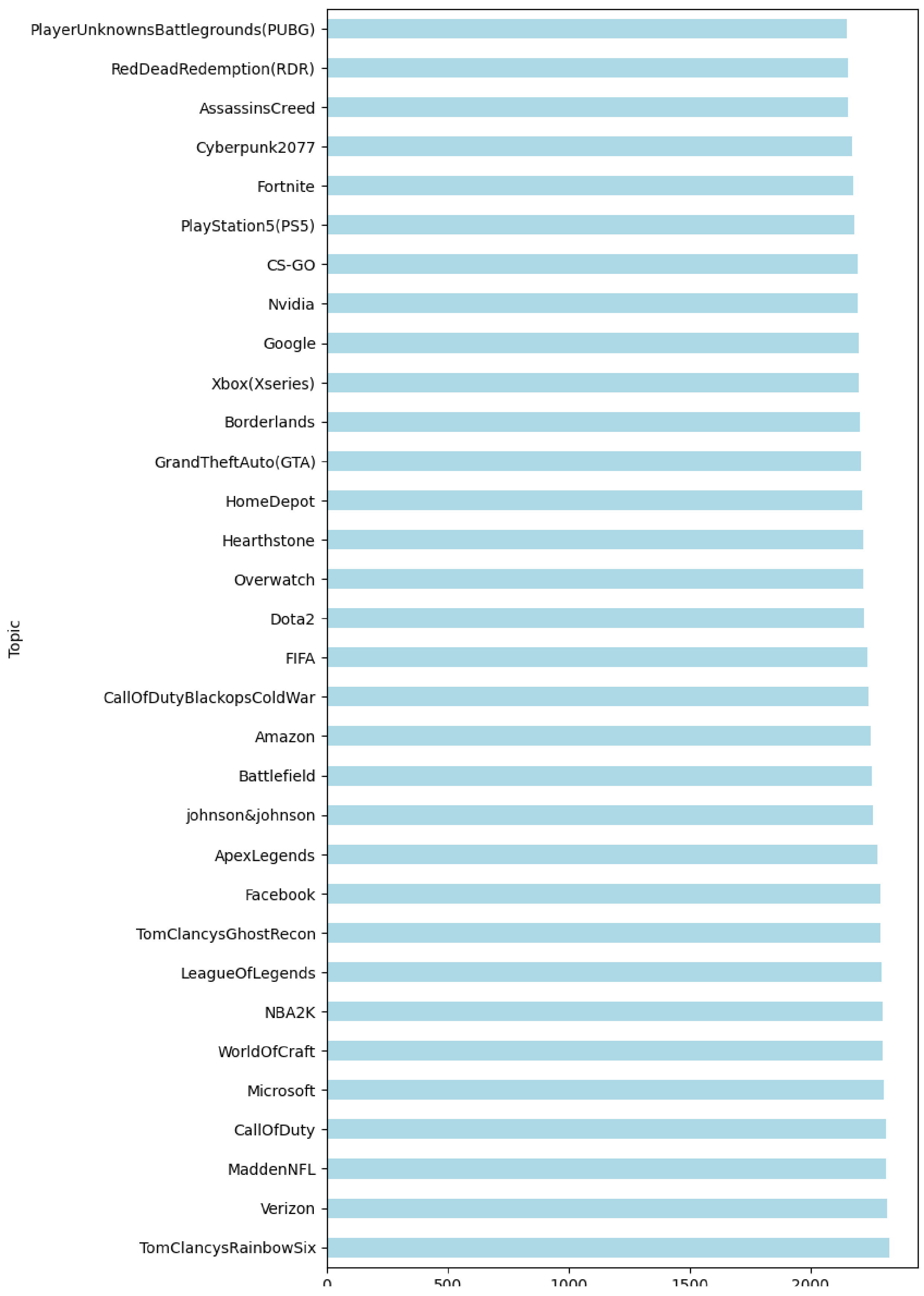
```
df.isnull().sum()
```

```
ID           0
Topic        0
Sentiment    0
Text         0
dtype: int64
```

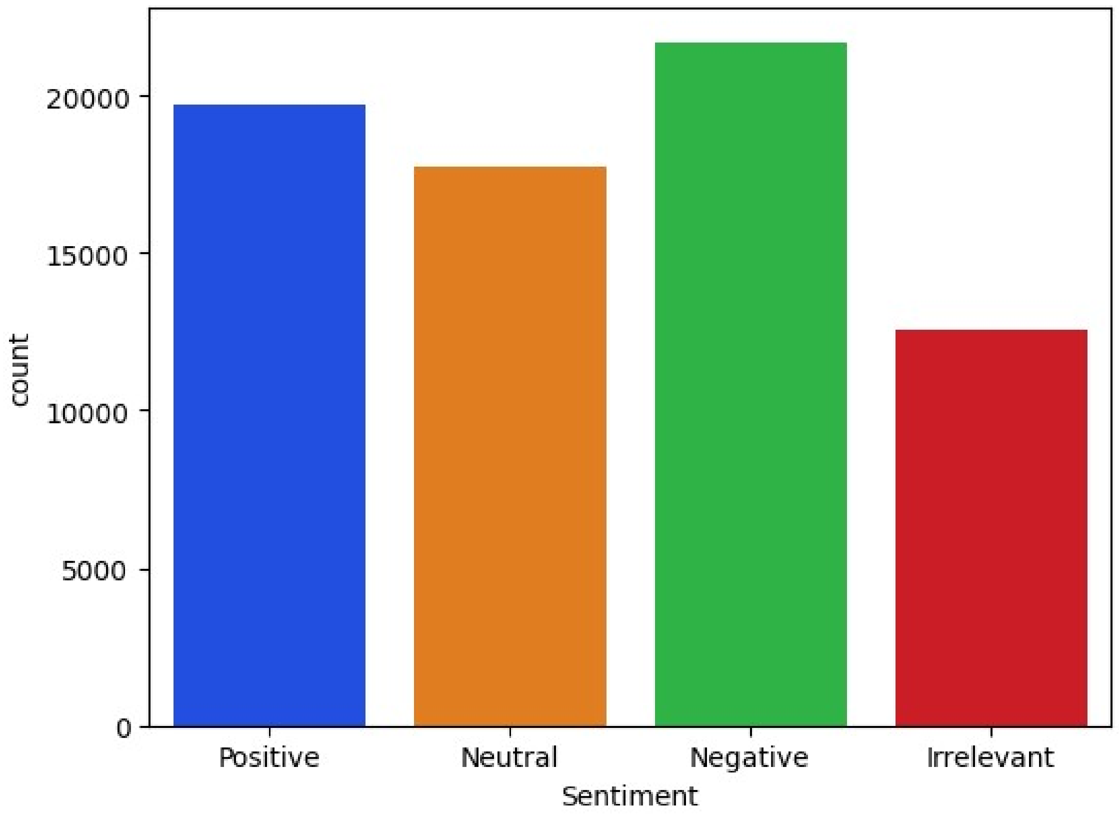
```
#checking for duplicate values
df.duplicated().sum()
```

```
2341
```

```
df.drop_duplicates(inplace=True)

df.duplicated().sum()

0

#Visualization of count of different topics
plt.figure(figsize=(7,15))
df['Topic'].value_counts().plot(kind='barh',color= 'lightBlue')
plt.xlabel("Count")
plt.show()
```


Topic
PlayerUnknownsBattlegrounds(PUBG)
RedDeadRedemption(RDR)
AssassinsCreed
Cyberpunk2077
Fortnite
PlayStation5(PS5)
CS-GO
Nvidia
Google
Xbox(Xseries)
Borderlands
GrandTheftAuto(GTA)
HomeDepot
Hearthstone
Overwatch
Dota2
FIFA
CallOfDutyBlackopsColdWar
Amazon
Battlefield
johnson&johnson
ApexLegends
Facebook
TomClancysGhostRecon
LeagueOfLegends
NBA2K
WorldOfCraft
Microsoft
CallOfDuty
MaddenNFL
Verizon
TomClancysRainbowSix
0
500
1000
1500
2000

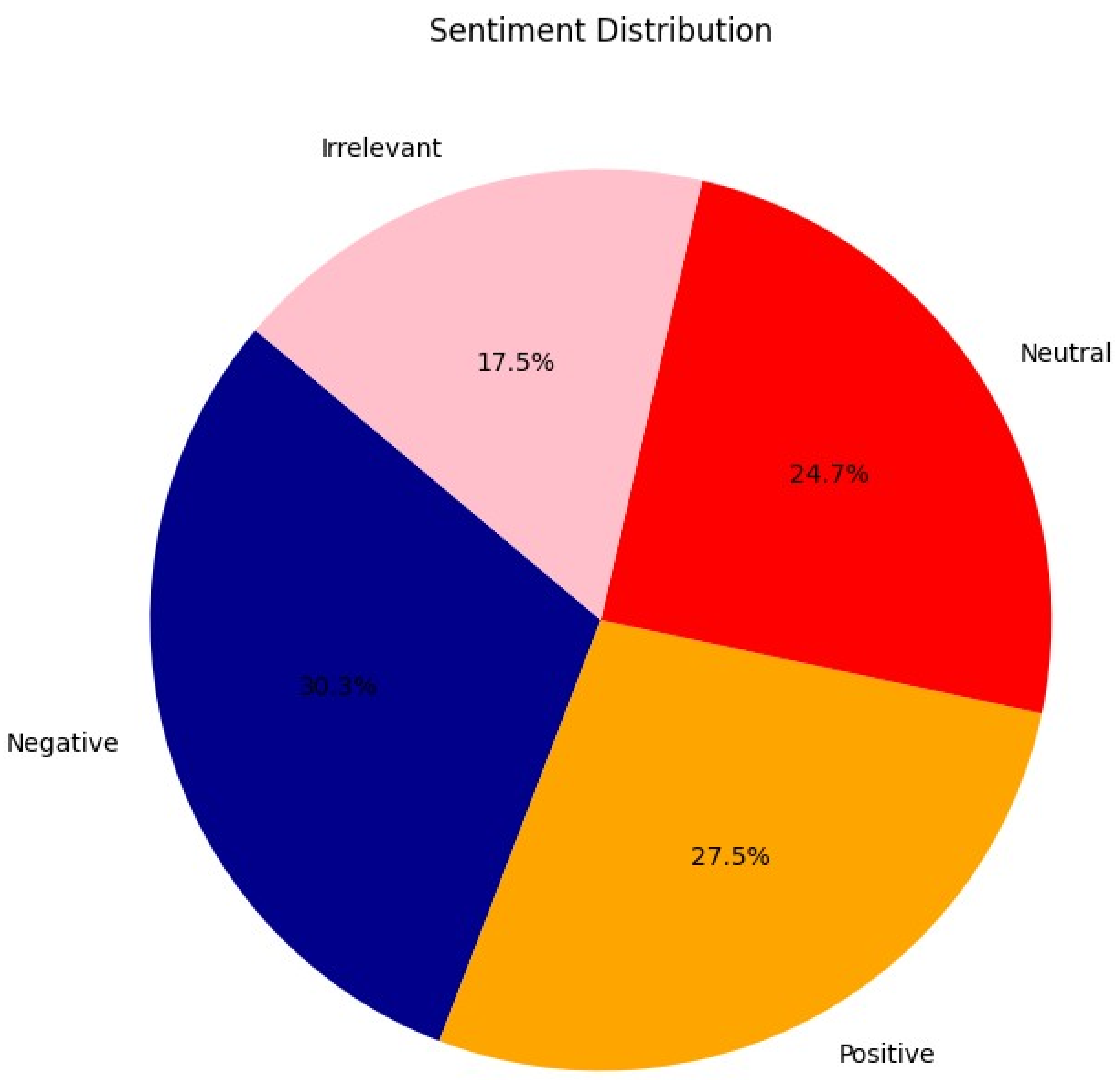
```
#Sentiment distribution
sns.countplot(x= 'Sentiment', data= df, palette= 'bright')
plt.show()
```



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

```
Sentiment
Negative      21698
Positive      19713
Neutral       17707
Irrelevant    12537
Name: count, dtype: int64
```

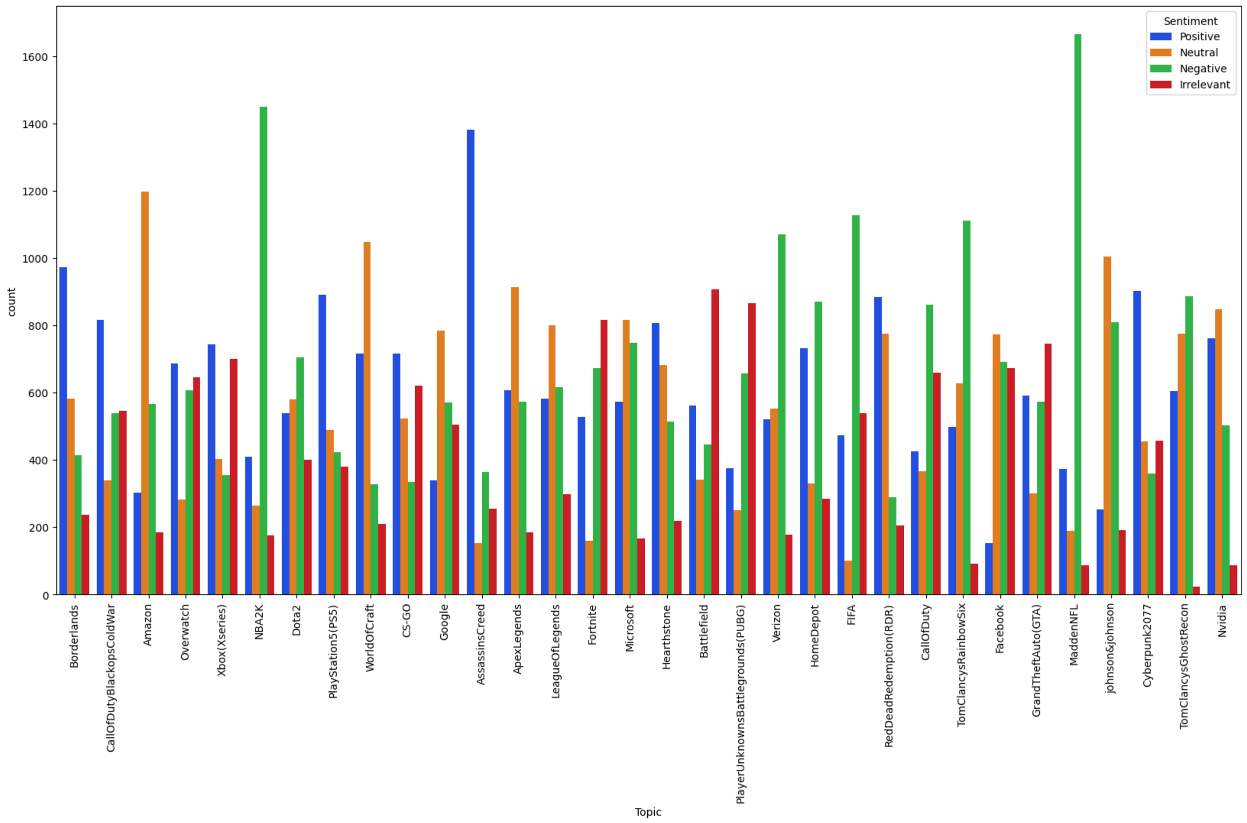
```
#Create the pie chart
plt.figure(figsize=(8,8))
plt.pie(sentiment_counts, labels= sentiment_counts.index,
autopct='%1.1f%%', startangle=140, colors=['darkblue','orange', 'red',
'pink'])
plt.title('Sentiment Distribution')
plt.show()
```



```
df
```

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 71655,\n  \"fields\":
[\n    {\n      \"column\": \"ID\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 3742,\n        \"min\": 1,\n
\"max\": 13200,\n        \"num_unique_values\": 12447,\n
\"samples\": [\n          1616,\n          2660,\n          2335\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Topic\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
32,\n        \"samples\": [\n          \"Cyberpunk2077\",\n
\"Microsoft\",\n          \"TomClancysRainbowSix\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Sentiment\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 4,\n        \"samples\": [\n
\"Neutral\",\n          \"Irrelevant\",\n          \"Positive\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Text\",\n      \"properties\":
{\n        \"dtype\": \"string\",\n        \"num_unique_values\":
69489,\n        \"samples\": [\n          \"I \\u00e2\\u0080\\u0099 m
totally not gonna spend any more money trying on\",\n
\"Bernthal is great as Walker in Breakpoint.  \",\n          \"And
they're awesome\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df"}
```

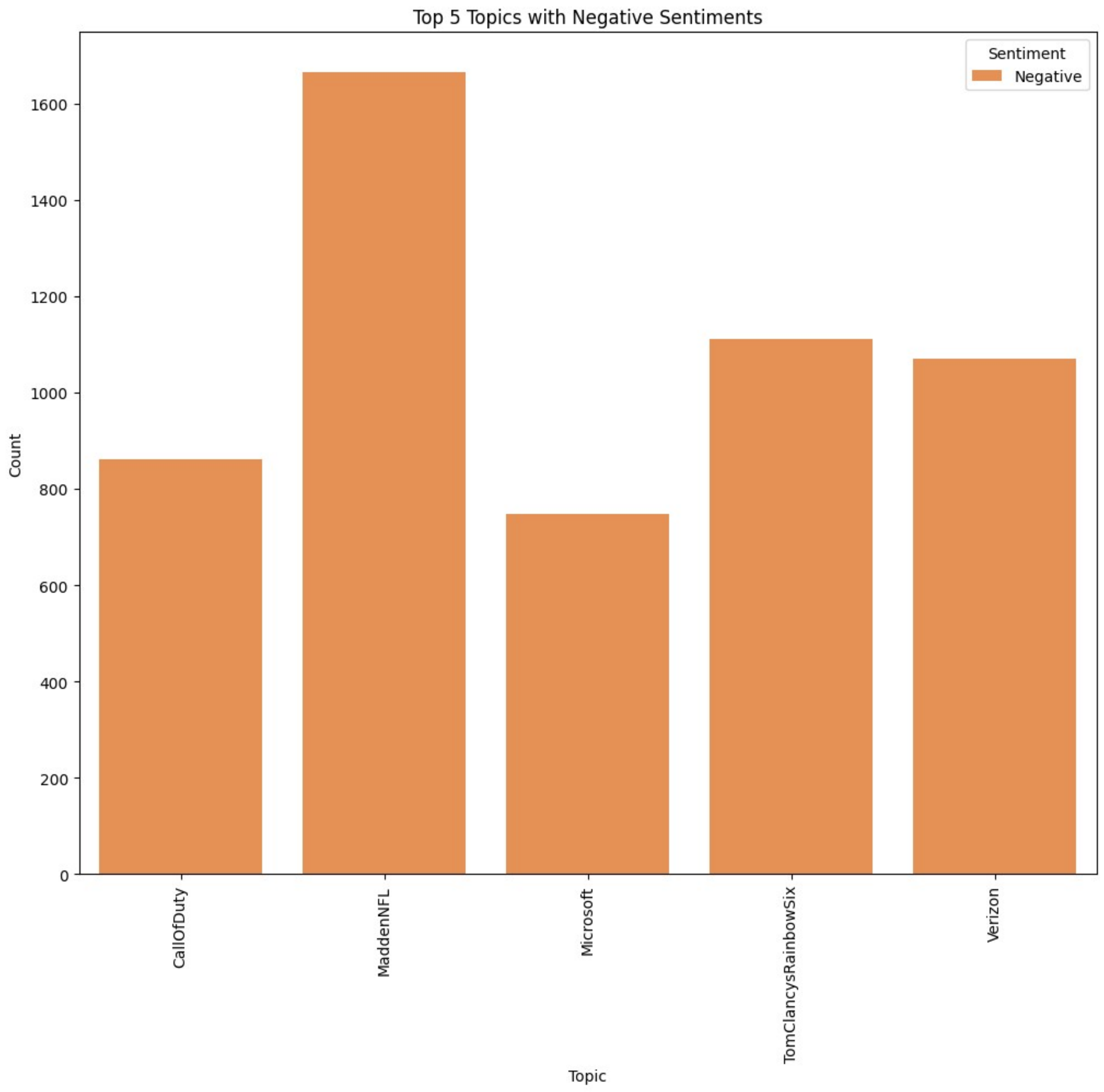
```
#Sentiment Distribution Topic-Wise
plt.figure(figsize = (20,10))
sns.countplot(x= 'Topic', hue= 'Sentiment', data= df, palette=
'bright')
plt.xticks(rotation=90)
plt.show()
```



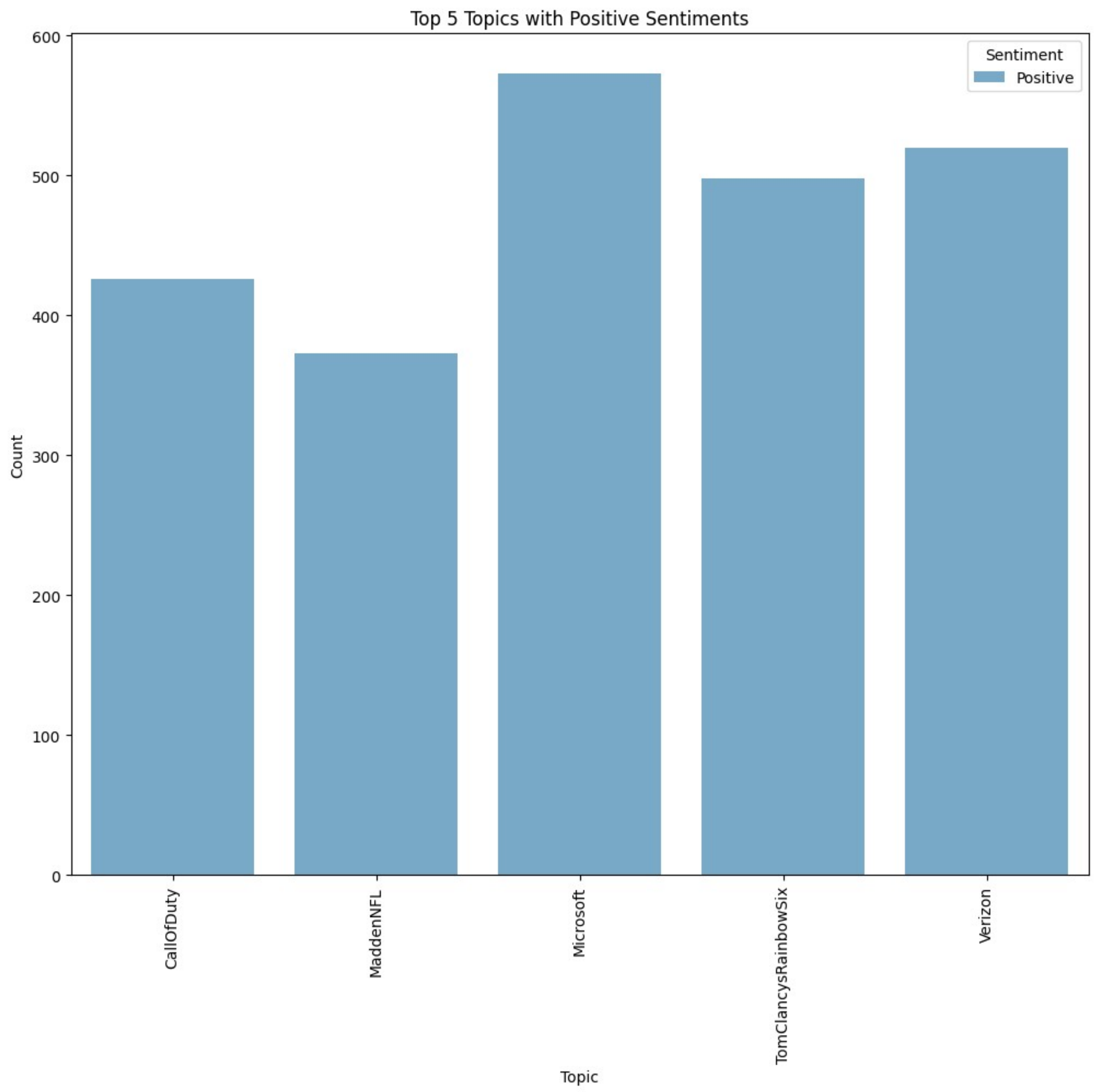
```
##Group by topic and Sentiment
topic_wise_sentiment = df.groupby(['Topic',
'Sentiment']).size().reset_index(name='Count')
topic_wise_sentiment

#step2: Select top 5 topics
topic_counts= df['Topic'].value_counts().nlargest(5).index
top_topics_sentiment=
topic_wise_sentiment[topic_wise_sentiment['Topic'].isin(topic_counts)]
top_topics_sentiment
```

{"summary":"{\n \"name\": \"top_topics_sentiment\",\n \"rows\": 20,\
n \"fields\": [\n   {\n   \"column\": \"Topic\",\n
\"properties\": {\n   \"dtype\": \"category\",\n
\"num_unique_values\": 5,\n   \"samples\": [\n
\"MaddenNFL\",\n   \"Verizon\",\n   \"Microsoft\"\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"Sentiment\",\n
\"properties\": {\n   \"dtype\": \"category\",\n
\"num_unique_values\": 4,\n   \"samples\": [\n
\"Negative\",\n   \"Positive\",\n   \"Irrelevant\"\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"Count\",\n   \"properties\":
{\n   \"dtype\": \"number\",\n   \"std\": 394,\n
\"min\": 86,\n   \"max\": 1665,\n   \"num_unique_values\":
20,\n   \"samples\": [\n   660,\n   1070,\n
498\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   }\n]\
n}","type":"dataframe","variable_name":"top_topics_sentiment"}

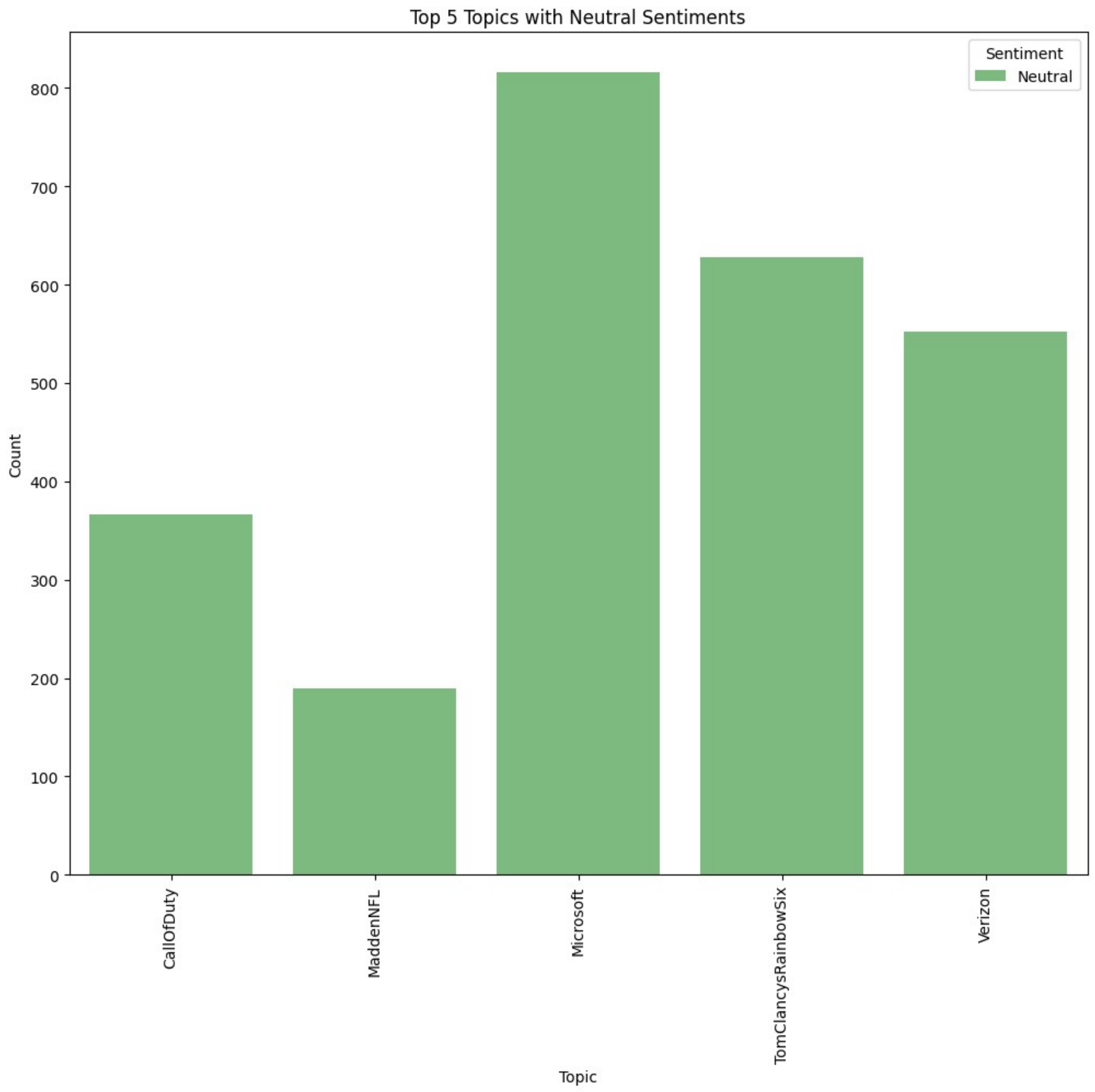
```
#TOP 5 topics with neagtive sentiment
plt.figure(figsize=(12,10))
sns.barplot(x='Topic', y='Count', hue='Sentiment',palette='Oranges',
data=
top_topics_sentiment[top_topics_sentiment['Sentiment']=='Negative'])
plt.xlabel('Topic')
plt.ylabel('Count')
plt.title('Top 5 Topics with Negative Sentiments')
plt.xticks(rotation=90)
plt.show

<function matplotlib.pyplot.show(close=None, block=None)>
```



```python
#top 5 topics with positive sentiments
plt.figure(figsize=(12,10))
sns.barplot(x='Topic', y='Count', hue='Sentiment',palette='Blues',
data=
top_topics_sentiment[top_topics_sentiment['Sentiment']=='Positive'])
plt.xlabel('Topic')
plt.ylabel('Count')
plt.title('Top 5 Topics with Positive Sentiments')
plt.xticks(rotation=90)
plt.show

<function matplotlib.pyplot.show(close=None, block=None)>
```

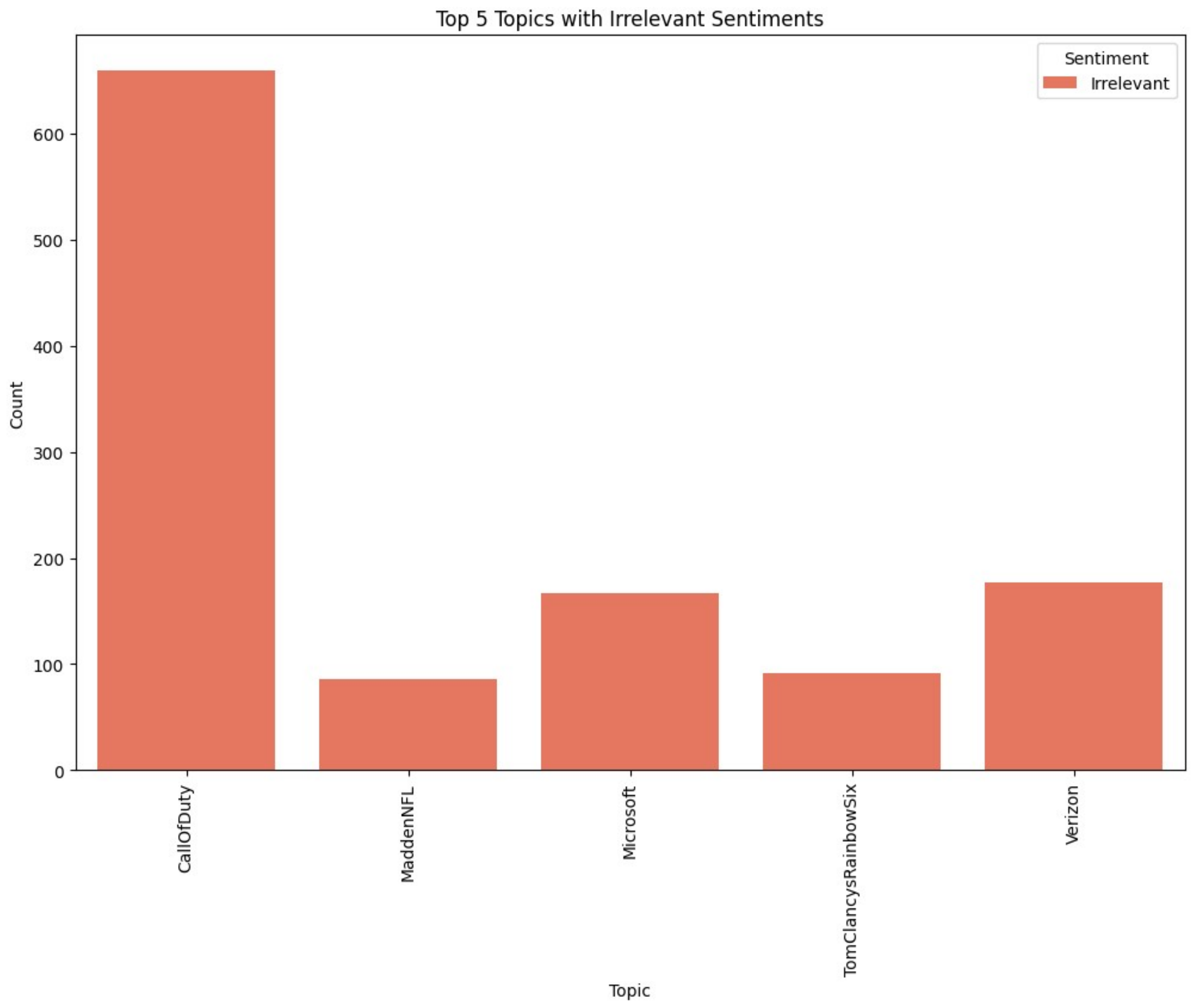


```
#Top 5 Topics with neutral sentiments
plt.figure(figsize=(12,10))
sns.barplot(x='Topic', y='Count', hue='Sentiment',palette='Greens',
data=
top_topics_sentiment[top_topics_sentiment['Sentiment']=='Neutral'])
plt.xlabel('Topic')
plt.ylabel('Count')
plt.title('Top 5 Topics with Neutral Sentiments')
plt.xticks(rotation=90)
plt.show

<function matplotlib.pyplot.show(close=None, block=None)>
```



```
#Top 5 Topics with Irrelevant Sentiments
plt.figure(figsize=(12,8))
sns.barplot(x='Topic', y='Count', hue='Sentiment',palette='Reds',
data=
top_topics_sentiment[top_topics_sentiment['Sentiment']=='Irrelevant'])
plt.xlabel('Topic')
plt.ylabel('Count')
plt.title('Top 5 Topics with Irrelevant Sentiments')
plt.xticks(rotation=90)
plt.show
```

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



```
#sentiment distribution in google
#filter the dataset to include only entries related to the topic
'Google'
google_data = df[df['Topic'] == 'Google']
google_data
```

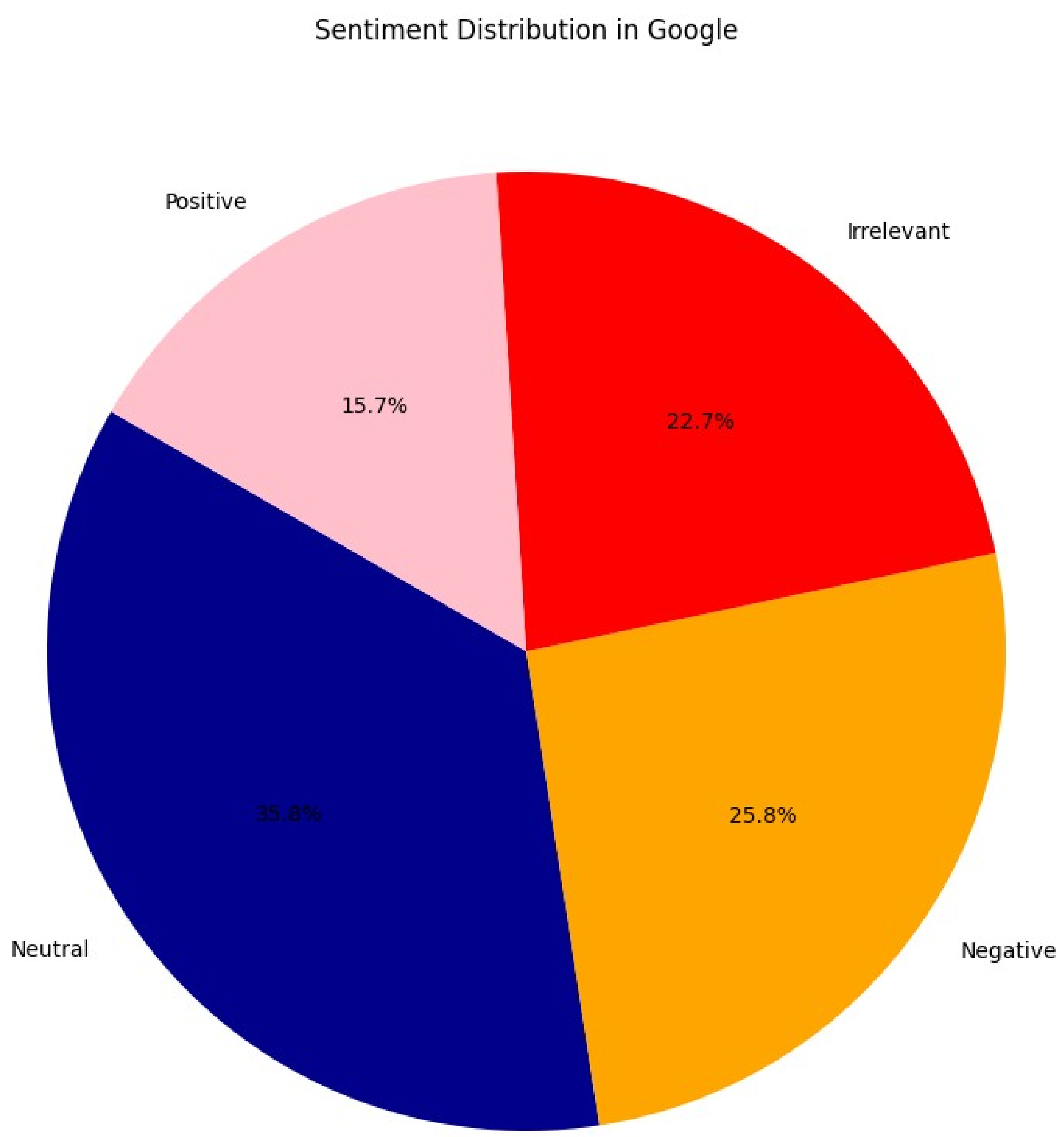
{"summary":"{\n \"name\": \"google_data\",\n \"rows\": 2298,\n \"fields\": [\n   {\n   \"column\": \"ID\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 116,\n   \"min\": 4401,\n   \"max\": 4800,\n   \"num_unique_values\": 383,\n   \"samples\": [\n   4684,\n   4666,\n   4734\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Topic\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 1,\n   \"samples\": [\n   \"Google\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Sentiment\",\n   \"properties\": {\n   \"dtype\":

```
\"category\",\n        \"num_unique_values\": 4,\n        \"samples\":
[\n          \"Irrelevant\"\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Text\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 2172,\n
\"samples\": [\n          \"Check out this review by Lopez & Co on
Google Maps. goo.gl / maps / 524QpyVKM..... THE MOST EXPERIENCED. TAX
PREPARATION CENTRE WE CAN TRUST.\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe","variable_name":"google_data"}
```

```
#count the occurences of each sentiment within the filtered dataset
sentiment_counts_google = google_data['Sentiment'].value_counts()
sentiment_counts_google
```

```
Sentiment
Neutral       822
Negative      594
Irrelevant    522
Positive      360
Name: count, dtype: int64
```

```
#plot the pie chart
plt.figure(figsize=(10,10))
plt.pie(sentiment_counts_google, labels=
sentiment_counts_google.index, autopct='%1.1f%%', startangle=150,
colors=['darkblue','orange', 'red', 'pink'])
plt.title('Sentiment Distribution in Google')
plt.show()
```



```
##sentiment distribution in Microsoft
microsoft_data = df[df['Topic'] == 'Microsoft']
microsoft_data
```

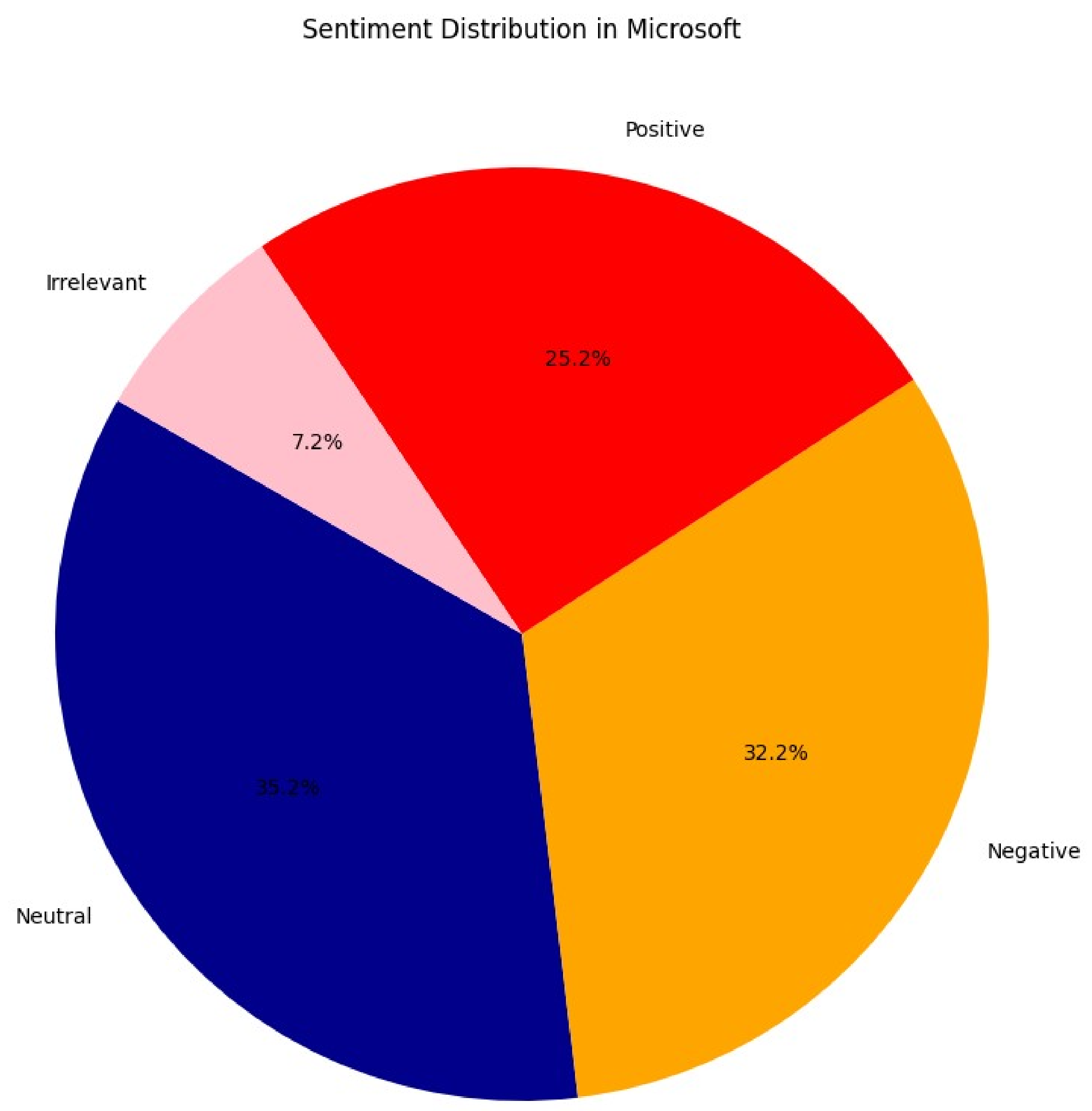
{"summary":"{\n \"name\": \"microsoft_data\",\n \"rows\": 2400,\n \"fields\": [\n   {\n   \"column\": \"ID\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 115,\n   \"min\": 8001,\n   \"max\": 8400,\n   \"num_unique_values\": 400,\n   \"samples\": [\n   8210,\n   8281,\n   8034\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\":

\"Topic\",\n   \"properties\": {\n   \"dtype\": \"category\",\
n   \"num_unique_values\": 1,\n   \"samples\": [\n
\"Microsoft\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Sentiment\",\n   \"properties\": {\n   \"dtype\":
\"category\",\n   \"num_unique_values\": 4,\n   \"samples\":
[\n   \"Neutral\"\n   ],\n   \"semantic_type\":
\"\",\n   \"description\": \"\"\n   }\n   },\n   {\n
\"column\": \"Text\",\n   \"properties\": {\n   \"dtype\":
\"string\",\n   \"num_unique_values\": 2263,\n
\"samples\": [\n   \"Well, it definitely cheered me up!.\"\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   }\n]\n}","type":"dataframe","variable_name":"microsoft_data"}

```
sentiment_counts_microsoft =
microsoft_data['Sentiment'].value_counts()
sentiment_counts_microsoft
```

```
Sentiment
Neutral       846
Negative      774
Positive      606
Irrelevant    174
Name: count, dtype: int64
```

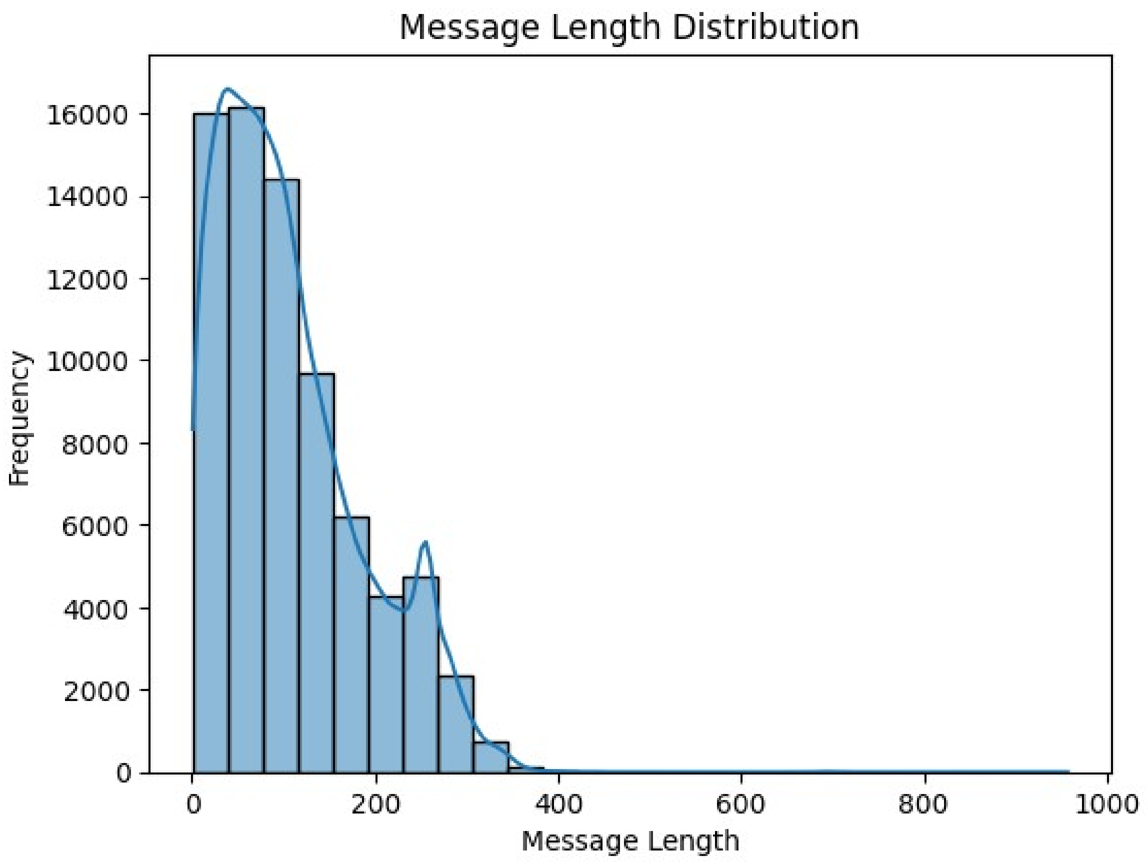
```
#plot the pie chart
plt.figure(figsize=(10,10))
plt.pie(sentiment_counts_microsoft, labels=
sentiment_counts_microsoft.index, autopct='%1.1f%%', startangle=150,
colors=['darkblue','orange', 'red', 'pink'])
plt.title('Sentiment Distribution in Microsoft')
plt.show()
```



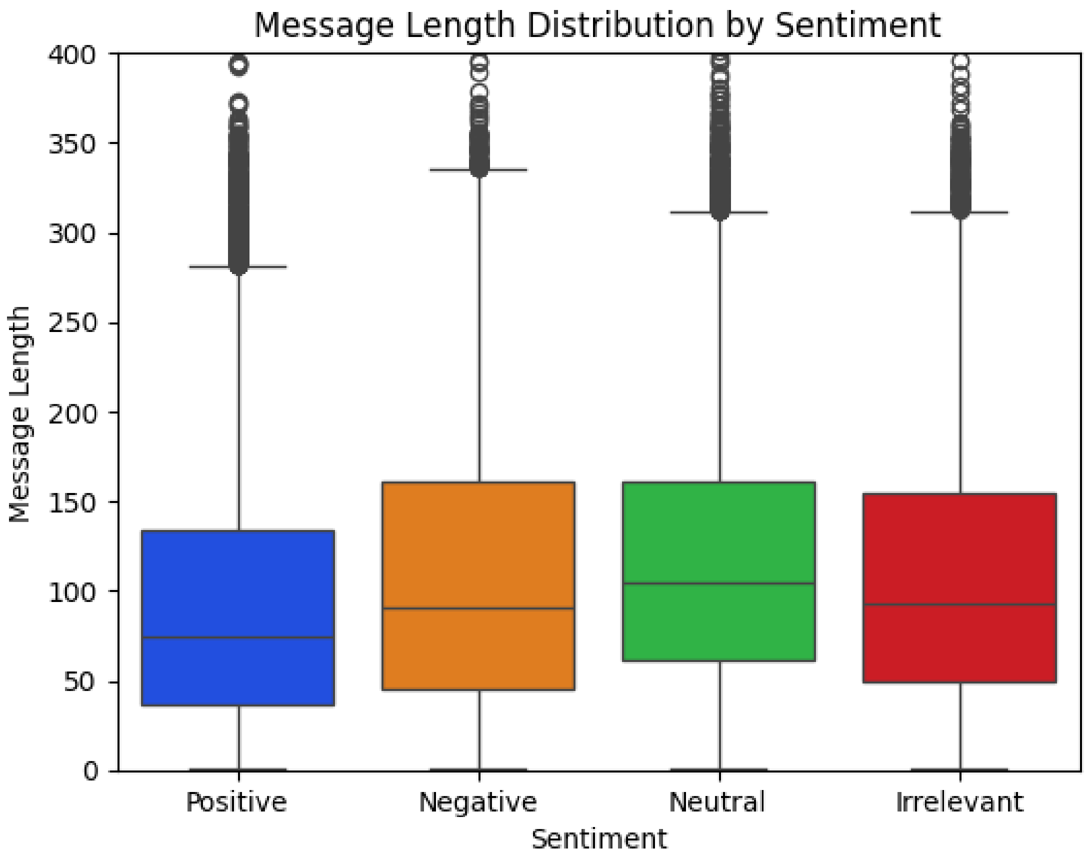
```
df['msg_len'] = df['Text'].astype(str).apply(len)
df
```

{"summary":"{\n \"name\": \"df\",\n \"rows\": 74682,\n \"fields\": [\n   {\n   \"column\": \"ID\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 3740,\n   \"min\": 1,\n   \"max\": 13200,\n   \"num_unique_values\": 12447,\n   \"samples\": [\n   1616,\n   2660,\n   2335\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Topic\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\":

```
32,\n        \"samples\": [\n          \"Cyberpunk2077\",\n
\"Microsoft\",\n          \"TomClancysRainbowSix\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Sentiment\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 4,\n        \"samples\": [\n
\"Neutral\",\n          \"Irrelevant\",\n          \"Positive\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Text\",\n      \"properties\":
{\n        \"dtype\": \"string\",\n        \"num_unique_values\":
69489,\n        \"samples\": [\n          \"I \\u00e2\\u0080\\u0099 m
totally not gonna spend any more money trying on\",\n
\"Bernthal is great as Walker in Breakpoint.  \",\n          \"And
they're awesome\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"msg_len\",\n      \"properties\": {\n        \"dtype\": \"number\",\
n        \"std\": 79,\n        \"min\": 1,\n        \"max\": 957,\n
\"num_unique_values\": 405,\n        \"samples\": [\n          18,\n
259,\n          408\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df"}

#Plot of message length disrtibution for training data
sns.histplot(df['msg_len'], kde=True, bins = 25)
plt.title('Message Length Distribution')
plt.xlabel('Message Length')
plt.ylabel('Frequency')
plt.show()
```



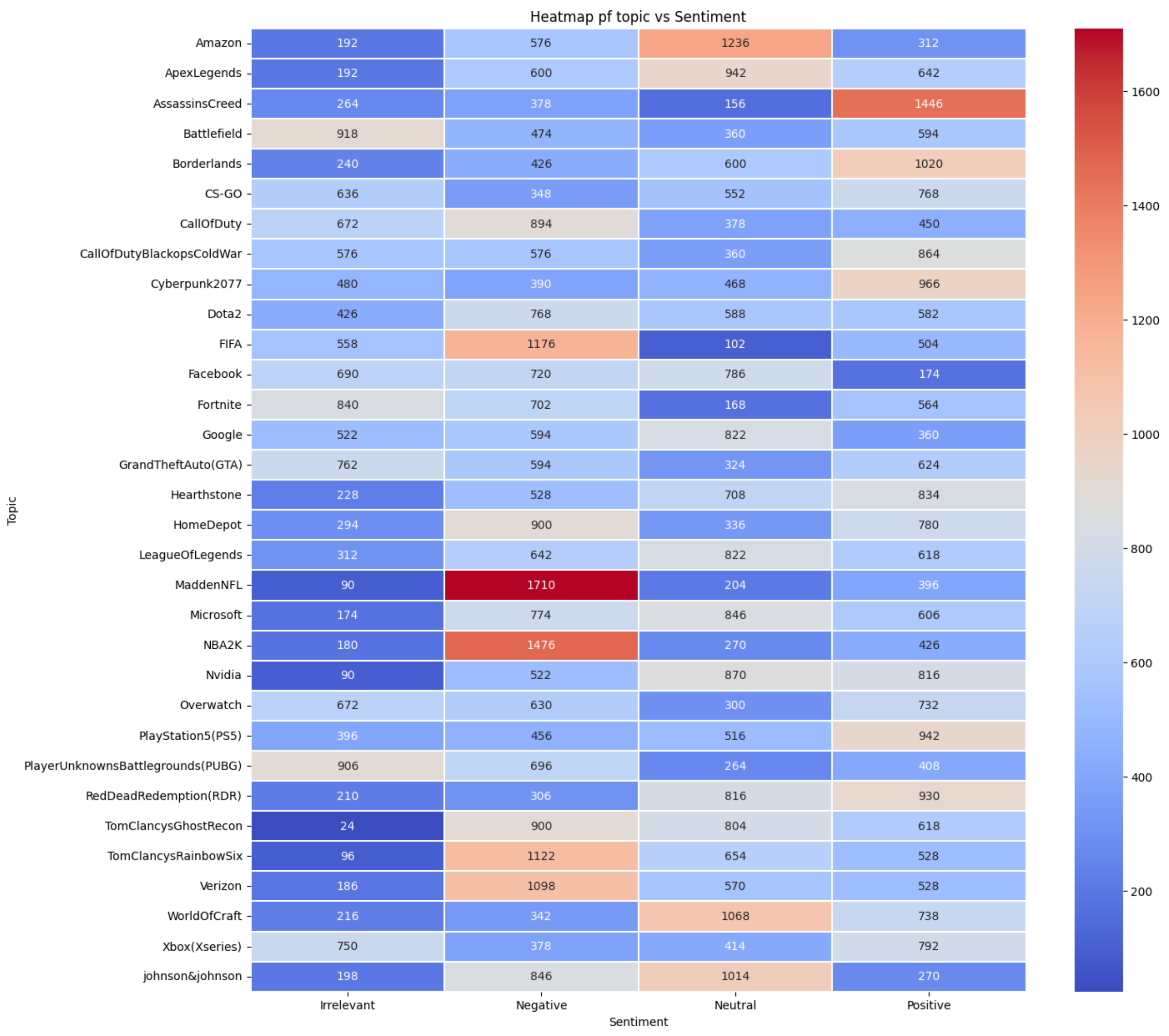
```
#Plot message length distribution by sentiment for training data
sns.boxplot(x='Sentiment', y='msg_len', data=df, palette = 'bright',
order=['Positive', 'Negative', 'Neutral', 'Irrelevant'])
plt.title('Message Length Distribution by Sentiment')
plt.xlabel('Sentiment')
plt.ylabel('Message Length')
plt.ylim(0,400)
plt.show()
```



```
#Create the Crosstab
crosstab = pd.crosstab(index=df['Topic'], columns=df['Sentiment'])
crosstab
```

```
{"summary":"{\n  \"name\": \"crosstab\",\n  \"rows\": 32,\n
\"fields\": [\n    {\n      \"column\": \"Topic\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 32,\n        \"samples\": [\n
\"WorldOfCraft\",\n          \"Hearthstone\",\n
\"PlayerUnknownsBattlegrounds(PUBG)\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Irrelevant\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
266,\n        \"min\": 24,\n        \"max\": 918,\n
\"num_unique_values\": 29,\n        \"samples\": [\n          750,\n
312,\n          522\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Negative\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 329,\n        \"min\": 306,\n
\"max\": 1710,\n        \"num_unique_values\": 28,\n
\"samples\": [\n          1176,\n          1098,\n          768\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Neutral\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
296,\n        \"min\": 102,\n        \"max\": 1236,\n
\"num_unique_values\": 30,\n        \"samples\": [\n          1068,\n
336,\n          816\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Positive\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 257,\n        \"min\": 174,\n
\"max\": 1446,\n        \"num_unique_values\": 30,\n
\"samples\": [\n          738,\n          834,\n          942\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    }\n  ]\n}","type":"dataframe","variable_name":"crosstab"}
```

```
#Plot the heatmap
plt.figure(figsize=(15,15))
sns.heatmap(crosstab, annot=True, cmap='coolwarm', fmt= 'd',
linewidths=.10)
plt.title('Heatmap pf topic vs Sentiment')
plt.xlabel('Sentiment')
plt.ylabel('Topic')
plt.show()
```



```
topic_list = ' '.join(crosstab.index)
topic_list
```

```
{"type":"string"}
```

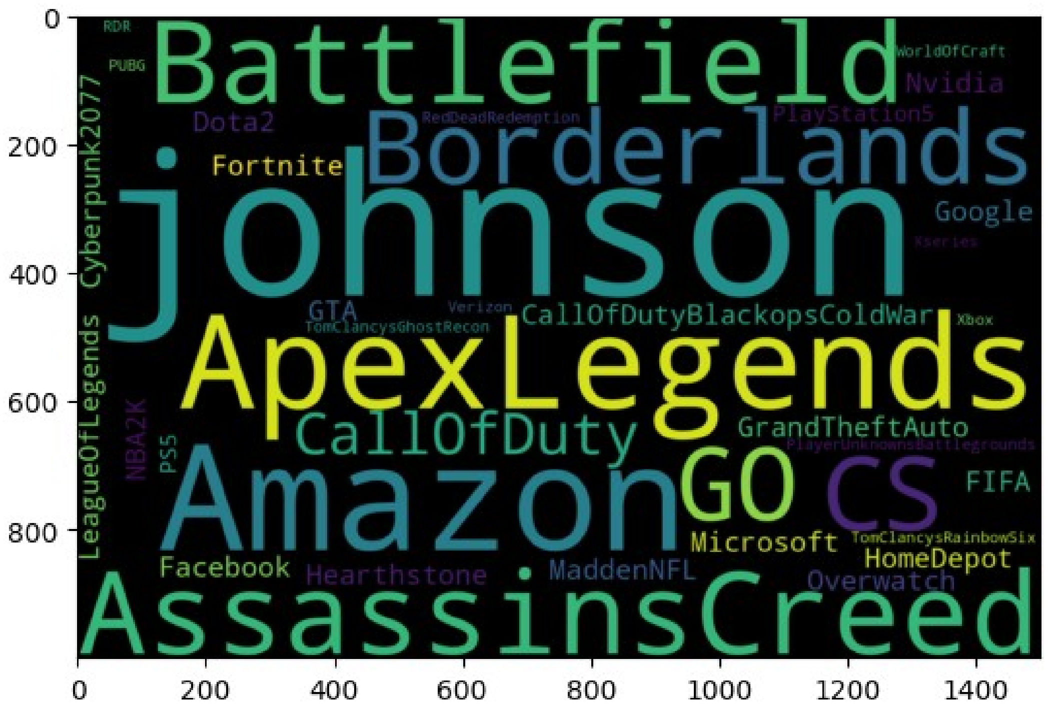
```
pip install WordCloud
```

```
Requirement already satisfied: WordCloud in
/usr/local/lib/python3.10/dist-packages (1.9.3)
Requirement already satisfied: numpy>=1.6.1 in
/usr/local/lib/python3.10/dist-packages (from WordCloud) (1.26.4)
Requirement already satisfied: pillow in
/usr/local/lib/python3.10/dist-packages (from WordCloud) (9.4.0)
Requirement already satisfied: matplotlib in
/usr/local/lib/python3.10/dist-packages (from WordCloud) (3.7.1)
Requirement already satisfied: contourpy>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->WordCloud)
```

```
(1.3.0)
Requirement already satisfied: cycler>=0.10 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->WordCloud)
(0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->WordCloud)
(4.53.1)
Requirement already satisfied: kiwisolver>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->WordCloud)
(1.4.7)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->WordCloud)
(24.1)
Requirement already satisfied: pyparsing>=2.3.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->WordCloud)
(3.1.4)
Requirement already satisfied: python-dateutil>=2.7 in
/usr/local/lib/python3.10/dist-packages (from matplotlib->WordCloud)
(2.8.2)
Requirement already satisfied: six>=1.5 in
/usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7-
>matplotlib->WordCloud) (1.16.0)
```

```python
from wordcloud import WordCloud

wc= WordCloud(width=1500, height=1000). generate(topic_list)
plt.imshow(wc, interpolation='bilinear')
```

```
<matplotlib.image.AxesImage at 0x784a28f8c4c0>
```

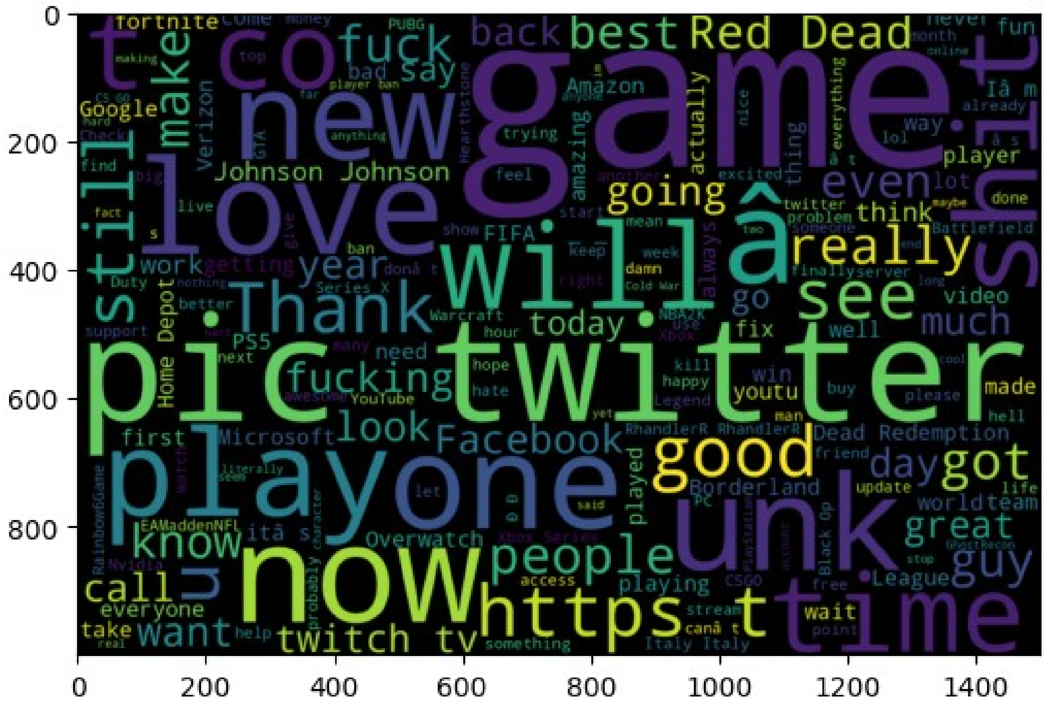


```
df['Text'] = df['Text'].astype(str)
corpus = ' '.join(df['Text'])
corpus
```

```
{"type":"string"}
```

```
wc2= WordCloud(width=1500, height=1000). generate(corpus)
plt.imshow(wc2, interpolation='bilinear')
```

```
<matplotlib.image.AxesImage at 0x784a28f8a0e0>
```


game
pic twitter
playone
unk
now
love
new
will
Thank
time
https
good
shit
co
fuck
Red Dead
best
really
see
still
make
Johnson Johnson
going
even
Facebook
people
fucking
twitch tv
know
got
great
guy
look
much
today
year
back
call
want
Microsoft
Dead Redemption
Borderland
Overwatch
Home Depot
Verizon
Google
Amazon
fortnite
0
200
400
600
800
1000
1200
1400